UNIVERSIDAD DE CANTABRIA

E.T.S. DE INGENIEROS DE CAMINOS, CANALES Y PUERTOS
Departamento de Matemática Aplicada y Ciencias de la Computación

Tesis Doctoral

Algunas herramientas estadísticas y matemáticas para la modelización del tráfico

Aida Calviño Martínez

Dirigido por **Enrique Castillo Ron**

Santander, Enero 2013

Fotografías de portada y contraportada: la autora

To $\boldsymbol{\pi}$,
for her endless and irrational love

Acknowledgments

This thesis would not have been possible without the support of many people who have helped me grow in academic and personal level. Thus, I am happy to have this section to thank all of them.

First of all, I am very grateful to Enrique Castillo, my thesis director, for sharing with me his way of understanding teaching, research and life in general. I was very fortunate to have him as a mentor. To him, for all the time he has spent with me, teaching and helping me grow. Enrique has also given me the opportunity to meet Maria, my officemate, whose help and little talks I will never forget. It has been a pleasure to share our little office and all these long hours with her.

I would also like to thank my mother, whom I owe everything and who taught me how to be a better person. For her never ending love and unconditional support, for making me feel capable of everything and being an example of hard work and perseverance, there is nothing left to say but thank you.

I cannot forget my partner, Raúl. Thank you for being there and encourage me every time I needed, for making me laugh and never stop smiling, and, overall, for leaving everything behind just to be with me.

To my *family*, Randi and Patricia, for acting like that. It is an honor for me to have people like them around me, that care about me so much.

To Professor Balakrishnan, who gave me the opportunity to work with him, learn different ways of working and know a new country. I would like to thank him and Debanjan for their warm welcome and for making easier and unforgettable the time I spent in Canada. I am also grateful to McMaster University for allowing me access and use of their facilities.

I would also like to thank every one that helped me find the balance between academic and personal life. To my friends, mainly Marta and Noelia, for the board games, the laughs and specially for the endless talks. To those who showed me how useful a coffee break in the middle of the morning is and share it with me also.

Finally, I am pleased to thank the Spanish Secretary of State of Research, Development and Innovation (former Ministry of Science and Innovation) for the economic support and for giving me the chance to work with Enrique Castillo. To the Department of Applied

Mathematics and Computational Science of the University of Cantabria for giving me the opportunity to start teaching.

To all of you, thank you very much.

Esta tesis no habría sido posible sin el apoyo de muchas personas que me han ayudado a crecer a nivel académico y personal. Por tanto, me alegro enormemente de poder contar con una sección para poder darles las gracias.

En primer lugar, estoy muy agradecida a Enrique Castillo, mi director de tesis, por compartir conmigo su forma de entender la enseñanza, la investigación y la vida en general. He sido una gran afortunada al tenerlo como mentor. A él, por todo el tiempo que me ha dedicado, enseñándome y ayudándome a crecer. Enrique también me ha dado la oportunidad de conocer a Maria, mi compañera de despacho, cuya ayuda y conversaciones nunca olvidaré. Ha sido un placer poder compartir con ella nuestra pequeña oficina y tantas horas de trabajo.

Me gustaría darle las gracias también a mi madre, a quien debo todo lo que soy y lo que tengo. Por su amor incesante y apoyo incondicional, por hacerme sentir capaz de cualquier cosa y por ser un ejemplo de trabajo duro y perseverancia, no tengo nada más que decir salvo gracias.

No puedo olvidarme de mi chico, Raúl. Gracias por estar ahí y animarme cada vez que lo he necesitado, por hacerme reir y nunca dejar de sonreir pero, sobretodo, por dejarlo todo para poder estar conmigo.

A mi familia*, Randi y Patricia, por actuar como tal. Es un honor para mí tener gente como ellas a mi alrededor que se preocupan tanto por mí.*

Al Profesor Balakrishnan, que me dió la oportunidad de trabajar con él, aprender nuevas formas de trabajo y conocer un nuevo país. Me gustaría agradecerle a él y a Debanjan la cálida acogida que me brindaron, así como por facilitar y hacer inolvidable mi estancia en Canadá. Asimismo, estoy agradecida a la McMaster University por permitirme el acceso y el uso de sus instalaciones.

Me gustaría agradecer también a todos aquellos que me han ayudado a encontrar el equilibrio entre mi vida personal y profesional. A mis amigas, especialmente a Marta y a Noelia, por los juegos de mesa, las risas y, especialmente, por las conversaciones interminables. A todos aquellos que me mostraron lo útil que puede ser un descanso en mitad de la mañana y además lo comparten conmigo.

Finalmente, debo agradecer a la Secretaría de Estado de Investigación, Desarrollo e Innovación (antiguo Ministerio de Ciencia e Innovación) por el apoyo económico y por darme la oportunidad de trabajar con Enrique Castillo. Al Departamento de Matemática Aplicada y Ciencias de la Computación de la Universidad de Cantabria por permitirme comenzar mi labor docente.

A todos ellos, muchísimas gracias.

Abstract

Traffic modeling is one of the most useful tools on traffic planning that permits, among others, evaluating the behavior of traffic networks and the adequacy of mobility programs before implanting them. For that reason, the development of new tools that solve new problems and model a wide variety of situations is important.

Several models have been proposed in the literature in order to solve problems with different characteristics. In this thesis, we also deal with various topics and we provide some statistical and mathematical tools to solve them.

This thesis provides the reader the following contributions:

- **Literature review.** A literature review about existing traffic problems and the most widely used models to solve them is done. In particular, the static traffic assignment, the matrix estimation, the observability and the dynamic network loading problems are dealt with. Some of these models are illustrated with examples for a better understanding of the main concepts and ideas.

- **A percentile traffic assignment model.** We first solve a conjecture proposed by Nie (2011) on the permutability of percentile and partial derivatives of route travel times with respect to route flows. Secondly, a percentile system optimal model is proposed, including both with and without path enumeration versions.

- **A traffic assignment model including overtaking classes.** A static traffic assignment model including overtaking classes is provided which is based on a new family of travel time functions. We also present equivalent optimization problems with and without path enumeration.

- **A Bayesian matrix estimation model.** We present a hierarchical optimization model to estimate the origin-destination flow matrix by means of link flow counts. This model, which is based on Bayesian statistical techniques, assumes that path flows, and hence link, node and origin-destination flows, belong to the Gamma distribution.

- **Upper bound of the number of sensors required for total observability.** The minimum set of links to be equipped with sensors in order to get total link

observability by means of link flows is derived. It is shown that this number is the rank of the link-path incidence matrix.

- **A model for the continuous dynamic network loading problem with overtaking class users.** A model for the continuous dynamic network loading problem including overtaking classes is proposed. This model considers different class users depending on their overtaking preferences and takes into account the interaction of flows of all paths and classes and their coincidence at different times and locations. The effect of downstream links, and hence of physical queues, is also considered.

- **Graphical methods to analyze traffic trajectories with and without overtaking.** Some graphical methods to analyze traffic trajectories with and without overtaking are given. We study the graphical characteristics of trajectories plots, such as slope or curvature, and explain their physical meaning, i.e., speed, acceleration, etc. We also analyze the plots obtained when trajectories of several overtaking classes are superimposed, producing color bands when overtaking takes place.

- **Practical application.** All the proposed methods are applied to fictitious and real networks in order to analyze their characteristics and performances, together with the associated computational requirements.

- **Program codes.** We present the computational implementations of the models presented in this thesis, which have been used to obtain the mentioned examples.

Contents

List of Figures

List of Tables

Part I

Resumen de la tesis en español

Chapter 1

Resumen de la tesis en español

Contents

1.1 Motivación

Millones de personas se desplazan a diario por motivos de trabajo, estudios, ocio, etc. dando lugar a atascos, contaminación, retrasos y accidentes. Por lo tanto, sería interesante poder contar con alguna herramienta que nos permita eliminar, o al menos minimizar, estos problemas derivados del tráfico. La ingeniería de tráfico es una herramienta de ese tipo mediante la cual se puede evaluar el estado de los sistemas de tráfico.

Tradicionalmente, los ingenieros de tráfico se han concentrado en mejorar los sistemas de tráfico a través de la construcción de nuevas infraestructuras. No obstante, ahora la tendencia consiste en evaluar los sistemas actuales para mejorarlos en términos de eficiencia, seguridad, rapidez, comodidad y sostenibilidad. Además, las técnicas de ingeniería de tráfico permiten evaluar los programas de movilidad antes de llevarlos a cabo para predecir

su utilidad, reduciendo así los costes de manera sustancial.

Entre otros campos de la ingeniería de tráfico, podemos destacar la predicción de viajeros, la cual requiere la definición y predicción de: la generación de viajes (cuántos viajes van a originarse), la distribución de los viajes (dónde se dirigen los viajeros), la elección del modo (qué modo de transporte utilizan) y la asignación de tráfico (qué rutas van a elegir).

Para resolver estas cuestiones es importante contar con información sobre el estado de los sistemas de tráfico. Por ello, se están desarrollando actualmente nuevos sistemas dinámicos que reciben el nombre de Sistemas Inteligentes de Transporte (más conocidos por sus siglas en inglés ITS). Estos sistemas incluyen sensores y detectores de velocidad y flujo de vehículos que se emplean para obtener información en tiempo real que permita tomar las decisiones adecuadas sobre el control del tráfico. Las ventajas derivadas del uso de los ITS no se limitan a reducir los tiempos de viaje o mejorar los flujos, sino que también incluyen la reducción de los accidentes, del consumo de combustibles y del impacto medioambiental.

El proceso de planificación y evaluación de los sistemas de tráfico requiere de herramientas que sean capaces de reproducir la realidad. Una herramienta muy útil son los modelos matemáticos y estadísticos, en los cuales se centra esta tesis. Por tanto, el principal objetivo de esta tesis es la implementación de modelos y herramientas estadístico-matemáticas que reflejen la realidad de los sistemas de tráfico, incorporen la información disponible y produzcan resultados fáciles de interpretar.

1.2 Contenido

En la literatura se han propuesto una gran variedad de modelos para resolver los diferentes problemas derivados de los sistemas de tráfico. En esta tesis, se abunda en la problemática y se proponen algunas herramientas estadístico-matemáticas para su resolución. Todos los métodos propuestos están acompañados de ejemplos ficticios o reales, lo que permite analizar las características y el comportamiento de los modelos.

El contenido de esta tesis está dividido en:

- **Revisión de la literatura.** Se realiza una revisión de la literatura que incluye algunos de los problemas de tráfico existentes, así como los modelos más utilizados para su resolución. En particular, se explican los siguientes problemas: asignación de tráfico, estimación de matrices, observabilidad de redes de tráfico y recarga dinámica de la red. Algunos de estos modelos son ilustrados con ejemplos para una mejor comprensión de los principales conceptos e ideas.

- **Un modelo de asignación de tráfico percentil.** En primer lugar se resuelve una conjetura propuesta por Nie (2011) sobre la permutación de los percentiles y las derivadas parciales de los tiempos de ruta con respecto a los flujos de éstas. A

continuación, se propone un modelo de asignación de tráfico percentil, incluyendo las versiones con y sinh enumeración de rutas.

- **Un modelo de asignación de tráfico que incluye adelantamientos.** Se propone un modelo estático de asignación de tráfico que incluye adelantamientos basado en una nueva familia de funciones de tiempo de viaje, así como dos problemas de optimización equivalente con y sin enumeración de rutas.

- **Un modelo bayesiano de estimación de matrices.** Presentamos un modelo jerárquico de optimización para estimar matrices de flujo origen-destino a partir de arcos escaneados o aforados. Este modelo, que se basa en técnicas estadísticas bayesianas, asume que los flujos de las rutas, y por tanto también los flujos en arcos, nodos y pares origen-destino, siguen una distribución Gamma.

- **Límite superior del número de sensores necesario para una observabilidad total.** Se calcula el mínimo conjunto de arcos que debe ser equipado con sensores para obtener observabilidad total a partir de los flujos en arcos. Se demuestra, además, que este número coincide con el rango de la matriz de incidencia arco-ruta.

- **Un modelo continuo para el problema dinámico de recarga de red incluyendo adelantamientos.** Se propone un modelo continuo para el problema dinámico de recarga de red incluyendo adelantamientos. Este modelo considera distintas clases de usuarios dependiendo de su inclinación al adelantamiento y tiene en cuenta la interacción en los arcos de los flujos de las distintas clases y rutas en distintos instantes y localizaciones. También se considera el efecto de los arcos aguas abajo y, por tanto, de las colas físicas.

- **Métodos gráficos para analizar trayectorias de tráfico con y sin adelantamiento.** Se desarrollan algunos métodos para analizar trayectorias de tráfico con y sin adelantamiento. Analizamos las caraterísticas de los gráficos de trayectorias (como son la pendiente y la curvatura) y explicamos su significado físico (velocidad, aceleración, etc.). También estudiamos los gráficos obtenidos al superponer trayectorias de distintas clases, dando lugar a bandas de distintos colores cuando los adelantamientos tienen lugar.

- **Implementación computacional.** Finalmente, presentamos la implementación computacional de los modelos propuestos en esta tesis y que han sido utilizados para obtener las aplicaciones prácticas ya mencionadas.

1.3 Contribuciones

Las principales contribuciones de esta tesis son: una revisión de la literatura, un modelo de asignación de tráfico percentil, un modelo de asignación de tráfico que incluye adelan-

tamientos, un modelo bayesiano de estimación de matrices, el límite superior del número de sensores necesario para una observabilidad total, un modelo continuo para el problema dinámico de recarga de red incluyendo adelantamientos y algunos métodos gráficos para analizar trayectorias de tráfico con y sin adelantamiento. En los siguientes apartados se ofrece una descripción detallada de las contribuciones mencionadas.

1.3.1 Revisión de la literatura

El propósito de esta parte es ubicar la tesis en el contexto de la modelización del tráfico y presentar un resumen del estado del arte de los modelos de tráfico, lo que representa la base de las soluciones propuestas en los capítulos restantes. En particular, esta parte está formada por cuatro capítulos: el problema de asignación de tráfico estático, el problema de estimación de matrices origen-destino, el problema de observabilidad de tráfico y los modelos dinámicos de tráfico.

En cada uno de los capítulos se detalla el problema y se exponen algunos de los modelos propuestos para su resolución. De esta forma, se proporciona al lector un mejor conocimiento del problema y una idea general de las soluciones ya propuestas en la literatura. Por último, dedemos destacar que los capítulos de la revisión de la literatura se corresponden con los capítulos posteriores de forma que los primeros pueden verse como una introducción a los segundos.

1.3.2 Un modelo de asignación de tráfico percentil

En este capítulo nos centramos en el problema de *fiabilidad del tiempo de viaje* y resolvemos una cuestión planteada por Nie (2011) sobre la permutación de percentiles y derivadas parciales de los tiempos de viaje en rutas con respecto a los flujos en las rutas. A partir de una familia de contraejemplos, se demuestra que las operaciones: (a) obtener percentiles y (b) derivar parcialmente los tiempos de viaje en ruta, no son intercambiables.

A continuación, proponemos un modelo que asume que los tiempos de viajes en las rutas pertenecen a una familia de localización y escala, cuyas medias y varianzas pueden ser evaluadas en términos de los tiempos de viaje de los arcos. Esto permite evitar el uso del teorema central del límite y de convoluciones. En oposición a la mayoría de los modelos que requieren enumeración de rutas, presentamos un modelo percentil de optimización del sistema incluyendo dos versiones con y sin enumeración de rutas. Este modelo puede considerarse como un modelo de fiabilidad del tiempo de viaje puesto que asume la existencia de clases de viajeros según su deseo de puntualidad. Por último, se muestran dos ejemplos de aplicación, uno de ellos real, para ilustrar la potencia del método propuesto.

1.3.3 Un modelo de asignación de tráfico que incluye adelantamientos

Este capítulo está dedicado al problema de asignación de tráfico cuando el adelantamiento está permitido. Presentamos una nueva familia de funciones de tiempo de viaje que permite reproducir el adelantamiento. Más concretamente, la nueva familia de funciones cumple una propiedad importante: reproduce el hecho de que las altas congestiones hacen imposible el adelantamiento pero proporciona distintos tiempos de viaje para usuarios de distintas clases cuando el nivel de congestión es bajo. Esta familia de funciones está basada en combinaciones lineales convexas de otras funciones de tiempo de viaje.

Recurrimos a dos problemas de desigualdad variacional (VIP) para modelizar el problema de asignación de tráfico uno con y otro sin enumeración de rutas. Además, se proponen dos problemas de optimización aquivalentes basados en una función de discrepancia, los cuales pueden ser resueltos a través de softwares generales de optimización. Se presenta también un conjunto de ejemplos que incluye la red real de Ciudad Real (España) para ilustrar los métodos propuestos. Entre dichos ejemplos, destacamos el caso analizado en el que automóviles y motos comparten la red bajo condiciones de baja y alta congestión.

1.3.4 Un modelo bayesiano de estimación de matrices

En este capítulo se propone un modelo jerárquico de optimización generado a partir de un método bayesiano de estimación para la predicción de matrices origen-destino.

El problema puede considerarse como un sistema de ecuaciones en el que tres de ellas son, a su vez, problemas de optimización: (1) un modelo de Wardrop de mínima varianza, que es utilizado para determinar las probabilidades de elección de las rutas, (2) un problema de mínimos cuadrados, utilizado para obtener la muestra de flujos de los pares origen-destino, y (3) un problema de máxima verosimilitud para estimar la moda a posteriori. Para resolver el modelo se propone un enfoque iterativo que permite resolver el problema multiobjetivo en pocas iteraciones.

Para finalizar, se muestran dos ejemplos de aplicación que permiten ilustrar el funcionamiento del método propuesto. Además, se incluye una comparación con otras técnicas existentes, las cuales producen resultados similares, probando así la validez de los métodos propuestos.

1.3.5 Límite superior del número de sensores necesario para una observabilidad total

Se demuestra que el número mínimo de sensores necesario para conocer los flujos en todos los arcos de una red de tráfico puede determinarse sólo si se posee información sobre las rutas. No obstante, no es necesaria la enumeración de todas las rutas, sino de un subconjunto que defina el rango r_W de la matriz de incidencia arco-ruta $\mathbf{W}$. Si el rango de dicha matriz para un subconjunto reducido de rutas es $m - n$, donde m es el número

de arcos y n, el número de nodos no centroides, se puede concluir que es suficiente con $m - n$ sensores. Es importante determinar el mínimo número de sensores puesto que la observación de más arcos de los estrictamente necesarios, puede provocar redundancia e incluso problemas de incompatibilidad.

También se muestra que el conjunto de todas las rutas puede obtenerse como las aristas de un cono y que las fórmulas que permiten determinar los flujos de los arcos dependientes en función de los independientes pueden obtenerse a partir de los enfoques basados en nodos y rutas, dando lugar a iguales resultados siempre que $r_w = m - n$.

Finalmente, se propone un algoritmo para la obtención de conjuntos de rutas linealmente independientes. Los métodos propuestos son ilustrados sobre la red paralela propuesta por Hu et al. (2009).

1.3.6 Un modelo continuo para el problema dinámico de recarga de red incluyendo adelantamientos

En este capítulo se presenta un modelo para resolver el problema dinámico de recarga de red cuando se asumen tipos de usuarios según su inclinación al adelantamiento, pero no se permite el adelantamiento entre usuarios del mismo tipo.

El modelo calcula las funciones de tiempo de viaje en los arcos en un conjunto finito de instantes igualmente espaciados, que son empleados para interpolar los tiempos de viaje en los instantes restantes. El modelo considera funciones de tiempo de viaje no lineales en los arcos que dependen del flujo en el propio arco, así como en los arcos aguas abajo, y toma en consideración el hecho de que los tiempos de viaje (y por consiguiente sus velocidades) deben coincidir para todas las clases cuando existe una alta congestión. Además, se consideran colas físicas al tener en cuenta el tiempo necesario para que se disipen las mismas.

Las demandas de las rutas en los nodos origen se asumen conocidas y se reproducen como combinaciones lineales de funciones de densidad. Las leyes de conservación son utilizadas para determinar la evolución de la onda asociada al flujo en rutas en toda la red.

Para finalizar, se fusiona toda la información disponible para hacer compatibles los tiempos de viaje a partir de un método iterativo hasta que se alcanza la convergencia. De nuevo, los métodos se ilustran a partir de ejemplos, algunos de ellos reales. Los resultados obtenidos reflejan las tendencias observadas en la realidad y los tiempos de computación son razonables, permitiendo su aplicación en redes reales de mayor tamaño.

1.3.7 Métodos gráficos para analizar trayectorias de tráfico con y sin adelantamiento

Este capítulo presenta algunos métodos gráficos para analizar trayectorias, esto es, t_0-familias de trayectorias $x = f(t; t_0)$ que representan la ubicación del usuario x en el

instante t cuando su tiempo de salida es t_0.

También trabajamos con la función inversa que representa el tiempo de llegada $t = g(x; t_0)$ a la localización x de un usuario que ha iniciado el viaje en el instante t_0. Diferenciamos entre dos tipos de gráficos: igualmente espaciados, en los cuales cada banda representa periodos idénticos de tiempo; y de flujos iguales, cuyas bandas representan igual número de usuarios. Se investiga la interpretación de las familias de curvas, sus inversas y las primeras y segundas derivadas parciales con respecto a la hora de salida, tiempo y ubicación, de forma que puedan conocerse estos valores a partir de los gráficos de trayectorias.

Por último, analizamos la superposición de trayectorias de distintas clases de usuarios y las bandas de colores que se producen como consecuencia de esta superposición. Asimismo, proponemos algunas normas para elegir el mejor instante para comenzar un viaje basado en los distintos gráficos. Los métodos propuestos se ilustran a través de su aplicación a una red sencilla.

Part II

Introduction and State-of-the-art

Chapter 2

Introduction and contributions

Contents

2.1 Introduction

Every day, millions of people move from one place to another because of work, studies, leisure, etc. leading to congestion, pollution, delays and accidents. Therefore, it would be interesting to have a tool that helps reducing these negative aspects of traffic. Traffic engineering is such a tool as it permits evaluating traffic systems.

Traditionally, traffic engineering has been focused on road improvements by means of building additional infrastructure. However, nowadays it is more centered in evaluating traffic systems so as to optimize them in terms of efficiency, safety, rapidness, comfortability, economy and environment. Moreover, traffic engineering techniques permit evaluating the adequacy of mobility programs ahead of their implantation in order to predict their usefulness and, thus, reducing costs.

Among other aspects of traffic, forecasting of passenger travel is an important area of

traffic engineering. It usually involves an urban transportation planning model, requiring the estimation of trip generation (how many trips for what purpose), trip distribution (destination choice, where is the traveler going), mode choice (what mode is being taken), and route assignment (which streets or routes are being used).

To solve all this problems, information on the state of the traffic system is important. For that reason, new dynamic systems, collectively called intelligent transportation systems, are being developed. These include automatic sensors, interconnected, guidance systems to manage traffic (for example, traffic signs which open a lane in different directions depending on the time of day). Also, traffic flow and speed sensors are used to obtain real-time information that is processed and decisions about traffic control are made. The advantages are not limited to decreasing the travel times or improving the traffic flow, but to congestion, accidents or fuel reduction, and avoiding or reducing environmental problems, etc.

The process of planning and evaluating traffic systems is usually made by means of mathematical and statistical models, that are capable of reproducing reality and hence are a very useful tool. This thesis focuses on this last point, i.e. implementation of different mathematical and statistical tools capable of reproducing traffic reality taking into account available data and providing easy to interpret results. There are many different models according to the objectives, namely, short or long term analysis, predicting or estimation, etc. A good engineering answer to these problems must be as simple as possible, but it must give a practical solution to the main problem in each case.

In the remaining of the thesis some statistical and mathematical models and tools are developed that serve to solve the aforementioned problems. In order to illustrate those tools, we apply them to illustrative and real networks (such as the Ciudad Real and Cuenca networks, which are medium size Spanish cities).

2.2 Contributions

The main contributions of this thesis are: a literature review, a percentile traffic assignment model, a traffic assignment model including overtaking classes, a Bayesian matrix estimation model, the computation of the upper bound of the number of sensors required for total observability, a model for the continuous dynamic network loading problem with overtaking class users and some graphical methods to analyze traffic trajectories with and without overtaking. A detailed description of each of these contributions is given in the following sections.

2.2.1 Literature review

The aim of this part is to locate this thesis on the transport modeling and to present a general summary of the state-of-the-art transport models, which is the basis of the

main proposed solutions. In particular, four chapters conform this part: the static traffic assignment problem, the matrix estimation problem, the traffic observability problem and the dynamic traffic models.

In each of these chapters the problem, as well as the models proposed to solve it, is presented. This way, we get the reader to have a better knowledge of the problem in study and a general idea of the solutions provided in the literature.

The literature review serves as an introduction to the models presented in the second part of the thesis, where the original contributions are explained. Chapter 3 introduces the static traffic assignment model, while Chapters 7 and 8 provide extensions to this topic. In Chapter 4 we explain several models proposed in the literature to solve the matrix estimation problem and in Chapter 9 we present a new model to solve the same problem. Chapter 5 summarizes recent studies on the link observability problem based on link flows and Chapter 10 proves that some of them can be improved. Finally, a new model for the dynamic network loading problem stated in Chapter 6 is given in Chapter 11.

2.2.2 A percentile traffic assignment model

In this chapter we deal with the travel time reliability problem and we answer an open question raised by Nie (2011) about the permutability of percentiles and partial derivatives of route travel times with respect to route flows. A family of counterexamples is given to demonstrate that the two operations: (a) obtain percentiles and (b) partial derivation of route travel times do not commute.

Next, we propose a model that assumes a location-scale family for the path travel times, whose means and variances are evaluated in terms of link travel times. This avoids the use of the central limit theorem and convolutions providing a flexible and simple alternative. Contrary to most existing models that require path enumeration or an iterative method to add paths sequentially, we present a PSO (percentile system optimization) alternative in its two versions: with and without path enumeration. This model can be classified into the travel time reliability problem as it assumes that there exist classes of users depending on their desire of punctuality. Two examples of applications, one of them is real, are used to illustrate the power of the proposed method.

2.2.3 A traffic assignment model including overtaking classes

This chapter is devoted to the traffic assignment problem when overtaking of vehicles is permitted. A new family of link travel time functions is presented that allows reproducing overtaking. In particular, this family of travel time functions have a very important property: to reproduce the fact that high congestions impede overtaking, it produces the same behavior of several overtaking classes under high congestion, but different travel times under mild congestion. This family is generated based on local linear convex combinations of travel time functions.

Two variational inequality problems (VIP) are used to solve the traffic assignment problem, one with and another without route enumeration. Equivalent optimization problems based on a gap function are provided, which can be solved by means of general optimization software. A set of examples including a real one (the city of Ciudad Real) is used to illustrate the proposed methods and techniques. In particular, a case in which cars and motorcycles share the network is analyzed under congested and uncongested conditions in order to see how congestion provokes equal travel times for all classes.

2.2.4 A Bayesian matrix estimation model

A hierarchical optimization problem generated by a Bayesian method for estimating origin-destination matrices, based on Gamma models, is given in this chapter.

The problem can be considered as a system of equations in which three of them are optimization problems: (1) a Wardrop minimum variance (WMV) assignment model, which is used to derive the route choice probabilities, (2) a least squares problem, used to obtain the Origin Destination pair flow sample data, and (3) a maximum likelihood problem to estimate the posterior modes. A multi-level iterative approach is proposed to solve the multi-objective problem that converges in a few iterations.

Finally, two examples of applications are used to illustrate the proposed methods and procedures, a simple and the medium size Ciudad Real networks. A comparison with existing techniques, which provide similar flows, seems to validate the proposed methods.

2.2.5 Upper bound of the number of sensors required for total observability

It is demonstrated that the minimum number of sensors required to know all link flows in a traffic network can be determined only if path information is available. However, not all paths need to be enumerated but a subset defining the rank r_W of the link-path incidence matrix $\mathbf{W}$. If this rank for a reduced subset of paths is already $m-n$, where m and n are the number of links and non-centroid nodes, respectively, we can conclude that $m-n$ counters are sufficient. It is important to determine the minimum number of sensors because observation of more links than those strictly necessary produce redundancy and this is known to lead to incompatibility problems.

It is also shown that the set of all paths can be obtained as the edges of a cone, and that the formulas providing the dependent link flows in terms of the independent link flows can be obtained by the node-based or the path-based approaches, with the same results only when the rank of the link-path incidence matrix is $m-n$.

Finally, an algorithm to obtain sets of linearly independent path vectors is given. The methods are illustrated by the parallel network example in Hu et al. (2009).

2.2.6 A model for the continuous dynamic network loading problem with overtaking class users

This chapter presents a model for solving the continuous loading network problem when different class users interact in the traffic network so that overtaking among different class users is permitted but the FIFO rule is satisfied for the same class users.

The model calculates the link travel time functions at a basic finite set of equally spaced times which are used to interpolate a monotone spline for all other times in order to preserve monotonicity and guarantee that the FIFO rule is satisfied at all points for the same class users. The model assumes non-linear link travel time functions of the link volumes including those ahead of the link being considered, takes into account that different class functions must be asymptotically coincident for high congestions and considers link physical-queues.

The path origin demands are reproduced as linear combinations of density functions and the conservation laws are used to determine the path flow wave evolution throughout the network. Different path flow waves are mixed together and a congestion equation is used to determine the link travel times.

Finally, all information is combined to make it compatible in times and locations using an iterative method until convergence. The method is illustrated by some examples of illustrative and real networks. The results seem to reproduce the observed trends closely. The resulting required cpu times are reasonable so that the method seems to be applicable to real networks.

2.2.7 Graphical methods to analyze traffic trajectories with and without overtaking

This chapter presents some graphical methods to analyze traffic trajectories, that is, t_0-families of trajectories $x = f(t; t_0)$ representing the user location x at time t when its departure time is t_0.

We also deal with the inverse function that represent arrival time $t = g(x; t_0)$ at location x of a user whose departure time is t_0. We distinguish between equally delayed plots, that represent trajectory bands corresponding to identical periods of time, and equal flow plots, whose bands represent equal number of users. The interpretations of the trajectory family curves and their inverse functions first partial derivatives (slopes) with respect to departure time, time or space and the corresponding second partial derivatives are given, so that the relative values of these derivatives and their physical meanings can be known by taking a look at the trajectory plots.

The superposition of trajectories of different classes, and the color bands that are produced consequently, are also studied. We also propose some rules to determine the best moments to start a journey based on different trajectory plots. Finally, the proposed methods are illustrated by their application to a simple traffic network.

2.3 Outline of the thesis

This thesis is organized as follows. Part II summarizes the fundamentals and relevant literature model of the problems considered. In Part III the original contributions are presented. Chapter 7 solves the conjecture proposed by Nie (2011) and provides a Percentile System Optimal model. In Chapter 8 a model to solve the static traffic assignment including overtaking classes and following Wardrop's first principle is given. In Chapter 9 the proposed methods for the estimation of OD-pair flows based on link counts and bayesian techniques are introduced. The upper bound on the number of link count sensors required for total link observability is derived in Chapter 10. Chapter 11 deals with the dynamic network loading problem and includes a model with overtaking classes to solve it. Some graphical methods to analyze traffic trajectories with and without overtaking are presented in Chapter 12. Finally, Part IV provides some conclusions and suggests future works and the Appendix includes the programming codes of the presented models.

Chapter 3

The traffic assignment problem

Contents

3.1 Introduction

The traffic assignment problem is one of the most studied problems in the field of transportation engineering, that has become a very useful tool for predicting traffic flow in congested areas. Given the graph representation of the transportation network, the associated link performance functions[1] and an origin-destination flow matrix, the aim of the problem is to find the flow of the paths[2] and, hence, of the network links. In other words, the issue is how to *assign* the OD matrix onto the network.

[1] A link performance function is a function that relates the flow on a link with the time required to travel through it. Some examples of performance function are given in Appendix A.

[2] Note that finding the flow of paths is equivalent to find the paths chosen by users. If the flow on a path is null, it means that no one has chosen it.

In this chapter, we consider the deterministic traffic assignment problem, in the sense that we assume that the distribution of the flow among alternatives is given by a certain analytical function. Contrary, stochastic models assume that the users behave in a random manner with a given density function.

It is also remarkable that the traffic assignment problem is static as it does not take into account the changes in the OD matrix that may occur with time. Because of this reason, traffic assignment models may be defined for short periods of time (such as morning peak hour, evening peak hour or midday) and assuming that the origin-destination flows within the period being studied is constant (so this problem must not be applied in situations where large demand fluctuations occur). Nevertheless, it is important to realize that the period of analysis cannot be very small since it has to be longer than the typical duration of trips at that time. Finally, traffic assignment models can be used for longer periods (such as a day or a week) when road users go through their routines repeatedly and at regular intervals (for example, the frequency of work-trips and the origin and destination of it are fixed).

To solve the traffic assignment problem, knowledge of the rules that users follow to choose their paths is required. The three existing principles to model route choice include:

- User-Equilibrium (UE) wherein no user can improve his/her travel time by unilaterally changing routes.
- System-Optimal (SO) wherein the total travel time of the system in minimized.
- Stochastic User-Equilibrium (SUE) wherein no user can uniterally change routes to improve his/her *perceived* travel times.

In this chapter we will focus on the User-Equilibrium approach and will describe the different methods used in the literature to solve this problem. The chapter is structured as follows. In Section 3.2 we introduce the User Equilibrium principle and analyze several formulations of this problem, namely, mathematical programming formulations in Section 3.2.1, nonlinear-complementary problems (NCP) in Section 3.2.2 and variational inequality problems (VIP) in Section 3.2.3. Section 3.3 is devoted to the System-Optimal approach and the conditions under which the UE and the SO approach are equivalent. Finally, in Section 3.4 the problem of user equilibrium with heterogeneous users is dealt with.

3.2 User Equilibrium (UE)

Suppose that the number of drivers who want to travel between a given origin-destination (OD) pair is known and that these OD-pairs are connected by several possible paths. The question, as it has been indicated in the previous section, is how drivers will be distributed among the different paths. If all of them chose the same shortest path, then congestion would develop on it. As a result, the travel time on this path might increase to a point

where it is no longer the minimum travel-time path. Some of the drivers would then use an alternative path that can, however, be congested too. This process can continue until an equilibrium situation is reached. This kind of equilibrium is called User-Equilibrium (UE) (or Wardrop's first principle) and, based on the ideas of the economic theory of supply-demand equilibrium, was first introduced by Wardrop (1952):

Definition 1 (User Equilibrium (UE)) *A user-equilibrium is reached when no vehicle can improve its travel time by uniterally changing routes.*

Obviously, this definition of UE implies that every user has knowledge about the effect that his transfer onto a new route has upon travel time. To make this principle suitable for a mathematical formulation, Wardrop proposed an analogous law stated as:

"The journey times on all the paths actually used are equal, and less than those which would be experienced by a single vehicle on any unused path."

Mathematically, the UE principle can be stated as

$$f_{ksr}(c_{ksr} - \pi_{ks}) = 0; \quad \forall r, ks, \tag{3.1}$$

$$c_{ksr} - \pi_{ks} \geq 0; \quad \forall r, ks, \tag{3.2}$$

where f_{ksr} is the flow on path r with origin-destination ks, c_{ksr} is the travel cost associated with route r with origin-destination ks and π_{ks} is the equilibrium cost to travel from origin k to destination s.

Condition (3.1) forces that when the travel time of route r is longer than the shortest travel time of routes of the same OD, the flow on that route is zero, and when the travel time is equal to the shortest travel time of routes of the same OD, its flow is equal or greater than zero. On the other hand, condition (3.2) forces that the travel time of any route is greater or equal to the shortest travel time of the routes of the same OD.

The first attempt to solve this problem was proposed in Wardrop (1952)[3]: Suppose that τ_{ks} is the flow traveling from node k to node s and that the performance function, that is, the function that gives the travel time t_r of route r, is a function of the route flow f_r given by $t_r = t_{0r}\varphi_r(f_r)$, where t_{0r} is the free flow time of route r and $\varphi_r(f_r)$ is a monotone normalized function. Assume also that π_{ks} is the minimum travel time of OD ks. In that case, routes whose free travel time t_{0r} are bigger that π_{ks}, cannot be used. Rewriting the performance function in its inverse form, we get

$$f_r = \varphi_r^{-1}\left(\frac{\pi_{ks}}{t_{0r}}\right); \quad \forall r | t_{0r} \leq \pi_{ks}.$$

[3]It is important to emphasize that this first attempt does not take into account the fact that the travel time on a path is influenced by the flow on different paths, i.e. it assumes that travel times of different paths are independent.

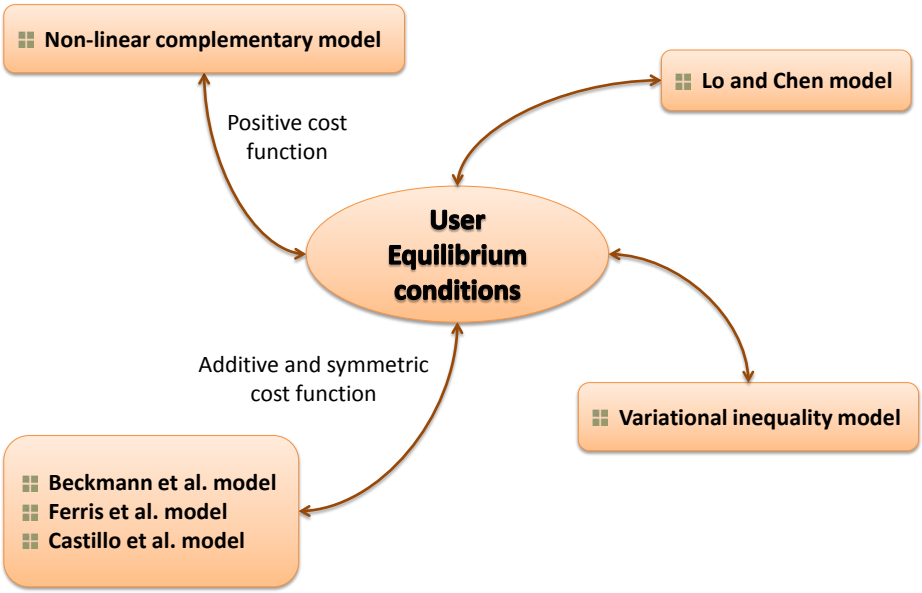


Figure 3.1: UE models equivalent to conditions (3.1)-(3.2) and the conditions that must hold.

Summing over the used routes gives:

$$\tau_{ks} = \sum_{r|t_{0r} \leq \pi_{ks}} f_r = \sum_{r|t_{0r} \leq \pi_{ks}} \varphi_r^{-1} \left(\frac{\pi_{ks}}{t_{0r}} \right).$$

This equation gives the value of τ_{ks} for which π_{ks} is the appropriate travel time. If τ_{ks} is calculated for each value of π_{ks}, it is possible to find the solution of the problem by picking out the value of π_{ks} which corresponds to the given τ_{ks}.

This procedure to solve the traffic assignment problem is not efficient and does not take into account the interactions between users of different routes. For this reason and in order to extend the problem, many different models have been proposed in the literature. Generally, the proposed methods can be classified in one of these groups: (i) mathematical programming (MP), (ii) nonlinear-complementary problem (NCP) and (iii) variational inequality problem (VIP). The following subsections are devoted to each of these approaches. Figure 3.1 shows a summary of the models that will be explained and the conditions that must hold so these models are equivalent to the UE conditions (3.1)-(3.2).

3.2.1 Mathematical programming approach

The Beckmann model

Some years after the UE principle was first stated, Beckmann et al. (1956) proposed to solve the traffic assignment problem (3.1)-(3.2) by means of the following optimization problem:

Consider a strongly connected network[4] $(\mathcal{N}, \mathcal{A})$*, where* $\mathcal{N}$ *is the set of nodes and* $\mathcal{A}$ *is the set of links* ℓ_{ij}*. For certain origin-destination node pairs,* $k, s \in \mathcal{OD}$*, where* $\mathcal{OD}$ *is a subset of* $\mathcal{N} \times \mathcal{N}$*, there are given positive demands* τ_{ks} *and a set* $\mathcal{R}$ *of routes joining them. Then, the traffic assignment problem can be stated as:*

$$\underset{\mathbf{f},\mathbf{w}}{\text{Minimize}}\, Z(\mathbf{w}) = \sum_{\ell_{ij} \in \mathcal{A}} \int_0^{w_{ij}} c_{ij}(s)ds = \sum_{\ell_{ij} \in \mathcal{A}} C_{ij}\left(w_{ij}\right) \tag{3.3}$$

subject to

$$\sum_{r \in \mathcal{R}} \xi_{ksr} f_r = \tau_{ks} : \eta_{ks}, \quad \forall k, s \in \mathcal{OD} \tag{3.4}$$

$$\sum_{r \in \mathcal{R}} f_r \delta_{ijr} = w_{ij} : \lambda_{ij}, \quad \forall \ell_{ij} \in \mathcal{A} \tag{3.5}$$

$$f_r \geq 0 : \mu_r \quad \forall r \in \mathcal{R}, , \tag{3.6}$$

where ℓ_{ij} is the link joining nodes i and j, $c_{ij}(\cdot)$ is the travel time function associated with link ℓ_{ij}, $C_{ij}(\cdot)$ is the integral of the travel time function associated with link ℓ_{ij}, f_r is the flow on route r, w_{ij} is the flow on link ℓ_{ij}, δ_{ijr} is the link-route incidence matrix ($\delta_{ijr} = 1$ if link ℓ_{ij} belongs to path r, and 0 otherwise), ξ_{ksr} is the OD-route incidence matrix and η_{ks}, λ_{ij} and μ_r are the dual variables of problem (3.3)-(3.6).

Theorem 1 (Equivalence) *The mathematical program (3.3)-(3.6) is equivalent to the UE principle (3.1)-(3.2).*

Proof. This equivalence can be proved by means of the Karush-Kuhn-Tucker (KKT) conditions (see Sheffi (1985)).

The Lagrangian function of the problem (3.3)-(3.6) is

$$\begin{aligned} \mathcal{L}(\mathbf{w}, \mathbf{f}, \boldsymbol{\lambda}, \boldsymbol{\eta}, \boldsymbol{\mu}) = & \sum_{\ell_{ij} \in \mathcal{A}} C_{ij}(w_{ij}) + \sum_{k,s \in \mathcal{OD}} \eta_{ks} \left(\tau_{ks} - \sum_{r \in \mathcal{R}} \xi_{ksr} f_r \right) \\ & + \sum_{\ell_{ij} \in \mathcal{A}} \lambda_{ij} \left(\sum_{r \in \mathcal{R}} f_r \delta_{ijr} - w_{ij} \right) - \sum_{r \in \mathcal{R}} \mu_r f_r, \end{aligned} \tag{3.7}$$

[4] A network is strongly connected if for any OD pair ks with positive demand, there is at least one path joining nodes k and s.

and the associated Karush-Kuhn-Tucker conditions are:

$$\frac{\partial \mathcal{L}}{\partial w_{ij}} = c_{ij}(w_{ij}) - \lambda_{ij} = 0; \quad \forall \ell_{ij} \in \mathcal{A} \tag{3.8}$$

$$\frac{\partial \mathcal{L}}{\partial f_r} = -\sum_{k,s \in \mathcal{OD}} \eta_{ks}\xi_{ksr} + \sum_{\ell_{ij} \in \mathcal{A}} \lambda_{ij}\delta_{ijr} - \mu_r = 0; \ \forall r \in \mathcal{R}, \tag{3.9}$$

$$\sum_{r \in \mathcal{R}} \xi_{ksr} f_r = \tau_{ks}; \quad \forall k, s \in \mathcal{OD} \tag{3.10}$$

$$\sum_{r \in \mathcal{R}} f_r \delta_{ijr} = w_{ij}; \quad \forall \ell_{ij} \in \mathcal{A} \tag{3.11}$$

$$f_r \geq 0; \quad \forall r \in \mathcal{R}, \tag{3.12}$$

$$f_r \mu_r = 0; \quad \forall r \in \mathcal{R}, \tag{3.13}$$

$$\mu_r \geq 0; \quad \forall r \in \mathcal{R}. \tag{3.14}$$

From (3.8) we get

$$c_{ij}(w_{ij}) = \lambda_{ij}, \quad \forall \ell_{ij} \in \mathcal{A}, \tag{3.15}$$

that is, the dual variable λ_{ij} is the travel time of link ℓ_{ij}. From Equations (3.9) and (3.15) we have

$$\begin{aligned} \mu_r &= -\sum_{k,s \in \mathcal{OD}} \eta_{ks}\xi_{ksr} + \sum_{\ell_{ij} \in \mathcal{A}} \lambda_{ij}\delta_{ijr} \\ &= \sum_{\ell_{ij} \in \mathcal{A}} c_{ij}(w_{ij})\delta_{ijr} - \sum_{k,s \in \mathcal{OD}} \eta_{ks}\xi_{ksr} \\ &= c_r - \sum_{k,s \in \mathcal{OD}} \eta_{ks}\xi_{ksr}; \quad \forall r \in \mathcal{R}, \forall k, s \in \mathcal{OD}, \end{aligned} \tag{3.16}$$

where c_r is the travel time of route r.

In addition, due to (3.13), if $f_r > 0$, then $\mu_r = 0$ and $c_r = \sum_{k,s \in \mathcal{OD}} \eta_{ks}\xi_{ksr}$, implying that $\eta_{ks} = \pi_{ks}$ is the equilibrium travel time from origin node k to destination node s. Finally, (3.13),(3.14) and (3.16) lead to

$$f_r(c_r - \sum_{k,s \in \mathcal{OD}} \eta_{ks}\xi_{ksr}) = 0; \quad \forall r \in \mathcal{R}, \tag{3.17}$$

$$c_r - \sum_{k,s \in \mathcal{OD}} \eta_{ks}\xi_{ksr} \geq 0; \quad \forall r \in \mathcal{R}, \tag{3.18}$$

which are the UE equations (3.1) and (3.2) in terms of f_r. ■

Note that the dual variables μ_r are associated with the overcost of route r with respect to the minimum cost, that is, if route r is being used ($f_r > 0$), its travel time is minimum and, then, the overcost is zero ($\mu_r = 0$).

Once we have proved that the solutions to the mathematical program (3.3)-(3.6) follow the UE principle (3.1)-(3.2), another important issue of this problem is the uniqueness of solution.

Theorem 2 (Uniqueness of solution) *The problem (3.3)-(3.6) has unique solution in terms of link flows w_{ij} if the link performance function c_{ij} is strictly monotone.*

Proof. In order to show that a minimization program has unique solution, it is sufficient to prove that the objective function and the feasible region are convex. The feasible region of the UE problem is given by constraints (3.4)-(3.6), which are all linear, implying that the feasible region is convex.

To prove that the objective function (3.3) is also convex, we need to see if the Hessian is positive definite. Since $\frac{\partial Z(\mathbf{w})}{\partial w_{ij}} = c_{ij}(w_{ij})$, $\frac{\partial Z(\mathbf{w})}{\partial^2 w_{ij}} = c'_{ij}(w_{ij})$ and $\frac{\partial Z(\mathbf{w})}{\partial w_{ij}\partial w_{i'j'}} = 0$; $\forall \ell_{ij} \neq \ell_{i'j'}$, the Hessian becomes

$$\nabla^2 Z(W_{ij}) = \begin{pmatrix} c'_{ij}(w_{ij}) & 0 & \cdots & 0 \\ 0 & c'_{i'j'}(w_{i'j'}) & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & c'_{i''j''}(w_{i''j''}) \end{pmatrix}, \tag{3.19}$$

and since $c_{ij}(w_{ij})$ has been assumed strictly monotone with respect to link flows ($c'_{ij}(w_{ij}) > 0$), the Hessian is positive definite, and therefore, the solution to the mathematical problem in terms of link flows is unique. ∎

On the contrary, we can have multiple solutions with respect to path flows. Following an analogous process to the one in the previous proof and taking into account that $c_{ij}(w_{ij}) = c_{ij}\left(\sum_{r\in\mathcal{R}} f_r \delta_{ijr}\right)$ we can realize that the link performance function is not strictly monotone with respect to path flows f_r[5], and therefore, there are multiple solutions to the problem (3.3)-(3.6) in terms of path flows.

The previous model has a practical inconvenience: it is necessary to have route information. Path enumeration is a hard and time-consuming task that may become impossible for large networks. For that reason, some authors have developed models without route enumeration, based on the disaggregation of link flows. In the following subsections, two models without route enumeration are presented.

The Ferris, Meeraus and Rutherford model

In Ferris et al. (1999) a multicommodity formulation is used to solve the static traffic assignment problem and it is shown that it can be implemented using standard modeling

[5]If the flow on a path that does not contain the link ℓ_{ij} is augmented, the cost of this link will remain constant meaning that the link performance function is not strictly monotone.

software. The idea is to associate a "commodity" with each destination such that the variables v_{ijs} represent the flow passing through link ℓ_{ij} with destination s. Considering again a network $(\mathcal{N}, \mathcal{A})$ where $\mathcal{N}$ is the set of nodes, $\mathcal{A}$ is the set of links and $\mathcal{D}$ is the set of destination nodes, the mathematical problem suggested by Ferris et al. (1999) is:

$$\underset{\mathbf{v}}{\text{Minimize}}\, Z(\mathbf{v}) = \sum_{\ell_{ij} \in \mathcal{A}} C_{ij} \left(\sum_{s \in \mathcal{D}} v_{ijs} \right) \tag{3.20}$$

subject to

$$\sum_{j|\ell_{ij} \in \mathcal{A}} v_{ijs} - \sum_{j|\ell_{ji} \in \mathcal{A}} v_{jis} = \tau_{is} : \eta_{is}, \quad \forall i \in \mathcal{N},\ \forall s \in \mathcal{D},\ i \neq s \tag{3.21}$$

$$v_{ijs} \geq 0 : \mu_{ijs} \quad \forall \ell_{ij} \in \mathcal{A},\ \forall s \in \mathcal{D}, \tag{3.22}$$

where τ_{is} are the OD flows going from origin node i to destination node s and η_{is} and μ_{ijs} are the dual variables associated with the optimization problem. We note that the balance equation for $i = s$ has no sense because origin and destination do not coincide.

In order to show that this problem is also equivalent to the UE conditions, we will use the Karush-Kuhn-Tucker conditions. The Lagrangian of the problem is:

$$\mathcal{L}(\mathbf{w}, \boldsymbol{\eta}, \boldsymbol{\mu}) = \sum_{\ell_{ij} \in A} C_{ij} \left(\sum_{s \in \mathcal{D}} v_{ijs} \right) + \sum_{\substack{i \in \mathcal{N} \\ s \in \mathcal{D} \\ i \neq s}} \eta_{is} \left(\tau_{is} - \sum_{j|\ell_{ij} \in \mathcal{A}} v_{ijs} + \sum_{j|\ell_{ji} \in \mathcal{A}} v_{jis} \right) - \sum_{\substack{s \in \mathcal{D} \\ \ell_{ij} \in \mathcal{A}}} \mu_{ijs} v_{ijs}, \tag{3.23}$$

and the associated KKT conditions are

$$c_{ij} \left(\sum_{s \in \mathcal{D}} v_{ijs} \right) - (1 - \delta'_{is})\eta_{is} + (1 - \delta'_{js})\eta_{js} - \mu_{ijs} = 0; \quad \forall \ell_{ij} \in \mathcal{A},\ \forall s \in \mathcal{D}, \tag{3.24}$$

$$\sum_{j|\ell_{ij} \in \mathcal{A}} v_{ijs} - \sum_{j|\ell_{ji} \in \mathcal{A}} v_{jis} = \tau_{is};\ \forall i \in \mathcal{N},\ \forall s \in \mathcal{D},\ i \neq s \tag{3.25}$$

$$v_{ijs} \geq 0; \quad \forall \ell_{ij} \in \mathcal{A},\ \forall s \in \mathcal{D}, \tag{3.26}$$

$$v_{ijs}\mu_{ijs} = 0; \quad \forall \ell_{ij} \in \mathcal{A},\ \forall s \in \mathcal{D}, \tag{3.27}$$

$$\mu_{ijs} \geq 0; \quad \forall \ell_{ij} \in \mathcal{A},\ \forall s \in \mathcal{D}, \tag{3.28}$$

where δ'_{is} is the Kronecker delta whose value is one if $i = s$ and zero, otherwise. Note that the Kronecker deltas in (3.24) have been included to take into account that the second summation in (3.23) does not include the case $i = s$.

From Equations (3.24) and (3.26)-(3.28) we get

$$c_{ij} \left(\sum_{s \in \mathcal{D}} v_{ijs} \right) + \rho_{is} \geq \rho_{js}; \quad \ell_{ij} \in \mathcal{A}, s \in \mathcal{D}; \quad \perp \quad v_{ijs} \geq 0, \tag{3.29}$$

where $\rho_{is} = (1 - \delta'_{is})\eta_{is}$ and $\perp$ represents complementary slackness i.e., both conditions must be satisfied but at least one with equality. Note that (3.29) shows that ρ_{is} are the minimum travel times between nodes i and s, and therefore, if $c_{ij}\left(\sum_{s\in\mathcal{D}} v_{ijs}\right) = \rho_{js} - \rho_{is}$, link ℓ_{ij} is in one of the least expensive path between nodes i and s and users traveling between those nodes can use link ℓ_{ij} ($v_{ijs} > 0$). Finally, μ_{ijs} represent the difference between the travel cost $c_{ij}(\cdot)$ of link ℓ_{ij} and the cost associated with traveling between nodes i and j through the minimum cost path.

We end this section by saying that a traffic model analogous to (3.20)-(3.22) can be obtained disaggregating the link flows by their origin nodes instead of the destination ones. However, none of these alternatives (disaggregating by origin or by destination) permits obtaining the routes that users have chosen.

The Castillo et al. model

In this section we propose a model for disaggregating link flows by their origin and destination nodes that will permit obtaining the paths actually used without the need of previous enumeration.

Given the origin-destination (OD) traffic flows, τ_{ks}, we can estimate the link flows associated with the different OD-pairs v_{ijks}, using the following optimization problem, which corresponds to the Wardrop equilibrium problem

$$\underset{\mathbf{v}}{\text{Minimize}}\, Z = \sum_{\ell_{ij}\in A} C_{ij}\left(\sum_{ks\in\mathcal{OD}} v_{ijks}\right) \tag{3.30}$$

subject to

$$\tau_{ks}(\delta'_{ik} - \delta'_{is}) = \sum_{\ell_{ij}\in\mathcal{A}} v_{ijks} - \sum_{\ell_{ji}\in\mathcal{A}} v_{jiks} : \lambda_{iks}; \quad \forall i \in \mathcal{N};\ ks \in \mathcal{OD}, \tag{3.31}$$

$$0 \leq v_{ijks} : \mu_{ijks}; \quad \forall \ell_{ij} \in \mathcal{A}, ks \in \mathcal{OD} \tag{3.32}$$

where δ'_{ik} are the Kronecker deltas ($\delta'_{ij} = 0$, if $i \neq j$ and $\delta'_{ii} = 1$). We have assumed that the cost on a link depends only on the flow on that link.

The problem (3.30)-(3.32) is a statement of the Beckmann et al. formulation of the Wardrop equilibrium problem, but stated for each OD pair (see Wardrop (1952) and Beckmann et al. (1956)). Note that equation (3.31) represents the flow balance associated with the OD-pair ks, for all nodes.

Once the values of the OD link flows v_{ijks} have been estimated using the optimization problem (3.30)-(3.32), we can easily calculate important flow information. Thus, the statement of the flow problem using the set of variables v_{ijks} has the following important advantages:

1. It avoids path enumeration.

2. The flows x_{ijk} coming from a given origin node k and using link ℓ_{ij} can be calculated as:

$$x_{ijk} = \sum_{s \in \mathcal{D}} v_{ijks}. \tag{3.33}$$

3. The flows y_{ijs} going to a given destination node s and using link ℓ_{ij} can be obtained as:

$$y_{ijs} = \sum_{k \in \mathcal{O}} v_{ijks}. \tag{3.34}$$

4. The total flow w_{ij} through link ℓ_{ij} is given by:

$$w_{ij} = \sum_{ks \in \mathcal{OD}} v_{ijks}. \tag{3.35}$$

5. The flow z_{iks} going from origin node k to destination node s and passing through node i is

$$z_{iks} = \sum_{i:\ell_{ij} \in \mathcal{A}} v_{ijks}. \tag{3.36}$$

6. We can identify and/or enumerate the flow paths very easily. To enumerate paths of an OD-pair (k, s), we can simply build the tree with accessible (with non-null flow v_{ijks} in some solution) branches (links) starting from the origin node k and ending with the destination node s.

7. Equations (3.33) to (3.36) and the selected variables v_{ijks} for the optimization problem (3.30)-(3.32) allow us incorporating new estimation techniques, based on information about $x_{ijk}, y_{ijs}, z_{iks}$ and/or v_{ijks}, which are not possible for other methods. This has important practical implications because new information based on traffic surveys (information about $x_{ijk}, y_{ijs}, z_{iks}$ and/or v_{ijks} data in Equations (3.33) to (3.36)) can be incorporated to the traffic flow estimation procedures.

We note that though the problem (3.30)-(3.32) has a unique solution in terms of total link flows, it can have infinitely many solutions in terms of v_{ijks}, that is, in terms of path flows, though they are equivalent in terms of link costs (they have the same link costs)[6].

The dual variables. Since it is important for practical purposes and for a better understanding of the next models, we will now analyze the meaning of the dual variables of problem (3.30)-(3.32).

[6]The proof of this statement can be made analogously to the one in Theorem 2.

The Lagrangian function of the problem is

$$\begin{aligned}\mathcal{L}(\mathbf{w},\boldsymbol{\lambda},\boldsymbol{\mu}) = &\sum_{\ell_{ij}\in\mathcal{A}} C_{ij}\left(\sum_{(k,s)\in\mathcal{OD}} v_{ijks}\right) - \sum_{\substack{\ell_{ij}\in\mathcal{A}\\(k,s)\in\mathcal{OD}}} v_{ijks}\mu_{ijks} \\ &+ \sum_{\substack{i\in\mathcal{N}\\(k,s)\in\mathcal{OD}}} \lambda_{iks}\left(\sum_{\ell_{ij}\in\mathcal{A}} v_{ijks} - \sum_{\ell_{ji}\in\mathcal{A}} v_{jiks} - \tau_{ks}(\delta'_{ik}-\delta'_{is})\right),\end{aligned} \tag{3.37}$$

and the associated KKT conditions:

$$\frac{\partial\mathcal{L}}{\partial v_{ijks}} = \sum_{\ell_{ij}\in A} c_{ij}\left(\sum_{(k,s)\in\mathcal{OD}} v_{ijks}\right) - \mu_{ijks} + \lambda_{iks} - \lambda_{jks} = 0; \quad \forall \ell_{ij}\in\mathcal{A},\ \forall ks\in\mathcal{OD} \tag{3.38}$$

$$\tau_{ks}(\delta'_{ik}-\delta'_{is}) = \sum_{\ell_{ij}\in\mathcal{A}} v_{ijks} - \sum_{\ell_{ji}\in\mathcal{A}} v_{jiks};\ \forall i\in\mathcal{N};\ \forall k\neq s, \tag{3.39}$$

$$v_{ijks} \geq 0; \quad \forall \ell_{i,j}\in\mathcal{A}, \forall k,s\in\mathcal{OD}, \tag{3.40}$$

$$\mu_{ijks}v_{ijks} = 0; \quad \forall \ell_{i,j}\in\mathcal{A}, \forall k,s\in\mathcal{OD}, \tag{3.41}$$

$$\mu_{ijks} \geq 0; \quad \forall \ell_{i,j}\in\mathcal{A}, \forall k,s\in\mathcal{OD}. \tag{3.42}$$

Analogously to dual variables of the Ferris et al. (1999) model, $\lambda_{iks}-\lambda_{jks}$ represent the minimum cost between nodes i and j. If link ℓ_{ij} is in one of the minimum cost paths between nodes k and s, then, its cost $c_{ij}(\cdot)$ equals the minimum cost between nodes i and j $(\lambda_{jks}-\lambda_{iks})$ and, therefore, it must be in use $(v_{ijks}>0)$. Furthermore, μ_{ijks} represents the overcost of traveling from node i to j through link ℓ_{ij} instead of using the least expensive path.

Looking for uniqueness

Following the ideas of the previous model, Castillo et al. (2008f) propose a model with unique solution, called the Wardrop-Minimum Variance equilibrium model (WMV). This model, which corresponds to a mixture of the Wardrop equilibrium and the minimum variance problems, can be stated as:

$$\underset{\mathbf{v}}{\text{Minimize}}\, Z = \sum_{\ell_{ij}\in\mathcal{A}} C_{ij}\left(\sum_{ks\in\mathcal{OD}} v_{ijks}\right) + \frac{\kappa}{m}\sum_{\ell_{ij}\in\mathcal{A}}\sum_{ks\in\mathcal{OD}}(v_{ijks}-\mu)^2 \tag{3.43}$$

subject to

$$\tau_{ks}(\delta'_{ik}-\delta'_{is}) = \sum_{\ell_{ij}\in\mathcal{A}} v_{ijks} - \sum_{\ell_{ji}\in\mathcal{A}} v_{jiks}; \quad \forall i\in\mathcal{N};\ ks\in\mathcal{OD}, \tag{3.44}$$

$$\mu = \frac{1}{m}\sum_{\ell_{ij}\in\mathcal{A}}\sum_{ks\in\mathcal{OD}} v_{ijks}, \tag{3.45}$$

$$0 \leq v_{ijks}; \quad \forall \ell_{ij}\in\mathcal{A}, ks\in\mathcal{OD} \tag{3.46}$$

where $\kappa > 0$ is a weighting factor, μ is the mean of the v_{ijks} variables, and m is its cardinal.

The problem (3.43)-(3.46) for $\kappa = 0$ becomes a pure Wardrop problem and has unique solution in terms of total link flows, but it can have infinitely many solutions in terms of v_{ijks}. Note that any exchange of users between equal cost subpaths does not alter the link flows nor the corresponding costs. So, given an optimal solution to the problem, exchanging different OD users from one subpath to the other leads to another optimal solution with different v_{ijks} values, though the same link flows w_{ij}. To solve this problem, one can choose small, but non-zero, value of κ (e.g., $\kappa = 0.000001$).

In this case, when $\kappa > 0$, (3.43) is strictly convex and the system (3.44)-(3.46) is compatible and convex. As a consequence, the problem (3.43)-(3.46) has a unique solution, which in addition is a global optimum.

We shall remark that in the problem (3.43)-(3.46) two objective functions are used (the two terms in (3.43)) and that they have a clear hierarchy, that is, the first term is the main function and the second term is the secondary function. Note also that we could have considered this problem in two steps: in the first step the first term could have been minimized, and in the second step we could have minimized the second term subject to no change in the first one. The selection of a very small value of κ permits us solving the problem in a single step.

In this case, the problem has a unique solution. The rationale of the problem (3.43)-(3.46) above consists of selecting among all the solutions of the pure Wardrop problem those minimizing the variance. This is achieved by using a very small value of κ, so that first a very low value of the first term in (3.43) is obtained, and next, a small (because κ is small) further improvement of the objective function is obtained by minimizing the variance (second term in (3.43)).

The Lo and Chen model

Although avoiding path enumeration is an important feature of the previous models, specially for large networks in which the number of routes is larger than the number of links, there are some cases where working with routes becomes necessary. This is the case of considering route costs as nonadditive, that is, the route costs are not the direct sum of link costs (e.g., in situations where people value travel time nonlinearly or in networks with nonlinear toll or fare structures) or when the travel cost function is not symmetric (e.g., when the link travel time depends not only on its own link flow but also on others link flows). An example of general cost route (Lo and Chen (2000) and Gabriel and Bernstein (1997)) is

$$c_{ksr} = \gamma_{ksr} + \sum_{\ell_{ij}} \rho_1 \delta_{ijr} \xi_{rks} c_{ij} + \varphi_{ksr} \left(\sum_{\ell_{ij}} \delta_{ijr} \xi_{rks} c_{ij} \right), \tag{3.47}$$

where γ_{ksr} denotes the financial costs (such as tolls) specific to route r, ρ_1 is the operating costs per travel time (e.g., fuel consumption), and φ_{ksr} is a function describing the value

of time for route r, which could be non linear.

In Lo and Chen (2000) this problem is discussed and a mathematical programming formulation with a smooth and convex gap function is developed. Furthermore, in order to avoid the procedure of exhaustive route enumeration, they proposed to work with a set of *a priori* paths based on travelers' preferences or interviews, which has the advantage of explicitly identifying those routes that would likely be used. In this section, we will explain Lo and Chen model but before we need to define what a gap function is.

Definition 2 (Gap function) *Let Ω be the set of solutions of a non-necessarily linear system of equations. A function $G : \mathbb{R}^n_+ \to \mathbb{R}^1_+$ is a gap function for the system of equations if*

(i) $G(\mathbf{x}) = 0 \Leftrightarrow \mathbf{x} \in \Omega$,

(ii) $G \geq 0$.

In essence, the gap function provides a measure of convergence of the system of equations at any point $\mathbf{x}$. By minimizing G over $\mathbf{x}$, a point in Ω is obtained.

Facchinei and Soares (1995) suggested three desirable properties of a gap function:

(i) smooth (or differentiable),

(ii) convex: every stationary point is a global solution,

(iii) bounded.

These properties are important from a computational point of view. If they are satisfied, the Mathematical Programming (MP) formulation can be solved efficiently by one of the existing optimization algorithms.

There are several examples of gap functions in the literature but they all failed in one of the previous properties. For example, Hearn (1982) proposed this gap function: $G(\tilde{\mathbf{v}}) = \max s(\tilde{\mathbf{v}})^T \cdot (\tilde{\mathbf{v}} - \mathbf{v})$, where $\mathbf{v}, \tilde{\mathbf{v}}, s(\tilde{\mathbf{v}})$ are, respectively, any set of link flows, the set of link flows at equilibrium, and the set of link travel times at equilibrium. This MP involves solving the minimax problem $\min_{\tilde{\mathbf{v}}} \max_{\mathbf{v}} s(\tilde{\mathbf{v}})^T \cdot (\tilde{\mathbf{v}} - \mathbf{v})$, which in general is not a smooth and convex programming problem. Smith (1983) used this gap function: $G = \sum_{ks} \sum_{r} \{f_{ksr}[c_{ksr} - c_{ksl}]_+\}^m$,where r and l are two paths between OD pair ks and $[c_{ksr} - c_{ksl}]_+ = \max\{0, c_{ksr} - c_{ksl}\}$. The Smith's gap function is not convex and, as $[c_{ksr} - c_{ksl}]_+$ is nondifferentiable, the term $\{f_{ksr}[c_{ksr} - c_{ksl}]_+\}$ is raised to the second or higher powers (i.e., $m \geq 2$) to make G differentiable.

Lo and Chen (2000) reformulate the traffic assignment problem (3.1)-(3.2) (including the demand constraint (3.4)) as:

$$\underset{\mathbf{f},\boldsymbol{\pi}}{\text{Minimize}} \; G(f_{ksr}, \pi_{ks}) = \sum_{k,s \in \mathcal{OD}} \sum_{r \in \mathcal{R}} f_{ksr} \left(c_{ksr} - \pi_{ks}\right) \tag{3.48}$$

subject to

$$\sum_{r\in\mathcal{R}} f_{ksr} = \tau_{ks}; \quad \forall k,s \in \mathcal{OD} \tag{3.49}$$

$$c_{ksr} - \pi_{ks} \geq 0; \quad \forall r \in \mathcal{R},\ \forall k,s \in \mathcal{OD} \tag{3.50}$$

$$\pi_{ks} \geq 0; \quad \forall k,s \in \mathcal{OD}, \tag{3.51}$$

$$f_{ksr} \geq 0; \quad \forall r \in \mathcal{R},\ \forall k,s \in \mathcal{OD}, \tag{3.52}$$

where $G(f_{ksr}, \pi_{ks})$ is the proposed gap function.

Proposition 1 (Equivalency between models) *Let Ω be the set of solutions to the UE conditions (3.1), (3.2), (3.4) with non-negative variables f_{ksr} and π_{ks}. Function $G(f_{ksr}, \pi_{ks}) : \mathbb{R}^n_+ \to \mathbb{R}^1_+$ is a gap function for these conditions.*

Proof.

(i) This gap function G satisfies the condition: $G(\mathbf{x}) = 0 \Leftrightarrow \mathbf{x} \in \Omega$, where $\mathbf{x} = (\mathbf{f}, \boldsymbol{\pi})$, $\mathbf{f}$ denotes the vector of $\{f_{ksr}\}$, and $\boldsymbol{\pi}$ denotes the vector of $\{\pi_{ks}\}$.

Necessity. Given the UE condition $f_{ksr}(c_{ksr} - \pi_{ks}) = 0$, summing it for all ks and for all r yields that $G = 0$.

Sufficiency. Constraints (3.50)-(3.52) ensure that each term $f_{ksr}(c_{ksr} - \pi_{ks})$ of G is nonnegative. If $G = 0$, then each term $f_{ksr}(c_{ksr} - \pi_{ks}) = 0$, which is the UE condition.

(ii) The gap function satisfies this second condition: $G(\mathbf{x}) \geq 0$, as ensured by constraints (3.50)-(3.52).

■

Furthermore, this gap function G satisfies the three desirable properties recommended by Facchinei and Soares (1995), as shown in Lo and Chen (2000).

Proposition 2 *The objective function of the mathematical program (3.48)-(3.52) is*

(i) differentiable,

(ii) convex, and

(iii) bounded,

if the route cost function is convex and monotonic with respect to path flows.

Proof. First we remind the reader of the mathematical definition of convexity and monotonicity.

- The route cost function $\mathbf{n}$ is convex with respect to path flows $\mathbf{f}$ if

$$\mathbf{n}(\theta \cdot \mathbf{f_1} + (1-\theta) \cdot \mathbf{f_2}) \leq \theta \cdot \mathbf{n}(\mathbf{f_1}) + (1-\theta) \cdot \mathbf{n}(\mathbf{f_2}). \tag{3.53}$$

- The route cost function $\mathbf{n}$ is monotonic with respect to path flows $\mathbf{f}$ if[7]

$$(\mathbf{f_1} - \mathbf{f_2})^T \cdot (\mathbf{n_1} - \mathbf{n_2}) \geq 0,$$

or simplifying,

$$\mathbf{f_1}^T \cdot \mathbf{n_2} + \mathbf{f_2}^T \cdot \mathbf{n_1} \leq \mathbf{f_1}^T \cdot \mathbf{n_1} + \mathbf{f_2}^T \cdot \mathbf{n_2}. \tag{3.54}$$

(i) The partial derivatives of G are $\dfrac{\partial G}{\partial f_{ksr}} = (c_{ksr} - \pi_{ks}) + \sum_{mn} \sum_{l} f_{mnl} \left(\dfrac{\partial c_{mnl}}{\partial f_{ksr}} \right)$ and $\dfrac{\partial G}{\partial \pi_{ks}} = \sum_{r} f_{ksr} = \tau_{ks}$. For differentiable route travel cost functions, (i.e., $\dfrac{\partial c_{mnl}}{\partial f_{ksr}}$ is smooth), G is differentiable with respect to f_{ksr}. Moreover, G is differentiable with respect to π_{ks} if τ_{ks} is a differentiable function. For the fixed-demand case, τ_{ks} is a given constant which is differentiable.

(ii) To prove that G is convex, we separate G into two parts, so that $G = G_1 + G_2$:

$$G_1 = \sum_{k,s \in \mathcal{OD}} \sum_{r \in \mathcal{R}} f_{ksr} c_{ksr}, \qquad G_2 = -\sum_{k,s \in \mathcal{OD}} \sum_{r \in \mathcal{R}} f_{ksr} \pi_{ks}. \tag{3.55}$$

G is convex if both G_1 and G_2 are convex (since the sum of convex functions is convex). G_1 is the total travel time of all the route flows which is a function of route flows only and equivalent to the total system cost. By definition of convexity, G_1 is convex if

$$G_1(\theta \cdot \mathbf{f_1} + (1-\theta) \cdot \mathbf{f_2}) \leq \theta \cdot G_1(\mathbf{f_1}) + (1-\theta) \cdot G_1(\mathbf{f_2}), \tag{3.56}$$

where $\mathbf{f_1}, \mathbf{f_2}$ are two route flow vectors satisfying constraints (3.49) and (3.52), and $0 \leq \theta \leq 1$. Since $G_1 = \mathbf{f}^T \cdot \mathbf{n}$, the left-hand side (LHS) of (3.56) can be written as

$$LHS = [\theta \cdot \mathbf{f_1} + (1-\theta) \cdot \mathbf{f_2}]^T \cdot \mathbf{n}(\theta \cdot \mathbf{f_1} + (1-\theta) \cdot \mathbf{f_2}). \tag{3.57}$$

Using (3.53) and (3.54), equation (3.57) can be written as

$$\begin{aligned} LHS &\leq [\theta \cdot \mathbf{f_1} + (1-\theta) \cdot \mathbf{f_2}]^T \cdot [\theta \cdot \mathbf{n_1} + (1-\theta) \cdot \mathbf{n_2}] \\ &= \theta^2 \cdot \mathbf{f_1}^T \cdot \mathbf{n_1} + (1-\theta)^2 \cdot \mathbf{f_2}^T \cdot \mathbf{n_2} + \theta(1-\theta)[\mathbf{f_1}^T \cdot \mathbf{n_2} + \mathbf{f_2}^T \cdot \mathbf{n_1}] \\ &\leq \theta \cdot \mathbf{f_1}^T \cdot \mathbf{n_1} + (1-\theta) \cdot \mathbf{f_2}^T \cdot \mathbf{n_2} = \theta \cdot G_1(\mathbf{f_1}) + (1-\theta) \cdot G_1(\mathbf{f_2}) = RHS. \end{aligned} \tag{3.58}$$

[7] For notational simplicity, let $\mathbf{n}(\mathbf{f_1})=\mathbf{n_1}$ and $\mathbf{n}(\mathbf{f_2})=\mathbf{n_2}$.

This proves the convexity of G_1. Next, we prove that G_2 is convex. Using the demand constraint (3.49), note that

$$G_2 = -\sum_{k,s\in\mathcal{OD}}\sum_{r\in\mathcal{R}} f_{ksr}\pi_{ks} = -\sum_{k,s\in\mathcal{OD}} \pi_{ks}\sum_{r\in\mathcal{R}} f_{ksr} = -\sum_{k,s\in\mathcal{OD}} \pi_{ks}\tau_{ks} = -\boldsymbol{\pi}^T\cdot\mathbf{q}. \tag{3.59}$$

Therefore, G_2 is a function only of $\boldsymbol{\pi}$ with $\mathbf{q}$ being a set of given demands. G_2 is convex if

$$G_2(\theta\cdot\boldsymbol{\pi}_1 + (1-\theta)\cdot\boldsymbol{\pi}_2) \le \theta\cdot G_2(\boldsymbol{\pi}_1) + (1-\theta)\cdot G_2(\boldsymbol{\pi}_2), \tag{3.60}$$

where $\boldsymbol{\pi}_1$, $\boldsymbol{\pi}_2$ are two arbitrary vectors of nonnegative $\{\pi_{ks}\}$. The LHS can be written as

$$\begin{aligned} LHS &= -[\theta\cdot\boldsymbol{\pi}_1 + (1-\theta)\cdot\boldsymbol{\pi}_2]^T\cdot\mathbf{q} = [-\theta\cdot{\boldsymbol{\pi}_1}^T\mathbf{q} - (1-\theta)\cdot{\boldsymbol{\pi}_2}^T\mathbf{q}] \\ &= [\theta\cdot G_2(\boldsymbol{\pi}_1) + (1-\theta)\cdot G_2(\boldsymbol{\pi}_2)] = RHS, \end{aligned} \tag{3.61}$$

which satisfies (3.60). This proves the convexity of G_2 and, hence, of G.

(iii) G is bounded if every term $f_{ksr}(c_{ksr} - \pi_{ks})$ is. The upper bound value of each term is obtained by loading the entire demand on the longest route and taking $\pi_{ks} = 0$. That is, $f_{ksr}(c_{ksr} - \pi_{ks}) \le \tau_{ks}\tilde{c}_{ksr}$, where $\tilde{c}_{ksr}$ is the longest route cost. Thus, there is always a large enough constant α such that $\tau_{ks}\tilde{c}_{ksr} \le \alpha$.

■

For computational efficiency, this formulation puts three requirements on the route cost function: smoothness, monotonicity and convexity. A basic route cost function which fulfills this requirements is one with additive link costs (e.g. the one used in the previous models), such as $c_{ksr} = \sum_{\ell_{ij}} \delta_{ijr}\psi_{ksr}c_{ij}$, where the route cost function is monotonic and convex if the link flow functions are. Nevertheless, as stated in Lo and Chen (2000), the real benefit of this formulation is its ability to model nonlinear or general route costs. The general route cost in (3.47) is a money based function but other kind of functions, based on time, have been proposed. For instance, Larsson et al. (2002) proposed the following function:

$$c_{ksr} = \sum_{\ell_{ij}} \delta_{ijr}\psi_{ksr}c_{ij} + \phi_{ksr}\left(m_{ksr}\right), \tag{3.62}$$

where m_{ksr} is the monetary outlay (e.g., route-specific financial cost which is allowed to vary according to route) and the function ϕ_{ksr} converts money into time.

Although both function (3.47) and (3.62) may look similar, it has been noted by Bernstein and Wynter (2000) that even if one chooses $\phi_{ksr} = \varphi_{ksr}^{-1}$ in (3.62), this will not make the two formulations equivalent.

3.2.2 Nonlinear-complementary problems

As it has been said in Section 3.1, several different formulations have been proposed in the literature to solve the traffic assignment problem. In Section 3.2.1, some mathematical programming approaches have been shown. In this section, we will focus on Nonlinear Complementary Problems (NCP), that were first used to state the traffic assignment problem in Aashtiani (1979). Before continuing with this section, we remind the reader of the UE equations in Section 3.2:

$$
\begin{aligned}
f_r(c_r - \pi_{ks(r)}) &= 0; \quad \forall r, ks, && (3.63)\\
c_r - \pi_{ks(r)} &\geq 0; \quad \forall r, ks, && (3.64)\\
\sum_{r \in \mathcal{R}} \xi_{ksr} f_r &= \tau_{ks}; \quad \forall k, s \in \mathcal{OD} && (3.65)\\
\sum_{\ell_{ij} \in \mathcal{A}} c_{ij} \delta_{ijr} &= c_r; \quad \forall r \in \mathcal{R}, && (3.66)\\
f_r &\geq 0; \quad \forall r \in \mathcal{R}, && (3.67)\\
\pi_{ks} &\geq 0; \quad \forall ks \in \mathcal{OD}, && (3.68)
\end{aligned}
$$

Definition 3 (Non-linear Complementary Problem (NCP)) *Let* $F(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_n(\mathbf{x}))$ *be a vector-valued function* $F : \mathbb{R}^n \to \mathbb{R}^n$. *Then a vector* $\mathbf{x} \in \mathbb{R}^n$ *is called a complementary solution of the NCP if it satisfies the following conditions:*

$$
\begin{aligned}
\mathbf{x} \cdot F(\mathbf{x}) &= 0, && (3.69)\\
F(\mathbf{x}) &\geq 0, && (3.70)\\
\mathbf{x} &\geq 0. && (3.71)
\end{aligned}
$$

In this section we show that the traffic equilibrium problem (3.63)-(3.68) has a complementary nature and, hence, can be stated by means of a nonlinear complementary problem. It is clear that equations (3.63), (3.64) and (3.67) are complementary in nature but it is not the case for the remaining equations. Therefore, we now show that the rest of the equations can also be expressed in a complementary form.

First, we need to do some simplification in the formulation, using the notation in Aashtiani (1979). Let $\mathbf{x} = (\mathbf{f}, \boldsymbol{\pi}) \in \mathbb{R}^n$ where $n = n_1 + n_2$, $n_1 = |\mathcal{R}|$ is the number of routes, $n_2 = |\mathcal{OD}|$ is the number of OD pairs, $\mathbf{f}$ denotes the vector of $\{f_r\}$, and $\boldsymbol{\pi}$ denotes the vector of $\{\pi_{ks}\}$. Furthermore, let

$$
p_{ksr}(\mathbf{x}) = c_r(\mathbf{f}) - \pi_{ks}, \ \forall r \in \mathcal{R}, \ \forall ks \in \mathcal{OD} \text{ and } g_{ks}(\mathbf{x}) = \sum_{r \in \mathcal{R}} \xi_{ksr} f_r - \tau_{ks}(\boldsymbol{\pi}), \ \forall ks \in \mathcal{OD},
$$

where $\tau_{ks}(\boldsymbol{\pi})$ is the demand function that depends on the minimum travel time[8] and we have added the argument to the route cost function $c_r(\mathbf{f})$ in order to clarify its dependence structure. Finally, let $F(\mathbf{x}) = (p_{ksr}(\mathbf{x})\ \forall r \in \mathcal{R}\,,\ \forall ks \in \mathcal{OD};\ g_{ks}(\mathbf{x})\ \forall ks \in \mathcal{OD}) \in \mathbb{R}^n$. Once F is defined as a vector-valued function from $\mathbb{R}^n$ into itself, we can write the UE problem as the following nonlinear complementary problem:

$$\mathbf{x} \cdot F(\mathbf{x}) = 0, \tag{3.72}$$

$$F(\mathbf{x}) \geq 0, \tag{3.73}$$

$$\mathbf{x} \geq 0. \tag{3.74}$$

Theorem 3 *Suppose that for all $\ell_{ij} \in \mathcal{A}$, $c_{ij} : \mathbb{R}^{n_1}_+ \to \mathbb{R}^1_+$ is a positive function and, for all $ks \in \mathcal{OD}$, $\tau_{ks} : \mathbb{R}^{n_2}_+ \to \mathbb{R}^1_+$ is a nonnegative function. Then, the user equilibrium system (3.63)-(3.68) is equivalent to the nonlinear complementary system (3.72)-(3.74).*

Proof. The proof of this theorem consists of two parts: any solution to the user-equilibrium system (3.63)-(3.68) is a solution to the nonlinear complementary system (3.72)-(3.74); and, any solution to (3.72)-(3.74) is a solution of (3.63)-(3.68). But before continuing with the demonstration, note that the only equation of the system (3.72)-(3.74) that is not included in (3.63)-(3.68) is $\pi_{ks}(\sum_{r\in\mathcal{R}} \xi_{ksr} f_r - \tau_{ks}(\boldsymbol{\pi})) = 0$ (which can be drawn from equation (3.72)). Therefore, the proof can be restated as proving that: 1) every solution to (3.63)-(3.68) fulfills $g_{ks}(\mathbf{x})\pi_{ks} = 0$; and 2) any solution to (3.72)-(3.74) is a solution of (3.63)-(3.68).

(i) Since $g_{ks}(\mathbf{x}) = \sum_{r\in\mathcal{R}} \xi_{ksr} f_k - \tau_{ks}(\boldsymbol{\pi}) = 0$ in the user equilibrium because of equation (3.65), it is obvious that any solution to (3.63)-(3.68) is a solution to (3.72)-(3.74).

(ii) We will use here the proof by contradiction. Suppose that there is a $\mathbf{x} = (\mathbf{f}, \boldsymbol{\pi})$ satisfying (3.72)-(3.74), but that $g_{ks}(\mathbf{x}) = (\sum_{r\in\mathcal{R}} \xi_{ksr} f_r - \tau_{ks}(\boldsymbol{\pi})) > 0$ (and hence it is not a solution to (3.63)-(3.68)). Then $g_{ks}(\mathbf{x})\pi_{ks} = 0$ implies that $\pi_{ks} = 0$. Also, since τ_{ks} is non-negative $\sum_{r\in\mathcal{R}} \xi_{ksr} f_r > \tau_{ks}(\boldsymbol{\pi}) \geq 0$ which implies that $f_r > 0$ for some r. But, for this particular r, equation $p_{ksr}(\mathbf{x}) f_r = 0$ implies that $p_{ksr} = c_r(\mathbf{f}) - \pi_{ks} = 0$ and, hence, $c_r(\mathbf{f}) = \pi_{ks}$. But, since $\pi_{ks} = 0$, $c_r = \sum_{\ell_{ij}\in\mathcal{A}} c_{ij}\delta_{ijr} = 0$ which contradicts the assumption $c_{ij} > 0$.

■

Remark 1 *For the constant demand function case, τ_{ks} is also a non-negative function and, hence, this theorem is also applicable.*

[8] In the previous sections, the demand function has been assumed constant ($\tau_{ks}(\mathbf{u}) = \tau_{ks}$). Nevertheless, in the rest of this section it will be considered variable and results to the constant case will be extended.

Next, we study the existence and uniqueness of solutions of the complementary problem (3.72)-(3.74). Note that, as this problem is equivalent to the UE problem, the conclusions drawn in the following pages about the existence and uniqueness of solutions must agree with the ones in Section 3.2.1. For the following results we will need to introduce some concepts.

Definition 4 (Monotonicity) *Let $F : \mathcal{D} \to \mathcal{E}^n,\ \mathcal{D} \subset \mathcal{E}^n$.*

(i) The function is said to be monotone on $\mathcal{D}$ if, for every pair $x \in \mathcal{D}$ and $y \in \mathcal{D}$, we have

$$(x - y)\,(F(x) - F(y)) \geq 0.$$

(ii) F is said to be strictly monotone on $\mathcal{D}$ if for every pair $x \in \mathcal{D}$ and $y \in \mathcal{D}$ with $x \neq y$, we have

$$(x - y)\,(F(x) - F(y)) > 0.$$

Theorem 4 (Existence of solution) *Suppose that $(\mathcal{N}, \mathcal{A})$ is a strongly connected network. Suppose that for all $\ell_{ij} \in \mathcal{A}$, $c_{ij} : \mathbb{R}^{n_1}_+ \to \mathbb{R}^1_+$ is a nonnegative function and, for all $ks \in \mathcal{OD}$, $\tau_{ks} : \mathbb{R}^{n_2}_+ \to \mathbb{R}^1_+$ is a continuous function that is bounded from above. Then, the nonlinear complementary problem (3.72)-(3.74), and hence the user equilibrium system (3.63)-(3.68), has a solution.*[9]

Proof. This proof has two parts: 1) the NCP has a solution, 2) the user equilibrium system has also a solution. The first part of the proof can be found in Aashtiani (1979) and, once that the nonlinear complementary problem has been shown to have a solution, with the equivalence Theorem 3, the existence of solutions to system (3.63)-(3.68) becomes proved. ∎

Remark 2 *For the constant demand function case, τ_{ks} is obviously a continuous and bounded from above function. Thus, this theorem is also applicable and the problem (3.72)-(3.74) when the demands are constant has also a solution.*

As it has been shown in Section 3.2.1, the solution to the user equilibrium problem is not unique in terms of path flows but it is in terms of link flows. In the following theorem we will prove that this is also the case when the UE approach is stated as a NCP. The reason why in terms of path flows the solution is not unique is based on the fact that the link travel time function is not strictly monotone with respect to route flows. Note that the cost of a link $c_{ij}\left(\sum_{r \in \mathcal{R}} f_r \delta_{ijr}\right)$ will remain unchanged if the flow of a route that does

[9] Note that these assumptions are not restrictive as the link travel cost functions are always nonnegative, and the demand function is bounded from above as the users willing to travel between an OD pair is always finite.

not contain that link is augmented. On the contrary, if the link cost function is stated in terms of link flows w_{ij}, then this cost function $c_{ij}(w_{ij})$ is strictly monotone.

For the following theorems, we need to introduce some new notation. Let $\mathbf{w}$ be the vector of link flows w_{ij}, $c(\mathbf{w})$, the vector of volume delay functions[10] and $\boldsymbol{\tau}(\boldsymbol{\pi})$, the vector of demand functions. Let also $\Delta = (\delta_{ijr})$ and $\Gamma = (\xi_{ksr})$ be the arc-path and path-OD incidence matrices, respectively. Then, the system (3.63)-(3.68) can be restated as:

$$\begin{aligned} \left(\Delta^T \cdot c\left(\Delta\mathbf{f}\right) - \Gamma \cdot \boldsymbol{\pi}\right) \cdot \mathbf{f} &= 0 && (3.75)\\ \Delta^T \cdot c\left(\Delta\mathbf{f}\right) - \Gamma \cdot \boldsymbol{\pi} &\geq 0 && (3.76)\\ \Gamma^T \cdot \mathbf{f} - \boldsymbol{\tau}(\boldsymbol{\pi}) &= 0 && (3.77)\\ \mathbf{f} &\geq 0 && (3.78)\\ \boldsymbol{\pi} &\geq 0 && (3.79) \end{aligned}$$

Now, let $G(\mathbf{x}) = \left(c\left(\Delta\mathbf{f}\right), -\boldsymbol{\tau}(\boldsymbol{\pi})\right)$,where $\mathbf{x} = (\mathbf{f}, \boldsymbol{\pi})$ and $G : \mathbb{R}^n_+ \to \mathbb{R}^m_+$ with $n = n_1 + n_2$ and $m = |\mathcal{A}| + n_2$, and

$$\bar{\Delta} = \begin{pmatrix} \Delta & \mathbf{0} \\ \mathbf{0} & \mathbf{I} \end{pmatrix} \quad \text{and} \quad \bar{\Gamma} = \begin{pmatrix} \mathbf{0} & -\Gamma \\ \Gamma^T & \mathbf{0} \end{pmatrix},$$

with dimensions $m \times n$ and $n \times n$, respectively, and $\mathbf{I}$ is the identity matrix with dimensions $n_2 \times n_2$. Then, the corresponding nonlinear complementary problem can be written as follows:

$$\begin{aligned} \left(\bar{\Delta}^T G\left(\bar{\Delta}\mathbf{x}\right) + \bar{\Gamma}\mathbf{x}\right)\mathbf{x} &= 0 && (3.80)\\ \bar{\Delta}^T G\left(\bar{\Delta}\mathbf{x}\right) + \bar{\Gamma}\mathbf{x} &\geq 0 && (3.81)\\ \mathbf{x} &\geq 0. && (3.82) \end{aligned}$$

It is easy to show that $\bar{\Delta}^T G\left(\bar{\Delta}\mathbf{x}\right) + \bar{\Gamma}\mathbf{x} = F(\mathbf{x})$ where F is the one used in equations (3.72)-(3.74). Therefore, (3.80)-(3.82) is equivalent to (3.72)-(3.74). The following lemma has been proved in Aashtiani (1979).

Lemma 1 *Let $K \subset \mathbb{R}^n$, let B be an $m \times n$ matrix and let $L = \{B\mathbf{x} | \mathbf{x} \in K\} \subset \mathbb{R}^m$. Suppose that $g : L \to \mathbb{R}^m$ is strictly monotone on L. Let A be an $n \times n$ positive semi-definite matrix. Define $f : \mathbb{R}^n \to \mathbb{R}^m$ by $f(\mathbf{x}) = B^T g(B\mathbf{x}) + A\mathbf{x}$. Then, $B\mathbf{x}$ has the same value for all of these solutions.*

Proof. Suppose that $\mathbf{x}^1$ and $\mathbf{x}^2$, $\mathbf{x}^1 \neq \mathbf{x}^2$, solve the nonlinear complementary problem, i.e.,

$$\mathbf{x}^i \geq 0,\ f(\mathbf{x}^i) \geq 0, \text{ and } \mathbf{x}^i f(\mathbf{x}^i) = 0, \quad \text{for } i = 1, 2,$$

[10]Note that $c(\mathbf{w}) = c(\mathbf{f})$ as there exist formulas providing $\mathbf{w}$ from $\mathbf{f}$.

then

$$\mathbf{x}^2 f(\mathbf{x}^1) \geq 0\ ,\ \mathbf{x}^1 f(\mathbf{x}^2) \geq 0,$$

and consequently

$$\begin{cases} \left(\mathbf{x}^2 - \mathbf{x}^1\right) f\left(\mathbf{x}^1\right) \geq 0 \\ \left(\mathbf{x}^1 - \mathbf{x}^2\right) f\left(\mathbf{x}^2\right) \geq 0 \end{cases}$$

which implies that

$$\left(\mathbf{x}^1 - \mathbf{x}^2\right)\left(f\left(\mathbf{x}^1\right) - f\left(\mathbf{x}^2\right)\right) \leq 0$$

or

$$\left(\mathbf{x}^1 - \mathbf{x}^2\right)\left(B^T g(B\mathbf{x}^1) + A\mathbf{x}^1 - B^T g(B\mathbf{x}^2) - A\mathbf{x}^2\right) \leq 0$$

or

$$\left(\mathbf{x}^1 - \mathbf{x}^2\right)\left[B^T\left(g(B\mathbf{x}^1) - g(B\mathbf{x}^2)\right)\right] + \left(\mathbf{x}^1 - \mathbf{x}^2\right) A \left(\mathbf{x}^1 - \mathbf{x}^2\right) \leq 0.$$

Since A is positive semi-definite, $\left(\mathbf{x}^1 - \mathbf{x}^2\right) A \left(\mathbf{x}^1 - \mathbf{x}^2\right) \geq 0$, implying that

$$\left(\mathbf{x}^1 - \mathbf{x}^2\right)\left[B^T\left(g(B\mathbf{x}^1) - g(B\mathbf{x}^2)\right)\right] \leq 0,$$

or

$$\left(B\mathbf{x}^1 - B\mathbf{x}^2\right)\left(g(B\mathbf{x}^1) - g(B\mathbf{x}^2)\right) \leq 0. \tag{3.83}$$

But g is strictly monotone on L, therefore $B\mathbf{x}^1 = B\mathbf{x}^2$. This concludes the proof. ■

Theorem 5 (Uniqueness) *For a strongly connected network $(\mathcal{N}, \mathcal{A})$, suppose that $\mathbf{c}$ is the vector of the volume delay functions, and $-\boldsymbol{\tau}$, the vector of the negative demand functions, are strictly monotone. Then, the arc volumes, $\mathbf{w}$, and the vector of travel times in equilibrium, $\boldsymbol{\pi}$, are unique.*

Proof. With the notation explained previously, we have that $\mathbf{G} = (\mathbf{c}, -\boldsymbol{\tau})$ is strictly monotone on $L = \{\bar{\Delta}\mathbf{x} = (\mathbf{w}, \boldsymbol{\pi}) : \mathbf{x} = (\mathbf{f}, \boldsymbol{\pi}) \in \mathcal{R}^n\}$ and that $\bar{\Gamma}$ is a positive semi-definite matrix. Thus, with $g = G$, $f = F$, $B = \bar{\Delta}$ and $A = \bar{\Gamma}$, by Lemma 1, $\bar{\Delta}\mathbf{x} = (\mathbf{w}, \boldsymbol{\pi})$ is unique for the nonlinear complementary system (3.72)-(3.74) which implies that the arc volumes $\mathbf{w}$ and the travel times in equilibrium π_{ks} are unique for the user equilibrium problem (3.63)-(3.68). ■

Note that both of the functions $\mathbf{c}$ and $-\boldsymbol{\tau}$ are required to be strictly monotone to insure the uniqueness of $(\mathbf{w}, \boldsymbol{\pi})$. In the next theorem we show that this restriction can be relaxed, and that uniqueness of $\boldsymbol{\pi}$ is maintained if either $\mathbf{c}$ or $-\boldsymbol{\tau}$ is strictly monotone.

Theorem 6 *Suppose that $\mathbf{c}$ and $-\boldsymbol{\tau}$ are monotone functions. If either of $\mathbf{c}$ or $-\boldsymbol{\tau}$ is strictly monotone, then $\boldsymbol{\pi}$ is unique. Also, if $\mathbf{c}$ is strictly monotone, then $(\mathbf{w}, \boldsymbol{\pi})$ is unique.*

Proof. Suppose that $\mathbf{x}^1 = (\mathbf{f}^1, \boldsymbol{\pi}^1)$ and $\mathbf{x}^2 = (\mathbf{f}^2, \boldsymbol{\pi}^2)$, $\mathbf{x}^1 \neq \mathbf{x}^2$, are two solutions. As in Lemma 1, with $g = G$, $f = F$, $B = \bar{\Delta}$ and $A = \bar{\Gamma}$, we have by equation (3.83)

$$\left(\bar{\Delta}\mathbf{x}^1 - \bar{\Delta}\mathbf{x}^2\right)\left(G(\bar{\Delta}\mathbf{x}^1) - G(\bar{\Delta}\mathbf{x}^2)\right) \leq 0.$$

But $\mathbf{G} = (\mathbf{c}, -\boldsymbol{\tau})$ is monotone as it has monotone components, i.e.,

$$\left(\bar{\Delta}\mathbf{x}^1 - \bar{\Delta}\mathbf{x}^2\right)\left(G(\bar{\Delta}\mathbf{x}^1) - G(\bar{\Delta}\mathbf{x}^2)\right) \geq 0.$$

Therefore,

$$\left(\bar{\Delta}\mathbf{x}^1 - \bar{\Delta}\mathbf{x}^2\right)\left(G(\bar{\Delta}\mathbf{x}^1) - G(\bar{\Delta}\mathbf{x}^2)\right) = 0, \tag{3.84}$$

that can be rewritten as

$$\left(\Delta\mathbf{f}^1 - \Delta\mathbf{f}^2\right)\left(\mathbf{c}\left(\Delta\mathbf{f}^1\right) - \mathbf{c}\left(\Delta\mathbf{f}^2\right)\right) + \left(\boldsymbol{\pi}^1 - \boldsymbol{\pi}^2\right)\left(-\boldsymbol{\tau}\left(\boldsymbol{\pi}^1\right) + \boldsymbol{\tau}\left(\boldsymbol{\pi}^2\right)\right) = 0. \tag{3.85}$$

But both $\mathbf{c}$ and $-\boldsymbol{\tau}$ are monotone functions, thus each term in (3.85) is zero; that is,

$$\left(\Delta\mathbf{f}^1 - \Delta\mathbf{f}^2\right)\left(\mathbf{c}\left(\Delta\mathbf{f}^1\right) - \mathbf{c}\left(\Delta\mathbf{f}^2\right)\right) = 0 \tag{3.86}$$

$$-\left(\boldsymbol{\pi}^1 - \boldsymbol{\pi}^2\right)\left(\boldsymbol{\tau}\left(\boldsymbol{\pi}^1\right) - \boldsymbol{\tau}\left(\boldsymbol{\pi}^2\right)\right) = 0. \tag{3.87}$$

If $-\boldsymbol{\tau}$ is strictly monotone, then equation 3.87 implies that $\boldsymbol{\pi}^1 = \boldsymbol{\pi}^2$ and, therefore, $\boldsymbol{\pi}$ is unique.

Now, suppose that $\mathbf{c}$ is the strictly monotone function. Then, equation (3.86) implies that $\mathbf{w}^1 = \Delta\mathbf{f}^1 = \Delta\mathbf{f}^2 = \mathbf{w}^2$, or that the arc volume vector $\mathbf{w}$ is unique. But uniqueness of the arc volume vector implies that the travel times, $c_{ij}(\mathbf{w})$, on each arc are unique, which obviously implies that $\boldsymbol{\pi}$ is unique. ∎

Remark 3 *For the constant demand function case, τ_{ks} is a monotone and continuous function so, if $\mathbf{c}$ is strictly monotone on link flows, Theorem 6 is applicable, and both $\mathbf{w}$ and $\boldsymbol{\pi}$ are unique.*

3.2.3 Variational inequality problems

In the previous sections different approaches (including mathematical programming and nonlinear complementary problems) to solve the static traffic assignment problem have been discussed. In this section, we will present the Variational Inequality Problems (VIP) that were first used to solve the traffic assignment problem in Dafermos (1980). VIPs can be seen as an extension to nonlinear complementary problems, as we will show later. Therefore, we give only a brief introduction to variational inequality problems and some demonstrations will be skipped because of its similarity to the NCP ones.

Definition 5 (Variational Inequality Problem (VIP)) *A variational inequality problem, VI(F,K), is to determine a vector $\mathbf{x}^* \in K \subset \mathbb{R}^n$, such that*

$$\langle F\left(\mathbf{x}^*\right)^T, \mathbf{x} - \mathbf{x}^* \rangle \geq 0, \quad \forall \mathbf{x} \in K, \tag{3.88}$$

where F is a given continuous function from K to $\mathcal{R}^n$ and K is a given closed convex set.

In geometric terms, variational inequality (3.88) states that $F(\mathbf{x}^*)^T$ is orthogonal to the feasible set K at the point $\mathbf{x}^*$.

The main advantage of the variational inequality formulation is that it allows for a unified treatment of equilibrium problems. Indeed, many mathematical problems, such as optimization or nonlinear complementary problems, can be formulated as variational inequality problems (see Nagurney (1999)) .

Theorem 7 (Equivalence between VIP and MP) *Let $\mathbf{x}^*$ be a solution to the variational inequality problem:*

$$\langle F(\mathbf{x}^*)^T, \mathbf{x} - \mathbf{x}^* \rangle, \quad \forall \mathbf{x} \in K, \tag{3.89}$$

where F is a continuously differentiable function whose Jacobian matrix $(\nabla F(\mathbf{x}))$ is symmetric and positive semi-definite and K is closed and convex. Then $\mathbf{x}^*$ is also a solution to the optimization problem:

$$\begin{aligned} &\textit{Minimize} \quad f(\mathbf{x}) \\ &\textit{subject to:} \quad \mathbf{x} \in K, \end{aligned} \tag{3.90}$$

where f is a convex function from K to $\mathbb{R}^1$ that satisfies $\nabla f(\mathbf{x}) = F(\mathbf{x})$.

The contrary (if $\mathbf{x}^*$ is a solution to the optimization problem (3.90), then $\mathbf{x}^*$ is a solution to $VI(F, K)$) also holds.

Proof. Firstly, we have to emphasize that, because of symmetry and positive semi-definiteness assumptions, it follows that $f(\mathbf{x}) = \int F(\mathbf{x})dx$. This proof has two parts (see Nagurney (1999)):

(i) If $\mathbf{x}^*$ is a solution to (3.90), then it is also a solution to (3.89). Let $\phi(t) = f(\mathbf{x}^* + t(\mathbf{x} - \mathbf{x}^*)), \text{for} \quad t \in [0, 1]$. Since $\phi(t)$ achieves its minimum at $t = 0$, $0 \leq \phi'(0) = \langle \nabla f(\mathbf{x}^*)^T, \mathbf{x} - \mathbf{x}^* \rangle = \langle F(\mathbf{x}^*)^T, \mathbf{x} - \mathbf{x}^* \rangle$, that is, $\mathbf{x}^*$ is a solution to (3.89).

(ii) If $\mathbf{x}^*$ is a solution to (3.89), then it is also a solution to (3.90). Since $f(\mathbf{x})$ is convex,

$$f(\mathbf{x}) \geq f(\mathbf{x}^*) + \langle \nabla f(\mathbf{x}^*)^T, \mathbf{x} - \mathbf{x}^* \rangle, \quad \forall \mathbf{x} \in K. \tag{3.91}$$

But $\langle \nabla f(\mathbf{x}^*)^T, \mathbf{x} - \mathbf{x}^* \rangle \geq 0$, since $\mathbf{x}^*$ is a solution to $VI(\nabla f, K) \equiv VI(F, K)$. Therefore, from (3.91) it follows that

$$f(\mathbf{x}) \geq f(\mathbf{x}^*), \quad \forall \mathbf{x} \in K,$$

that is, $\mathbf{x}^*$ is a minimum point of the mathematical problem (3.90).

■

Note that Theorem 7 implies that a variational inequality problem can be reformulated as a convex optimization problem, only when the symmetry condition and the positive semi-definiteness condition hold.

Historically, many equilibrium problems have been formulated as optimization problems under such a symmetry assumption (this is the case of the Beckmann et al. (1956) model presented in Section 3.2.1). The assumption, however, in terms of applications is restrictive and excludes the more realistic modeling of multiple modes or class users. Moreover, the objective function that results is often artificial, without a clear interpretation. These are some of the reasons why alternative approaches different from optimization problems have been proposed.

Next, we will see that the complementary problem shown in Section 3.2.2 is a special case of the variational inequality problem.

Theorem 8 *$VI(F, \mathbb{R}^n_+)$ and nonlinear problem (3.69)-(3.71) have precisely the same solution.*

Proof. Again, this demonstration has two parts (see Nagurney (1999)):

(i) If $\mathbf{x}^*$ satisfies $VI(F, \mathbb{R}^n_+)$, then it also satisfies the complementary problem (3.69)-(3.71). Substituting $\mathbf{x} = \mathbf{x}^* + e_i$ into $VI(F, \mathbb{R}^n_+)$, where e_i denotes the n-dimensional vector with 1 in the i-th position and 0, elsewhere, one concludes that $F(\mathbf{x}^*) \geq 0$.

Substituting now $\mathbf{x} = 2\mathbf{x}^*$ into the variational inequality, one obtains

$$\langle F\left(\mathbf{x}^*\right)^T, \mathbf{x}^* \rangle \geq 0. \tag{3.92}$$

Substituting then $\mathbf{x} = 0$ into the variational inequality, one obtains

$$\langle F\left(\mathbf{x}^*\right)^T, -\mathbf{x}^* \rangle \geq 0. \tag{3.93}$$

(3.92) and (3.93) together imply that $\langle F\left(\mathbf{x}^*\right)^T, \mathbf{x}^* \rangle = 0$.

(ii) Conversely, if $\mathbf{x}^*$ satisfies the complementary problem, then

$$\langle F\left(\mathbf{x}^*\right)^T, \mathbf{x} - \mathbf{x}^* \rangle \geq 0$$

since $\mathbf{x} \in \mathcal{R}^n_+$ and $F(\mathbf{x}^*) \geq 0$.

■

Theorem 8 states the equivalence between NCP and VIP in the case that $\mathbf{x}$ is positive. As in the traffic problems, we usually assume that the flows are positive, in practice, there would be no big differences between NCP and VIP in our case. For that reason, we will only give a general idea on VIP in this section and we will not develop all demonstrations. In the remaining of the section, we will give some results on existence and uniqueness of solutions of variational inequality problems and the equivalent UE problem in terms of VIP.

Theorem 9 *If K is a compact convex set and $F(\mathbf{x})$ is continuous on K, then the variational inequality problem admits at least one solution $\mathbf{x}^*$.*[11]

Proof. The demonstration of this theorem can be found in Nagurney (1999). ∎

Theorem 10 *Suppose that $F(\mathbf{x})$ is strictly monotone on K. Then the solution is unique, if one exists.*

Proof. Suppose that $\mathbf{x}^1$ and $\mathbf{x}^*$ are both solutions and $\mathbf{x}^1 \neq \mathbf{x}^*$. Then, since $\mathbf{x}^1$ and $\mathbf{x}^*$ are solutions, they must satisfy:

$$\langle F\left(\mathbf{x}^1\right)^T, \mathbf{x}' - \mathbf{x}^1\rangle \geq 0, \quad \forall \mathbf{x}' \in K \tag{3.94}$$

$$\langle F\left(\mathbf{x}^*\right)^T, \mathbf{x}' - \mathbf{x}^*\rangle \geq 0, \quad \forall \mathbf{x}' \in K. \tag{3.95}$$

Equation (3.95) is equivalent to $\langle -F\left(\mathbf{x}^*\right)^T, \mathbf{x}^* - \mathbf{x}'\rangle \geq 0$. Adding this last equation to (3.94), one obtains:

$$\langle \left(F\left(\mathbf{x}^1\right) - F\left(\mathbf{x}^*\right)\right)^T, \mathbf{x}^* - \mathbf{x}^1\rangle \geq 0. \tag{3.96}$$

But inequality (3.96) is in contradiction of strict monotonicity (Definition 4 (ii)). Hence, $\mathbf{x}^* = \mathbf{x}^1$. ∎

To end this section, we will show the equivalent UE variational inequality formulation.

Theorem 11 *The user equilibrium system (3.63)-(3.68) is equivalent to the following variational inequality problem:*

$$\langle c\left(\mathbf{f}^*\right), \mathbf{f} - \mathbf{f}^*\rangle, \quad \mathbf{f} \in K, \tag{3.97}$$

where $\mathbf{f}^*$ is the equilibrium flow pattern and $K = \{\mathbf{f} \geq 0 : \sum_{r \in \mathcal{R}} \xi_{ksr} f_{ksr} = \tau_{ks}\}$.

Proof. (see Ran and Boyce (1996)) Firstly, we have to remark that equation (3.97) is equivalent, by definition, to

$$\sum_{r \in \mathcal{R}} \sum_{ks \in \mathcal{OD}} c^*_{ksr}[f_{ksr} - f^*_{ksr}] \geq 0. \tag{3.98}$$

This proof has two parts: (i) any solution to (3.63)-(3.68) satisfies (3.98); and (ii) any solution to (3.98) also satisfies (3.63)-(3.68).

(i) For any route r, a feasible flow is

$$f_{ksr} \geq 0. \tag{3.99}$$

[11] Note that the conditions for the VIP to have a solution are the same as for the NCP.

Multiplying equilibrium condition (3.2)[12] by the above equation, we have

$$f_{ksr}[c^*_{ksr} - \pi^*_{ks}] \geq 0, \quad \forall r, k, s. \tag{3.100}$$

We subtract equation (3.1) from equation (3.100) and obtain

$$[f_{ksr} - f^*_{ksr}][c^*_{ksr} - \pi^*_{ks}] \geq 0, \quad \forall r, k, s. \tag{3.101}$$

Summing equation (3.101) for all routes r and all OD pairs ks, it follows that

$$\begin{aligned} &\sum_{r\in\mathcal{R}} \sum_{ks\in\mathcal{OD}} [f_{ksr} - f^*_{ksr}][c^*_{ksr} - \pi^*_{ks}] \\ &= \sum_{r\in\mathcal{R}} \sum_{ks\in\mathcal{OD}} [f_{ksr} - f^*_{ksr}]c^*_{ksr} - \sum_{ks\in\mathcal{OD}} \pi^*_{ks} \sum_{r\in\mathcal{R}} [f_{ksr} - f^*_{ksr}] \\ &= \sum_{r\in\mathcal{R}} \sum_{ks\in\mathcal{OD}} [f_{ksr} - f^*_{ksr}]c^*_{ksr} \geq 0, \end{aligned} \tag{3.102}$$

where the flow conservation equation

$$\sum_{r\in\mathcal{R}} f_{ksr} = \sum_{r\in\mathcal{R}} f^*_{ksr} = \tau_{ks}$$

holds for each OD pair. Note that equation (3.102) is the VIP equation proposed (3.98), which concludes this part of the proof.

(ii) We need to prove that any solution f^*_{ksr} to variational inequality (3.98) satisfies UE conditions (3.1)-(3.2). We know that (3.2) holds by definition, therefore, we need to prove that condition (3.1) also holds.

Assume that condition (3.1) does not hold only for a route p in OD pair zn, i.e.,

$$f^*_{znp} > 0 \quad \text{and} \quad c^*_{znp} - \pi^*_{zn} > 0.$$

Since condition (3.1) holds for all routes other than p, it follows that

$$\sum_{r\in\mathcal{R}} \sum_{ks\in\mathcal{OD}} f^*_{ksr}[c^*_{ksr} - \pi^*_{ks}] > 0. \tag{3.103}$$

On the contrary, for each OD pair KS, we can always find one minimal actual travel time route l, where route l was evaluated under the optimal flow pattern $\mathbf{f}^*$. For this route l, condition (3.2) becomes an equality by definition. It follows that:

$$c^*_{ksl} - \pi^*_{ks} = 0, \quad \forall l, k, s.$$

[12] In this demonstration the superscript $*$ has been added in equations (3.1)-(3.2) to denote the value of the variables when the equilibrium has been reached.

Next, we need to find a set of feasible route flows so that the following equation

$$f^*_{ksr}[c^*_{ksr} - \pi^*_{ks}] = 0, \quad \forall r, k, s \tag{3.104}$$

always holds. For each OD pair ks, we assign OD departure flow τ_{ks} to the minimal travel time route l, which was evaluated under the optimal flow pattern $\mathbf{f}^*$. This generates a set of feasible route flows $\mathbf{f}$ which always satisfy (3.104) because flows are not assigned to routes with non-minimal travel times. Summing equation (3.104) for all routes and OD pairs, it follows that

$$\sum_{r\in\mathcal{R}} \sum_{ks\in\mathcal{OD}} f_{ksr}[c^*_{ksr} - \pi^*_{ks}] = 0. \tag{3.105}$$

We subtract equation (3.103) from equation (3.105) and obtain

$$\begin{aligned} &\sum_{r\in\mathcal{R}} \sum_{ks\in\mathcal{OD}} [f_{ksr} - f^*_{ksr}][c^*_{ksr} - \pi^*_{ks}] \\ &= \sum_{r\in\mathcal{R}} \sum_{ks\in\mathcal{OD}} [f_{ksr} - f^*_{ksr}]c^*_{ksr} - \sum_{ks\in\mathcal{OD}} \pi^*_{ks} \sum_{r\in\mathcal{R}} [f_{ksr} - f^*_{ksr}] \\ &= \sum_{r\in\mathcal{R}} \sum_{ks\in\mathcal{OD}} [f_{ksr} - f^*_{ksr}]c^*_{ksr} < 0, \end{aligned} \tag{3.106}$$

where again the flow conservation equation

$$\sum_{r\in\mathcal{R}} f_{ksr} = \sum_{r\in\mathcal{R}} f^*_{ksr} = \tau_{ks}$$

holds for each OD pair. The above equation contradicts VIP (3.98), thus, any optimal solution $\mathbf{f}^*$ to variational inequality (3.98) satisfies condition (3.1).

■

Theorems 9 and 11 imply that VIP (3.97) has always a solution as long as the route cost function is continuous, as K is always compact and convex. Furthermore, VIP (3.97) has a unique solution if the route cost function is strictly monotone.

Finally, note that all the conditions imposed to the cost function in order to have uniqueness are the same (i.e. strictly monotone cost functions), independently of the approach used to model the UE traffic assignment problem, which is consistent with the fact that all approaches are equivalent.

In Section 3.2 different approaches to solve the User Equilibrium problem have been explain in order to show that different mathematical tools can be used to solve this problem. Figure 3.2 shows a scheme of the existing relations between the models presented and the conditions that have to hold for this equivalences. This wide variety of models have permitted modeling different versions of the classical UE problem, such as equilibrium between modes (see Aashtiani (1979)) or different class users (see Nie (2011)).

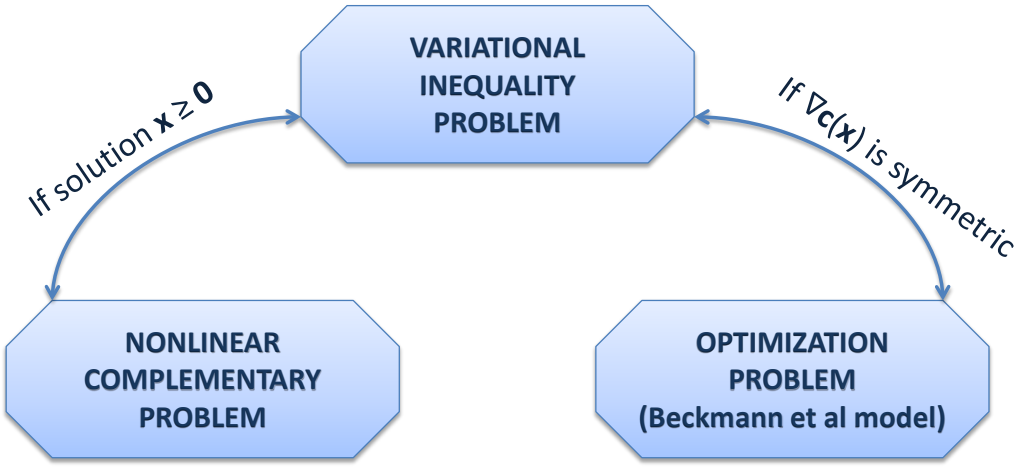


Figure 3.2: Equivalence relations of the models presented in Section 3.2.

3.3 System-Optimal (SO)

As mentioned in Section 3.1, the UE approach is not the only one to solve the static traffic assignment problem. In fact, the first Wardrop's principle does not lead to the best possible use of the network. It assumes that users behave individually in their own interest, but not necessarily in the interest of the system as a whole. For that reason, Wardrop (1952) proposed his second principle:

"The total (or average) travel time should be minimized",

that leads to the so called system optimal (SO) formulation, which can be formulated mathematically as a minimization problem of the total travel time spent by all users in the network:

$$\underset{\mathbf{f},\mathbf{w}}{\text{Minimize}}\, Z(\mathbf{w}) = \sum_{\ell_{ij}\in A} w_{ij} c_{ij}\left(w_{ij}\right) \tag{3.107}$$

subject to

$$\sum_{r\in\mathcal{R}} \xi_{ksr} f_r = \tau_{ks} : \zeta_{ks}, \quad \forall k,s \in \mathcal{OD} \tag{3.108}$$

$$\sum_{r\in\mathcal{R}} f_r \delta_{ijr} = w_{ij} : \psi_{ij}, \quad \forall \ell_{ij} \in \mathcal{A} \tag{3.109}$$

$$f_r \geq 0 : \mu_r \quad \forall r \in \mathcal{R}. \tag{3.110}$$

Under the system-optimal, some travels may be assigned to routes that have travel costs higher than the minimal that they could achieve by deciding by themselves

independently. For that reason, the flow pattern that minimizes the total travel time does not generally represent an equilibrium situation (i.e., users may be able to decrease their travel time by uniterally changing routes). Such situation is unlikely to sustain itself and consequently the SO flow pattern is not stable and should not be used as a model of actual behavior and equilibrium. Nevertheless, the SO model is frequently used as a bound in many mathematical programs dealing with network design.

The dual variables. As in the UE case, the dual variables of the SO problem (3.107)-(3.110) have a physical meaning which will be explained next. The Lagrangian function of the problem (3.107)-(3.110) is:

$$\begin{aligned}\mathcal{L}(\mathbf{w},\mathbf{f},\boldsymbol{\zeta},\boldsymbol{\psi},\boldsymbol{\mu}) = \sum_{\ell_{ij}\in A} w_{ij}c_{ij}\left(w_{ij}\right) + \sum_{ks\in\mathcal{OD}} \zeta_{ks}\left(\tau_{ks} - \sum_{r\in\mathcal{R}} \xi_{ksr}f_r\right) \\ + \sum_{\ell_{ij}\in\mathcal{A}} \psi_{ij}\left(\sum_{r\in\mathcal{R}} f_r\delta_{ijr} - w_{ij}\right) - \sum_{r\in\mathcal{R}} \mu_r f_r,\end{aligned} \tag{3.111}$$

and the associated Karush-Kuhn-Tucker conditions are:

$$\frac{\partial\mathcal{L}}{\partial w_{ij}} = c_{ij}\left(w_{ij}\right) + w_{ij}c'_{ij}\left(w_{ij}\right) - \psi_{ij} = 0; \quad \forall\ell_{ij}\in\mathcal{A} \tag{3.112}$$

$$\frac{\partial\mathcal{L}}{\partial f_r} = -\zeta_{ks}\xi_{ksr} + \sum_{\ell_{ij}\in\mathcal{A}} \psi_{ij}\delta_{ijr} - \mu_r = 0;\ \forall r\in\mathcal{R} \tag{3.113}$$

$$\sum_{r\in\mathcal{R}} \xi_{ksr}f_r = \tau_{ks}; \quad \forall ks\in\mathcal{OD} \tag{3.114}$$

$$\sum_{r\in\mathcal{R}} f_r\delta_{ijr} = w_{ij}; \quad \forall\ell_{ij}\in\mathcal{A} \tag{3.115}$$

$$f_r \geq 0; \quad \forall r\in\mathcal{R} \tag{3.116}$$

$$f_r\mu_r = 0; \quad \forall r\in\mathcal{R} \tag{3.117}$$

$$\mu_r \geq 0; \quad \forall r\in\mathcal{R}, \tag{3.118}$$

where $c'_{ij}\left(w_{ij}\right)$ is the first derivative of the link cost function $c_{ij}\left(w_{ij}\right)$. From the previous KKT conditiones, we can draw the following conclusions:

1. The dual variable ψ_{ij} can be interpreted as the marginal contribution of an infinitesimal additional traveler on link ℓ_{ij} to the total travel time on this link (equation (3.112)).

2. If the flow on route r is not null ($f_r > 0$), then (because of equation (3.117)) the dual variable μ_r equals zero. In that case, from equation (3.113) we get that $\zeta_{ks}\xi_{ksr} = \sum_{\ell_{ij}\in\mathcal{A}} \psi_{ij}\delta_{ijr}$, in other words, ζ_{ks} can be interpreted as the route marginal contribution to total travel time in the network of an additional traveler.

3. All the used routes of the same OD have the same marginal contribution to the total cost of the system of a new user. This contribution is larger for non-used routes (see equations (3.113) and (3.118)).

4. The dual variable μ_r can be interpreted as the over-cost on the minimum marginal contribution of route r For that reason, for used routes μ_r equals zero, meaning that the marginal contribution is minimal for that route.

Finally, we will see under what circumstances the User Equilibrium approach and the System Optimal one coincide.

Theorem 12 *Wardrop's first and second principle lead to the same solution for uncongested networks (see Sheffi (1985)).*

Proof. We will next prove that for uncongested networks, i.e. when congestion effects are ignored, both UE and SO programs ((3.3)-(3.6) and (3.107)-(3.110), respectively) will produce identical results. First, we remark that for uncongested networks the link travel times are not a function of the flow on that (or any other) link, i.e., $c_{ij}(w_{ij}) = c_{ij}$. Then the objective function (3.3) can be rewritten as:

$$\sum_{\ell_{ij}\in\mathcal{A}} \int_0^{w_{ij}} c_{ij} ds = \sum_{\ell_{ij}\in\mathcal{A}} c_{ij} \int_0^{w_{ij}} ds = \sum_{\ell_{ij}\in A} w_{ij} c_{ij}, \tag{3.119}$$

which is identical to the SO objective function (3.107) when $c_{ij}(w_{ij}) = c_{ij}$. Since the constraints are common for both problems, they coincide and, hence, they share the same optimal solution. ∎

Finally, we note that the SO problem can be stated without path enumeration using disaggregated link flow variables, as it is done in the Castillo et al. model in Section 3.2.1.

Example 1 (UE and SO models) *With the purpose of illustrating the proposed concepts and methods, they are applied to the well known Nguyen-Dupuis network (see Nguyen and Dupuis (1984)). We use an example of bidirectional flows. It consists of* 13 *nodes and* 38 *links and we assume the existence of symmetric links, i.e. any pair of nodes i and j are connected in both directions by links ℓ_{ij} and ℓ_{ji} (see Figure 3.3). In this example we use the BPR (Bureau of Public Roads (1964)) cost function (see the Appendix for a review of cost functions):*

$$t_{ij} = t_{0ij}\left[1 + \beta_{ij}\left(\frac{w_{ij}}{q_{ij}}\right)^{\gamma_{ij}}\right],$$

where for a given link ℓ_{ij}, t_{ij} refers to its travel time, t_{0ij} is the travel time associated with free flow conditions, w_{ij} is the flow on that link, q_{ij} is a constant measuring the flow producing congestion, and β_{ij} and γ_{ij} are constants defining how the cost increases with traffic flow.

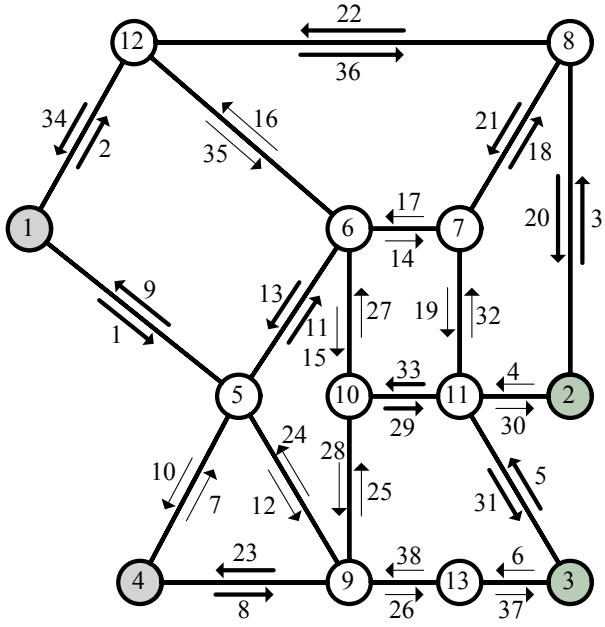


Figure 3.3: The Nguyen-Dupuis network.

The network data used in this example are $\beta_{ij} = 1$, $\gamma_{ij} = 3$, $\forall \ell_{ij}$ *and the* t_{0ij} *and* q_{ij} *travel cost constants used for every link* ℓ_{ij} *are in Table 3.1. In this section, we illustrate the traffic assignment problem, i.e., the OD-pair flows are given, and the flows* f_r *are to be determined. In particular, we solve the traffic assignment problem from the user equilibrium and the system optimum perspectives. We consider the following OD-pair flows:*

$$\tau_{12} = 350 \quad \tau_{13} = 448 \quad \tau_{42} = 336 \quad \tau_{43} = 210$$

The UE problem is solved using the different models explained previously obtaining the same results, as it was expected given that they are all equivalent. First, it is solved without path enumeration using the Castillo et al. model so the utilized paths can be obtained. Afterwards, some paths are added to the set of paths and the problem is solved using the remaining models. The resulting link flows disaggregated by OD are shown in the even columns of Table 3.3.

Furthermore, the SO solution is also computed and its link flows disaggregated by OD are shown in the remaining columns of Table 3.3. Note that the results of both perspectives are different. Some links are used in a solution but not in the other. It is also remarkable that the total travel time (defined as the sum of the travel times of all users) associated with the UE and SO models is 61238.034 *and* 59178.625*, respectively. As expected, the total travel time is lower under the SO, but it leads to an overcost for some users that is incompatible with the equilibrium idea, as it has been mentioned before. Finally, the*

Link	t_{0ij}	q_{ij}
7 - 11 and 11- 7	9	700
8 - 2 and 2 - 8	9	700
9 - 10 and 10- 9	10	280
9 - 13 and 13- 9	9	280
10- 11 and 11- 10	6	700
11- 2 and 2 - 11	9	280
11- 3 and 3 - 11	8	560
12- 6 and 6 - 12	7	140
12- 8 and 8 - 12	14	560
13- 3 and 3 - 13	11	560

Link	t_{0ij}	q_{ij}
1 - 5 and 5 - 1	7	700
1 - 12 and 12- 1	9	560
4 - 5 and 5 - 4	9	560
4 - 9 and 9 - 4	12	280
5 - 6 and 6 - 5	3	420
5 - 9 and 9 - 5	9	420
6 - 7 and 7 - 6	5	700
6 - 10 and 10- 6	5	280
7 - 8 and 8 - 7	5	700

Table 3.1: Network parameters for the Nguyen-Dupuis network.

used routes can be observed in Figure 3.4. The upper figure corresponds with UE problem, whereas the lower correspond to the SO. Again, note that the used routes do not coincide for both types of solution. Finally, the costs associated with the used paths are shown in Table 3.2. We emphasize again the fact that the SO model implies some costly paths to be used. It is the case of paths $6, 7, 8$ *and* 12*. In a real situation, users in more expensive paths will tend to change their path to get lower travel times and, hence, in the long term, an equilibrium situation may be reached.*

3.4 User Equilibrium with heterogeneous users

The problem of user interaction in traffic networks where travelers compete for space (traffic assignment problem) has been dealt with in the existing literature for several decades, as it has been shown in the previous section. The most common approach assumes that users behave in a homogeneous way in the sense that all of them behave in the same form or assume a mean behavior (speed, travel time, etc.). However, recently, some heterogeneous cases have arisen as is the case of the *travel time reliability problem* (see Asakura and Kashiwadani (1991), Lo and Tung (2003), Lo et al. (2006) or Nie and Wu (2009)), in which different users perceive the problem from a different perspective. This occurs when travelers are concerned about reaching the destination on time because of possible consequences in terms of prestige, money losses, etc. but the perception of these consequences is not the same for all of them, so that they can be grouped in different classes. Consequently, users in different classes choose different routes based on different criteria and we face *heterogeneity*. Apart from the travel time reliability problem, in its multiple versions, traffic problems with heterogeneous users are not very frequent in the existing literature. This section is devoted to the *travel time reliability problem* showing some of the models proposed in the literature.

OD	Routes	path links	UE	SO
1 - 2	1	1 5 7 9 11	44.338	**39.894**
1 - 2	2	2 18 11	**43.414**	**42.144**
1 - 3	3	1 5 7 10 16	**45.539**	**42.766**
1 - 3	4	1 5 8 14 16	**45.539**	**40.516**
1 - 3	5	2 17 8 14 16	**45.539**	**45.016**
1 - 3	6	1 6 12 14 16	**45.539**	**48.766**
1 - 3	7	1 6 13 19	**45.539**	**45.766**
1 - 3	8	2 17 7 10 16	**45.539**	**47.266**
4 - 2	9	3 5 7 9 11	**46.501**	**42.753**
4 - 2	10	3 5 7 10 15	**46.501**	**45.753**
4 - 2	11	3 6 12 14 15	46.501	**51.753**
4 - 2	12	4 12 14 15	**46.501**	**47.253**
4 - 3	13	3 6 13 19	47.702	**48.625**
4 - 3	14	3 5 8 14 16	**47.702**	**43.375**
4 - 3	15	4 13 19	**47.702**	**44.125**

Table 3.2: Used routes for the UE and SO model classified by OD for the Nguyen-Dupuis network. Used routes are boldfaced.

The travel time reliability problem was introduced in order to overcome one of the major limitations of the classic traffic assignment: it ignores the uncertainties that take place in transportation systems. Transportation systems are affected by uncertainties of various sorts, which can de broadly classified as those affecting the supply of transportation (e.g. weather, incidents, works) and those associated with the demand for transportation (e.g. travel and activity behavior, special events). The first type of uncertainties is associated with the impossibility to predict the real capacity of the link, whereas the last is associated with the variability in the number of users of the network. Generally, travel time reliability models are concerned with the first type of uncertainties and, hence, the capacity of the arcs in the network are considered as random.

These uncertainties lead to unpredictable traffic conditions on the available paths to reach the desired destinations and, therefore, users are not able to know in advance the travel time of their trips leading to undesired delays or early arrivals. In that scenario and in order to avoid delays and early arrivals, users are concerned about the reliability of the different paths, in the sense that they would prefer paths that reduce the risk of arriving late, rather than to minimize the expected travel time. Empirical studies (Jackson and Jucker (1981), Hall (1983) and Abdel-Aty et al. (1995)) have shown that travelers'

	OD:1 - 2		OD:1 - 3		OD:4 - 2		OD:4 - 3	
Link	UE	SO	UE	SO	UE	SO	UE	SO
1 -5	-	14.09	398.64	369.74	-	-	-	-
1 -12	350.00	335.91	49.36	78.26	-	-	-	-
4 -5	-	-	-	-	260.32	296.55	44.81	43.34
4 -9	-	-	-	-	75.68	39.45	165.19	166.66
5 -6	-	14.09	283.96	282.79	260.32	178.57	44.81	29.39
5 -9	-	-	114.68	86.94	-	117.98	-	13.95
6 -7	-	14.09	133.47	212.82	260.32	178.57	-	-
6 -10	-	-	199.85	148.24	-	-	44.81	29.39
7 -8	-	14.09	-	-	214.98	160.04	-	-
7 -11	-	-	133.47	212.82	45.35	18.53	-	-
8 -2	350.00	350.00	-	-	214.98	160.04	-	-
9 -10	-	-	22.45	18.08	75.68	157.43	-	-
9 -13	-	-	92.24	68.87	-	-	165.19	180.61
10-11	-	-	222.30	166.32	75.68	157.43	44.81	29.39
11-2	-	-	-	-	121.02	175.96	-	-
11-3	-	-	355.76	379.13	-	-	44.81	29.39
12-6	-	-	49.36	78.26	-	-	-	-
12-8	350.00	335.91	-	-	-	-	-	-
13-3	-	-	92.24	68.87	-	-	165.19	180.61

Table 3.3: UE and SO models. Link flows disaggregated by OD for the Nguyen-Dupuis network.

decisions are known to be largely influenced by travel time variability and reliability, which have been recognized by users as two of the main criteria for route choice. Specifically, Abdel-Aty et al. (1995) noted that about 54% of the responders of the survey indicated that travel time reliability was either the most important or second most important reason for choosing their paths. Nevertheless, there exist different types of users depending on their attitude towards risk and the trip purpose. Risk-averse users (or users whose trip purpose is important, e.g. interviews or exams) will tend to choose very reliable paths even if that means choosing slower paths, whereas risk-prone users will tend to choose less reliable paths that will provide them the chance of arriving before[13].

The models defined in the area of travel time reliability postulate various user behav-

[13]Note that in the classic traffic assignment problem, all users are supposed to be risk-neutral.

iors when facing uncertainties and propose different models that take into account that behavior. For example, Lo and Tung (2003) postulate that drivers would select routes to lower their travel time variabilities, just as they would to lower their mean travel times. Meanwhile, Watling (2006) postulates that travelers are more likely to make the routing decisions based on a latest acceptable arrival time.

The Lo and Tung model

As already mentioned, Lo and Tung (2003) postulate that drivers would select routes to lower their travel time variabilities, just as they would to lower their mean travel times. Overtime, users learn the routes' travel time variabilities based on past experiences, factor such variabilities into their route considerations, and settle into a long-term equilibrium pattern, named by the author as Probabilistic User Equilibrium (PUE).

Assuming that the capacities Q_{ij} are independent random variables (and hence the link and path travel times in the network are random variables too), they define the PUE solution as the vector of flows $\mathbf{f} = \{\mathbf{f_r}\}_{\mathbf{r}\in\mathcal{R}}$ such that:

(i) The flow f_r on route r is positive if its mean travel time is equal to the minimum mean OD ks travel time π_{ks}, defined as the minimum of the mean travel times of all paths joining OD ks. Furthermore, all unused routes have equal or higher mean travel times. Mathematically:

$$f_r[E(T_r) - \sum_{ks\in\mathcal{OD}} \xi_{ksr}\pi_{ks}] = 0, \quad \forall r \in \mathcal{R} \tag{3.120}$$

$$E(T_r) \geq \sum_{ks\in\mathcal{OD}} \xi_{ksr}\pi_{ks}, \quad \forall r \in \mathcal{R}, \tag{3.121}$$

where T_r is the travel time random variable of route r and $E(T_r)$ is its mean.

(ii) The travel time of a used route satisfies the following reliability condition:

$$P(-\epsilon \sum_{ks\in\mathcal{OD}} \xi_{ksr}\pi_{ks} \leq T_r - \sum_{ks\in\mathcal{OD}} \xi_{ksr}\pi_{ks} \leq \epsilon \sum_{ks\in\mathcal{OD}} \xi_{ksr}\pi_{ks}) \geq \rho, \quad \forall r \in \mathcal{R}, \tag{3.122}$$

where ϵ is the fractional derivation from π_{ks} and ρ is the OD travel time reliability measure. ϵ and $0 \leq \rho \leq 1$ are performance related positive parameters.

The second condition imposes dispersion to be small so, routes with mean travel time equal to π_{ks} will not be used if their variabilities are higher than specified in part (ii). Likewise, routes with small variabilities will remain unused if their mean travel times are larger than specified in (i). If one desires the travel time of a network to be very reliable, then a small ϵ and a large ρ should be defined. For example, let $\epsilon = 1\%$ and $\rho = 99\%$ then, we will require that the travel times of the used routes have a probability of at least

99% of lying within a 1% deviation from the minimum mean OD travel time. Operating and simplifying equation (3.122), the following equivalent condition is obtained:

$$cov(T_r) \leq \lambda, \tag{3.123}$$

where $cov(T_r)$ is the coefficient of variation of the route travel time random variable and λ is a performance parameter that can be established in terms of ϵ and ρ. Indeed, a small value of λ correspond to a small ϵ or a large ρ and hence a tighter route travel time distribution. So, a vector of route flows $\mathbf{f}$ is a PUE solution if it is a solution to the following system of equations:

$$\begin{aligned}
\sum_{r\in\mathcal{R}} \xi_{ksr} f_r &= \tau_{ks}, \quad \forall ks \in \mathcal{OD}, \\
\sum_{r\in\mathcal{R}} f_r \delta_{ijr} &= w_{ij}; \quad \forall \ell_{ij} \in \mathcal{A}, \\
f_r[E(T_r) - \sum_{ks\in\mathcal{OD}} \xi_{ksr}\pi_{ks}] &= 0, \quad \forall r \in \mathcal{R}, \\
E(T_r) &\geq \sum_{ks\in\mathcal{OD}} \xi_{ksr}\pi_{ks}, \quad \forall r \in \mathcal{R}, \\
f_r(cov(T_r) - \lambda) &\leq 0, \quad \forall r \in \mathcal{R}, \\
f_r &\geq 0, \quad \forall r \in \mathcal{R}.
\end{aligned}$$

Assuming that the capacities Q_{ij} follow a uniform distribution and making use of the Central Limit Theorem (CLT), the authors determine that the route travel times belong to the normal family, compute the expressions for the mean travel times and coefficients of variation and apply the model to the Nguyen-Dupuis network.

This model does not reproduce heterogeneity among users the way it was defined by the authors but establishes the basic ideas of the travel time reliability problem. Nevertheless, the model can be generalized to the case of heterogeneous users by introducing different values of λ for the user classes based on their risk aversion. In that case, each user class would choose its own routes taking into account the route variabilities, i.e. one route might be too risky for one user class but be suitable for another leading to different configuration of paths for the different risk classes.

The Lo et al. model

Following the ideas of Lo and Tung (2003), Lo et al. (2006) extend the concept of PUE by defining the term *travel time budget*. As the transportation system is subject to relative minor events of stochastic link capacity variations, the link travel times are not deterministic anymore and users cannot know a priori their trip travel times. In that case, Lo et al. (2006) postulate that, to hedge against uncertainty, users generally add a travel time margin to the expected trip time to form their *travel time budget*. In other words,

the *travel time budget* is the travel time allocated to travel between an origin-destination pair. The travel time budget is a function of several factors: (a) the probability required to reach destination on time (which depends on the risk aversion of the users and the trip purpose), referred to as within budget time reliability (WBTR) and (b) the variability of the paths in the network (very unreliable paths will have large associated travel time budgets). Anyway, regardless of their required WBTR, all travelers want to reduce their travel time budgets.

Again, the authors assume independent and random link capacities Q_{ij} and develop the formulas for the travel time budget in the case of BPR performance function and uniform capacities. Making use of the CLT, Lo et al. (2006) conclude that the route travel times belong to normal distributions with parameters that depend on link flows and that degenerate properly into the typical deterministic form when the link capacity is assumed deterministic.

As it has been said, the key part of this model is the travel time budget, that the authors define as

$$[\text{Travel Time Budget}] = [\text{Expected Travel Time}] + [\text{Travel Time Margin}]. \tag{3.124}$$

Mathematically, the travel time budget associated with path r, b_r, can de expressed as

$$b_r = E(T_r) + \lambda \sigma_{T_r}, \quad \forall r \in \mathcal{R}, \tag{3.125}$$

where λ is a parameter and $E(T_r)$ and σ_{T_r} are the mean and standard deviation of the random variable of the travel time on route r, T_r, respectively. The parameter λ is related to the requirement on punctuality. For trips that have a high penalty for lateness, λ is expected to be large as the margin is defined to cover possible delays. Formally, the value of λ can be related mathematically with the on-time probability as

$$P\{T_r \leq b_r\} = P\{T_r \leq E(T_r) + \lambda \sigma_{T_r}\} = \rho, \tag{3.126}$$

where ρ is the probability that the travel time is within the travel time budget and is referred to as the within budget time reliability (WBTR). Under normality assumptions, as is the case of the paper, λ is given by

$$\lambda = \Phi(\rho), \tag{3.127}$$

where Φ is the CDF of the standard normal distribution. As stated before, for a large WBTR, a large value of λ is obtained and, hence, a larger travel time budget. It is also assumed that there are different types of users, each one with a different WBTR, and, hence, a different value of λ.

Lo et al. (2006) also postulate that all the users seek to minimize their travel time budget and, therefore, a long-term equilibrium will be reached where all the used routes for a class of users have equal and minimal associated travel time budget, whereas unused

routes have equal or higher travel time budget. Note that PUE is an extension to classical UE when stochastic fluctuations are considered.

Let $\mathcal{H}$ be the set of user classes and let α label their different degree of risk aversion. The model can be defined via a gap function as discussed in Lo and Chen (2000) as:

$$\underset{\mathbf{f},\mathbf{w},\boldsymbol{\pi}}{\text{Minimize}}\, g = \sum_{\alpha\in\mathcal{H}}\sum_{r\in\mathcal{R}} f_r^\alpha \left(b_r^\alpha - \sum_{ks\in\mathcal{OD}} \xi_{ksr}\pi_{ks}^\alpha \right) \tag{3.128}$$

subject to

$$\sum_{r\in\mathcal{R}} \xi_{ksr} f_r^\alpha = \tau_{ks}^\alpha, \quad \forall k,s \in \mathcal{OD},\ \forall \alpha \in \mathcal{H}, \tag{3.129}$$

$$\sum_{\alpha\in\mathcal{H}}\sum_{r\in\mathcal{R}} f_r^\alpha \delta_{ijr} = w_{ij}, \quad \forall \ell_{ij} \in \mathcal{A}, \tag{3.130}$$

$$b_r^\alpha - \sum_{ks\in\mathcal{OD}} \xi_{ksr}\pi_{ks}^\alpha \geq 0, \quad \forall r \in \mathcal{R},\ \forall \alpha \in \mathcal{H}, \tag{3.131}$$

$$f_r^\alpha \geq 0, \quad \forall r \in \mathcal{R},\ \forall \alpha \in \mathcal{H}, \tag{3.132}$$

where b_r^α is the travel time budget of a user of class α, f_r^α is the flow on route r of users of class α and τ_{ks}^α is the flow of users of class α that travel between node k and s.

The solution of this nonlinear mathematical program (3.128)-(3.132) provides the route choice pattern and the corresponding travel time budgets that satisfy the equilibrium complementary conditions and the within budget time reliability requirements. As it has been shown in Section 3.2, the objective function (3.128) is smooth and convex. Therefore, the mathematical program (3.128)-(3.132) can be solved easily using software packages. Furthermore, in the case of deterministic link capacities, this formulation will fall back nicely into the classical UE.

Finally, Lo et al. (2006) state a very important property of the previous formulation.

Proposition 3 *In the mathematical program (3.128)-(3.132) with multi-class travelers, if a set of routes (with at least two routes) is used by more than one user class, then all the routes in the set have the same mean travel time and standard deviation.*

Proof. Consider that two different classes with associated parameters λ_1 and λ_2 use routes 1 and 2 of the same OD pair. At equilibrium, the travel time budget of each class must be the same for both routes:

$$E(T_1) + \lambda_1\sigma_{T_1} = E(T_2) + \lambda_1\sigma_{T_2} \tag{3.133}$$

$$E(T_1) + \lambda_2\sigma_{T_1} = E(T_2) + \lambda_2\sigma_{T_2}. \tag{3.134}$$

Subtracting (3.134) from (3.133) we get:

$$(\lambda_1 - \lambda_2)\sigma_{T_1} = (\lambda_1 - \lambda_2)\sigma_{T_2} \Longrightarrow \sigma_{T_1} = \sigma_{T_2}. \tag{3.135}$$

Substituting (3.135) into (3.133), we obtain:

$$E(T_1) + \lambda_1 \sigma_{T_1} = E(T_2) + \lambda_1 \sigma_{T_1} \Longrightarrow E(T_1) = E(T_2). \tag{3.136}$$

■

Corollary 1 *Two or more different user classes do not share more than one route unless the shared routes have the same mean travel time and standard deviation.*

As the WBTR is defined in terms of path travel times[14], a path-based approach is required. As the authors mention, for large networks pre-enumerating all the paths can be computationally exhausted, so some simplifications should be done. For instance, they suggest to use a priori path set through interviews, as routes actually in use has been shown to be limited. This has the additional benefit of producing more real results as they will be based on users actual preferences.

The Watling model

A different point of view of how users choose their paths when facing variability is proposed by Watling (2006). In this case, an extension of the classical user equilibrium approach, termed as Late Arrival Penalized UE (LAPUE), is developed. Instead of assuming a disutility function depending only on the expected travel time, a penalty term based on preferred arrival time is added.

Whereas in the previous model, the capacity of the links are assumed to be random variables, Watling (2006) assumes that the link travel times are the source of variability. Therefore, the arc travel times T_{ij} are represented as random variables and it is supposed that the joint density of arc travel times $\mathbf{T} = \{T_{ij}\}_{\ell_{ij} \in \mathcal{A}}$ has a known distributional form[15] which is parameterized by a function ϱ of the arc flow vector $\mathbf{w}$ that also gives the mean link travel times. Furthermore, the path travel times T_r are also random variables given by

$$T_r = \sum_{\ell_{ij} \in \mathcal{A}} \delta_{ijr} T_{ij}, \quad \forall r \in \mathcal{R},$$

where δ_{ijr} is the link-path incidence matrix.

The key difference of this model is the introduction of a latest acceptable arrival time κ_{ks} associated with each OD pair that is defined as the longest possible travel time for a journey which, if exceeded, would incur some inconvenience. This new concept permits defining a path disutility function u_r as:

$$u_r = \theta_0 d_r + \theta_1 E[T_r] + \theta_2 E[\max(0, T_r - \kappa_{ks})], \quad \forall r \in \mathcal{R}, \tag{3.137}$$

[14]Note that the definition of travel time budget involves the use of path standard deviation which can not be computed as the sum of the link standard deviations.

[15]It should be emphasized that this assumption permits including a dependency structure among the links by choosing an appropriate joint distribution.

where d_r represents the attributes that are independent of time/flow, such as tolls or distance, θ_0 is the value placed on this attributes, θ_1 is the value of time, and θ_2 reflects the value of being one time unit latter than acceptable. If ϑ_r denotes the marginal density function of T_r, then (3.137) can be written

$$u_r = \theta_0 d_r + \theta_1 E[T_r] + \theta_2 \int_{\kappa_r}^{\infty} (t - \kappa_{ks}) \vartheta_r(t) dt, \quad \forall r \in \mathcal{R}. \tag{3.138}$$

It should be noted that this disutility function permits reflecting:

1. The user's valuation of path's attributes (distance, expected travel time, tolls, etc.).
2. The extent to which the path is likely to satisfy a traveler in achieving an acceptable arrival time at destination.

Making use of the expected disutility functions $u_r(\boldsymbol{f})$ in (3.138), Watling (2006) defines the Late Arrival Penalized User Equilibrium (LAPUE) using a variational inequality problem formulation and proves the existence and uniqueness of solutions.

Definition 6 (LAPUE) *A path flow vector $\boldsymbol{f}^*$ is termed a Late Arrival Penalized User Equilibrium, if $\boldsymbol{f}^*$ is a Wardrop equilibrium based on path cost functions $u_r(\boldsymbol{f})$. Mathematically, if $\boldsymbol{u}(\boldsymbol{f})$ denotes the vector with elements $u_r(\boldsymbol{f})$, then $\boldsymbol{f}^*$ is a LAPUE if and only if:*

$$\boldsymbol{u}(\boldsymbol{f}^*)'(\boldsymbol{f} - \boldsymbol{f}^*) \geq 0, \quad \forall \boldsymbol{f} \in D, \tag{3.139}$$

where $D = \left\{ \boldsymbol{f} : \sum_{r \in \mathcal{R}} \xi_{ksr} f_r = \tau_{ks},\ \forall k, s \in \mathcal{OD} \text{ and } f_r \geq 0\ \forall r \in \mathcal{R} \right\}$.

Proposition 4 *Suppose that the functions $F_r = \int_{\kappa_r}^{\infty} (t - \kappa_{ks}) \vartheta_r(t) dt$ exist and are continuous and that $\varrho(\mathbf{w})$ is a continuous mapping. Then LAPUE solutions exist.*

Proof. The proof to this proposition is based on the proof of Smith (1979) and can be found in Watling (2006). ∎

Proposition 5 *Suppose that the conditions of existence hold and that in (3.138), $\theta_1 > 0$ and $\theta_2 \geq 0$. Suppose further that the arc travel time functions ϱ are strictly monotone (as defined in Definition 4 (ii)). Finally, suppose that the functions F_r are non-decreasing. Then, there is a unique induced LAPUE arc flow solution.*

Proof. Consider any two LAPUE solutions $\boldsymbol{f}$ and $\boldsymbol{g}$, with distinct induced arc flows, i.e. $\Delta \boldsymbol{f} \neq \Delta \boldsymbol{g}$. Consider the function $\Delta' \boldsymbol{\varrho}(\Delta \mathbf{f})$ that gives the mean path travel times from the path flows. Then, for such $\boldsymbol{f}$ and $\boldsymbol{g}$,

$$\left(\Delta' \boldsymbol{\varrho}(\Delta \boldsymbol{f}) - \Delta' \boldsymbol{\varrho}(\Delta \boldsymbol{g})\right)' (\boldsymbol{f} - \boldsymbol{g}) = (\boldsymbol{\varrho}(\Delta \boldsymbol{f}) - \boldsymbol{\varrho}(\Delta \boldsymbol{g}))' (\Delta \boldsymbol{f} - \Delta \boldsymbol{g}) > 0,$$

since ϱ is strictly monotone and by hypothesis $\Delta \boldsymbol{f} \neq \Delta \boldsymbol{g}$. Since the mean path travel times are, then, strictly monotone and, by assumption, the functions F_r are non-decreasing, and $\theta_1 > 0$ and $\theta_2 \geq 0$, then the path disutility functions $\mathbf{u}(\mathbf{f})$ also satisfies strict monotonicity:

$$(\mathbf{u}(\mathbf{f}) - \mathbf{u}(\mathbf{g}))'(\mathbf{f} - \mathbf{g}) > \mathbf{0}, \quad \forall \mathbf{f}, \mathbf{g} \in \mathbf{D}. \tag{3.140}$$

On the other hand, consider the following expression:

$$\mathbf{u}(\mathbf{f})'(\mathbf{f} - \mathbf{g}) = \mathbf{u}(\mathbf{g})'(\mathbf{f} - \mathbf{g}) + (\mathbf{u}(\mathbf{f}) - \mathbf{u}(\mathbf{g}))'(\mathbf{f} - \mathbf{g}) > \mathbf{0}, \tag{3.141}$$

where the > 0 condition holds because the first term is non-negative since $\mathbf{g}$ is a LAPUE solution and the second term is positive by (3.141). But since $\mathbf{f}$ is also a LAPUE solution, $\mathbf{u}(\mathbf{f})'(\mathbf{g} - \mathbf{f}) \geq \mathbf{0}$, i.e. $\mathbf{u}(\mathbf{f})'(\mathbf{f} - \mathbf{g}) \leq \mathbf{0}$, implying that the original hypothesis, of two LAPUE solutions must be false. ∎

Once the LAPUE model has been defined, and its existence and uniqueness of solutions discussed, the next question is how to compute this kind of solutions. In general, the expected disutility function (3.138) is not expressible as a sum of arc disutilities and, as a consequence, standard shortest path methods cannot be applied and storage of path flows cannot be avoided. Following the ideas of Lo and Chen (2000), Watling (2006) propose to solve the LAPUE problem via a smooth gap function, which allows using standard algorithms for unconstrained optimization.

Let $\boldsymbol{x}$ and $\boldsymbol{F}(\boldsymbol{x})$ be two $|\mathcal{R}| + |\mathcal{OD}|$ vectors such that:

$$\boldsymbol{x} = \begin{pmatrix} \boldsymbol{f} \\ \boldsymbol{y} \end{pmatrix}, \quad \boldsymbol{F}(\boldsymbol{x}) = \begin{pmatrix} \boldsymbol{F}^f(\boldsymbol{x}) \\ \boldsymbol{F}^y(\boldsymbol{x}) \end{pmatrix},$$

where $\boldsymbol{y}$ is the vector of minimum OD disutilities, $F^f(\cdot)$ is the vector of elements $u_r(\boldsymbol{f}) - y_{ks}$, and $F^y(\cdot)$ is the vector of elements $\sum_{r \in \mathcal{R}} \xi_{ksr} f_r - \tau_{ks}$.

LAPUE solutions may then be determined by minimizing the gap function:

$$G(\boldsymbol{x}) = \sum_{i=1}^{|\mathcal{R}|+|\mathcal{OD}|} \iota(x_i, F_i(\boldsymbol{x})), \quad \text{where } \iota(a,b) = \frac{1}{2}\left(\sqrt{a^2 + b^2} - (a+b)\right)^2. \tag{3.142}$$

The author also develops the formulas for the cases of multivariate normal distribution and mixture of normal distributions as the joint density of arc travel times. The advantages and disadvantages of these distributions are discussed and some numerical results are shown. It is noted that both distributions lead to unique solutions as they follow the assumptions of the proposition and that when the acceptable times tend to infinity or the variabilities of the path travel times tend to zero, the LAPUE solutions converge to the UE ones.

Finally, it must be said that this formulation can be generalized to the case of different user classes (with different trip purposes or different levels of risk aversion), each one with

a different latest acceptable time. In that case, the actual ODs can be divided into a larger number of virtual OD pairs with different values of $\boldsymbol{\kappa}$ and proceed as usual. This generalization has not been develop earlier as it would have made the notation more complex.

The Nie model

Similarly to Lo et al. (2006), Nie (2011) proposes an extension to the UE model, called *multi-class percentile UE*, where all the used routes have the same percentile travel time[16]. In this case, link capacities are assumed to be random variables and are called Service Flow Rates (SFR). The main advantages of this model are that it permits assuming any theoretical distribution, does not make use of the CLT and enables flow-dependent stochasticity, i.e. random road capacity distribution may vary with congestion levels. The model further assumes that link travel times are independent random variables and, thus, route travel times can be expressed as the sum of its link travel times.

It is supposed that the link capacities (or SFR) Q_{ij} are random variables with $F_{Q_{ij}}$ being its cumulative distribution function (CDF). It is also assumed that travel time on a link T_{ij} is strictly increasing with link volume w_{ij} and strictly decreasing with service flow rate Q_{ij}. T_{ij} also depends on a vector of other parameters $\boldsymbol{\theta}_{ij}$, written as $T_{ij} = g(w_{ij}, Q_{ij}, \boldsymbol{\theta}_{ij})$, where $g(\cdot)$ is the link performance function. Therefore, the CDF of the link travel times can be obtained as:

$$\begin{aligned} F_{T_{ij}}(y) &= P\{T_{ij} \leq y\} = P\{g(w_{ij}, Q_{ij}, \boldsymbol{\theta}_{ij}) \leq y\} = P\{g^{-1}(w_{ij}, y, \boldsymbol{\theta}_{ij}) \leq Q_{ij}\} \\ &= 1 - F_{Q_{ij}}(g^{-1}(w_{ij}, y, \boldsymbol{\theta}_{ij})). \end{aligned} \tag{3.143}$$

As mentioned, this model is based on percentiles. The α-percentile link travel time and α-percentile link SFR can be computed as: $t_{ij}^{\alpha} = F_{T_{ij}}^{-1}(\alpha)$ and $q_{ij}^{\alpha} = F_{Q_{ij}}^{-1}(1-\alpha)$, respectively. An important property of these percentiles is given in Proposition 6.

Proposition 6 *Given a CDF $F_{Q_{ij}}(y)$ and a link performance function $g(\cdot)$, the α-percentile link travel time is given by: $t_{ij}^{\alpha} = g(w_{ij}, q_{ij}^{\alpha}, \boldsymbol{\theta}_{ij})$.*

Proof. Note that $t_{ij}^{\alpha} = F_{T_{ij}}^{-1}(\alpha)$ and, hence,

$$\alpha = F_{T_{ij}}(t_{ij}^{\alpha}) = 1 - F_{Q_{ij}}(g^{-1}(w_{ij}, t_{ij}^{\alpha}, \boldsymbol{\theta}_{ij})).$$

Thus, we have

$$g^{-1}(w_{ij}, t_{ij}^{\alpha}, \boldsymbol{\theta}_{ij}) = F_{Q_{ij}}^{-1}(1-\alpha) \Rightarrow t_{ij}^{\alpha} = g(w_{ij}, F_{Q_{ij}}^{-1}(1-\alpha), \boldsymbol{\theta}_{ij}) = g(w_{ij}, q_{ij}^{\alpha}, \boldsymbol{\theta}_{ij}).$$

■

[16] It should be mentioned that, as shown in Wu and Nie (2011), the percentile travel time is equivalent to the travel time budget such as discussed in Lo et al. (2006) when the travel time is normally distributed.

Proposition 6 suggests that the α-percentile link travel time can be computed using a deterministic link performance function with a nominal SFR that reflects the reliability requirements. Furthermore, it permits proving the following proposition.

Proposition 7 *The derivative of the α-percentile link travel time is the corresponding α-percentile derivative of link travel time.*

Proof. The derivative of the α-percentile link travel time is given by

$$\frac{\partial t_{ij}^{\alpha}}{\partial w_{ij}} = g'(w_{ij}, q_{ij}^{\alpha}, \boldsymbol{\theta}_{ij}).$$

On the other hand, the derivative of the link travel time is

$$\frac{\partial T_{ij}}{\partial w_{ij}} = g'(w_{ij}, Q_{ij}, \boldsymbol{\theta}_{ij}),$$

and, similarly to the result in Proposition 6, the α-percentile derivative of link travel time can be computed as $\left(\partial \frac{T_{ij}}{\partial w_{ij}}\right)^{\alpha} = g'(w_{ij}, q_{ij}^{\alpha}, \boldsymbol{\theta}_{ij})$. This concludes the proof. ∎

We shall next define the model in Nie (2011). Let Υ_r^{α} be the α-percentile route travel time of path r. If, as mentioned before, all the users were to minimize their percentile route travel time to their own α, the UE conditions imply that any used path has the identical and minimum percentile route travel time, i.e.,

$$f_r^{\alpha} > 0 \Rightarrow \Upsilon_r^{\alpha} = \sum_{ks \in \mathcal{OD}} \xi_{ksr} \pi_{ks}^{\alpha}; \qquad \Upsilon_r^{\alpha} \geq \sum_{ks \in \mathcal{OD}} \xi_{ksr} \pi_{ks}^{\alpha}, \quad \forall r \in \mathcal{R}, \forall \alpha \in \mathcal{H}, \boldsymbol{f} \in \Omega, \tag{3.144}$$

where $\Omega = \{\boldsymbol{f} | \sum_{r \in \mathcal{R}} \xi_{ksr} f_r^{\alpha} = \tau_{ks}^{\alpha}, \ \boldsymbol{f} \geq 0\}$. This route flow pattern is called a multi-class percentile UE solution.

Once the model has been defined, we need to focus on the computation of the α-percentile route travel times. Let T_r be the random travel time on path r. Clearly the CDF of T_r can be constructed from the CDFs of its path links:

$$T_r = \sum_{\ell_{ij} \in \mathcal{A}} \delta_{ijr} T_{ij} = \sum_{\ell_{ij} \in \mathcal{A}} \delta_{ijr} g(w_{ij}, Q_{ij}, \boldsymbol{\theta}_{ij}). \tag{3.145}$$

Assuming that all Q_{ij} are independently distributed, F_{T_r} can be evaluated recursively by convolution. Let $\hat{r}$ be a subpath of r, that is, $r = \ell_{ij} \cup \hat{r}$ where ℓ_{ij} is the first link on path r. Then,

$$F_{T_r}(y) = \int_0^y \nu_{T_{ij}}(z) F_{T_{\hat{r}}}(y - z) dz, \tag{3.146}$$

where $\nu_{T_{ij}}$ is the link travel time pdf. Furthermore, we have $\Upsilon_r^{\alpha} = F_{T_r}^{-1}(\alpha)$, that is not available in closed form but can be evaluated numerically[17] using (3.146).

[17]Note that the main contribution of this work is the use of convolutions to compute the CDF of the path travel times, instead of relying on the CLT.

It is also demonstrated in Nie (2011) that the percentile route travel time mapping function is monotone under very strong conditions, i.e., when all routes are separable (in other words, when routes that shared one or more links do not exist). Therefore, the author postulates that monotonicity does not hold in general and, hence, uniqueness of solutions cannot be dealt with.

Finally, Nie (2011) develops a gradient projection algorithm to solve the multi-class percentile UE problem. This implies that gradients, that is, partial derivatives of route travel time percentiles with respect to route flows need to be calculated. This algorithm is based on the idea that, as it happens with link travel times in Proposition 7, percentiles and derivatives of path travel times permute, i.e., the derivative of the α-percentile route travel time coincides with the α-percentile of the path travel time derivative. The author postulates that this relation holds but leaves it as an open question. However, as mentioned in the paper, the quality of the algorithm largely depends on the validity of that conjecture.

There are multiple ways to model the reliability problem in the literature. As it has been explained, not all of them coincide or have the same philosophy. In this section, we have explained four different models but other models such as the ones in Nie and Wu (2009), Uchida and Iida (1993) or Szeto et al. (2006) must be mentioned.

Example 2 (Travel time reliability) *With the purpose of showing the differences between the proposed travel time reliability models, they are applied to the Nguyen-Dupuis network (Figure 3.1). Again, the BPR travel time function is assumed and the parameters and OD flows are the ones in Example 1. As all the proposed models are path based, a set of paths is needed. The one used in this example is shown in Table 3.4.*

We have solved the travel time reliability problem by means of the Lo and Tung (2003), Lo et al. (2006) and Watling (2006) models[18]*. For the sake of simplicity, only one class of users has been assumed in all the models. For the three models, different parameters have been assumed. For the Lo and Tung (2003) model,* $\lambda = 0.07$*; for Lo et al. (2006),* $\rho = 0.95$ *and hence* $\lambda = 1.645$ *and for Watling (2006),* $d_r = 0,\ \forall r \in \mathcal{R}, \theta_1 = 1$ *and* $\theta_2 = 0.4$*. As it can be seen, the value of the parameters are associated with risk averse users. We have decided to choose them in order to show bigger differences with respect to the classical UE model.*

In Table 3.5 the used routes for the different models (including the classical UE) and the values of their corresponding measures are shown. Note that the path configuration is different for all the models as the objective pursued by each of them is not the same. For example, Lo and Tung (2003) postulate that paths will be used if they have minimum mean travel time and variance under a threshold. For that reason, path 5 *and* 14 *are not in use even if they have minimum mean travel times. Lo et al. (2006) propose to*

[18]The Nie (2011) model has not been solved as it is equivalent to the Lo et al. (2006) when normal distributions are assumed.

OD	Routes	path links
1 - 2	1	1 5 7 9 11
1 - 2	2	2 18 11
1 - 3	3	1 5 7 10 16
1 - 3	4	1 5 8 14 16
1 - 3	5	2 17 8 14 16
1 - 3	6	1 6 12 14 16
1 - 3	7	1 6 13 19
1 - 3	8	2 17 7 10 16
4 - 2	9	3 5 7 9 11
4 - 2	10	3 5 7 10 15
4 - 2	11	3 6 12 14 15
4 - 2	12	4 12 14 15
4 - 3	13	3 6 13 19
4 - 3	14	3 5 8 14 16
4 - 3	15	4 13 19

Table 3.4: Routes used for the travel time reliability model for the Nguyen Dupuis network.

minimize the budget travel time (which under certain circumstances is equivalent to the percentile). Paths 5 *and* 6 *have bigger travel time budgets than the other paths in the same OD. For that reason, there is no flow on them. Finally, Watling (2006) seeks to minimize a function that takes into account the mean travel time plus a penalization term on the expected time over a latest acceptable travel time. For that reason, paths with large mean travel times or big variabilities will not be in use. That is the case of paths* 1 *and* 11.

OD	Routes	Wardrop (1952)	Lo & Tung (2003)	Lo et al. (2006)	Watling (2006)
1 - 2	1	44.338	**48.965**	**52.534**	44.383
1 - 2	2	**43.414**	**48.965**	**52.534**	**43.516**
1 - 3	3	**45.539**	**50.575**	**54.350**	**45.669**
1 - 3	4	**45.539**	**50.575**	**54.350**	**45.669**
1 - 3	5	**45.539**	50.575	54.425	45.675
1 - 3	6	45.539	**50.575**	54.579	**45.669**
1 - 3	7	**45.539**	**50.575**	**54.350**	**45.669**
1 - 3	8	45.539	**50.575**	**54.350**	**45.669**
4 - 2	9	**46.501**	**51.734**	**55.666**	**46.538**
4 - 2	10	**46.501**	**51.734**	**55.666**	**46.538**
4 - 2	11	46.501	**51.734**	**55.666**	46.709
4 - 2	12	**46.501**	**51.734**	**55.666**	**46.538**
4 - 3	13	**47.702**	**53.345**	57.494	47.931
4 - 3	14	47.702	53.345	57.482	47.930
4 - 3	15	**47.702**	**53.345**	**56.821**	**47.916**

Table 3.5: Route variables for the different models classified by OD for the Nguyen-Dupuis network. Used routes are boldfaced.

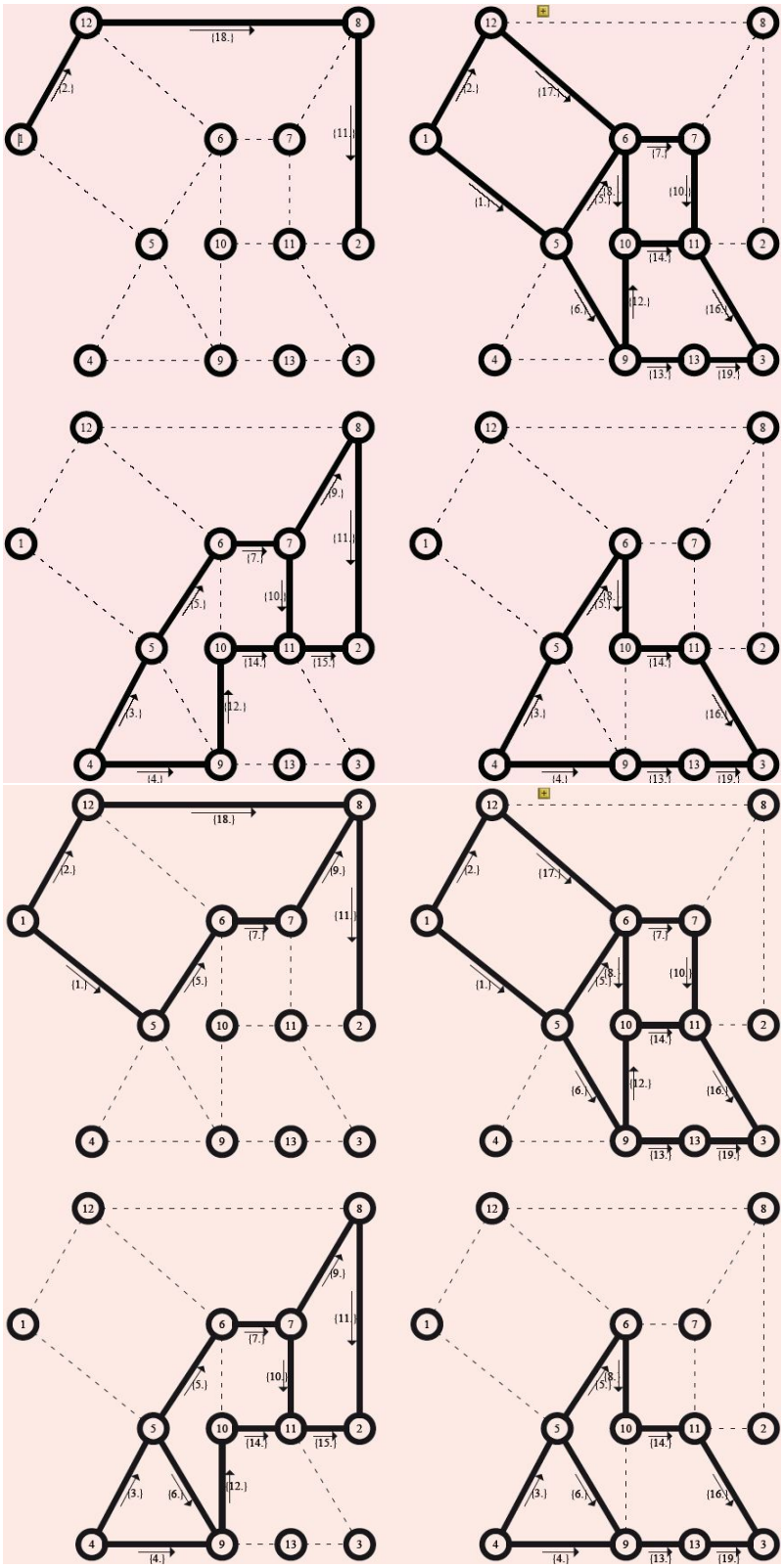


Figure 3.4: The Nguyen-Dupuis example. Used routes by different OD pairs for UE and SO models, respectively.

Appendix

A Link performance function

In this section, some link performance functions (also called volume-delay functions) that have been proposed in the literature are explained. In most traffic assignment methods, the effect of road capacity on travel times is specified by means of the so called volume-delay or link travel time functions $t(v/q)$ which express the travel time on a link as a function of the link traffic volume v and q is related to the link capacity. Many different types of volume-delay functions have been proposed and used in practice in the past. A lot of link cost-flow functions were developed during the $1960s - 70's$ but as shown in Boyce and Janson (1981), different link congestion functions should be used depending on the city, region or country, and most importantly, depending on the assignment procedure.

One of the most common link travel time formulas is the Bureau of Public Roads (BPR) (Bureau of Public Roads (1964)) cost function

$$t_{BPR}(v; t_0, \beta, \gamma, q) = t_0 \left[1 + \beta \left(\frac{v}{q}\right)^{\gamma}\right], \tag{3.147}$$

where for a given link, t_0 is the travel time associated with free flow conditions, v is the flow on that link, q is a constant measuring the flow producing congestion, and β and γ are constants defining how the cost increases with traffic flow. Some examples of the BPR functions are given in the left graph of Figure 3.5.

One alternative to the BPR function is the Spiess cost function (see Spiess (1990)):

$$t_{Spiess}(v; \rho, q) = t_0 \left[-\frac{2\rho - 1}{2\rho - 2} - \rho\left(1 - \frac{v}{q}\right) + \sqrt{\frac{(2\rho-1)^2}{(2\rho-2)^2} + \rho^2\left(1 - \frac{v}{q}\right)^2 + 2}\right], \tag{3.148}$$

where ρ is a constant. Some examples of the Spiess functions are given in the right graph of Figure 3.5 for $\rho = 2, 3, 4, 5, 6, 10, 15, 20$.

Other formulas have been developed in the transportation field. A lot of them are empirical and others result from experimental works.

For example, Mosher (1963), suggested the following logarithmic and hyperbolic functions:

$$t_{Mosher1}(v) = t_0 + \ln(q/(q - v)); \quad v \leq q, \tag{3.149}$$

$$t_{Mosher2}(v) = t_0 - \rho + \rho q/(q - v); \quad v \leq q, \tag{3.150}$$

where ρ is a parameter and q is the traffic volume leading to traffic collapse.

These functions predict travel times that would be infinite for some feasible link flows. That is why the functions must be limited by $v < q$.

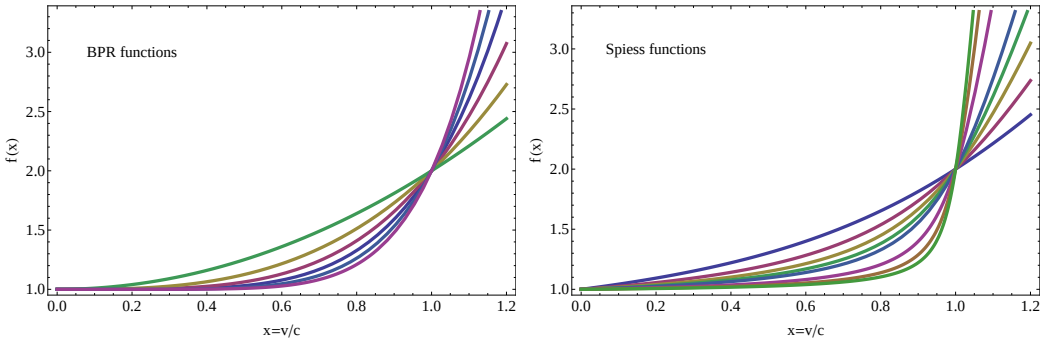


Figure 3.5: Some examples of the BPR ($\beta = 1; \gamma = 2, 3, 4, 5, 6, 7$) and Spiess ($\rho = 2, 3, 4, 5, 6, 10, 15, 20$) link travel time functions.

The Traffic-Research-Corporation (1966) proposed the following function to relate journey per unit distance and flow on links of the Winnipeg's streets network:

$$t_{TRC}(v) = \rho + \beta(v - \gamma) + \sqrt{[\beta^2(v - \gamma)^2 + q]}, \tag{3.151}$$

where ρ, β, γ and q are parameters to be estimated from data.

This is a complicated formula which can, in fact, be replaced by the BPR function (3.147) with little loss of accuracy (see Branston (1976)).

Davidson (1966), proposed a link congestion function that includes a delay parameter k, and he assumes that it varies with route type and location in a metropolitan area. The link travel time function is:

$$t_{Davidson}(v) = t_0 \left[1 + \rho \left(\frac{v}{q - v}\right)\right], \tag{3.152}$$

where ρ is a parameter that varies from 0.01 to 0.5, depending on the type of road.

B Notation

α	risk class.
β	cost function parameter.
β_{ij}	parameter of the BPR function.
γ	cost function parameter.
γ_{ij}	parameter of the BPR function.
γ_{ksr}	parameter of a cost function associated with route r with origin-destination ks.
$\mathbf{\Gamma}$	OD-route incidence matrix.
$\mathbf{\Delta}$	path-link incidence matrix.
δ_{ijr}	element of the link-route incidence matrix.
δ'_{ij}	Kronecker delta.
ϵ	parameter associated with risk aversion.
$\boldsymbol{\eta}$	vector with elements η_{ks}.
η_{is}	dual variable.
η_{ks}	dual variable.
ℓ_{ij}	link joining nodes i and j.
θ_0	cost function parameter.
θ_1	cost function parameter.
θ_2	cost function parameter.
$\boldsymbol{\theta}_{ij}$	vector of parameters associated with link ℓ_{ij}.
ϑ_r	marginal density function of T_r.
κ	weighting factor.
κ_{ks}	latest acceptable arrival time associated with OD pair ks.
λ	parameter associated with risk aversion.
$\boldsymbol{\lambda}$	vector with elements λ_{ij}.
λ_{ij}	dual variable.
λ_{iks}	dual variable.
μ	mean of the v_{ijks} variables.
$\boldsymbol{\mu}$	vector with elements μ_r.
μ_{ijs}	dual variable.
μ_{ijks}	dual variable.
μ_r	dual variable.
ξ_{rks}	element of the OD-route incidence matrix.
π_{ks}	equilibrium cost to travel from origin k to destination s.
ρ	parameter.
ρ_1	cost function parameter.
ρ_{is}	auxiliary variable associated link ℓ_{ij} and destination s.
$\varrho(\cdot)$	arc travel time function.
σ_{T_r}	standard deviation of the random variable of the travel time on route r.

τ_{ks}	flow traveling from node k to node s.
Υ_r^α	α-percentile route travel time of path r.
$\phi_{ksr}(\cdot)$	function converting money into time.
φ_{ksr}	function describing the value of time for route r.
$\varphi_r(\cdot)$	monotone normalized function.
ζ_{ks}	dual variable.
ψ_{ij}	dual variable.
$\mathcal{A}$	set of links.
b_r	travel time budget associated with path r.
$c_{ij}(\cdot)$	cost associated with traversing link ℓ_{ij}.
$C_{ij}(\cdot)$	integral of the travel time function associated with link ℓ_{ij}.
c_{ksr}	travel cost associated with route r with origin-destination ks.
$\tilde{c}_{ksr}$	longest route cost.
c_r	travel time of route r.
$c'_{ij}(\cdot)$	first derivative of $c_{ij}(\cdot)$.
$\mathcal{D}$	set of destination nodes.
d_r	parameter of a cost function.
$\mathbf{f}$	vector of route flows.
f_{ksr}	flow on path r with origin-destination ks.
$F_{Q_{ij}}$	cumulative distribution function of Q_{ij}.
f_r	flow on path r.
$\mathbf{f}^*$	equilibrium flow pattern.
$G(\cdot)$	gap function.
$g_{ks}(\cdot)$	auxiliary function associated with origin-destination ks.
$\mathcal{H}$	set of user classes.
i	link begin node.
$\mathbf{I}$	identity matrix.
j	link end node.
k	origin node.
K	set of feasible route flows.
ks	origin-destination pair from nod k to node$s$.
l	route.
m	cardinal of the v_{ijks} variables.
m_{ksr}	cost function parameter associated with route r with origin-destination ks.
$\mathbf{n}$	route cost vector.
$\mathcal{N}$	set of nodes.
n_1	number of routes.
n_2	number of OD pairs.
$\mathcal{O}$	set of origin nodes.
$\mathcal{OD}$	set of origin-destination pairs.

$p_{ksr}(\cdot)$	auxiliary function associated with route r with origin-destination ks.
q	cost function parameter.
$\mathbf{q}$	vector of flow demands.
q_{ij}	parameter of the BPR function.
Q_{ij}	random variable associated with the capacity on link ℓ_{ij}.
r	route.
$\mathbb{R}$	set of real numbers.
$\mathcal{R}$	set of routes.
s	destination node.
t_{0r}	free flow time of route r.
t_{0ij}	parameter of the BPR function.
T_{ij}	travel time random variable of link ℓ_{ij}.
t_{ij}^{α}	α-percentile link travel time of link ℓ_{ij}.
t_r	performance function of route r.
T_r	travel time random variable of route r.
u_r	path disutility function.
v_{ijks}	flow through link i, j with origin node r and destination node s.
v_{ijs}	flow passing through link ℓ_{ij} with destination s.
$\mathbf{w}$	vector of link flows.
w_{ij}	flow through link ℓ_{ij}.
x_{ijr}.	flow coming from a given origin node r and using link ℓ_{ij}.
y_{ijs}	flow going to a given destination node s and using link ℓ_{ij}.
$Z(\cdot)$	objective function.
z_{irs}	flow going from origin node r to destination node s and passing through node i.

Chapter 4

Origin-Destination matrix estimation models

Contents

4.1 Introduction

The Origin-Destination (OD) flow matrix is a fundamental input for most problems regarding planning and management of transportation systems, e.g., the assignment problem dealt with in the previous chapter. In practice, the "true" OD matrix is seldom, if ever, available and various methods can be used for its estimation (see, for example, Bell (1983), Ashok and Ben Akiva (2000), Zhou and Mahmassani (2006), Nie and Zhang (2008), or the review of Praskher and Bekhor (2004)). According to Cascetta (1984) and Doblas and Benítez (2005), the matrix estimation (ME) models can be divided into the following categories:

(a) *Direct sample estimation*, such as home-based or roadside surveys.

(b) *Traffic counts based methods*, such as the least squares or maximum entropy models.

Among the above two approaches, the first one yields the most accurate results, as it deals with very informative data, but it is very expensive and time consuming and

cannot be undertaken frequently. The second alternative is the one that has received more attention in the literature because of the great economic advantages it offers, which are derived from the carrying out of flow measurements instead of the more expensive surveys. The aim of this chapter is to review some models of the second category developed in the literature.

The chapter is organized as follows. Section 4.2 introduces the notation and concepts of traffic count based methods and develops some of the models presented in the literature to solve the ME problem, namely, generalized least squares based methods are dealt with in Section 4.2.1, entropy or information based methods in Section 4.2.2 and statistical based methods in Section 4.2.3. Finally, Section 4.3 is devoted to the bi-level problem.

4.2 Traffic count based methods

Since the number of OD pairs is normally much larger than the number of links, this problem (estimate OD flows based on link flows) is under-specified, i.e., there is an infinite set of solutions for the OD pair flows satisfying the conservation laws (see Castillo et al. (2008b)) and many of them can be far from the actual ones. Since we look for OD flow estimates close to the real ones, more information is needed. To this end, we normally use a prior (or out-of-date) OD matrix and contemplate, as a reasonable set of solutions, the set of matrices close to it.

To obtain the OD flow estimates there exist a wide range of possibilities. However, all of them can be formulated, in general, as follows (see Yang et al. (1992) and Doblas and Benítez (2005)):

$$\underset{\mathbf{W},\mathbf{T}}{\text{Minimize}}\, Z = F_1(\mathbf{T}, \bar{\mathbf{T}}) + F_2(\mathbf{W}, \bar{\mathbf{W}}) \tag{4.1}$$

subject to

$$\mathbf{W} = M(\mathbf{T}), \tag{4.2}$$

where $\mathbf{T}$ is the OD matrix to be estimated, $\bar{\mathbf{T}}$ is the reference (or prior) OD matrix, $\mathbf{W}$ is the vector containing the link flows to be estimated, $\bar{\mathbf{W}}$ is the vector containing the observed link flows, and $F_1(\mathbf{T}, \bar{\mathbf{T}})$ and $F_2(\mathbf{W}, \bar{\mathbf{W}})$ are functions of a generalized distance measurement, or errors between $\mathbf{T}$ and $\bar{\mathbf{T}}$, and $\mathbf{W}$ and $\bar{\mathbf{W}}$, respectively. Functions F_1 and F_2 might already include factors which relative weight one function over the other. $M(\mathbf{T})$ refers to the assignment map[1], which describes the relationship (not necessarily linear) between the predicted link flows ($\mathbf{W}$) and OD matrix ($\mathbf{T}$).

If the route choice proportions are determined independently from the estimating process, proportional assignment is employed and constraint (4.2) reduces to the linear equation system:

$$\mathbf{W} = \mathcal{B}\mathbf{T}, \tag{4.3}$$

[1] M assigns the OD flows T to the different links in the network W. A kind of assignment map is the User Equilibrium explained in Chapter 3.

where $\mathcal{B} = [\beta_{ijks}]$ is the assignment proportion matrix whose elements β_{ijks} denote the proportion of trips in the OD pair ks using link ℓ_{ij}. If a model with path enumeration is being used, $\mathcal{B}$ can be replaced by $\mathbf{\Delta}P$, where $\mathbf{P}$ is a matrix defining the probabilities of the users to select the different paths associated with all OD pairs, and $\mathbf{\Delta}$ is the path-link incidence matrix.

On the other hand, if link counts are assumed error free, the above formulation reduces to:

$$\underset{\mathbf{T}}{\text{Minimize}}\, Z = F_1(\mathbf{T}, \bar{\mathbf{T}}) \tag{4.4}$$

subject to

$$\mathbf{W} = \mathcal{B}\mathbf{T}, \tag{4.5}$$

because the observed flows can be considered as the true ones, and hence $F_2(\mathbf{W}, \bar{\mathbf{W}}) = 0$. It should be emphasized that this formulation is only valid for full observability of link flows.

However, the assumption of a linear assignment map has inherent shortcomings, as stated by Yang et al. (1992):

> *"Because the OD matrix is estimated from observed link flows with fixed route choice proportions, and the OD matrix is assigned to the network making use of the user equilibrium concept, there is an inconsistency in using one set of route choice proportions to obtain an OD matrix from link flows, and another to obtain the link flow distribution by assigning the OD matrix to the network".*

Furthermore, if a linear assignment map is assumed, congestion effects would not be taken into account correctly. These deficiencies are overcome by many researchers by relating matrix $\mathcal{B}$ (or $\mathbf{P}$) to travel demands in an iterative manner of assigning pre-estimated OD matrices to the network from which a proportion matrix is obtained exogenously and to proceed to the OD matrix estimation problem, repeating this loop till convergence is reached. This approach is called bi-level model[2] and is developed in Section 4.3.

Based on the link flow observations and the prior matrix, the OD matrix estimate can be obtained by many different methods. In particular, generalized least squares methods (GLS), entropy or information based methods and statistical based methods.

4.2.1 Generalized least squares based methods

The generalized least squares (GLS) method seeks to estimate the OD flows by minimizing the sum of squares of the differences between the predicted and the prior OD trip matrices,

[2]The bi-level models combine the OD matrix estimation and the network equilibrium assignment into one process so that, the effects of traffic congestion on travel times and on route choices are taken into account explicitly.

correcting or weighting by the variances and covariances of each flow. The GLS problem can be formulated as:

$$\underset{\mathbf{W},\mathbf{T}}{\text{Minimize}}\, Z_{GLS} = (\bar{\mathbf{T}} - \mathbf{T})'\mathbf{U^{-1}}(\bar{\mathbf{T}} - \mathbf{T}) + (\bar{\mathbf{W}} - \mathbf{W})'\mathbf{Y^{-1}}(\bar{\mathbf{W}} - \mathbf{W}) \tag{4.6}$$

subject to

$$\mathbf{W} = \mathcal{B}\mathbf{T}, \tag{4.7}$$

where Z_{GLS} is the objective function to be minimized, the apostrophe stands for the transpose, $\mathbf{U}^{-1}$ and $\mathbf{Y}^{-1}$ are the precision (inverse variance-covariance) matrices of $\bar{\mathbf{T}}$ and $\bar{\mathbf{W}}$, respectively; and $\mathbf{T}$, $\bar{\mathbf{T}}$, $\mathbf{W}$ and $\bar{\mathbf{W}}$ are as defined previously.

These methods have been discussed by many authors such as Cascetta (1984), Cascetta and Nguyen (1988) or Doblas and Benítez (2005), and have important advantages because of their good statistical bases, which also allows us treating them from a statistical point of view. Under the hypothesis that the target trip and traffic counts are unbiased, the generalized least squares estimator, also called Aitken estimator, is the best linear unbiased estimate (BLUE) of the OD matrix. Moreover, if those random vectors can be considered distributed according to a multivariate normal, the Aitken estimator coincides with the maximum likelihood one.

Other authors add more constraints to the generalized least squares method, as, for example, Doblas and Benítez (2005), Bell (1991) and Carey and Revelli (1986). In particular, Doblas and Benítez (2005) suggest to add the following ones:

$$l_{ks} \leq t_{ks} \leq u_{ks}, \quad \forall k, s \in \mathcal{OD} \tag{4.8}$$

$$l_k^O \leq \sum_s t_{ks} \leq u_k^O, \quad \forall k \in \mathcal{O} \tag{4.9}$$

$$l_s^D \leq \sum_k t_{ks} \leq u_s^D, \quad \forall s \in \mathcal{D} \tag{4.10}$$

$$l \leq \sum_k \sum_s t_{ks} \leq u, \tag{4.11}$$

where l_{ks}, l_k^O, l_s^D, l and u_{ks}, u_k^O, u_s^D, u are the lower and upper bounds of different magnitudes, respectively. More precisely, Equation (4.8) refers to the number of trips between OD pairs; Equation (4.9), to the number of trips generated in each zone (origin); Equation (4.10), to the number of trips attracted by each zone (destination); and, Equation (4.11), to the total number of trips in the network. The rationale behind these constraints is to preserve as much information as possible from the target matrix, and not only the trip distribution between pairs of transport zones. Furthermore, Doblas and Benítez (2005) state that there are not universal values for the bounds and they will depend on the features of the real problem to be solved.

4.2.2 Entropy or information based methods

Analogously to the generalized least squares methods, they optimize an objective function subject to constraints, which try to measure how reasonable and close to reality are the different OD matrices (Castillo et al. (2008g)). In this case, they determine the optimal OD flow matrix that is consistent with the information contained in the observed link flow data, by maximizing the entropy or minimizing the information with respect to a prior matrix[3]. Relevant contributions in this group are those of Willumsen (1978), Van Zuylen (1978), Van Zuylen and Willumsen (1980), and Rossi et al. (1989).

Since the information available in the traffic counts on the links is insufficient to determine a complete matrix, Van Zuylen (1978) proposes to choose the OD matrix that adds as little information as possible, using Brillouin (1956)'s information measure. Therefore, the ME problem can be stated as[4]:

$$\underset{\mathbf{T}}{\text{Minimize}}\, Z_{INF} = \sum_{\ell_{ij}} \sum_{ks} t_{ks} \beta_{ijks} \log_e \frac{t_{ks} S_{ij}}{w_{ij} \bar{t}_{ks}} \tag{4.12}$$

subject to

$$w_{ij} = \sum_{ks} \beta_{ijks} t_{ks}, \quad \forall \ell_{ij} \in \mathcal{A} \tag{4.13}$$

where w_{ij} are the link flows and $S_{ij} = \sum_{ks} \beta_{ijks} \bar{t}_{ks}$.

Similarly, Willumsen (1978) proposes to solve the problem following an entropy maximizing approach. According to the definition of entropy, the most likely OD matrix is the one having a greatest number of micro-states associated with it. The entropy maximization model can be stated as (Brenninger-Gothe and Jornsten (1989)):

$$\underset{\mathbf{W},\mathbf{T}}{\text{Minimize}}\, Z_{ENT} = \gamma_1 \sum_{ks} t_{ks} \left(\log_e \frac{t_{ks}}{\bar{t}_{ks}} - 1 \right) + \gamma_2 \sum_{\ell_{ij}} w_{ij} \left(\log_e \frac{w_{ij}}{\bar{w}_{ij}} - 1 \right) \tag{4.14}$$

subject to

$$w_{ij} = \sum_{ks} \beta_{ijks} t_{ks}, \quad \forall \ell_{ij} \in \mathcal{A} \tag{4.15}$$

where γ_1 and γ_2 are the weighting factors of the OD matrix and the traffic counts, respectively. For example, both factors can be the inverse of the variance of $\bar{t}_{ks}$ and $\bar{w}_{ij}$, which are natural measures of uncertainty. Finally, note that only one weighting factor, i.e. γ_1, could be used if the other is set equal to $\gamma_2 = 1$. In that case, γ_1 would represent the relative weight of the first term in Equation (4.17) over the second term.

[3] According to Van Zuylen and Willumsen (1980), further tests with real data are required to settle the question of which model (entropy or information based models) is superior; and according to Brenninger-Gothe and Jornsten (1989), there is no valid theoretical argument that can be used to give preference to either of the two models.

[4] Note that in this case, error free link counts have been assumed.

4.2.3 Statistical based methods

All the methods explained up to this point have been developed with the aim of reproducing the observed data. However, there could be sampling and measurement errors in the data that lead to inconsistencies. By introducing the concept of population parameters, which are defined as the expected values across the transportation network, statistical models can accommodate the variations or errors in the observed link flows. Furthermore, statistical models recognize the stochastic nature of the data and supply information on the variability of the estimates, in the form of probability or credibility intervals. These models conform the largest group and can be classified into two subgroups: Classical and Bayesian methods.

Classical methods

They assume that the traffic flows are multivariate random variables from a given family that satisfy some convenient properties (reproductivity, positive domain, positive skewness, etc.), such as multivariate normal, Poisson or Gamma. For example, Vardi (1996), Lo et al. (1996) and Hazelton (2000) calculate the likelihood from the data based on the assumption that link and OD flows are independent Poisson random variables.

Assuming that the link flows are sufficiently large, these authors approximate the joint distribution of $\mathbf{W}$ by an appropriate multivariate normal distribution:

$$\mathbf{W} \sim \mathbf{N}\left(\boldsymbol{\Delta}\boldsymbol{P}\boldsymbol{T}, \boldsymbol{\Sigma}_{\boldsymbol{W}}\right), \tag{4.16}$$

where the variance-covariance matrix of the link flows is $\boldsymbol{\Sigma}_{\boldsymbol{W}} = \boldsymbol{\Delta} diag(\boldsymbol{P}\boldsymbol{T})\boldsymbol{\Delta}'$.

The corresponding optimization problem based on the log-likelihood is:

$$\underset{\mathbf{W},\mathbf{T}}{\text{Minimize}}\, Z_{EST} = \frac{1}{2}\left(\bar{\mathbf{W}} - \boldsymbol{\Delta}\boldsymbol{P}\boldsymbol{T}\right)' \boldsymbol{\Sigma}_{\boldsymbol{W}}^{-1}\left(\bar{\mathbf{W}} - \boldsymbol{\Delta}\boldsymbol{P}\boldsymbol{T}\right) \tag{4.17}$$

subject to

$$\mathbf{W} = \boldsymbol{\Delta}\boldsymbol{P}\boldsymbol{T}. \tag{4.18}$$

Although they all propose similar optimization problems, these authors make some different assumptions, e.g. Vardi (1996) assumes fixed routing, that is, only one given route between each OD pair is used. As noted by Hazelton (2000), this procedure is not particularly appropriate for traffic systems as Vardi (1996) was principally interested in inference for computer networks. Moreover, Lo et al. (1996) and Hazelton (2000) assume that the traffic system under study remains uncongested and, hence, the probabilities of choosing the different paths can be regarded as independent of traffic flow.

Finally, it should be noted that other authors use constrained maximum likelihood to estimate the model parameters, such as Spiess (1987) and Cascetta and Nguyen (1988), or Bayesian networks, such as Sun et al. (2006) and Castillo et al. (2008g).

Bayesian methods

Bayesian methods are well known statistical tools that consider parametric families of distributions whose parameters are themselves random variables. For example, Tebaldi and West (1998) consider Poissonian flows, Maher (1983) combines a multivariate normal distribution with a multivariate normal prior and Mahmassani and Sinha (1981) deal with the problem of updating trip generation parameters.

According to Cascetta and Nguyen (1988), in the Bayesian inference framework, a priori information on the trip matrix $\mathbf{T}$ is expressed as a prior probability function $g(\mathbf{T})$. On the other hand, the traffic counts represent an additional source of information about $\mathbf{T}$ with given probability $p(\bar{\mathbf{W}}|\mathbf{T})$. Bayes's theorem allows these two sources of information to be combined to provide the posterior probability function $f(\mathbf{T}|\bar{\mathbf{W}})$, i.e., the probability of observing $\mathbf{T}$ conditional to the traffic count $\bar{\mathbf{W}}$:

$$f(\mathbf{T}|\bar{\mathbf{W}}) = \frac{\mathbf{p}(\bar{\mathbf{W}}|\mathbf{T})\mathbf{g}(\mathbf{T})}{\int \mathbf{p}(\bar{\mathbf{W}}|\mathbf{T})\mathbf{g}(\mathbf{T})\mathbf{dT}} \propto \mathbf{p}(\bar{\mathbf{W}}|\mathbf{T})\mathbf{g}(\mathbf{T}). \tag{4.19}$$

When the families of priors and posteriors coincide, we say that this family and the likelihoods are *conjugate*. In this case, the posterior parameters can be easily obtained in terms of the prior parameters and the sample values. Maher (1983) makes use of this property assuming multivariate normal (MVN) distribution for the traffic counts and the prior distribution. Therefore, the posterior distribution of the OD flows $\mathbf{T}$ is $MVN(\mu_{\mathbf{T}}, \Sigma_{\mathbf{T}})$, where:

$$\mu_{\mathbf{T}} = \bar{\mathbf{T}} + \mathbf{\Sigma}_{\bar{\mathbf{T}}} (\mathbf{\Delta P})' \left(\mathbf{\Sigma}_{\bar{\mathbf{W}}} + (\mathbf{\Delta P})\mathbf{\Sigma}_{\bar{\mathbf{T}}}(\mathbf{\Delta P})'\right)^{-1} (\bar{\mathbf{W}} - (\mathbf{\Delta P})\bar{\mathbf{T}}) \tag{4.20}$$

$$\Sigma_{\mathbf{T}} = \Sigma_{\bar{\mathbf{T}}} - \Sigma_{\bar{\mathbf{T}}} (\mathbf{\Delta} \boldsymbol{P})' \left(\Sigma_{\bar{\mathbf{W}}} + (\mathbf{\Delta} \boldsymbol{P})\Sigma_{\bar{\mathbf{T}}}(\mathbf{\Delta} \boldsymbol{P})'\right)^{-1} (\mathbf{\Delta} \boldsymbol{P})\Sigma_{\bar{\mathbf{T}}}, \tag{4.21}$$

$\Sigma_{\bar{\mathbf{T}}}$ and $\Sigma_{\bar{\mathbf{W}}}$ are the variance-covariance matrices of the prior and traffic counts distributions, respectively (see Maher (1983) for more details on the computation of the posterior distribution parameters). As it can be seen, the use of conjugate families permits updating the parameters of the posterior distribution in a simple fashion.

It should be noted that Maher (1983) states that, as the random variables are concerned with counts, the true distribution is some multivariate form of the Poisson distribution. Nevertheless, for counts with means which are not too small[5], the multivariate normal provides an accurate approximation. As already explained, the assumption of normal distribution permits the use of conjugate distributions. Similarly to Maher (1983), Tebaldi and West (1998) consider Poissonian flows but make use of Markov Chain Monte Carlo (MCMC) simulations to compute the posterior distribution, as posterior computation are analytically difficult.

[5]This condition can always be ensured by increasing the period of study so as to have larger traffic counts.

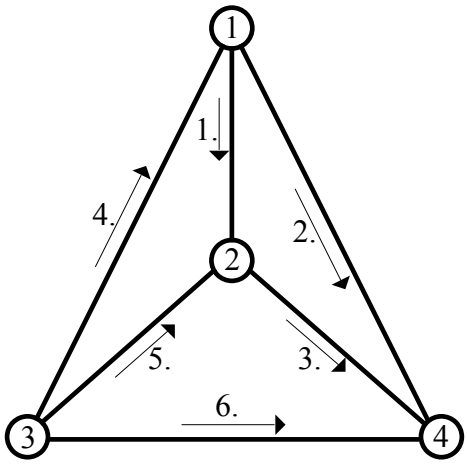


Figure 4.1: The elementary example network used for illustrative purposes, showing the nodes and links.

On the other hand, Hazelton (2001) makes an interesting and deep distinction between reconstruction and estimation of OD pair flows, and indicates that both techniques are useful for estimation purposes, though he mentions that the dissimilarities between both solutions can be very large if the data contradicts the prior information. In particular, according to Hazelton (2001), *the aim of the OD matrix reconstruction is to try and pinpoint the actual number of trips between each OD pair that occurred during the observational period; whereas, the aim of mean OD trip count estimation is to estimate the expected number of OD trips.* In that sense, the author classifies the firsts models explained as reconstruction methods (e.g., Bell (1983), Van Zuylen and Willumsen (1980) or Cascetta (1984)) and the statistical models as estimation methods (e.g., Vardi (1996), Lo et al. (1996), Maher (1983) or Tebaldi and West (1998)).

Example 3 (Matrix estimation models) *In order to illustrate the methods described above, we apply them to a simple network. It consists of* 4 *nodes and* 6 *links (see Figure 4.1).*

Table 4.1 shows the 3 *OD pairs considered and the corresponding* 6 *paths used in this example. For the sake of simplicity, it is assumed that they are the only possible OD pairs and paths.*

The Bureau of Public Roads (BPR) cost function (see Equation (3.147) in the Appendix of Chapter 3) has been used where the parameter t_{0ij} *and* q_{ij} *are shown in Table 4.2, together with real and prior flows for OD pairs and link flows, and parameters* $\beta_{ij} = 1$ *and* $\gamma_{ij} = 4$ *for all the links in the network.*

Table 4.3 contains the $\boldsymbol{\Delta}$ *and* $\boldsymbol{P}$ *matrices. To obtain the elements of* $\boldsymbol{P}$*, a System*

OD	path code (r)	Links
1-4	1	2
	2	1 3
3-4	3	6
	4	5 3
	5	4 2
2-4	6	3

Table 4.1: Set of 3 OD pairs and 6 paths considered in the simple example.

OD matrix		
OD pair	Real	Prior ($\bar{\mathbf{T}}$)
1	100	70
2	120	84
3	80	56

Link parameters and real flows			
Link	t_{0ij}	q_{ij}	Real flows
1	3	50	30
2	2	40	**70**
3	3	100	**180**
4	2	70	0
5	1	50	**70**
6	1	20	50

Table 4.2: Data of the matrix estimation example.

$\boldsymbol{\Delta}$ matrix						
Link	Path					
	1	2	3	4	5	6
1	0.00	1.00	0.00	0.00	0.00	0.00
2	1.00	0.00	0.00	0.00	1.00	0.00
3	0.00	1.00	0.00	1.00	0.00	1.00
4	0.00	0.00	0.00	0.00	1.00	0.00
5	0.00	0.00	0.00	1.00	0.00	0.00
6	0.00	0.00	1.00	0.00	0.00	0.00

$\boldsymbol{P}$ matrix						
OD pair	Path					
	1	2	3	4	5	6
1	0.753	0.247	0.000	0.000	0.000	0.000
2	0.000	0.000	0.399	0.570	0.031	0.000
3	0.000	0.000	0.000	0.000	0.000	1.000

Table 4.3: $\boldsymbol{\Delta}$ and $\boldsymbol{P}$ matrices used in the matrix estimation example.

Optimal assignment model has been used (see Section 3.3 for details on this kind of assignment).

Consider that flows on links $2, 3$ *and* 5 *have been observed, which values are the true flows shown in Table 4.2 and assume a coefficient of variation of* 0.1*. With these data we can obtain the* $\mathbf{Y}^{-1}$ *and* γ_2 *values for the Generalized Least Squares (GLS) and Entropy (ENT) based methods, i.e. the inverse of the variance-covariance matrices of link flows.*

Estimated OD flows				
OD pair	GLS	ENT	CLA	BAY
1	82.85	82.66	89.07	93.52
2	111.68	112.95	102.79	101.52
3	64.97	59.69	63.90	65.68
RMSE	8.09	9.21	8.66	8.09

Estimated link flows				
Link	GLS	ENT	CLA	BAY
1	20.46	20.42	22.00	23.10
2	65.85	65.74	69.05	73.57
3	149.10	144.49	161.21	146.65
4	3.46	3.50	1.98	3.15
5	63.66	64.38	36.43	57.87
6	44.56	45.07	25.50	40.51
RMSE	5.64	6.32	7.73	6.28

Table 4.4: OD matrix and link flows resulting from the different matrix estimation methods.

For $\mathbf{U}^{-1}$ *and* γ_1*, we have considered the elements of the prior OD matrix with a coefficient of variation of* 0.3*. In both cases, for the sake of simplicity, independence among OD pairs and link flows have been assumed.*

The estimated flows with the different matrix estimation methods are shown in Table 4.4, together with the corresponding root mean squared errors (RMSE). This error is defined as:

$$RMSE = \sqrt{\frac{\sum_i (\bar{t}_i - t_i)^2}{N}},$$

where N *is the number of OD pairs or links. It allows us to compare the estimated OD and link flows with the real ones. For example, Table 4.4 shows that the best estimates*[6] *come from the Bayesian Methods (BAY), although the results obtained from the GLS model are also very precise. On the other hand, the Entropy based methods (ENT) lead to the worst results.*

As already mentioned, the main shortcoming of Matrix Estimation methods is that they work with a fixed $\mathbf{P}$ *matrix. If this matrix is far from the real one, the results will not reproduce the reality correctly. In the example, the values in this matrix make the models lead to positive flows on link* 4 *when the real flow through this link is null, as it can be seen in Tables 4.2 and 4.4. This kind of problems can be solved when the bi-level methods, explained in the next section, are applied.*

[6]Note that the aim of these models is to predict the OD matrix so, in order to determine the quality of the predictions, the $RMSE$ values associated with the OD matrix are more important than those of the link flows.

4.3 Bi-level models

As already explained, ME models assume that route choice proportions are given constants but, in a network with realistic congestion levels, this assumption does not hold. In order to overcome this deficiency, bi-level models have been developed which combine the OD matrix estimation and the network equilibrium assignment into one process. While the models above are appropriate for uncongested networks (where flows do not significantly impact on travel times), when the elements of $\mathbf{T}$ are sufficiently large, congestion need to be taken into account as matrix $\mathbf{P}$ (or $\mathcal{B}$) depends on $\mathbf{T}$.

For that purpose, ME models have been extended by switching from the linear assignment map to a user equilibrium assignment map. The resulting formulation has the form of a bi-level optimization problem where the upper level seeks to minimize the sum of distance measurements, while the lower level represents a user equilibrium assignment which guarantees that the estimated OD matrix and corresponding link flows satisfy the user equilibrium conditions. We shall remark that in this case, the output of the model consist of, not only the trip flow matrix, but also the route choice proportions and link flows.

Yang et al. (1992), following Bard (1988), define a bi-level programming problem as:

$$\underset{x}{\text{Min}}\, F(x, y) \tag{4.22}$$

subject to

$$G(x, y) \leq 0, \tag{4.23}$$

where y is obtained by solving the problem

$$\underset{y}{\text{Min}}\, g(x, y) \tag{4.24}$$

subject to

$$g(x, y) \leq 0. \tag{4.25}$$

In game theory, this problem is called a Stackelberg game. It has a hierarchical structure in which an upper-level (the leader) and a lower-level (the follower) decision makers must select their strategies in order to optimize their objective functions, respectively. It is assumed that the leader is given the first choice and selects an x in accordance, while taking into account the reaction of the follower. Finally, in light of this decision, the follower selects a y so as to minimize his objective function f. This kind of games has been widely used to model transportation systems.

Nguyen (1977) was the first to propose a method to estimate OD matrices from traffic counts when congestion effects are considered. The model proposed permits estimating an OD matrix that reproduces observed travel costs, that are derived from traffic count data in conjunction with link cost functions. However, the objective function of the problem

is not strictly convex and, hence, the solutions are not unique. Moreover, it requires a complete set of link traffic counts.

Since then, alternative user equilibrium based estimation techniques have been developed by a number of authors. Fisk (1988) proposes an extended version of the entropy maximizing model of Van Zuylen and Willumsen (1980) where a new constraint is added that represents the UE conditions. The author suggests different formulations of the problem, including the NCP formulation of the UE problem explained in Section 3.2.2. Nevertheless, the observed link flows are included in the model as deterministic and, therefore, existence of a trip matrix solution is not guaranteed as there may be inconsistencies on the link flows.

Yang et al. (1992) show how the generalized least squares (GLS) and entropy function (ENT) models can be integrated with an equilibrium traffic assignment in the form of a convex bi-level optimization problem. According to the authors, by combining the GLS and UE subproblems, we obtain the following bi-level optimization problem:

$$\underset{\mathbf{T}}{\text{Minimize}}\, Z_{BI-GLS} = (\bar{\mathbf{T}} - \mathbf{T})'\mathbf{U}^{-1}(\bar{\mathbf{T}} - \mathbf{T}) + (\hat{\mathbf{W}} - \mathbf{W})'\mathbf{Y}^{-1}(\hat{\mathbf{W}} - \mathbf{W}) \tag{4.26}$$

subject to

$$\mathbf{T} \geq \mathbf{0}, \tag{4.27}$$

where $\mathbf{W}$ solves[7]

$$\underset{\mathbf{f},\mathbf{w}}{\text{Minimize}} \sum_{\ell_{ij} \in \mathcal{A}} \int_0^{w_{ij}} c_{ij}(s)ds \tag{4.28}$$

subject to

$$\sum_{r \in \mathcal{R}} \xi_{rks} f_r = t_{ks}, \quad \forall k, s \in \mathcal{OD} \tag{4.29}$$

$$\sum_{r \in \mathcal{R}} f_r \delta_{ijr} = w_{ij}, \quad \forall \ell_{ij} \in \mathcal{A} \tag{4.30}$$

$$f_r \geq 0 \quad \forall r \in \mathcal{R}. \tag{4.31}$$

In the case of the entropy function, the bi-level optimization problem presents a similar scheme:

$$\underset{\mathbf{T}}{\text{Minimize}}\, Z_{BI-ENT} = \gamma_1 \sum_{ks} t_{ks} \left(\log_e \frac{t_{ks}}{\bar{t}_{ks}} - 1 \right) + \gamma_2 \sum_{\ell_{ij}} w_{ij} \left(\log_e \frac{w_{ij}}{\bar{w}_{ij}} - 1 \right) \tag{4.32}$$

subject to

$$t_{ks} \geq 0, \quad \forall k, s \in \mathcal{OD} \tag{4.33}$$

where $\mathbf{W}$ solves (4.28) subject to (4.29)-(4.31).

[7]The UE formulation used in this model corresponds to the Beckmann model explained in Section 3.2.1.

It should be pointed out that any model that fit into the ME category (such as the minimum information model) can be used as the upper-level objective function of the bi-level optimization model presented.

As the upper-level objective functions in both problems are strictly convex with respect to $\mathbf{T}$ and $\mathbf{W}$ and, for $\mathbf{T}$ fixed, the lower level objective function of UE problem is also strictly convex with respect to its decision variables $\mathbf{W}$, these bi-level optimization methods are convex and, hence, have a unique feasible solution. Furthermore, they do not require full observability of link flows.

We note that the model in Fisk (1988) can be seen as a special case of model (4.32)-(4.1). If the observed link flow pattern is at equilibrium and a large enough parameter γ_2 is chosen, we get the extreme case where no deviation between $\mathbf{W}$ and $\hat{\mathbf{W}}$ is allowed, and therefore the model will produce the same results as Fisk's model.

As the bi-level models make use of the UE models, something must be said about the non-uniqueness of solutions in terms of path flows. As already explained in the previous chapter, UE models lead to unique solutions in terms of link flows (see Theorem 2), but generally not in term of path flows. In that sense, the model has a flaw as matrix $\mathbf{P}$ is constructed from path flows that are not uniquely determined. However, the value of the upper-level objective function is defined by vectors $\mathbf{T}$ and $\mathbf{W}$ and, hence, is independent of the values of path flows.

Bi-level programming problems are generally difficult to solve because evaluation of the upper-level objective function requires solving the lower-level optimization problem. For that reason, a heuristic solution approach has been considered that involves iteratively solving the ME and UE problems in sequence until convergence (see Fisk (1988, 1989) or Yang et al. (1992)). This heuristic is a close representation of the actual decision making process in terms of a Stackelberg game. The general scheme of the algorithm is described as follows (see, for example, Castillo et al. (2008b), Yang (1995) for more details[8]).

Algorithm 1 (Bi-level heuristic) *This algorithm consists of solving the lower-level and upper-level problems iteratively.*

INPUT. A prior OD matrix, the observed link flows and the link cost functions.

OUTPUT. The estimated OD matrix, link flows and route choice proportions (matrix $\mathbf{P}$*).*

Step 0: Initialization. Initialize $\mathbf{T_0}$*. This initial trip matrix can normally be the prior OD matrix (*$\mathbf{T_0} = \bar{\mathbf{T}}$*).*

Step 1: Lower level solution. Given the matrix $\mathbf{T_0}$*, obtain* $\mathbf{P_0}$ *solving the UE assignment problem.*

[8] Yang (1995) presents some other heuristic algorithms for the bi-level origin destination matrix estimation problem.

Estimated OD flows		
OD pair	GLS	ENT
1	78.19	81.24
2	112.98	111.92
3	74.82	68.77
RMSE	7.83	7.77

Estimated link flows		
Link	GLS	ENT
1	7,24	11,24
2	70,95	70,01
3	151,48	148,93
4	0,00	0,00
5	69,42	68,92
6	43,56	43,00
RMSE	6.18	6.16

Table 4.5: OD matrix and link flows resulting from the different bilevel methods.

Step 2: Upper level solution. Using $\mathbf{P_0}$*, the observed link flows and the prior OD matrix, obtain the OD matrix* $\mathbf{T}$*.*

Step 3: Testing convergence. Compute the error by means of

$$\epsilon = (\mathbf{T_0} - \mathbf{T})'(\mathbf{T_0} - \mathbf{T}).$$

If the error ϵ *is less than a given tolerance, stop the process and return the values of* $\mathbf{T}$ *and* $\mathbf{W}$*. Otherwise, update the initial matrix (*$\mathbf{T_0} = \mathbf{T}$*), and go to Step 1.*

The following example, where the previous algorithm is applied, illustrates the described bi-level methods.

Example 4 (Bi-level models) *In order to show how these methods work, we apply them on the network used in Example 3. The same parameters, prior OD matrix and real flows are used.*

The algorithm explained before has been implemented with a convergence tolerance of 10^{-6} *and the results obtained are shown in Table 4.5.*

In this case, the best results come from the entropy model but both models perform similarly. In general, the results are better than those obtained by the matrix estimation procedures (see Table 4.4). Specifically, note that these models, contrary to the ME ones, lead to a null flow for link 4*, which is the real flow on that link. This is due to the fact that the* $\mathbf{P}$ *matrix is updated in each iteration of the algorithm and the choice proportions are calculated depending on the level of demand; in other words, the congestion effects are included in the model. Finally, as the level of congestion is not high in this network, the difference in the RMSE between conventional and bi-level models is not very large. If the network in study is highly congested, the gain in terms of RMSE can be larger.*

Appendix

A Notation

β_{ij}	parameter of the BPR function.
β_{ijks}	proportion of trips in the OD pair ks using link ℓ_{ij}.
γ_1	weighting factor.
γ_2	weighting factor.
γ_{ij}	parameter of the BPR function.
$\mathbf{\Delta}$	path-link incidence matrix.
δ_{ijr}	element of the link-route incidence matrix.
ϵ	convergence error.
ℓ_{ij}	link joining nodes i and j.
$\mu_{\mathbf{T}}$	mean of the multivariate normal distribution of $\mathbf{T}$.
ξ_{rks}	element of the OD-route incidence matrix.
$\mathbf{\Sigma}_{\boldsymbol{W}}$	variance-covariance matrix of the link flows.
$\Sigma_{\mathbf{T}}$	variance of the multivariate normal distribution of $\mathbf{T}$.
$\Sigma_{\bar{\mathbf{T}}}$	variance-covariance matrix of the prior distribution.
$\Sigma_{\bar{\mathbf{W}}}$	variance-covariance matrix of the traffic counts distribution.
$\mathcal{A}$	set of links.
$\mathcal{B}$	assignment proportion matrix.
c_{ij}	cost associated with traversing link ℓ_{ij}.
$\mathcal{D}$	set of destination nodes.
$f(\mathbf{T}\|\bar{\mathbf{W}})$	posterior probability function.
$F_1(\mathbf{T}, \bar{\mathbf{T}})$	function of a generalized distance measurement between $\mathbf{T}$ and $\bar{\mathbf{T}}$.
$F_2(\mathbf{W}, \bar{\mathbf{W}})$	function of a generalized distance measurement between $\mathbf{W}$ and $\bar{\mathbf{W}}$.
f_r	flow on route r.
$g(\mathbf{T})$	prior probability function.
i	link begin node.
j	link end node.
k	origin node.
ks	origin-destination pair from nod k to node$s$.
l	lower bound.
l_{ks}	lower bound associated with OD ks.
l_k^O	lower bound associated with origin node k.
l_s^D	lower bound associated with destination node s.
$M(\mathbf{T})$	assignment map.
$\mathcal{O}$	set of origin nodes.
$\mathcal{OD}$	set of origin-destination pairs.
$p(\bar{\mathbf{W}}\|\mathbf{T})$	probability function of traffic counts.

$\mathbf{P}$	matrix defining the probabilities to select the different paths associated with all OD pairs.
q_{ij}	parameter of the BPR function.
r	route.
$\mathcal{R}$	set of routes.
s	destination node.
$\mathbf{T}$	OD matrix to be estimated.
$\bar{\mathbf{T}}$	prior OD matrix.
t_{0ij}	parameter of the BPR function.
$\mathbf{T_0}$	initial matrix for the algorithm.
t_{ks}	OD flow associated with OD pair ks; element of matrix $\mathbf{T}$.
$\bar{t}_{ks}$	element of matrix $\bar{\mathbf{T}}$.
u	upper bound.
$\mathbf{U}$	variance-covariance matrix of $\bar{\mathbf{T}}$.
u_{ks}	upper bound associated with OD ks.
u_k^O	upper bound associated with origin node k.
u_s^D	upper bound associated with destination node s.
$\mathbf{W}$	vector containing the link flows to be estimated.
$\bar{\mathbf{W}}$	vector containing the observed link flows.
w_{ij}	flow through link ℓ_{ij}.
$\bar{w}_{ij}$	observed flow through link ℓ_{ij}.
$\mathbf{Y}$	variance-covariance matrix of $\bar{\mathbf{W}}$.
Z	objective function.

Chapter 5

Observability problem in traffic models

Contents

5.1 Introduction

One of the aims of traffic models is to estimate traffic flows. These can be link flows, origin-destination flows (as the models explained in Chapter 4), path flows, node flows, etc. The observability problem, dealt with in this chapter, consists of determining if a given subset of available flow measurements is sufficient to estimate another subset of traffic flows (not necessarily of the same type).

Observability analysis is a previous step to flow estimation. It addresses the question: do we have enough information to estimate the flows in a network?

Note that the flow values themselves are not needed to solve the observability problem. One wants to know only if knowledge of these flows would provide enough information to obtain all or a given subset of flows (link, path or OD flows), i.e., if the subset is observable.

If a traffic network is observable, it is relevant to identify critical measurements, that is, measurements that, if missing, make the network unobservable. In other words, to identify measurements whose elimination lead to non-observability. On the contrary, if the state of the system is unobservable, it is relevant to identify the flow subsets that can be estimated.

As done in Castillo et al. (2010) and Castillo et al. (2011), we classify the observability problem on the basis of the target parameters (OD, route and link flows):

OD-pair flow observability: The most common observability problem in the traffic literature is the OD-pair observability problem, in which the OD-pair flows are estimated in terms of other flows. Because the number of independent OD-pair flows is much larger than the number of independent link flows, the problem normally becomes under-specified. In this case, the OD flow observability problem has an infinite set of solutions satisfying the conservation laws, as already seen in Chapter 4. In this case of under-specification, the only possibility of solving the problem and get a unique solution consists of adding extra information, which normally comes in the form of a combination of prior information of OD-pair flows together with some optimization property (maximum entropy or generalized least squares, among others).

However, there are other ways to deal with this problem of under-specification. In particular, some authors, such as Castillo et al. (2007) or Castillo et al. (2008c), dealt with the unusual case of over-specification, that is, when the number of independent link flows is larger than the number of independent OD-pair flows; which is not a very realistic situation. Finally, another way to overcome the problem of under-specification is to make use of scanned links, which supply more information than the counted link flows, this is the case of Anagnostopoulos et al. (2006) and Castillo et al. (2008e). By scanned links we mean links on which a camera has been installed to capture the plate number of the circulating cars, providing information on the links traversed by the vehicles.

Route flow observability: A more difficult observability problem is the route flow observability problem in which we aim to observe all route flows. Due to the fact that knowledge of the route flows immediately leads to the knowledge of the OD pairs and link flows, through the conservation or balance equations, the observability problem of route flows can be considered as the full observability problem. However, it is the most difficult of the observability problems as it requires the maximum amount of information. Thus, the route flow observability problem presents the same under-specification difficulties as does the OD pair observability problem. Similarly to the OD pair observability problem, methods to solve this problem include the use of prior information or more powerful techniques, such as the plate scanning technique.

Link flow observability: The simplest observability problem is the link observability based on link flows. This problem consists of observing all or a subset of link flows on the basis of the observation of the flows on a subset of them. An important property of this problem, which distinguishes it from the other problems considered above, is that a solution always exists; that is, any subset of link is observable on the basis of a subset of links, which, in some extreme cases, can be the subset itself.

In spite of its simplicity, this problem, on which we focus in the remaining of the

chapter, has been suggested recently by Hu et al. (2009). Nevertheless, a somehow different problem is dealt with in Bianco et al. (2001, 2006), who analyze the observability of all arcs in a network based on known turning ratios and observed node flows and discuss the associated complexity.

According to the type of technique used to solve the observability problem, it can also be classified as follows: (a) algebraic, which considers the algebraic relations between all flows and operate them algebraically to draw observability conclusions (see Monticelli and Wu (1985a,b), Abur and Gómez Expósito (2004), Gomez Exposito and Abur (1998), Castillo et al. (2008c)); and (b) topological, which considers only qualitative relations among flows (mainly through the use of graph theory) to derive observability conclusions (see Clements and Wollenberg (1975), Nucera and Gilles (1991), Castillo et al. (2008c), Castillo et al. (2007)). Methods in the first group normally supply more information, but those in the second are faster and require less memory and CPU resources.

For example, Castillo et al. (2008c) present an algebraic model to solve the observability problem and a topological version of it. The algebraic approach makes use of matrix $\mathcal{B}$[1] and presents and algorithm that permits expressing the observable flows in terms of the actually observed flows. Contrary, the topological approach requires modifying matrix $\mathcal{B}$ by replacing any non-zero value by 1, such a way that qualitative dependencies are considered but quantitative dependencies are not. Since the topological version works with binary number, it has no numerical problems and requires less memory, making it useful for large networks. The main limitation of these models is that they require knowledge of matrix $\mathcal{B}$, which is generally unknown, so by selecting these data some errors can be included in the model.

Since all these techniques are based on mathematical properties of the system of equations and have the same structure for traffic problems, these approaches applied to "physical" networks are equally applicable to traffic networks.

In the next section some algebraic models on link flow observability on the basis of counted link flows are explained, which will be useful for a better understanding of Part IV.

5.2 Some algebraic link flow observability models

As already seen, the link flow observability problem consists of determining whether a subset of link flows is sufficient to estimate another given subset of link flows. This apparently simple problem was first stated by Hu et al. (2009).

In practical applications, the installation of sensors on all links is generally unrealistic because of budgetary constraints. Therefore, it is interesting to know the smallest subset

[1]We remind the reader that $\mathcal{B} = [\beta_{ijks}]$ is the assignment proportion matrix whose elements β_{ijks} denote the proportion of trips in the OD pair ks using link ℓ_{ij}, used in Chapter 4.

of links in a network on which to locate sensors that enables the accurate estimation of traffic flows on all links of the network under steady-state conditions. Furthermore, in the presence of measurement errors, duplicity of information would lead to incompatibilities. Hu et al. (2009) propose a simple linear algebra based method to identify the minimum subset of links to be equipped with sensors so as to estimate the flows on all links. Their model does not require the knowledge of matrix $\mathcal{B}$, which is an important advantage because it does not impose any assumptions on the O-D matrices, turning ratios or route choice behavior, but make use of the link-path incidence matrix $\mathbf{\Delta}$. The proposed model is based on the idea of basis of the vector space generated by the link-path incidence matrix.

The basis of the vector space associated with matrix $\mathbf{\Delta}$ consists of z^2 linearly independent column vectors, and the links corresponding to these columns are called the *basis* links. The remaining links in the network are called the *non-basis* links. If the flows on the basis links are observed using sensors, then by definition of algebraic basis, the flow on all links can be inferred through linear combinations of the basis link flows.

In order to determine the rank of matrix $\mathbf{\Delta}$, i.e. the minimum number of counted links, Hu et al. (2009) propose to obtain the reduced row echelon form[3] (RREF) of matrix $\mathbf{\Delta}$ via the Gaussian elimination algorithm, and take r as the number of nonzero rows. Furthermore, the links to be counted are those associated with a unit column vector and the coefficients of the linear combinations that relates non-basis with basis link flows are those shown in the RREF of the link-path incidence matrix.

Because any minor submatrix with the same rank leads to a subset of independent link flows, this means that there can be several solutions. Due to the steps of the Gaussian elimination algorithm, the column positions of the links in matrix $\mathbf{\Delta}$ can decide the set of basis links, giving priority to the links associated with the leftmost columns. Hence, if some links were to be prioritized over others for some external reasons, the higher priority links should be assign to the leftmost columns of the link-path incidence matrix, giving rise to the same network link flows.

Finally, Hu et al. (2009) perform a sensitivity analysis on the effects of network topology and number of OD-pairs and paths on the minimum subset of links to be installed with vehicle sensors and conjecture, as a result, that "there may be an upper bound on the number of basis links that is governed by the network topology irrespective of the total number of links in the network".

[2]Note that z is the rank of the link-path incidence matrix $\mathbf{\Delta}$.

[3]A matrix is said to be in its reduced row echelon form if it satisfies:

1. Any row containing a nonzero entry precedes any row in which all the entries are zero (if any).
2. The first nonzero entry in each row is the only nonzero entry in this column.
3. The first nonzero entry in each row is 1 and it appears in a column to the right of the leading 1 in any preceding row. By definition, if the first nonzero number in a row is 1, it is called leading 1.

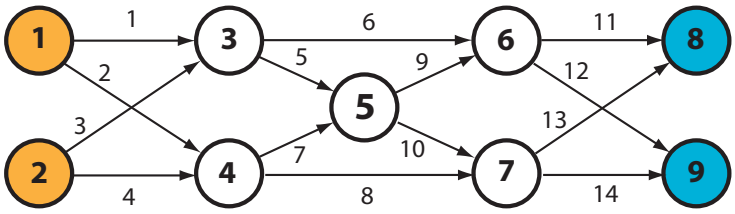


Figure 5.1: The parallel highway network in Hu et al. (2009), showing the nodes and links.

Example 5 (The parallel highway network) *In this example we will apply the method proposed by Hu et al. (2009) to the parallel highway network presented in Hu et al. (2009) and shown in Figure 5.1. It consists of* 9 *nodes,* 14 *links and four OD-pairs. Nodes* 1 *and* 2 *are trip origin nodes, whereas nodes* 8 *and* 9 *are the destination nodes. Tables 5.1 and 5.2 illustrate the link-path incidence matrix for the network and its RREF, respectively. The matrix in Table 5.2 has* 9 *nonzero rows and, hence, the rank of the link-path incidence matrix and the number of links to be counted is* 9*. The basis links correspond to the* 9 *columns in the table that denote unit columns (for a better understanding, they have been boldfaced) and represent the links on which to install vehicle sensors.*

Note also that all non-basis link flows can be inferred through linear combinations of the basis link flows using as coefficients the values in the associated column of the RREF matrix. For example, the flow through links 6 *and* 14*, which are non-basis links, can be computed as:*

$$\begin{aligned} w_6 &= w_1 + w_3 - w_5 \\ w_{14} &= w_2 + w_4 + w_5 - w_9 - w_{13}. \end{aligned}$$

Finally, note that all links need not be equipped with sensors in order to estimate the flows on all links. In this example, we need to install vehicle sensors on 64% *of the links.*

Motivated by the work of Hu et al. (2009), Castillo et al. (2010) and Castillo et al. (2011) present some theorems and algorithms to solve the observability problem. For example, Castillo et al. (2010) present a theorem and several algorithms to solve the following problems:

Problem 1: Given a subset $\mathbf{V_2^0}$ of traffic-flow observations and another subset $\mathbf{V_1^0}$ of flows on the network, determine the subset $\mathbf{V_1}$ of flows in $\mathbf{V_1^0}$ that can be calculated from those in $\mathbf{V_2^0}$ and provide formulas to perform these calculations.

Problem 2: Given a subset $\mathbf{V_1}$ of traffic flows and another subset $\mathbf{V_2^0}$ of flows on the network, determine a minimum subset $\mathbf{V_2}$ of flows in $\mathbf{V_2^0}$, such that all flows in $\mathbf{V_1}$ can be calculated from those in $\mathbf{V_2}$, and provide the formulas to perform these calculations.

Path (by nodes)	Link number													
	1	2	3	4	5	6	7	8	9	10	11	12	13	14
$1-3-6-8$	1	0	0	0	0	1	0	0	0	0	1	0	0	0
$1-3-5-7-8$	1	0	0	0	1	0	0	0	0	1	0	0	1	0
$1-4-7-8$	0	1	0	0	0	0	0	1	0	0	0	0	1	0
$1-3-6-9$	1	0	0	0	0	1	0	0	0	0	0	1	0	0
$1-3-5-6-9$	1	0	0	0	1	0	0	0	1	0	0	1	0	0
$1-4-7-9$	0	1	0	0	0	0	0	1	0	0	0	0	0	1
$2-4-5-6-8$	0	0	0	1	0	0	1	0	1	0	1	0	0	0
$2-4-7-8$	0	0	0	1	0	0	0	1	0	0	0	0	1	0
$2-3-6-8$	0	0	1	0	0	1	0	0	0	0	1	0	0	0
$2-4-5-6-9$	0	0	0	1	0	0	1	0	1	0	0	1	0	0
$2-4-7-9$	0	0	0	1	0	0	0	1	0	0	0	0	0	1
$2-3-6-9$	0	0	1	0	0	1	0	0	0	0	0	1	0	0

Table 5.1: Link-path incidence matrix of the parallel highway network.

Path (by nodes)	Link number													
	1	2	3	4	5	6	7	8	9	10	11	12	13	14
$1-3-6-8$	**1**	**0**	**0**	**0**	**0**	1	**0**	0	**0**	0	**0**	1	**0**	0
$1-3-5-7-8$	**0**	**1**	**0**	**0**	**0**	0	**0**	1	**0**	0	**0**	0	**0**	1
$1-4-7-8$	**0**	**0**	**1**	**0**	**0**	1	**0**	0	**0**	0	**0**	1	**0**	0
$1-3-6-9$	**0**	**0**	**0**	**1**	**0**	0	**0**	1	**0**	0	**0**	0	**0**	1
$1-3-5-6-9$	**0**	**0**	**0**	**0**	**1**	-1	**0**	0	**0**	1	**0**	-1	**0**	1
$1-4-7-9$	**0**	**0**	**0**	**0**	**0**	0	**1**	-1	**0**	1	**0**	0	**0**	0
$2-4-5-6-8$	**0**	**0**	**0**	**0**	**0**	0	**0**	0	**1**	-1	**0**	1	**0**	-1
$2-4-7-8$	**0**	**0**	**0**	**0**	**0**	0	**0**	0	**0**	0	**1**	-1	**0**	0
$2-3-6-8$	**0**	**0**	**0**	**0**	**0**	0	**0**	0	**0**	0	**0**	0	**1**	-1
$2-4-5-6-9$	**0**	**0**	**0**	**0**	**0**	0	**0**	0	**0**	0	**0**	0	**0**	0
$2-4-7-9$	**0**	**0**	**0**	**0**	**0**	0	**0**	0	**0**	0	**0**	0	**0**	0
$2-3-6-9$	**0**	**0**	**0**	**0**	**0**	0	**0**	0	**0**	0	**0**	0	**0**	0

Table 5.2: RREF of link-path incidence matrix of the parallel highway network.

As it can be seen, these problems refer not only to link flow, but to any kind of flow (link, OD, node, etc.). The authors consider that all flows can be written in terms of route flows and work with the path-link incidence matrix, path-OD incidence matrix, etc. A very interesting theorem, which is the basic tool for the proposed algorithms, is demonstrated that states the necessary and sufficient conditions for a subset of flows to be linearly dependent on another subset of flows.

Theorem 13 (Observability theorem) *Assume that the column matrices* $\mathbf{V_1}$ *and* $\mathbf{V_2}$ *are a subset of unobserved flows and a subset of observed flows, respectively, which can be written in terms of route flow columns* $\mathbf{R_1}$ *and* $\mathbf{R_2}$*. More precisely, assume that:*

$$\left(\begin{array}{c}\mathbf{V_1}\\ \hline \mathbf{V_2}\end{array}\right) = \left(\begin{array}{c|c}\mathbf{A_{11}} & \mathbf{A_{12}}\\ \hline \mathbf{A_{21}} & \mathbf{A_{22}}\end{array}\right)\left(\begin{array}{c}\mathbf{R_1}\\ \hline \mathbf{R_2}\end{array}\right), \tag{5.1}$$

where $\mathbf{A_{11}}, \mathbf{A_{12}}, \mathbf{A_{21}}$ *and* $\mathbf{A_{22}}$ *are the corresponding matrices of coefficients, and* $\mathbf{R_1}$ *and* $\mathbf{R_2}$ *are a partition of all route flows.*

If $\mathbf{A_{21}}$ *is invertible, then the necessary and sufficient conditions for the flows in* $\mathbf{V_1}$ *to be observable in terms of* $\mathbf{V_2}$ *is that*

$$\mathbf{A_{11}A_{21}^{-1}A_{22} - A_{12} = 0}, \tag{5.2}$$

where $\mathbf{0}$ *is the zero matrix, and this condition guarantees that the flows in* $\mathbf{V_1}$ *can always be calculated in terms of the flows in* $\mathbf{V_2}$ *as*

$$\mathbf{V_1 = A_{11}A_{21}^{-1}V_2}. \tag{5.3}$$

Proof. *Sufficiency.* We assume that $\mathbf{A_{21}}$ is invertible and that (5.1) and (5.2) hold. In this case, from (5.1) we have

$$\mathbf{V_1} = \mathbf{A_{11}R_1 + A_{12}R_2} \tag{5.4}$$

$$\mathbf{V_2} = \mathbf{A_{21}R_1 + A_{22}R_2} \tag{5.5}$$

and if $\mathbf{A_{21}}$ is invertible, from (5.5), we have

$$\mathbf{R_1 = A_{21}^{-1}V_2 - A_{21}^{-1}A_{22}R_2} \tag{5.6}$$

and replacing into (5.4) and considering (5.2), we get

$$\mathbf{V_1 = A_{11}A_{21}^{-1}V_2}. \tag{5.7}$$

Sufficiency. We assume that $\mathbf{V_1}$ is observable in terms of $\mathbf{V_2}$, $\mathbf{A_{21}}$ is invertible and (5.1) holds. If $\mathbf{V_1}$ is observable in terms of $\mathbf{V_2}$, then there exists a matrix $\mathbf{K}$ such that

$$\mathbf{V_1 = KV_2} \tag{5.8}$$

and from (5.4) and (5.5), we obtain

$$\mathbf{V_1 = A_{11}R_1 + A_{12}R_2 = KV_2 = K(A_{21}R_1 + A_{22}R_2)} \tag{5.9}$$

that is

$$\mathbf{(KA_{21} - A_{11})R_1 + (KA_{22} - A_{12})R_1 = 0} \tag{5.10}$$

which, due to the fact that the route flows are linearly independent, implies

$$\mathbf{KA_{21}} = \mathbf{A_{11}} \qquad \mathbf{KA_{22}} = \mathbf{A_{12}}. \tag{5.11}$$

If $\mathbf{A_{21}}$ is invertible, then from (5.11), we get $\mathbf{K} = \mathbf{A_{11}A_{21}^{-1}}$, and finally, using (5.11), (5.2) again leads to

$$\mathbf{A_{11}A_{21}^{-1}A_{22} - A_{12}} = \mathbf{KA_{22} - A_{12}} = \mathbf{0}. \tag{5.12}$$

■

Theorem 13 can be used to test observability when solving Problems 1 and 2 stated previously, as condition (5.2) provides the necessary and sufficient conditions for the flows in $\mathbf{V_1}$ to be observable in terms of $\mathbf{V_2}$. Based on this result, Castillo et al. (2010) develop different algorithms to solve the observability problem and show some example of application. Among this examples, the authors apply their methods to the problem stated by Hu et al. (2009), leading to the same results.

Castillo et al. (2011) also propose several algorithms to solve some observability problems. In this case, they present two algorithms to solve the observability problem on the basis of counted links and scanned links, respectively. The second algorithm is based on the information contained in scanned data.

The authors state that the first algorithm is equivalent to Hu et al. (2009)'s reduced row echelon form methods but, in addition to determining the subset of independent link flows, it allows determining the subset of observable routes and OD-pairs. It is based on the pivoting strategy explained in Castillo et al. (2008c).

Castillo et al. (2011) apply the methods to several networks and, in particular, to the parallel highway network obtaining the same results in terms of link flows as Hu et al. (2009) and Castillo et al. (2010). Nevertheless, their method offers information about the observable routes and OD-pairs. In the example, it is shown that no OD-pais flows are observable in terms of the link flows $\{w_1, w_2, w_3, w_4, w_5, w_7, w_9, w_{11}, w_{13}\}$ but route flows r_2 and r_5 are. In particular, the algorithm permits obtaining these route flows as: $r_2 = w_5 + w_7 - w_9$ and $r_5 = w_9 - w_7$.

As noted by Castillo et al. (2011), the methods proposed by Hu et al. (2009), Castillo et al. (2010) and themselves have one possible limitation: they require full path enumeration, which implies that they are hardly applicable for large-sized networks. Ng (2012) analyzes the contributions of those works and argues that:

1. The problem is particularly elegant because of its assumption-free character. More specifically, it does not impose any assumptions on the O-D matrices, turning ratios or route choice behavior.

2. The approach used relies on full path enumeration and this is infeasible in real-world networks, unless we resort to simplifying assumptions such as "most likely paths" destroying the assumption-free nature of the problem.

3. A node-based approach can be used to solve the problem.

4. The conjecture made in Hu et al. (2009) that "there may be an upper bound on the number of basis links that is governed by the network topology irrespective of the total number of links in the network" does not hold.

5. This upper bound is $m-n$, where m and n are the number of links and non-centroid nodes, respectively, which is dependent on the total number of links and nodes but not on the network topology.

Ng (2012) proposes to solve the link observability problem making use of the modified node-link incidence matrix $\mathbf{A}$. Firstly, the author proposes to divide the set of nodes into centroids and non-centroids. Centroid nodes are the nodes where traffic originates/is destined to, and non-centroid nodes denote all other nodes in $\mathcal{N}$. The set of all non-centroid nodes is denoted $\mathcal{N}^*$. Furthermore, the cardinality of sets $\mathcal{N}^*$ and $\mathcal{A}$ are n and m, respectively. Finally, let $O(i)$ and $I(i)$ denote, respectively, the set of outgoing and incoming links at node $i \in \mathcal{N}$. The node-link incidence matrix $\mathbf{A}^*$ is defined as the matrix with entries given by:

$$A^*_{ij} = \begin{cases} -1 & \text{if } j \in O(i) \\ 1 & \text{if } j \in I(i) \\ 0 & \text{otherwise.} \end{cases}$$

The modified node-link incidence matrix $\mathbf{A}$ (henceforth simply referred to as node-link incidence matrix) is obtained by deleting the rows in the node-link incidence matrix $\mathbf{A}^*$ associated with the centroid nodes. The proposed node based approach is based on the notion of flow conservation at non-centroid nodes, which can be simply expressed as:

$$\mathbf{A}\mathbf{w} = 0, \tag{5.13}$$

where $\mathbf{w}$ denotes the vector of link flows. Now suppose that $\mathbf{A}$ and $\mathbf{w}$ can be partitioned into two subsets, i.e., one can find an invertible matrix $\mathbf{B}$, a matrix $\mathbf{N}$ and vectors $\mathbf{w}_B$ and $\mathbf{w}_N$ such that (5.13) can be rewritten as:

$$\begin{pmatrix} \mathbf{B} & \mathbf{N} \end{pmatrix} \begin{pmatrix} \mathbf{w}_B \\ \mathbf{w}_N \end{pmatrix} = \mathbf{0}, \tag{5.14}$$

or equivalently,

$$\mathbf{B}\mathbf{w}_B = -\mathbf{N}\mathbf{w}_N.$$

Since $\mathbf{B}$ is invertible,

$$\mathbf{w}_B = -\mathbf{B}^{-1}\mathbf{N}\mathbf{w}_N. \tag{5.15}$$

That is, if one observes $\mathbf{w}_N$ through sensor measurements, the link flows $\mathbf{w}_B$ become observable.

The following proposition, proved by Ng (2012), states that, assuming that there are at least one centroid node and $n < m$, it is always possible to partition $\mathbf{A}$ into matrices $\mathbf{B}$ and $\mathbf{N}$.

Proposition 8 (Existence of B) *It is always possible to partition* $\mathbf{A}$ *into two matrices* $\mathbf{B}$ *and* $\mathbf{N}$*, where* $\mathbf{B}$ *is an* n*-by-*n *invertible matrix.*

Proof. From the network optimization literature, it is well known that the deletion of one row from $\mathbf{A}^*$ will result in a matrix of full rank (see Ahuja et al. (1993) for details). As $\mathbf{A}$ consists of a strict subset of the rows of $\mathbf{A}^*$, all rows in $\mathbf{A}$ are linearly independent. Combining this with the fact that $rank(\mathbf{A}) \leq \min\{m, n\}$, it follows that $rank(\mathbf{A}) = n$, completing the proof. ∎

In order to find a matrix $\mathbf{B}$, Ng (2012) adopts the same technique as in Hu et al. (2009): use Gaussian elimination, put $\mathbf{A}$ in its reduced row echelon form and determine $\mathbf{B}$ *by inspection*. That is, matrix $\mathbf{B}$ is constituted by the columns in $\mathbf{A}$ that form an n-by-n identity matrix. The flows on the links associated with the columns in $\mathbf{B}$ can thus be inferred by from the sensor measurements on the remainder of the links via (5.15). It is important to remark that in Ng (2012)'s model the link flows associated with the basis matrix $\mathbf{B}$ are to be inferred from other sensor measurements, whereas in Hu et al. (2009), Castillo et al. (2011) and Castillo et al. (2010) basis links are to be equipped with sensors themselves.

Finally, Ng (2012) solves Hu et al. (2009)'s conjecture: "*there may be an upper bound on the number of basis links that is governed by the network topology irrespective of the total number of links in the network*". In the following proposition it is demonstrated that the first part of the conjecture is correct but the second part is not.

Proposition 9 (Minimum number of sensors for full observability) *In order to have full observability, at a minimum, sensors need to be installed on* $m - n$ *of the links.*

Proof. From the previous proposition, it follows that at most n links flows can be inferred from sensor measurements. Hence, at least $m - n$ link flows need to be measured. ∎

As it can be seen, Ng (2012) shows that it is possible to derive an explicit expression for the upper bound on the number of links to be counted (proving the first part of the conjecture) and that this upper bound is dependent on n and m (disproving the second part of the conjecture).

Example 6 (The parallel highway network) *We will now apply the methods proposed by Ng (2012) to the parallel highway network in Figure 5.1. We note first that the number of links and non-centroid nodes is* 14 *and* 5*, respectively. Following Proposition 9, a total of* $14 - 5 = 9$ *links need to be measured, the same value obtained by Hu et al. (2009), Castillo et al. (2011) and Castillo et al. (2010).*

	Link number													
	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Node 3	1	0	1	0	-1	-1	0	0	0	0	0	0	0	0
Node 4	0	1	0	1	0	0	-1	-1	0	0	0	0	0	0
Node 5	0	0	0	0	1	0	1	0	-1	-1	0	0	0	0
Node 6	0	0	0	0	0	1	0	0	1	0	-1	-1	0	0
Node 7	0	0	0	0	0	0	0	1	0	1	0	0	-1	-1

Table 5.3: Node-path incidence matrix of the parallel highway network.

	Link number													
	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Node 3	**1**	**0**	1	0	**0**	**0**	1	**0**	0	-1	-1	-1	0	0
Node 4	**0**	**1**	0	1	**0**	**0**	-1	**0**	0	1	0	0	-1	-1
Node 5	**0**	**0**	0	0	**1**	**0**	1	**0**	-1	-1	0	0	0	0
Node 6	**0**	**0**	0	0	**0**	**1**	0	**0**	1	0	-1	-1	0	0
Node 7	**0**	**0**	0	0	**0**	**0**	0	**1**	0	0	0	0	-1	-1

Table 5.4: RREF of the node-path incidence matrix of the parallel highway network.

Tables 5.3 and 5.4 show the modified node-link incidence matrix and its RREF, respectively. It can be seen that the columns in $\mathbf{A}$ *associated with links* $1, 2, 5, 6$ *and* 8 *constitute a possible* $\mathbf{B}$ *since their corresponding columns in the Table 5.4 form a* 5*-by-*5 *identity matrix (shown in bold). The flows on these links can thus be inferred from the sensor measurements on the remainder of the links via (5.15).*

Finally, it should be noted that the resulting links without sensors do not coincide with those in Hu et al. (2009) (their solution determines that sensors are not installed on links $6, 8, 10, 12$ *and* 14*). However, as matrix* $\mathbf{B}$ *is not unique, the same solution can be reached using the same modeling approach but rearranging the columns in* $\mathbf{A}$*, placing the columns associated with the links obtained by Hu et al. (2009) at the left most positions.*

Appendix

A Notation

β_{ijks}	proportion of trips in the OD pair ks using link ℓ_{ij}.
$\mathbf{\Delta}$	link-path incidence matrix.
ℓ_{ij}	link joining nodes i and j.
$\mathcal{A}$	set of links.
$\mathbf{A}$	modified node-link incidence matrix.
$\mathbf{A}^*$	node-link incidence matrix.
$\mathbf{A_{11}}$	matrix of coefficients.
$\mathbf{A_{12}}$	matrix of coefficients.
$\mathbf{A_{21}}$	matrix of coefficients.
$\mathbf{A_{22}}$	matrix of coefficients.
A^*_{ij}	elements of matrix $\mathbf{A}^*$.
$\mathcal{B}$	assignment proportion matrix.
$\mathbf{B}$	invertible matrix.
i	link begin node.
$I(i)$	set of incoming links at node i.
j	link end node.
k	origin node.
$\mathbf{K}$	generic matrix.
ks	origin-destination pair from nod k to node$s$.
m	cardinality of $\mathcal{A}$.
n	cardinality of $\mathcal{N}^*$.
$\mathcal{N}$	set of nodes.
$\mathcal{N}^*$	set of non-centroid nodes.
$\mathbf{N}$	generic matrix.
$O(i)$	set of outgoing links at node i.
r	route.
$\mathbf{R_1}$	partition of all route flows.
$\mathbf{R_2}$	partition of all route flows.
r_r	flow through route r.
s	destination node.
$\mathbf{V_1}$	set of traffic flows.
$\mathbf{V_1^0}$	set of traffic flows.
$\mathbf{V_2}$	set of traffic flows.
$\mathbf{V_2^0}$	set of traffic flows.
$\mathbf{w}$	vector of link flows.
$\mathbf{w_B}$	vector of link flows.

$w_{\ell_{ij}}$	flow through link ℓ_{ij}.
$\mathbf{w_N}$	vector of link flows.
z	rank of the link-path incidence matrix.

Chapter 6

Dynamic traffic models

Contents

6.1 Introduction

Dynamic traffic models (also referred to as dynamic traffic assignment) consist of describing the time-varying traffic flows on each road of a traffic network. They can be seen as a generalization of the static traffic assignment problem dealt with in Chapter 3 when given a set of time-varying origin-destination demands is known. They are a very useful tool in the development of real time traffic control techniques, as they consider the traffic flow patterns over time and space.

These problems present two components: (i) the travel choice principle (see, for example, Janson (1991), Smith (1979), Friesz et al. (1993), Lo and Szeto (2002), Ban et al. (2008) or Ran et al. (1996)) and (ii) the traffic-flow component (see, for example, Hopf (1950) or Friesz et al. (2011)). The travel choice principle models travelers' propensity to travel, and if so, how they select their routes, departure times, modes or destinations. Contrary, the traffic-flow component depicts how traffic propagates on a transport network and hence governs the network performance in terms of travel time.

The commonly adopted travel choice principle is the dynamic extension of Wardrop (1952)'s first principle called the Dynamic User Optimal (or Dynamic User Equilibrium) principle. However, under certain assumptions, a solution under the DUO principle may

not exist and other principles, such as the tolerance-based DUO principle where the travel time of used routes is required to be within a tolerance from the minimum OD travel time, must be applied (see Szeto (2003) or Szeto and Lo (2006)). Some authors aim at finding only the used routes (Lo and Szeto (2002) or Lam and Huang (1995)) whereas others propose simultaneous route and departure time choice models (Szeto and Lo (2004), Bernstein et al. (1993) or Huang and Lam (2002)).

Several types of approaches have been adopted to model the DUO problem: mathematical program-based (see Janson (1991) or Janson and Robles (1995)), variational inequality-based (see Ran et al. (1996) or Friesz et al. (1993)), and optimal control-based (see Ran et al. (1993)) approaches.

Among the traffic-flow component models, also called Network Loading Problem (NLP), we can distinguish between two categories: the simulation-based and the analytical-based approaches. The simulation-based approach emphasizes each individual driver's behavior and hence the microscopic traffic flow characteristics (see Chandler et al. (1958), Gazis et al. (1961) or Wagner et al. (1996)). One advantage of the simulation approach is its detailed description of traffic phenomena. However, it implies a great computational effort and lacks well-defined solution properties, as it sheds no light on the relationship between user choices and path flow times and volumes.

The analytical-based approach concerns the average driver's behavior, and is essentially macroscopic. This approach has well-defined properties in terms of optimal conditions. In this group we can mention the following models: the hydrodynamic, the Merchant-Nemhauser, the cell transmission and the point and physical queue models. This chapter is devoted to the analytical network loading problem.

On the other hand, dynamic models can be classified as discrete and continuous, depending on the way space and time are dealt with. Since in reality both magnitudes are continuous, those models that consider the traffic evolution as continuous functions (e.g, the hydrodynamic model) are the most adequate. However, they are also the most complex. In order to simplify the models, in many cases they are discretized in time, in space or in both variables (e.g. the cell transmission model). A very interesting way of treating this problem is proposed by Castillo et al. (2012), where they work with discretized times but make use of monotone splines in order to obtain continuous results. The rationale behind the use of monotone splines is that the FIFO (First In First Out) condition is ensured.

In summary, existing analytical and continuous type methods dealing with dynamic traffic flow lead to complicated problems that involve a lot of computations and make them not very appropriate for real networks. Thus, discrete or simplified versions of these models have been suggested.

The remaining of the chapter is structured as follows. In Section 6.2 we present some basic concepts that are necessary to treat the dynamic traffic approach. In addition, we define the FIFO rule, providing a mathematical way to observe this principle. In Section

6.3 we review some of the network loading models proposed in the literature, focussing on the cell transmission and the point and physical queue models.

6.2 Dynamic traffic flow concepts

This section describes some basic concepts needed for a better understanding of dynamic traffic models.

Let $u_{ij}(t)$ and $v_{ij}(t)$ be the inflow and outflow rates for link ℓ_{ij} at time t, respectively, and $U_{ij}(t)$ and $V_{ij}(t)$ their corresponding cumulative variables. By definition, it holds that:

$$U_{ij}(t) = \int_0^t u_{ij}(s)ds, \qquad \text{or} \qquad u_{ij}(t) = \frac{\partial U_{ij}(t)}{\partial t}, \tag{6.1}$$

$$V_{ij}(t) = \int_0^t v_{ij}(s)ds, \qquad \text{or} \qquad v_{ij}(t) = \frac{\partial V_{ij}(t)}{\partial t}. \tag{6.2}$$

The **flow conservation condition** requires the number of vehicles on a link (link occupancy) at a particular time to be equal to the total inflow at the entry of that link at that time minus the corresponding total outflow at the exit. Mathematically,

$$x_{ij}(t) = U_{ij}(t) - V_{ij}(t), \tag{6.3}$$

where $x_{ij}(t)$ is the number of vehicles on link ℓ_{ij} at time t. By taking derivatives to both sides of Equation (6.3) we get the following alternative expression:

$$\frac{\partial x_{ij}(t)}{\partial t} = u_{ij}(t) - v_{ij}(t). \tag{6.4}$$

Assuming that vehicles leave links in the same order as they enter (i.e., the FIFO rule is assumed), we can derive the link travel times of each vehicle by means of the cumulative inflows and outflows. The vehicle entering link ℓ_{ij} at time t_1 exits this link at time t_2 if and only if

$$U_{ij}(t_1) = V_{ij}(t_2). \tag{6.5}$$

As no overtaking is allowed due to the FIFO condition, the link travel time of a vehicle is equal to the link exit time minus the corresponding entry time. Mathematically,

$$U_{ij}(t) = V_{ij}(t + \tau_{ij}(t)), \text{ or} \tag{6.6}$$

$$\tau_{ij}(t) = V_{ij}^{-1}\left[U_{ij}(t)\right] - t, \tag{6.7}$$

where $\tau_{ij}(t)$ is the travel time on link ℓ_{ij} at time t. Taking derivatives of (6.7) and rearranging gives:

$$v_{ij}(t + \tau_{ij}(t)) = \frac{u_{ij}(t)}{1 + \dfrac{d\tau_{ij}(t)}{dt}}. \tag{6.8}$$

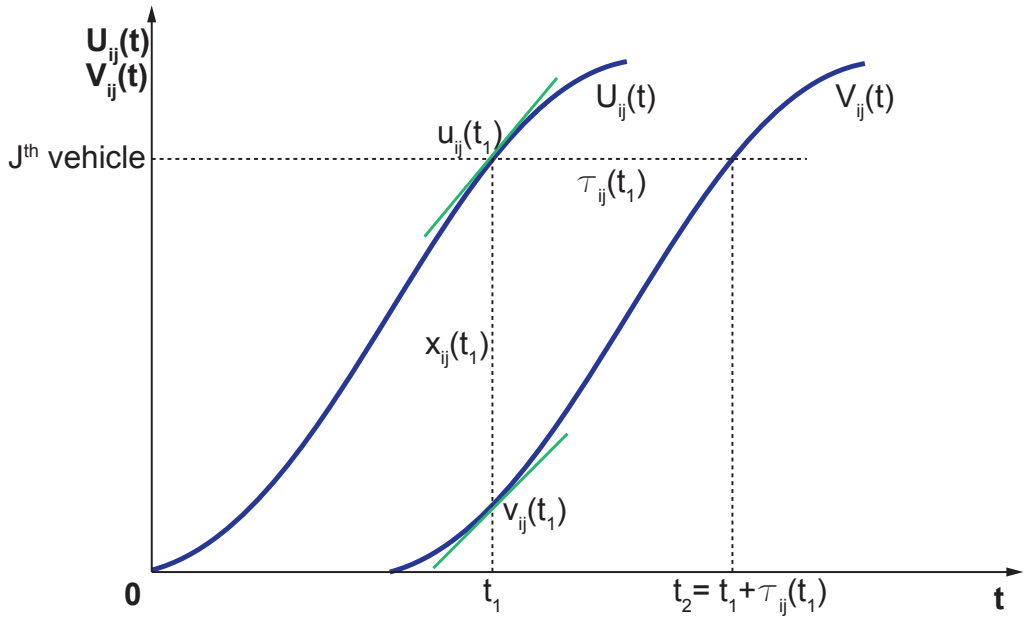


Figure 6.1: Relationships among inflow rate (u_{ij}), outflow rate (v_{ij}), travel time (τ_{ij}), occupancy (x_{ij}), cumulative inflow (U_{ij}) and cumulative outflow (V_{ij}).

Equations (6.7) and (6.8) show the relationship among inflow rate, outflow rate and travel time and is known as **time-flow consistency condition** or flow propagation condition.

Figure 6.1 shows the relationships explained throughout equations (6.1)-(6.8). In particular, the vertical distance between the cumulative inflow and outflow curves gives the number of vehicles in the link at that particular moment. Furthermore, the horizontal distance between both cumulative curves is the link travel time of a vehicle entering the link at that precise moment.

Most traffic models assume flow conservation and time-flow consistency conditions but those are not the only considerations made on the traffic flow component. FIFO, causality and queue spillback are three important properties also proposed in the literature.

6.2.1 Causality, FIFO rule and queue spillback

Causality refers to the property that the link travel times for traffic entering at time t_1 only depend on the traffic entering at time $t_2 \leq t_1$. This property means that the speed and travel time of a vehicle on a link is affected by the speed of vehicles ahead but not by vehicles behind.

FIFO (First In First Out) rule (at the link or path level) means that users who enter a link (or path) earlier will leave it sooner. The FIFO rule is also defined at the

origin-destination level. In that case, OD FIFO is satisfied if users on the same OD pair who depart the origin earlier, reach the destination sooner. Link FIFO can prevent unrealistic situations such as the fast traffic *jump over* the preceding slow traffic (Carey and Subrahmanian (2000)). Although FIFO does not allow any realistic overtaking on a microscope level, in reality, road traffic tends to behave in a FIFO manner. Traffic which embarks on a road first will *on average* exit first (Carey (1992)). In particular, on a single-lane road and in a queue, no overtaking can be occurred and capturing FIFO is a mandatory.

The FIFO condition can be stated mathematically in several ways. In the following lines, we explain how it can be observed in terms of the link travel time $\tau_{ij}(t)$.

Theorem 14 *The FIFO rule is equivalent to (Carey (2004)):*

$$\tau'_{ij}(t) > -1. \tag{6.9}$$

Proof. We start by demonstrating that equation (6.9) is necessary. Let $\omega > 0$, then the FIFO condition for two vehicles that enter the link at times t and $t+\omega$, respectively, in terms of link exit times can be written as:

$$t + \tau_{ij}(t) < t + \omega + \tau_{ij}(t+\omega), \tag{6.10}$$

that is,

$$\tau_{ij}(t+\omega) - \tau_{ij}(t) > -\omega \Leftrightarrow \frac{\tau_{ij}(t+\omega) - \tau_{ij}(t)}{\omega} > -1 \Leftrightarrow \tau'_{ij}(t) = \lim_{\omega \to 0} \frac{\tau_{ij}(t+\omega) - \tau_{ij}(t)}{\omega} > -1. \tag{6.11}$$

Condition (6.9) is sufficient too. If $\tau'_{ij}(t) > -1$, then we have

$$\tau_{ij}(t+\omega) = \tau_{ij}(t) + \int_t^{t+\omega} \tau'_{ij}(s)ds > \tau_{ij}(t) + \int_t^{t+\omega} -1 ds = \tau_{ij}(t) - \omega, \tag{6.12}$$

which implies (6.10). ∎

FIFO rule is an important condition imposed and studied in a large number of works. In particular, Friesz et al. (1993) prove that the FIFO condition is satisfied whenever the arc traversal time functions are affine[1]. Carey (2004) demonstrates that causality, time-flow consistency and FIFO are independent, i.e., capturing one in the model does not imply capturing the other.

Finally, **queue spillback** (or junction blockage) refers to the end of queue spilling backwards in the network. When the queue spills backward passing the junction, the

[1] A function is affine if it is of the form:

$$f(x) = ax + b, \quad a, b \geq 0.$$

junction gets blocked. When the streets are short and the demand for them is high, queue spillback often occurs and this phenomenon must be captured in the models.

Although the five considerations explained (flow conservation, time-flow consistency, causality, FIFO and queue spillback) are important to be captured in the dynamic traffic models, only flow conservation (expressed in Equation (6.3)) is considered in every traffic model.

6.3 The Network Loading Problem

As already explained, the network loading problem consists of determining, given some path departure rate functions, the arc and path traversal functions. Note that the time-dependent flows in every path are known and, thus, the path flows do not depend on the costs resulting from the solution to the NLP.

The analogies of fluid and gas dynamics with traffic flow has led engineers to model traffic flow using well known equations from hydraulics leading to macroscopic models for traffic simulation. More precisely, we can assume that the links are channels and that the traffic flow is replaced by a fluid (see Lax (1954), Lighthill and Whitham (1955), Richards (1956)). The key postulate of these models (also referred as to LWR) is that there exists some functional relation between the flow q and the density (or concentration) k, where the flow is defined as the rate at which vehicles pass some point and the density is the number of vehicles per unit length of road. It is also assumed that the following conservation equation (also called equation of continuity) holds

$$\frac{\partial k(x,t)}{\partial t} + \frac{\partial q(x,t)}{\partial x} = 0.$$

The LWR model is based on partial differential equations that follow from the equation of continuity. As noted by Newell (1993a) and Daganzo (1994), the solution to the differential equations is tedious and can lead to multiple solutions. Moreover, it does not take into account the downstream occupancy.

An important alternative to these models is in the Newell's kinematic wave trilogy Newell (1993a,b,c), in which instead of using the Lighthill-Whitham-Richards theory to evaluate flows or densities, the cumulative flow past any point at any time is evaluated. The proposed model studies the variation of traffic flow at one end of the link from the behavior of traffic at the other end, without evaluating the behavior at intermediate points. The main advantage of this model is that it requires less computer memory.

Due to the fact that fluid-based models have important limitations, some modifications proposed by several authors, led to complex models that involved well known equations, such as the Burgers (see Whitham (1974)), the Boltzmann (see Prigogine and Herman (1971) or Paveri Fontana (1975)) and the Navier-Stokes like equations (see Stokes (1845)). A complete state of the art corresponding to the initial and most active period of macroscopic models is provided by Helbing (1996).

However, since very complex models are required to reproduce the real and complicated traffic flow behavior and there are no analytical solutions for the general Navier-Stokes equation, simplified versions of the Navier-Stokes equation or complicated numerical methods must be used to deal with traffic problems. The final consequence is that they are not practical for real traffic networks.

Merchant and Nemhauser (1978a,b) proposed a model based on an exit-flow function that determines the share of the number of users leaving the link during a time interval. Their model was formulated as a discrete time, non linear and nonconvex mathematical programming problem.

The cell transmission model (see Daganzo (1994, 1995)) was developed as a discrete approximation to the hydrodynamic theory of traffic flow to reproduce some observed traffic situations, such as the processes of initiation, propagation, and dissipation of physical-queues that occur in real practice. The cell transmission model (CTM) seems to strike an appropriate balance between capturing sufficient details for modeling queue dynamics while leaving out microscopic features that would slow down computation. Some important extensions of the cell transmission model including the assignment problem are in Lo (1999), Lo and Szeto (2002), Szeto and Lo (2004) and Szeto and Lo (2006).

The queue models (point or physical) allow us to simulate the delay due to link saturation. Moreover, physical queue models include the shock-wave effect on this delay (Carey and Srinivasan (1993), Huang and Lam (2002) and Szeto and Lo (2005)).

In the following subsections the cell transmission and some point and physical models will be described more in detail.

6.3.1 The Cell Transmission model

The Cell Transmission model, which was proposed by Daganzo (1994, 1995), consists of evaluating flow at a finite number of carefully selected intermediate points of the links. As in Newell's model, the difference equations that form the basis for this procedure are discrete approximations to the differential equations of the hydrodynamic theory, as it is proved by the author.

Under the CTM, the road is divided into homogeneous sections (or cells) and time into intervals such that the cell length is equal to the distance traveled by free-flowing traffic in one time interval. If, for the sake of simplicity, we assume that we deal with a single road[2] and, hence, the cells can be numbered consecutively starting with the upstream end of the road, the system's evolution obeys:

$$n_c(t+1) = n_c(t) + y_c(t) - y_{c+1}(t), \tag{6.13}$$

where $n_c(t)$ is the number of vehicles in cell c at time t and $y_c(t)$ is the inflow of cell c at

[2]The results presented can be extended to more complex traffic networks following the ideas in Daganzo (1995).

time t and is given by:

$$y_c(t) = \min\{n_{c-1}(t), Q_c(t), N_c(t) - n_c(t)\}, \tag{6.14}$$

where $N_c(t)$ is the maximum number of vehicles that can be present in cell c at time t and $Q_c(t)$ is the maximum number of vehicles that can flow into cell c when the clock advances from t to $t + 1$[3].

Equation (6.13) permits calculating the number of vehicles in cell c at time $t + 1$ as the cell occupancy of that cell at time t, plus the inflow minus the outflow.

Moreover, (6.14) expresses that the number of vehicles that can flow from cell $c - 1$ to cell c for the time interval after t is the smallest quantity among: (i) the number of vehicles in cell $c - 1$ at time t ($n_{c-1}(t)$), (ii) the capacity flow into cell c for time interval t ($Q_c(t)$), and (iii) the amount of empty space in cell c at time t ($N_c(t) - n_c(t)$). This equation permits incorporating queueing in the model as queues would form whenever a cell has reached its capacity and no more vehicles can flow into it.

The occupancy constraint $y_c(t) \leq N_c(t) - n_c(t)$, derived from equation (6.14), is conservative because it assumes that no vehicles leave cell c at time t. This is equivalent to assume that density waves propagate backwards at free flow speed, as the time intervals are defined by means of the free flow speed. This is somewhat unrealistic because in reality waves move much more slowly than free flow traffic, changing the manner in which vehicles approach the bottleneck. Taking this into account, Daganzo proposes an extension to the original CTM model such that Equation (6.14) is replaced by:

$$y_c(t) = \min\{n_{c-1}(t), Q_c(t), \kappa[N_c(t) - n_c(t)]\}, \tag{6.15}$$

where $\kappa = 1$ if $n_{c-1}(t) \leq Q_c(t)$ and $\delta = w/v$, otherwise, with v and w being the free flow speed and the backward wave speed (the speed with which disturbances propagate backwards when traffic is congested), respectively.

The cell transmission model as recently defined offers four degrees of freedom: the free flow speed, the maximum flow, the jam density and the wave speed. As pointed by several authors, these are the most important determinants of traffic evolution and it is very unlikely that in any practical application an engineer would have reliable data beyond these parameters. Therefore, the CTM is a sufficiently flexible model that permits modeling a wide variety of traffic networks.

Example 7 *Finally, in order to illustrate the cell transmission model, we will show an example taken from Daganzo (1994). We consider a* 1.25 *mile homogeneous road with free flow speed of* 50 *mph and maximum flow capacity of* 3000 *vph. If we choose a 6 second clock tick (*$1/600^{th}$ *of an hour), then the length of a cell must be* 1/12 *mile and there will be* 15

[3]Note that constants $N_c(t)$ and $Q_c(t)$ are allowed to vary with time to be able to model traffic incidents that would reduce the capacity of the cells.

cells. The cell constants are: $N = 15$ *and* $Q = 5$*. Initially, each cell contains four vehicles* ($n_c(0) = 4,\ \forall c$) *and a constant flow of four vehicles enters the road* ($Q_0(t) = 4,\ \forall t$).

In order to see how the CTM models the queue spillback phenomena, we will assume that an incident occurs in cell 11 *at the beginning of the period in study lasting two minutes. This incident effectively reduces flow capacity to* 20% *of the maximum and is modeled by limiting the capacity of that cell for the first two minutes:* $Q_{11}(t) = 1,\ t \leq 20$.

With this information, Equations (6.13) and (6.14) are solved iteratively and the results are presented in the upper graph of Figure 6.2, where the cell occupancy levels along time are shown. Traffic propagates in the direction of vertical axis, whereas the horizontal axis is for time. The intensities of the shades correspond to the occupancy levels, as shown in the legend on the right side of the plot. It can be seen that a queue starts forming in cell 11 *at the beginning of the period and spills back through the upstream cells until the incident is over* ($t = 20$). *At that moment, the queue spills back as more vehicles can leave the conflictive cell. It is interesting to note the low density in the downstream cells during the incident as only one vehicle can leave cell* 11 *at each clock tick.*

Finally, the results obtained from the cell transmission model are compared with those resulting from the simplified CTM (SCTM), which ignores the storage capacity term in the CTM (i.e., $N = \infty$). *The SCTM can be seen as a point queue model, which will be further studied in the following subsection, as it assumes that the queue has no length*[4] *and, hence, it cannot spill backwards. The results obtained from the SCTM can be found in the lower graph of Figure 6.2. A very high congestion can be found on cell* 11*, due to the fact that there are more vehicles entering than exiting the cell. This congestion gets worse up to time* 20 *when the incident disappears and the situations starts improving. It is important to note that no queue is generated under the point queue paradigm but the low congestion in the downstream cells during the incident is still present.*

From these figures, one can see how introducing storage capacity affects the occupancies over time and queueing locations. The point-queue paradigm does not consider storage capacity when queues form, and hence cannot capture queue spillback, whereas the physical-queue paradigm considers that effect and hence queue spillback.

6.3.2 Point and Physical Queue models

In this subsection we will focus on the point and physical queue models[5]. In the previous example (Example 7) two versions of the cell transmission model with both kinds of queues have been solved in order to show the main differences between them. As already explained, the point-queue representation treats vehicles as points without physical lengths,

[4] Actually, the model assumes that the cells have infinity capacity and, thus, all vehicles can enter the link avoiding queues in the upstream cells.

[5] Note that the Cell Transmission Model and the Simplified CTM can also be classified as physical and point queue models, respectively.

whereas the physical-queue representation considers the vehicle lengths. Therefore, only the physical-queue representation can capture junction blockage and queue spillback.

- In the point queue model there exist three general formulations, namely, (a) the exit flow function approach (e.g., Carey and Srinivasan (1993), Lam and Huang (1995)), where the inflow rates and occupancies are given and the outflow rates are determined by the flow conservation condition (6.4) according to the predefined exit flow function and then, the travel time is calculated using the flow propagation condition (6.7); (b) the travel time function approach (e.g., Xu et al. (1999), Huang and Lam (2002)), where we determine the travel times based on the flow conservation (6.4) and the travel times functions, given the link inflows and occupancies and then, the outflow rate can be obtained by the flow propagation condition (6.7); and (c) the mixed approach (e.g., Yang and Huang (1997)), where we require both predefined travel time functions and exit flow functions, the travel times and the outflow rates are determined separately by their corresponding equation and the flow conservation condition (6.4), without satisfying the flow propagation condition (6.7).

- In the physical queue model there exist two general formulations, namely, (a) the exit flow function approach (e.g., Szeto and Lo (2004)), which is similar to the exposed exit flow function approach, but considers the storage capacity in the exit flow function to capture the effects of physical queues; and (b) the combined approach (e.g., Rubio-Ardanaz et al. (2001)), which divides a link into a running segment (based on the travel time function) and a queuing segment (based on an exit flow function, which considers the downstream storage capacity), combining flow conservation and propagation conditions.

Table 6.1 shows a comparative study of the solution properties of the point and physical queue model, according to Szeto and Lo (2005, 2006).

Finally, we will give a brief explanation of one point-queue model, namely, the one proposed by Huang and Lam (2002).

A point queue model: Huang and Lam (2002)

Huang and Lam (2002) propose a network loading model with discrete time that follows the FIFO rule at intersections. The model does not consider the physical queue length effects (or spillback effects) on link capacities. It is assumed that the time needed to pass through a capacity-constrained bottleneck can be modeled as a deterministic queueing process and, then, the link queue would develop linearly when the inflow rate exceeds capacity.

Let us consider that the time period T of interest is discretized to a finite set of time intervals, $\mathcal{L} = \{l : l = 1, \ldots, L\}$. Let δ be the interval length such that $\delta L = T$. The value

Solution properties	**Point-queue problems**	**Physical-queue problems**
Causality	May or may not satisfy causality, depending on the choice of travel time or exit flow functions.	Obey causality
Link FIFO	May or may not satisfy Link FIFO, depending on the choice of travel time or exit flow functions.	May or may not satisfy Link FIFO, depending on whether addition variables are introduced to capture Link FIFO.
Route FIFO	Satisfy Route FIFO if they satisfy Link FIFO.	
OD FIFO	Satisfy this property under the DUO condition and certain assumptions, but not satisfy under the Stochastic DUO conditions.	
Continuity w.r.t. route flows	Continuous under mild assumptions.	Possibly discontinuous.
Monotonicity w.r.t. route flows	Usually non-monotonic.	
Differentiability w.r.t. route flows	Differentiable under differentiable link travel time functions and non-differentiable under continuous exit flow functions.	Possibly non-differentiable.
Continuity of OD travel time w.r.t. demands	Continuous under mild assumptions	Possibly discontinuous.
Solution Existence	Must exist.	May not exist.
Solution Uniqueness	Non-unique in terms of route flows and link flows.	

Table 6.1: Comparison of the properties between Point-queue and Physical-queue dynamic models.

of δ is chosen to be small enough so that the proposed discrete-time model is close to its continuous-time counterpart.

Next, the equations used in the model to solve the network loading problem are explained.

Let $U_{ij}(l)$ be the cumulative inflow of link ℓ_{ij} up to interval l. Assuming constant flow rates during each interval, we have

$$U_{ij}(l) = U_{ij}(l-1) + \delta u_{ij}(l), \quad \ell_{ij} \in \mathcal{A},\ l \in \mathcal{L}, \tag{6.16}$$

where $u_{ij}(l)$ is the inflow rate on link ℓ_{ij} during interval l.

It is also supposed that the departure rates are constant during each interval and, hence,

$$V_{ij}(l + t_{ij}(l)) = V_{ij}(l - 1 + t_{ij}(l-1)) + \delta \left[l + t_{ij}(l) - (l - 1 + t_{ij}(l-1))\right] v_{ij}(l + t_{ij}(l)), \quad \ell_{ij} \in \mathcal{A},\ l \in \mathcal{L}, \tag{6.17}$$

where $v_{ij}(l)$ denotes the departure rate from link ℓ_{ij} during interval l, $V_{ij}(l)$ denotes the cumulative departures from link ℓ_{ij} up to interval time l, and $t_{ij}(l)$ is the travel time on link ℓ_{ij} for users entering this link at interval l. Equation (6.17) represents that the users that enter the link at interval $l-1$ leave that link before the end of interval $l-1+t_{ij}(l-1)$. Moreover, the flows entering at next interval, l, will leave the link during $[l-1+t_{ij}(l-1), l+t_{ij}(l)]$ by the constant departure rate $v_{ij}(l+t_{ij}(l))$.

Under the FIFO rule, a vehicle must leave link ℓ_{ij} in the same order as its order of arrival at the link ℓ_{ij}. So, $U_{ij}(l) = V_{ij}(l+t_{ij}(l))$. This condition, together with (6.16) and (6.17), leads to:

$$u_{ij}(l) = v_{ij}(l+t_{ij}(l))\left[1+t_{ij}(l)-t_{ij}(l-1)\right]. \tag{6.18}$$

It is further assumed that at the end of each link there is a bottleneck with the maximum exit rate, s_{ij}, $\ell_{ij} \in \mathcal{A}$. Then, the travel time of traversing link ℓ_{ij} for users entering at interval l can be computed as

$$t_{ij}(l) = t_{ij}^0 + \frac{q_{ij}(l)}{\delta s_{ij}}, \quad \ell_{ij} \in \mathcal{A},\ l \in \mathcal{L}, \tag{6.19}$$

where t_{ij}^0 is the free-flow travel time of link ℓ_{ij} and $q_{ij}(l)$ is the queue size experienced by vehicles entering link ℓ_{ij} at interval l.

By applying (6.19) into (6.18), we have

$$u_{ij}(l) = v_{ij}(l+t_{ij}(l))\left[1+\frac{q_{ij}(l)-q_{ij}(l-1)}{\delta s_{ij}}\right]. \tag{6.20}$$

According to the deterministic queueing theory, the outflow rate on link ℓ_{ij} is evaluated as follows:

$$v_{ij}(l+t_{ij}(l)) = \begin{cases} s_{ij} & \text{if } t_{ij}(l) > t_{ij}^0 \text{ or } \ u_{ij}(l) > s_{ij} \\ u_{ij}(l) & \text{otherwise.} \end{cases} \tag{6.21}$$

Combining (6.20) and (6.21) with the non-negativity constraint $q_{ij}(l) \geq 0$, we get

$$q_{ij}(l) = \max\{q_{ij}(l-1) + \delta(u_{ij}(l) - s_{ij}), 0\}. \tag{6.22}$$

Finally, equation (6.22) shows that if $q_{ij}(0)$ and $u_{ij}(l)$ for all l are given[6], then the queues for all l can be obtained recursively. Therefore, the link travel times and the outflow rates for all l can be determined by (6.19) and (6.20), respectively, solving the network loading problem.

[6] $u_{ij}(l)$ for the links starting at origin nodes are given and for the remaining links can be computed as $u_{ij}(l) = v_{j'i}(l)$, where it is assumed that link $j'i$ is the link prior to link ij.

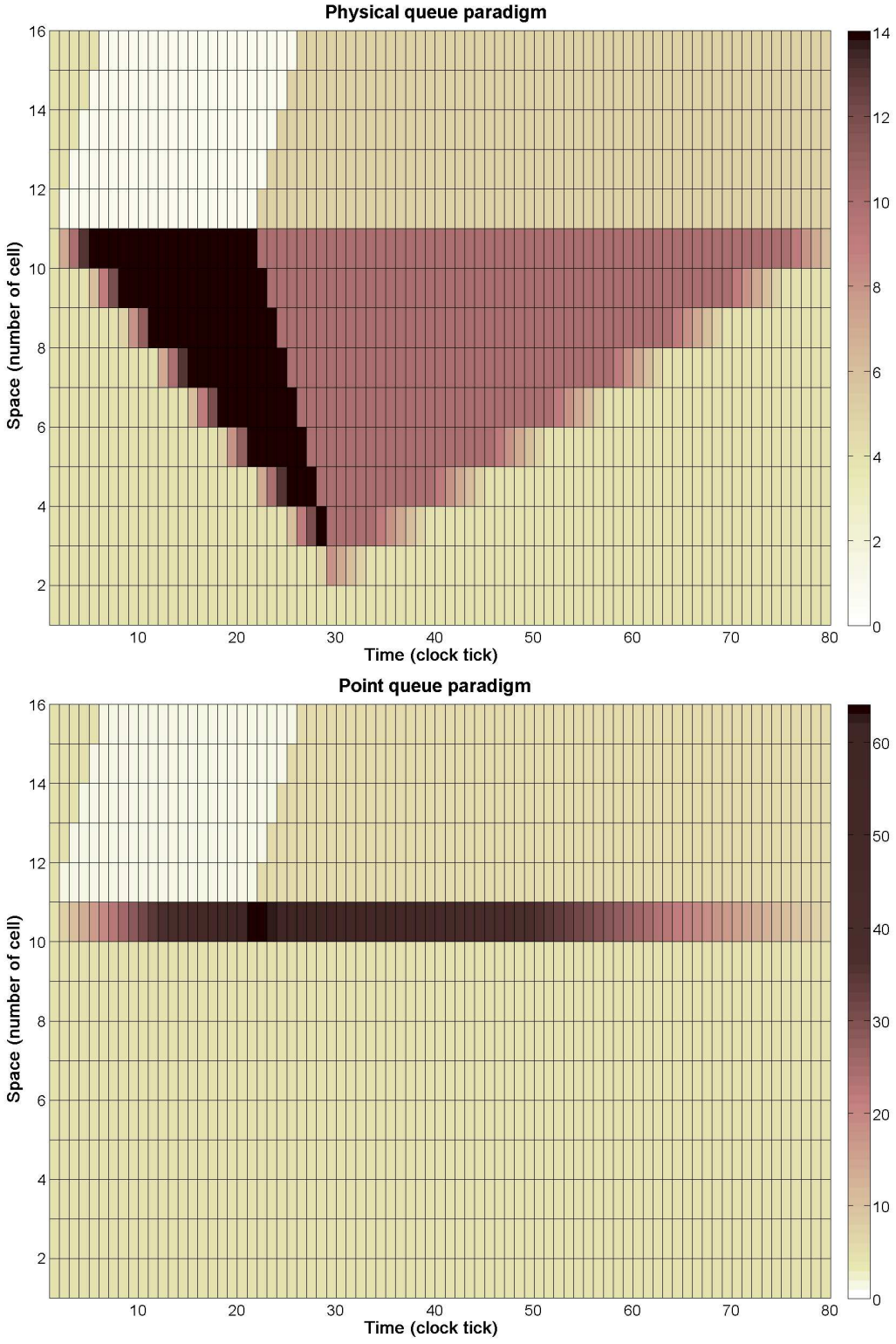


Figure 6.2: Occupancy plots under the physical and point queue cell transmission models.

Appendix

A Notation

δ	time interval length.
κ	auxiliary parameter.
ℓ_{ij}	link joining nodes i and j.
$\tau_{ij}(t)$	travel time on link ℓ_{ij} at time t.
ω	time index.
$\mathcal{A}$	set of links.
c	cell.
$\mathcal{N}$	set of nodes.
i	link begin node.
j	link end node.
k	traffic density.
l	time interval.
L	last time interval.
$\mathcal{L}$	set of time intervals.
$n_c(t)$	number of vehicles in cell c at time t.
$N_c(t)$	maximum number of vehicles that can be present in cell c at time t.
q	traffic flow.
$Q_c(t)$	maximum number of vehicles that can flow into cell c when the clock advances from t to $t+1$.
$q_{ij}(l)$	queue size experienced by vehicles entering link ℓ_{ij} at interval l.
s_{ij}	link ℓ_{ij} maximum exit rate.
t	time index.
T	time period.
t_{ij}^0	free-flow travel time of link ℓ_{ij}.
$t_{ij}(l)$	travel time on link ℓ_{ij} for users entering the link at interval l.
$u_{ij}(t)$	inflow rate for link ℓ_{ij} at time t.
$U_{ij}(t)$	cumulative inflow rate for link ℓ_{ij} at time t.
v	free flow speed.
$v_{ij}(t)$	outflow rate for link ℓ_{ij} at time t.
$V_{ij}(t)$	cumulative outflow rate for link ℓ_{ij} at time t.
w	backward wave speed.
x	space index.
$x_{ij}(t)$	number of vehicles on link ℓ_{ij} at time t.
$y_c(t)$	inflow of cell c at time t.

Part III

Original Contributions

Chapter 7

A Percentile Traffic Assignment model

Contents

7.1 Introduction

The problem of user interaction in traffic networks where travelers compete for space has been dealt with in the existing literature for several decades. The most common approach assumes that users behave in a homogeneous way. However, recently, some heterogeneous cases have arisen as is the case of the travel time reliability problem, in which different users perceive the problem from a different perspective. This occurs when travelers are concerned about reaching the destination on time because of possible consequences in terms of prestige, money losses, etc., but the perception of these consequences is not the same for all of them, so that they can be grouped in different classes. It is common to measure travel time reliability as the probability that a trip can be completed on time (for a review of traffic assignment models, including traveling time reliability ones, see Chapter 3). Consequently, users reduce the risk of late arrival and decide to start the trip with

sufficient time to guarantee a high probability of success (arrive at the target on time) according to their classes. However, making this decision is not easy because travel times are random in nature and its statistical properties are not well known.

In the existing literature there have been some approaches to solve the travel time reliability problem. Important references on this issue are Lo and Tung (2003), Lo et al. (2006), Nie and Wu (2009) or Nie (2011). To model the effects of the travelers risk averse route choice, these models assume that travelers choose routes to minimize the percentile travel time, i.e. the travel time budget that ensures their preferred probability of on-time arrival; in doing so, they drive the system to a percentile user equilibrium (PUE), which can be viewed as an extension of the classic Wardrop equilibrium, where travel times are replaced by α-percentile travel times, with α values that depend on the user class.

In order to reproduce the stochastic character of flows, some authors, such as Lo et al. (2006), provide a method in which they assume a probability distribution for the link travel time and based on the central limit theorem, they obtain route travel times as normally distributed random variables. Other authors, as Uchida and Iida (1993), assume link travel times as normally distributed and get normally distributed route travel times as a consequence. Finally, other authors, as Nie (2011), let the distribution of link travel times free and obtain route travel times by convolutions.

The problem arises because the distributional assumptions are made at the link level and then we must proceed to the route level. A good alternative consists of making a weak distributional assumption at the route level, but considering the fact that the route mean and variances can be written in terms of link means and variances, even in the case of dependent link travel times.

In this chapter we present a model in which the route travel times are assumed to belong to a location-scale family of distributions, which can be considered as a weak assumption, at least when compared with the assumption of normal random variables. So, we use neither the central limit theorem nor the assumption of normal random variables. We note that assuming satisfaction of the central limit theorem implies two inconveniences: (a) forcing the route travel time to be normally or approximately normally distributed, and (b) assuming a large number of links in all routes, which does not need to be true in real cases.

Nie (2011) proposes a gradient projection method to solve the variational formulation of the percentile UE problem. This implies that gradients, that is, partial derivatives of route travel time percentiles with respect to route flows need to be calculated. To this end he assumes that link travel times are independent random variables and suggests to calculate route travel times by an approximation through recursive convolution, which is of general application, but time consuming.

Since partial derivatives of link travel times with respect to route flows and percentiles permute (see Proposition 7 in Chapter 3), Nie (2011) conjectures that this property also holds for routes, though he states that the validity of this relation is an open question.

Motivated by this challenge, we have studied this problem and in this chapter we solve this question in the negative sense by providing a collection of counter examples. All existing methods dealing with the traffic assignment problem with travel time reliability considerations are path oriented, that is, they require a path enumeration, a path generation module or an algorithm to iteratively add routes leading to the smallest travel time percentiles. In this chapter, we also present a method that, to our knowledge, is the first that can be used without path enumeration. It must be mentioned that avoiding path enumeration is very difficult or impossible in some of this kind of problems, due to the presence of a square root in the formula that provides the path travel time α-percentile in terms of link variances.

The paper is organized as follows. In Section 7.2, we introduce the open question and give a solution to it. In Section 7.3, we introduce the PSO problem, which can be stated in terms of path travel times or link travel times, so that path enumeration can be avoided. In Section 7.4, we illustrate the proposed method by its application to the Nguyen-Dupuis and the Ciudad Real networks.

7.2 Open question raised by Nie (2011)

In this section we solve the open question in the negative sense. First, we introduce the question with some precision and next we solve it.

7.2.1 Statement of the open question

First, for the sake of clarity we introduce the problem using the material in Nie (2011). Consider a network consisting of a set of nodes $\mathcal{N}$, a set of links $\mathcal{A}$, and a set of OD pairs $\mathcal{OD} \subset \mathcal{N}^2$. Each OD pair $ks \in \mathcal{OD}$ is connected by a set of routes $\mathcal{R}_{ks}$. Let $\mathcal{R} = \cup_{ks\in\mathcal{OD}}\mathcal{R}_{ks}$ denote the set of all routes, and let $m = |\mathcal{A}|$, $o = |\mathcal{OD}|$ and $n = |\mathcal{R}|$ denote the cardinalities of $\mathcal{A}$, $\mathcal{OD}$ and $\mathcal{R}$, respectively. Let the matrix $(\Delta = [\delta_{ijr}]) \in \mathbb{R}^{m\times n}$ denote the link-route incidence matrix; here $\delta_{ijr} = 1$ if link ℓ_{ij} is on route r and $\delta_{ijr} = 0$ otherwise. Further, travelers with the same on-time arrival reliability α are grouped into class $\alpha \in \mathcal{H}$, where $h = |\mathcal{H}|$. We use column vectors $(f = [f^\alpha]) \in \mathbb{R}^{n\times h}$, $(\xi = [\xi^\alpha]) \in \mathbb{R}^{n\times h}$ and $(w = [w^\alpha]) \in \mathbb{R}^{m\times h}$ to denote the route flow, the percentile route travel time and the link flow, respectively, where $(f^\alpha = [f^\alpha_r]) \in \mathbb{R}^n$, $(\xi^\alpha = [\xi^\alpha_r]) \in \mathbb{R}^n$ and $(w^\alpha = [w^\alpha_{ij}]) \in \mathbb{R}^m$ are the corresponding column vectors for class α. Finally, let f^α_{rks} and ξ^α_{rks} denote the flow and the associated percentile travel time for class α travelers from OD pair ks on route r, respectively. If all travelers choose routes based on the percentile route travel time according to their own α, the user equilibrium (UE) conditions imply that any used route has the identical and minimum percentile route travel time π^α_{ks}, i.e.,

$$f^\alpha_{rks} > 0 \rightarrow \xi^\alpha_{rks} = \pi^\alpha_{ks}; \quad \xi^\alpha_{rks} \geq \pi^\alpha_{ks}, \quad r \in \mathcal{R},\ ks \in \mathcal{OD},\ \alpha \in \mathcal{H};\ f \in \Omega, \tag{7.1}$$

where $\Omega = \{f | \sum_r \delta_{ijr} f^{\alpha}_{rks} = w^{\alpha}_{ij}, f^{\alpha} \geq 0, \alpha \in \mathcal{H}\}$.

Let $t_{ij} = g(w_{ij}, q_{ij}, \boldsymbol{\theta}_{ij})$ be the ℓ_{ij} link travel time, $g(w_{ij}, q_{ij}, \boldsymbol{\theta}_{ij})$ the link performance function and $t^{\alpha}_{ij} = g(w_{ij}, q^{\alpha}_{ij}, \boldsymbol{\theta}_{ij})$ its α percentile, where $\boldsymbol{\theta}_{ij}$ and q_{ij} are deterministic and random parameters, respectively, and q^{α}_{ij} is the α percentile of q_{ij}. Then, Nie (2011) obtains the relation

$$\tau^{\alpha}_{ij} = \frac{dg(w_{ij}, q^{\alpha}_{ij}, \boldsymbol{\theta}_{ij})}{dw_{ij}} = \frac{dt^{\alpha}_{ij}}{dw_{ij}}, \tag{7.2}$$

where τ^{α}_{ij} is the α percentile of the derivative $\dfrac{\partial g(w_{ij}, q_{ij}, \boldsymbol{\theta}_{ij})}{\partial w_{ij}}$.

Relation (7.2) expresses that percentile and partial derivatives of link travel time with respect to link flow operations permute (Proposition 7 in Chapter 3 states and demonstrates this interesting property). Whether or not this property holds also for routes is the problem raised by Nie (2011).

7.2.2 Solving the open question

In this section we solve the open question[1]. In the following lines we assume, the BPR function in (3.147) and normal and independent link travel time random variables. Let ξ_{rks} be the random travel time on route $r \in \mathcal{R}_{ks}$, then we have

$$\begin{aligned} \xi_{rks} &= \sum_{\ell_{ij}} \delta_{ijr} g(w_{ij}, q_{ij}, \boldsymbol{\theta}_{ij}) = \sum_{\ell_{ij}} \delta_{ijr} t_{0ij} \left[1 + \beta_{ij} \left(\frac{w_{ij}}{q_{ij}}\right)^{\gamma}\right] \\ &= \sum_{\ell_{ij}} \delta_{ijr} t_{0ij} + \sum_{\ell_{ij}} \delta_{ijr} \frac{t_{0ij} \beta_{ij} w^{\gamma}_{ij}}{q^{\gamma}_{ij}} = M_r + \sum_{\ell_{ij}} \alpha_{ijr} \frac{w^{\gamma}_{ij}}{q^{\gamma}_{ij}}, \end{aligned} \tag{7.3}$$

where t_{0ij} is the free travel time on link ℓ_{ij}, and β_{ij} and γ are non-negative parameters of the BPR function, and

$$M_r = \sum_{\ell_{ij}} \delta_{ijr} t_{0ij}, \tag{7.4}$$

$$\alpha_{ijr} = \delta_{ijr} t_{0ij} \beta_{ij}. \tag{7.5}$$

Taking derivatives in (7.3) with respect to route r' we get

$$\frac{\partial \xi_{rks}}{\partial f_{r'ks}} = \sum_{\ell_{ij}} \frac{\partial \xi_{rks}}{\partial w_{ij}} \frac{\partial w_{ij}}{\partial f_{r'ks}} = \sum_{\ell_{ij}} \frac{\alpha_{ijr} \gamma w^{\gamma-1}_{ij}}{q^{\gamma}_{ij}} \delta_{ijr'} = \sum_{\ell_{ij}} \rho_{ijrr'} \frac{w^{\gamma-1}_{ij}}{q^{\gamma}_{ij}}, \tag{7.6}$$

where

$$\rho_{ijrr'} = \delta_{ijr'} \alpha_{ijr} \gamma = \delta_{ijr'} \delta_{ijr} t_{0ij} \beta_{ij} \gamma. \tag{7.7}$$

[1]To demonstrate that the conjecture does not hold, it is sufficient to prove it for any particular case.

If the random variable $\chi_{ij} = 1/q_{ij}^{\gamma}$ is assumed to be normal $N(\mu_{ij}, \sigma_{ij}^2)$ and we assume that the χ_{ij} variables are independent, we have

$$\xi_{rks} \sim N\left(M_r + \sum_{\ell_{ij}} \alpha_{ijr} w_{ij}^{\gamma} \mu_{ij}, \sum_{\ell_{ij}} (\alpha_{ijr} w_{ij}^{\gamma} \sigma_{ij})^2\right), \tag{7.8}$$

$$\frac{\partial \xi_{rks}}{\partial f_{r'ks}} \sim N\left(\sum_{\ell_{ij}} \rho_{ijrr'} w_{ij}^{\gamma-1} \mu_{ij}, \sum_{\ell_{ij}} \left(\rho_{ijrr'} w_{ij}^{\gamma-1} \sigma_{ij}\right)^2\right). \tag{7.9}$$

Since the α-percentile x_α of a normal random variable $N(\mu, \sigma)$ is $x_\alpha = \mu + \sigma z_\alpha$, where z_α is the α-percentile of the $N(0,1)$, the α-percentile ξ_{rks}^{α} of ξ_{rks} is

$$\xi_{rks}^{\alpha} = M_r + \sum_{\ell_{ij}} \alpha_{ijr} w_{ij}^{\gamma} \mu_{ij} + z_\alpha \sqrt{\sum_{\ell_{ij}} \left(\alpha_{ijr} w_{ij}^{\gamma} \sigma_{ij}\right)^2} \tag{7.10}$$

and the α-percentile $\psi_{rr'ks}^{\alpha}$ of $\dfrac{\partial \xi_{rks}}{\partial f_{r'ks}}$ is

$$\begin{aligned} \psi_{rr'ks}^{\alpha} &= \sum_{\ell_{ij}} \rho_{ijrr'} w_{ij}^{\gamma-1} \mu_{ij} + z_\alpha \sqrt{\sum_{\ell_{ij}} \left(\rho_{ijrr'} w_{ij}^{\gamma-1} \sigma_{ij}\right)^2} \\ &= \sum_{\ell_{ij}} \alpha_{ijr} \gamma w_{ij}^{\gamma-1} \mu_{ij} \delta_{ijr'} + z_\alpha \sqrt{\sum_{\ell_{ij}} \left(\delta_{ijr'} \alpha_{ijr} \gamma w_{ij}^{\gamma-1} \sigma_{ij}\right)^2}. \end{aligned} \tag{7.11}$$

Finally, the partial derivative of the α-percentile ξ_{rks}^{α} in (7.10) with respect to the route flow $f_{r'ks}$ becomes

$$\begin{aligned} \frac{\partial \xi_{rks}^{\alpha}}{\partial f_{r'ks}} &= \sum_{\ell_{ij}} \left[\frac{\partial \xi_{rks}^{\alpha}}{\partial w_{ij}} \frac{\partial w_{ij}}{\partial f_{r'ks}}\right] \\ &= \sum_{\ell_{ij}} \left[\alpha_{ijr} \gamma w_{ij}^{\gamma-1} \mu_{ij} \delta_{ijr'} + z_\alpha \sigma_{ij}^2 \frac{\alpha_{ijr}^2 \gamma w_{ij}^{2\gamma-1} \delta_{ijr'}}{\sqrt{\sum_{\ell_{ij}} \left(\alpha_{ijr} w_{ij}^{\gamma} \sigma_{ij}\right)^2}}\right] \\ &= \sum_{\ell_{ij}} \left[\alpha_{ijr} \gamma w_{ij}^{\gamma-1} \mu_{ij} \delta_{ijr'} + z_\alpha \sigma_{ij}^2 \frac{t_{0ij}^2 \beta_{ij}^2 \gamma w_{ij}^{2\gamma-1} \delta_{ijr} \delta_{ijr'}}{\sqrt{\sum_{\ell_{ij}} \delta_{ijr} t_{0ij}^2 \beta_{ij}^2 w_{ij}^{2\gamma} \sigma_{ij}^2}}\right]. \end{aligned} \tag{7.12}$$

In order $\psi_{rr'ks}^{\alpha}$ and $\dfrac{\partial \xi_{rks}^{\alpha}}{\partial f_{r'ks}^{\alpha}}$ to be identical, according to (7.11) and (7.12), the following

functional equation must be satisfied[2]

$$\sum_{\ell_{ij}} \delta_{ijr}\delta_{ijr'}\nu_{ij}^2 w_{ij}^{2\gamma-1} = \sqrt{\left(\sum_{\ell_{ij}} \delta_{ijr}\nu_{ij}^2 w_{ij}^{2\gamma}\right)\left(\sum_{\ell_{ij}} \delta_{ijr'}\delta_{ijr}\nu_{ij}^2 w_{ij}^{2\gamma-2}\right)}, \tag{7.13}$$

where $\nu_{ij} = t_{0ij}\beta_{ij}\sigma_{ij}$, which holds only when routes r and r' have no links in common (in such a case, the partial derivatives (7.6) and (7.12) are null) or when $r \subseteq r'$ and route r has a single link (a very special case). Thus, the two operations: (a) obtain percentiles and (b) partial derivation of route travel times do not commute in general.

In addition, even though Nie's algorithm uses only diagonal elements ($r = r'$), it has been proved that this algorithm must be modified in general, because it is valid only in very special cases.

7.3 Proposed PSO model

In this section we will propose a traffic assignment model called the *Percentile System Optimal (PSO)*. Let ψ_{ks}^{α} be the mean travel time of all users of OD ks and class α and ϕ_r the path r travel time. Then, we have

$$\begin{aligned}\psi_{ks}^{\alpha} &= \sum_{r\in ks} \frac{f_r^{\alpha}}{t_{ks}^{\alpha}}\phi_r = \sum_{r\in ks} \frac{f_r^{\alpha}}{t_{ks}^{\alpha}} \sum_{\ell_{ij}\in r} t_{ij} = \sum_{\ell_{ij}\in r}\sum_{r\in ks} \frac{f_r^{\alpha}}{t_{ks}^{\alpha}} t_{ij} \\ &= \sum_{\ell_{ij}\in\mathcal{A}}\sum_{r\in ks} \frac{f_r^{\alpha}}{t_{ks}^{\alpha}}\delta_{ijr}t_{ij} = \sum_{\ell_{ij}\in\mathcal{A}} \left(\sum_{r\in ks} \frac{f_r^{\alpha}}{t_{ks}^{\alpha}}\delta_{ijr}\right) t_{ij},\end{aligned} \tag{7.14}$$

where t_{ij} is the ℓ_{ij} link travel time and t_{ks}^{α} are the class OD travel demands.

From Equation (7.14) we obtain:

$$E\left[\psi_{ks}^{\alpha}\right] = \sum_{\ell_{ij}\in\mathcal{A}} \left(\sum_{r\in ks} \frac{f_r^{\alpha}}{t_{ks}^{\alpha}}\delta_{ijr}\right) E[t_{ij}], \tag{7.15}$$

and

$$Var\left[\psi_{ks}^{\alpha}\right] = \sum_{\ell_{ij}\in\mathcal{A}} \left(\sum_{r\in ks} \frac{f_r^{\alpha}}{t_{ks}^{\alpha}}\delta_{ijr}\right)^2 Var[t_{ij}], \tag{7.16}$$

where for the sake of simplicity, we have used the independence assumption.

If now we assume that the ψ_{ks}^{α} belong to a location-scale[3] and infinitely divisible family $H(\mu,\sigma)$, where μ and σ are the mean and standard deviation parameters, that is,

[2] Functional equations are dealt with, for example, in Aczél (1966), Castillo and Ruiz (1992) and Castillo et al. (2005).

[3] This is equivalent to assume that the total $t_{ks}^{\alpha}\psi_{ks}^{\alpha}$ belong to the same location-scale family.

the location and scale parameters, respectively[4], the ψ_{ks}^{α} α-percentile ξ_{ks}^{α} becomes:

$$\xi_{ks}^{\alpha} = \frac{1}{t_{ks}^{\alpha}} \left[\sum_{\ell_{ij} \in \mathcal{A}} \sum_{r \in ks} f_r^{\alpha} \delta_{ijr} E\left[t_{ij}\right] + z_{\alpha} \sqrt{\sum_{\ell_{ij} \in \mathcal{A}} \left(\sum_{r \in ks} f_r^{\alpha} \delta_{ijr} \right)^2 Var\left[t_{ij}\right]} \right], \tag{7.17}$$

where assuming the BPR function in (3.147) and taking into account that,

$$E\left[t_{ij}\right] = t_{0ij} + t_{0ij} \beta_{ij} x_{ij}^{\gamma} \mu_{ij} \tag{7.18}$$

$$Var\left[t_{ij}\right] = \left(t_{0ij} \beta_{ij} x_{ij}^{\gamma} \sigma_{ij} \right)^2, \tag{7.19}$$

where $x_{ij} = w_{ij}/q_{ij}$, leads to

$$\xi_{ks}^{\alpha} = \frac{1}{t_{ks}^{\alpha}} \left[\sum_{\ell_{ij} \in \mathcal{A}} \left(t_{0ij}(1 + \beta_{ij} x_{ij}^{\gamma} \mu_{ij}) \sum_{r \in ks} f_r^{\alpha} \delta_{ijr} \right) + z_{\alpha} \sqrt{\sum_{\ell_{ij} \in \mathcal{A}} \left(t_{0ij} \beta_{ij} x_{ij}^{\gamma} \sigma_{ij} \right)^2 \left(\sum_{r \in ks} f_r^{\alpha} \delta_{ijr} \right)^2} \right]. \tag{7.20}$$

Finally, since we aim at a global decision of all users in class α, we propose to minimize the sum of percentiles ξ_{ks}^{α} of all users in all classes, that is, the following PSO optimization problem:

$$\min_{\mathbf{x},\mathbf{f}} \sum_{k,s,\alpha} \left(\sum_{\ell_{ij} \in \mathcal{A}} \left(t_{0ij}(1 + \beta_{ij} x_{ij}^{\gamma} \mu_{ij}) \sum_{r \in ks} f_r^{\alpha} \delta_{ijr} \right) + z_{\alpha} \sqrt{\sum_{\ell_{ij} \in \mathcal{A}} \left(t_{0ij} \beta_{ij} x_{ij}^{\gamma} \sigma_{ij} \right)^2 \left(\sum_{r \in ks} f_r^{\alpha} \delta_{ijr} \right)^2} \right) \tag{7.21}$$

subject to:

$$x_{ij} = \sum_{r,\alpha} f_r^{\alpha} \delta_{ijr}; \quad \ell_{ij} \in \mathcal{A} \tag{7.22}$$

$$t_{ks}^{\alpha} = \sum_{r \in ks} f_r^{\alpha}; \quad k, s \in \mathcal{OD},\ \alpha \in \mathcal{H} \tag{7.23}$$

$$f_r^{\alpha} \geq 0; \quad r \in \mathcal{R},\ \alpha \in \mathcal{H}, \tag{7.24}$$

which is a path based model and the function (7.21) is not convex.

7.3.1 Avoiding path enumeration

As indicated in the introduction, avoiding path enumeration is crucial for the methods to be useful for large or very large networks. In this section we state the above model without path enumeration.

[4]If the $H(\mu, \sigma)$ family is reproductive, in the case of independent link travel times this assumption is equivalent to assuming that the link travel times belong to family $H(\mu_{ij}, \sigma_{ij}^2)$. One particular case of these families is the normal distribution family, but other families have this property too.

To this end, we realize that making the following change of variables

$$v_{ijks}^{\alpha} = \sum_{r \in ks} f_r^{\alpha} \delta_{ijr}, \tag{7.25}$$

where v_{ijks}^{α} is the α-class flow through link ℓ_{ij} with origin node k and destination node s, and rewriting constraints (7.22) and (7.23) in terms of v_{ijks}^{α}, we get the following PSO optimization problem:

$$\underset{\mathbf{v}}{\text{Minimize}}\, Z = \sum_{k,s,\alpha} \left\{ \sum_{\ell_{ij} \in \mathcal{A}} v_{ijks}^{\alpha} (t_{0ij} + t_{0ij}\beta_{ij} x_{ij}^{\gamma} \mu_{ij}) + z_{\alpha} \sqrt{\sum_{\ell_{ij} \in \mathcal{A}} \left(v_{ijks}^{\alpha} t_{0ij} \beta_{ij} x_{ij}^{\gamma} \sigma_{ij} \right)^2} \right\} \tag{7.26}$$

subject to

$$t_{ks}^{\alpha}(\delta_{ik}^{*} - \delta_{is}^{*}) = \sum_{\ell_{ij} \in \mathcal{A}} v_{ijks}^{\alpha} - \sum_{\ell_{ji} \in \mathcal{A}} v_{jiks}^{\alpha}; \quad i \in \mathcal{N};\ k, s \in \mathcal{OD};\ \alpha \in \mathcal{H} \tag{7.27}$$

$$x_{ij} = \sum_{k,s,\alpha} v_{ijks}^{\alpha}; \quad \ell_{ij} \in \mathcal{A} \tag{7.28}$$

$$v_{ijks}^{\alpha} \geq 0; \quad \ell_{ij} \in \mathcal{A};\ k, s \in \mathcal{OD};\ \alpha \in \mathcal{H}, \tag{7.29}$$

where δ_{ij}^{*} are the Kronecker delta.

Since (7.26) is non-convex, solvers can provide undesirable local minima instead of global minima. To avoid this problem, we can add the extra constraint

$$Z \leq z_{bound}, \tag{7.30}$$

and solve the optimization problem (7.26)–(7.30) several times, starting with $z_{bound} = \infty$ and making $z_{bound} = Z^{*} - \varepsilon$ with ε a small number after each iteration, until unfeasibility is obtained. Then, the last feasible solution obtained can be taken as the global optimum. Our experience with this method reveals that a very few iterations (2 or 3) are normally required.

Note that the problem (7.26)-(7.29) avoids path enumeration and since (7.14) can be written as

$$\psi_{ks}^{\alpha} = \frac{1}{t_{ks}^{\alpha}} \sum_{\ell_{ij}} \left(\sum_{r \in ks} f_r^{\alpha} \delta_{ijr} \right) t_{ij} = \frac{1}{t_{ks}^{\alpha}} \sum_{\ell_{ij}} v_{ijks}^{\alpha} t_{ij}, \tag{7.31}$$

the objective function (7.26) using (7.18) and (7.19) could have been obtained directly avoiding path enumeration too.

We note that the formulation without path enumeration requires large number of constraints. The number of constraints is roughly in the order of $|\mathcal{N}| \times |\mathcal{OD}| \times |\mathcal{H}|$, where $|\mathcal{N}|$ is the number of nodes, $|\mathcal{OD}|$ is the number of OD pairs, and $|\mathcal{H}|$ is the number of user classes. For a real-sized network in the current practice, the number of constraints can be very large. Thus, the proposed formulation would work fine for small to medium size problems (with hundreds of nodes and OD pairs). For large instances, other ad hoc algorithms such as those proposed by Nie (2011) would be more promising.

7.4 Examples of applications

The proposed method is illustrated by its application to some networks.

7.4.1 The Nguyen-Dupuis network

In order to understand the behavior of the proposed methods, consider the Nguyen-Dupuis network topology, with 13 nodes and 38 bidirectional links as shown in Figure 3.3. A total of 11 $\mathcal{OD}$ pairs are considered, whose origins and destination nodes are provided in Table 7.1 together with the corresponding total $\mathcal{OD}$ flows.

$\mathcal{OD}$	$\mathcal{O}$	$\mathcal{D}$	$\mathcal{OD}$ flow		$\mathcal{OD}$	$\mathcal{O}$	$\mathcal{D}$	$\mathcal{OD}$ flow
1	1	2	210		7	3	1	430
2	1	3	430		8	3	4	110
3	1	8	320		9	3	12	40
4	2	1	210		10	4	2	320
5	2	4	320		11	4	3	110
6	2	12	50					

Table 7.1: $\mathcal{OD}$ pairs and corresponding flows used in the Nguyen-Dupuis example.

The link parameters used in this example are those shown in Table 7.2 and the considered 43 routes for solving the Problem (7.21)-(7.24) are shown in Table 7.3.

Four classes of users are considered with on-time arrival probabilities $\alpha = 0.1, 0.5, 0.7$ and 0.9, respectively. We use identical flow share for all classes, that is, $0.25, 0.25, 0.25$ and 0.25.

The optimization problems with path enumeration (7.21)-(7.24) and without path enumeration (7.26)-(7.29) have been implemented in GAMS using the IPOPT solver. The cpu times required were 0.19 sec and 0.86 sec, respectively, on a HP Z200 Workstation, Intel Core i7-870 2.93 8MB/1333 QC, RAM: 8GB (2x4GB).

Table 7.4 shows the percentile PSO solutions for the Nguyen-Dupuis example. It includes the ξ_{ks}^{α} OD travel time percentiles and the corresponding OD flows. It also includes the resulting path travel time percentiles and flows. We can see that different α users utilize the same or different routes of the same OD, and that some classes can use several routes and ignore other of the same OD. Note that when only one path is used in one OD, the OD and the path α travel time percentiles coincide.

It is interesting to see that paths $16, 20, 38, 41$ and 43 are not used by any user. It is also important to realize that the model (7.26)-(7.29) without path enumeration permits identifying the used paths by any class user.

Figure 7.1 shows the resulting paths used by different classes corresponding to a selected set of ODs ($2, 4, 5$ and 7). Each group of 4 graphs (one per class) corresponds to

Link	i	j	t_{0ij}	q_{ij}	β_{ij}	γ_{ij}	Link	i	j	t_{0ij}	q_{ij}	β_{ij}	γ_{ij}
1	1	5	7.00	700	1.00	2	20	8	2	9.00	700	1.00	2
2	1	12	9.00	560	1.00	2	21	8	7	5.00	700	1.00	2
3	2	8	9.00	700	1.00	2	22	8	12	14.00	560	1.00	2
4	2	11	9.00	280	1.00	2	23	9	4	12.00	280	1.00	2
5	3	11	8.00	560	1.00	2	24	9	5	9.00	420	1.00	2
6	3	13	11.00	560	1.00	2	25	9	10	10.00	280	1.00	2
7	4	5	9.00	560	1.00	2	26	9	13	9.00	280	1.00	2
8	4	9	12.00	280	1.00	2	27	10	6	5.00	280	1.00	2
9	5	1	7.00	700	1.00	2	28	10	9	10.00	280	1.00	2
10	5	4	9.00	560	1.00	2	29	10	11	6.00	700	1.00	2
11	5	6	3.00	420	1.00	2	30	11	2	9.00	280	1.00	2
12	5	9	9.00	420	1.00	2	31	11	3	8.00	560	1.00	2
13	6	5	3.00	420	1.00	2	32	11	7	9.00	700	1.00	2
14	6	7	5.00	700	1.00	2	33	11	10	6.00	700	1.00	2
15	6	10	5.00	280	1.00	2	34	12	1	9.00	560	1.00	2
16	6	12	7.00	140	1.00	2	35	12	6	7.00	140	1.00	2
17	7	6	5.00	700	1.00	2	36	12	8	14.00	560	1.00	2
18	7	8	5.00	700	1.00	2	37	13	3	11.00	560	1.00	2
19	7	11	9.00	700	1.00	2	38	13	9	9.00	280	1.00	2

Table 7.2: Parameters of the Nguyen-Dupuis network.

one OD, which have been sorted from top to bottom and left to right with increasing α values $(0.1, 0.5, 0.7, 0.9)$.

These figures illustrate that users with smaller α values are more selective in using paths (they use one or very few paths) while users with large α values have no problem in using several path alternatives.

Finally, we indicate that, as expected, the two problems: with (7.21)-(7.24) and without path enumeration (7.26)-(7.29) lead to exactly the same solution.

7.4.2 The Ciudad Real network

To test the proposed models in a real network, we have used the Ciudad Real network used in Castillo et al. (2008g) and shown in Figure 7.2, which consists of 218 links, 105 nodes, 380 OD pairs, and 590 routes. The results are very similar to those described for the Nguyen-Dupuis network, but we cannot show all of them because of its size and the lack of space. The cpu times required to reach the solution with four α classes were 19 sec and 58 min, for the problems (7.21)-(7.24) and (7.26)-(7.29), respectively on a HP Z200 Workstation, Intel Core i7-870 2.93 8MB/1333 QC, RAM: 8GB (2x4GB).

Route (r)	Links (ℓ)
OD pair 1	
1	1 11 14 18 20
2	2 35 14 18 20
3	2 36 20
4	1 12 25 29 30
5	1 11 15 29 30
6	2 35 15 29 30
OD pair 2	
7	1 11 14 19 31
8	1 11 15 29 31
9	1 12 25 29 31
10	1 12 26 37
11	2 35 14 19 31
12	2 35 15 29 31
OD pair 3	
13	1 11 14 18
14	2 35 14 18
15	2 36
OD pair 4	
16	3 21 17 13 9
17	3 22 34
OD pair 5	
18	3 21 17 13 10
19	4 33 28 23
OD pair 6	
20	3 21 17 16
21	3 22

Route (r)	Links (ℓ)
OD pair 7	
22	5 32 17 13 9
23	5 32 17 16 34
24	5 33 27 13 9
25	5 33 27 16 34
26	5 33 28 24 9
27	6 38 24 9
28	5 32 18 22 34
OD pair 8	
29	5 33 28 23
30	6 38 23
31	5 32 17 13 10
OD pair 9	
32	5 32 17 16
33	5 33 27 16
34	5 32 18 22
OD pair 10	
35	7 11 14 18 20
36	8 25 29 30
37	7 11 15 29 30
38	7 12 25 29 30
OD pair 11	
39	8 25 29 31
40	8 26 37
41	7 12 25 29 31
42	7 11 15 29 31
43	7 12 26 37

Table 7.3: Set of $\mathcal{OD}$-pairs and routes considered in the Nguyen-Dupuis network.

Table 7.5 shows some of the percentile PSO solutions for the Ciudad Real example. The results are very similar to those obtained in Table 7.4 for the Nguyen-Dupuis example.

Figure 7.3 shows the resulting paths used by different classes corresponding to a selected set of ODs (46, 192, 193 and 206). Each group of 4 graphs (one per class) corresponds to one OD, which have been sorted from top to bottom and left to right. This figure allows us to understand better the results in Table 7.5 and how users with smaller α values are more selective in choosing routes than users with larger α values. Path origins and destinations have been indicated by green and blue circles, respectively, and used paths are indicated by thick red segments. The remaining links and nodes are shown by thin

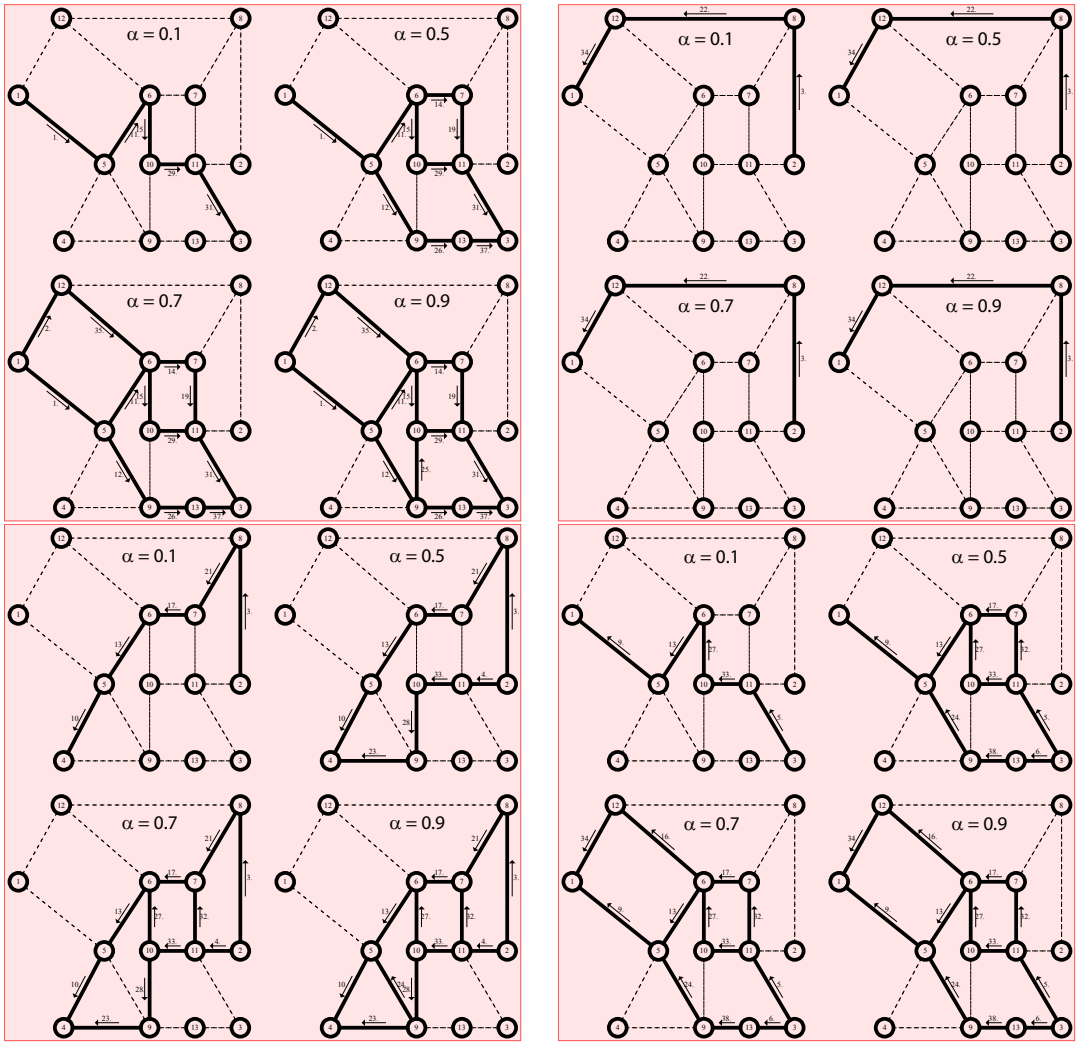


Figure 7.1: Routes used by different α classes corresponding to ODs $2, 4, 5$ and 7 in the Nguyen-Dupuis example obtained without path enumeration.

segments and small circles, respectively.

It is interesting to see that most paths are not used by any user. This is due to the fact that most streets are one-way and changing the shortest path implies increasing the path length. It is also important to realize that the model (7.26)-(7.29) without path enumeration permits identifying the few paths used by any class user.

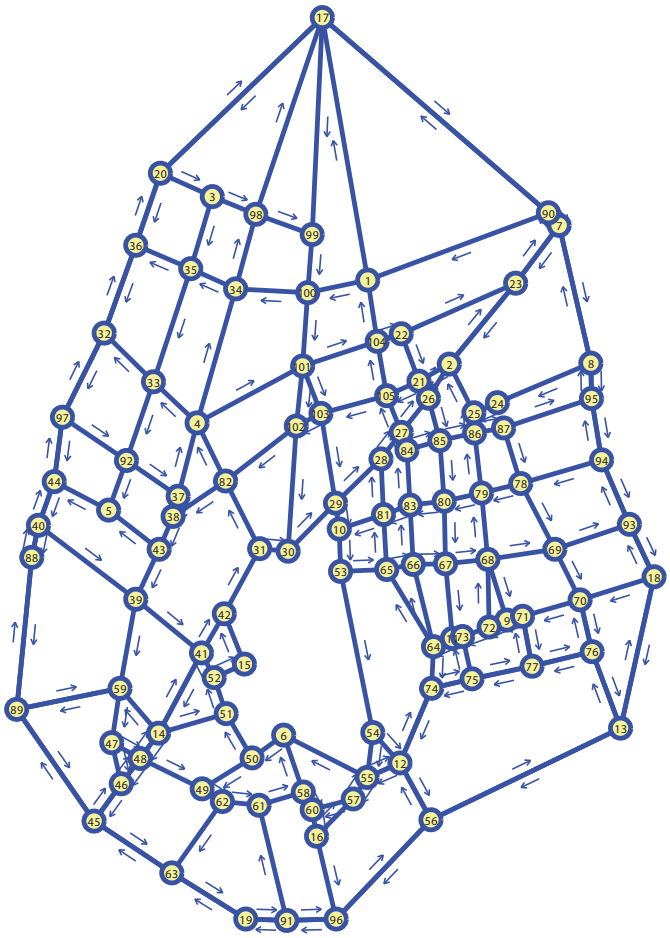


Figure 7.2: Ciudad Real traffic network used in the Example of application.

O-D	Route	(α =, 0.10)		(α = 0.50)		(α = 0.70)		(α = 0.90)	
		Flow	Percentile	Flow	Percentile	Flow	Percentile	Flow	Percentile
1 -2	2	-	-	-	-	14.34	45.85	13.00	46.10
	3	-	-	-	-	-	-	0.03	48.70
	5	-	-	-	-	7.27	46.52	5.55	46.76
	7	-	-	-	-	10.41	53.52	14.10	53.70
	11	-	-	-	-	-	-	1.18	50.13
	12	-	-	-	-	-	-	0.03	52.73
	14	-	-	-	-	-	-	1.05	50.79
	17	52.50	45.25	52.50	46.71	20.48	48.17	17.55	48.46
OD		52.50	45.25	52.50	46.71	52.50	47.95	52.50	48.67
1 -3	4	-	-	39.82	46.48	19.22	47.73	17.84	47.97
	6	107.50	43.26	4.16	44.52	14.07	45.79	13.04	46.04
	8	-	-	-	-	-	-	2.32	52.97
	9	-	-	63.52	49.06	42.17	50.26	40.35	50.50
	13	-	-	-	-	18.41	51.78	19.85	52.00
	15	-	-	-	-	13.63	49.84	14.09	50.07
OD		107.50	43.26	107.50	47.93	107.50	49.01	107.50	49.31
2 -1	20	52.50	39.72	52.50	40.55	52.50	41.39	52.50	41.56
OD		52.50	39.71	52.50	40.55	52.50	41.39	52.50	41.56
2 -4	18	80.00	41.56	60.26	42.54	25.14	43.53	23.45	43.73
	21	-	-	-	-	8.96	46.03	7.50	46.20
	22	-	-	-	-	6.40	44.13	4.04	44.30
	23	-	-	19.74	46.30	39.51	47.20	33.88	47.37
	24	-	-	-	-	-	-	11.13	51.23
OD		80.00	41.55	80.00	43.47	80.00	45.39	80.00	46.24
3 -1	25	-	-	50.07	43.85	30.95	44.83	26.91	45.02
	28	-	-	-	-	10.83	48.70	16.14	48.85
	29	107.50	40.88	11.36	41.90	12.44	42.92	11.32	43.12
	32	-	-	-	-	7.12	46.80	8.84	46.96
	36	-	-	46.07	46.36	46.17	47.37	44.30	47.56
OD		107.50	40.88	107.50	44.86	107.50	45.92	107.50	46.23
3 -4	26	-	-	-	-	8.92	46.52	8.55	46.71
	30	-	-	-	-	3.94	44.61	3.39	44.81
	33	-	-	-	-	1.27	47.69	2.36	47.88
	34	-	-	-	-	-	-	0.91	51.80
	35	27.50	42.61	27.50	43.82	13.37	45.03	10.98	45.27
	37	-	-	-	-	-	-	1.31	49.26
OD		27.50	42.61	27.50	43.82	27.50	45.25	27.50	45.89
3 -12	27	10.00	36.33	5.83	37.06	6.81	37.80	6.81	37.94
	31	-	-	4.17	35.11	3.19	35.91	3.19	36.06
OD		10.00	34.32	10.00	36.57	10.00	37.16	10.00	37.30
4 -2	38	80.00	43.41	68.09	44.56	35.72	45.71	33.79	45.93
	41	-	-	-	-	-	-	5.82	53.45
	44	-	-	11.91	48.26	44.28	49.30	40.39	49.50
OD		80.00	43.41	80.00	45.11	80.00	47.38	80.00	47.89
4 -3	39	-	-	-	-	6.60	47.58	6.28	47.80
	40	-	-	-	-	3.67	45.64	2.73	45.87
	42	-	-	-	-	-	-	1.95	52.74
	43	-	-	-	-	-	-	2.29	50.32
	45	-	-	-	-	4.61	48.57	5.32	48.77
	46	27.50	43.41	27.50	44.71	12.62	46.01	8.92	46.27
OD		27.50	43.41	27.50	44.71	27.50	46.40	27.50	47.39

Table 7.4: Percentile PSO solution for the Nguyen-Dupuis example.

O-D	Route	(α =, 0.10)		($\alpha = 0.50$)		($\alpha = 0.70$)		($\alpha = 0.90$)	
		Flow	Percentile	Flow	Percentile	Flow	Percentile	Flow	Percentile
3 -9	79	11.25	6.24	4.91	6.28	4.30	6.33	4.30	6.34
	82	-	-	-	-	1.77	5.92	1.52	5.93
	85	-	-	-	-	1.01	6.00	1.26	6.01
	89	-	-	6.34	5.79	4.17	5.83	4.17	5.84
OD 46		11.25	6.23	11.25	6.00	11.25	6.03	11.25	6.04
4 -14	95	-	-	-	-	-	-	0.14	4.29
	96	-	-	-	-	-	-	0.14	4.45
	97	-	-	-	-	1.08	4.46	2.74	4.47
	10	7.50	3.85	7.50	3.88	3.64	3.91	2.35	3.92
	10	-	-	-	-	2.78	4.08	2.13	4.08
OD 70		7.50	3.85	7.50	3.88	7.50	4.05	7.50	4.18
10-3	25	-	-	-	-	0.38	7.17	0.49	7.18
	26	-	-	-	-	0.78	6.93	0.89	6.94
	26	-	-	-	-	0.54	7.08	0.66	7.10
	27	3.75	6.72	3.75	6.78	2.05	6.84	1.72	6.86
OD 174		3.75	6.72	3.75	6.78	3.75	6.92	3.75	6.95
11-2	27	-	-	-	-	0.01	2.95	0.10	2.95
	27	-	-	-	-	0.02	2.90	0.17	2.91
	27	-	-	0.93	2.80	0.91	2.82	0.81	2.83
	29	-	-	-	-	0.08	2.90	0.17	2.91
	29	2.00	2.81	1.07	2.83	0.98	2.86	0.75	2.86
OD 192		2.00	2.81	2.00	2.81	2.00	2.84	2.00	2.85
11-3	28	-	-	-	-	0.37	6.94	0.40	6.95
	28	2.00	6.62	0.93	6.66	0.57	6.70	0.54	6.71
	29	-	-	-	-	0.40	6.98	0.44	6.99
	30	-	-	1.07	6.70	0.66	6.74	0.62	6.75
OD 193		2.00	6.60	2.00	6.67	2.00	6.79	2.00	6.81
11-8	27	-	-	-	-	1.78	4.09	2.20	4.09
	27	-	-	-	-	1.07	4.00	1.08	4.01
	27	-	-	-	-	1.88	4.01	2.23	4.02
	28	-	-	-	-	2.50	4.04	2.31	4.05
	29	11.25	3.91	7.07	3.93	1.30	3.96	1.10	3.96
	30	-	-	4.18	3.94	2.72	3.97	2.34	3.97
OD 198		11.25	3.92	11.25	3.94	11.25	4.01	11.25	4.02
11-17	28	-	-	-	-	0.22	5.33	0.66	5.34
	28	-	-	3.43	5.26	4.48	5.30	4.05	5.31
	29	-	-	-	-	0.23	5.37	0.67	5.38
	30	10.00	5.26	6.57	5.30	5.07	5.34	4.62	5.35
OD 206		10.00	5.26	10.00	5.29	10.00	5.32	10.00	5.33
11-20	28	-	-	-	-	0.61	6.34	0.66	6.35
	28	-	-	1.44	6.06	0.92	6.10	0.87	6.11
	29	-	-	-	-	0.66	6.38	0.72	6.38
	30	3.25	6.06	1.81	6.10	1.06	6.14	1.00	6.15
OD 209		3.25	6.05	3.25	6.07	3.25	6.20	3.25	6.21

Table 7.5: Percentile PSO solution for the Ciudad Real example.

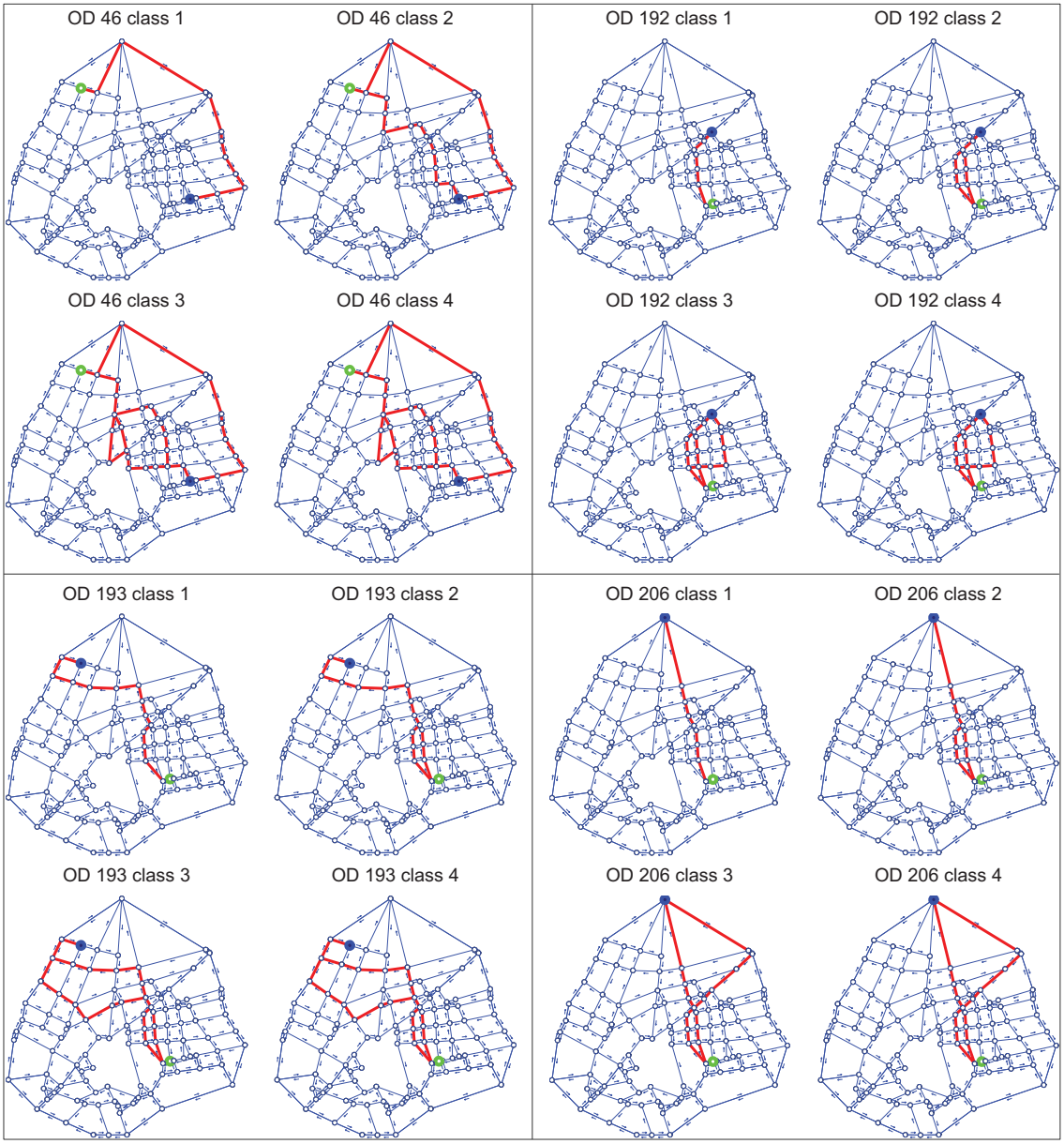


Figure 7.3: Routes used by different α classes corresponding to the ODs 46, 192, 193 and 206 in the Ciudad Real example obtained without path enumeration. Path origins and destinations have been indicated by green and blue circles, respectively, and used paths are indicated by thick red segments. The remaining links and nodes are shown by thin segments and small circles, respectively.

Appendix

A Notation

α	class.
α_{ijr}	auxiliary parameter associated with route r and link ℓ_{ij}.
β_{ij}	BPR parameter associated with link ℓ_{ij}.
γ	BPR exponent parameter.
Δ	link-route incidence matrix.
δ^*_{ij}	Kronecker delta.
δ_{ijr}	element of the link-route incidence matrix.
ϵ	small scalar.
$\boldsymbol{\theta}_{ij}$	deterministic parameters.
μ	mean of the normal distribution.
μ_{ij}	mean of the Normal distribution associated with link ℓ_{ij}.
ν_{ij}	auxiliary parameter associated with link ℓ_{ij}.
ξ	percentile route travel time vector.
ξ^α	percentile route travel time vector associated with class α.
ξ^α_{ks}	α-percentile of ψ^α_{ks}.
ξ^α_r	percentile route travel time of route r associated with class α.
ξ_{rks}	random travel time on route r of OD pair ks.
ξ^α_{rks}	percentile route travel time of route r of OD pair ks associated with class α.
π^α_{ks}	minimum travel time of class α and OD ks.
σ	standard deviation of the normal distribution.
σ_{ij}	standard deviation of the Normal distribution associated with link ℓ_{ij}.
τ^α_{ij}	α percentile of the partial derivative of t_{ij} with respect to w_{ij}.
ϕ_r	path r travel time.
χ_{ij}	inverse of the link capacity random variable.
ψ^α_{ks}	mean travel time of all users of OD ks and class α.
$\psi^\alpha_{rr'ks}$	α-percentile of $\dfrac{\partial \xi_{rks}}{\partial f_{r'ks}}$.
Ω	set of feasible path flows.
$\mathcal{A}$	set of links.
f	route flow vector.
$\mathbf{f}$	vector of route flows f^α_r.
f^α	route flow vector associated with class α.
f^α_r	r route flow of class α.
f^α_{rks}	flow through route r of OD pair ks associated with class α.
$g(\cdot)$	link performance function.
h	cardinality of set $\mathcal{H}$.

$\mathcal{H}$	set of all α classes.
i	link begin node.
j	link end node.
k	origin node.
ℓ_{ij}	link joining nodes i and j.
m	cardinality of set $\mathcal{A}$.
M_r	auxiliary parameter associated with route r.
n	cardinality of set $\mathcal{R}$.
$\mathcal{N}$	set of nodes.
o	cardinality of set $\mathcal{OD}$.
$\mathcal{OD}$	set of all OD-pairs.
q_{ij}	random link capacity.
q_{ij}^{α}	α percentile of q_{ij}.
r	route.
$\mathcal{R}_{ks}$	set of routes joining OD pair ks.
s	destination node.
t_{0ij}	link free travel time associated with link ℓ_{ij}.
t_{ij}	ℓ_{ij} link travel time.
t_{ij}^{α}	travel time α percentile on link ℓ_{ij}.
t_{ks}^{α}	α-class OD flow.
$\mathbf{v}$	vector of disaggregated class link flows v_{ijks}^{α}.
v_{ijks}^{α}	α-class flow through link i,j with origin node k and destination node s.
w	link flow vector.
w^{α}	link flow vector associated with class α.
w_{ij}^{α}	flow through link ℓ_{ij} associated with class α.
x_{α}	α-percentile of a normal random variable $N(\mu,\sigma)$.
x_{ij}	congestion ratio in link ℓ_{ij}.
Z	objective function.
z_{α}	α-percentile of the standard normal distribution.
z_{bound}	bound for the objective function.

Chapter 8

A Traffic Assignment problem including overtaking classes

Contents

8.1 Introduction

The problem of traffic assignment in which different travelers compete for space in a network has been treated in the existing literature for a long time (for a review of traffic assignment models see Chapter 3). Normally, models assume homogeneous users in the sense that all of them behave in the same form or assume a mean behavior (speed, travel time, etc.), but in reality we face a different situation because users are essentially heterogeneous. However, traffic problems with heterogeneous users are not very frequent in the existing literature.

One of the problems in which heterogeneous users have been considered is in the

travel time reliability problem dealt with in Section 3.4, in which different users perceive the traffic from different perspectives. This occurs when travelers are concerned about punctual arrivals, so that we can classify users in different classes depending on this perception (see Asakura and Kashiwadani (1991), Lo and Tung (2003) or Nie and Wu (2009)). Consequently, users in different classes choose different routes based on different criteria and we face *heterogeneity*.

Another interesting case that we study in this chapter arises when we consider overtaking and include classes of users who are prone to overtake, classes of users who avoid overtaking and intermediate classes. It is clear that the possibility of overtaking depends on the existing degree of congestion in the corresponding links (those in the user path), so that users choose route accordingly in order to reduce travel time variability and reliability. Overtaking is related to travel time reliability because if some undesired delays occur, overtaking offers a possibility to recover partially or totally these delays.

A model without consideration of overtaking is able to provide mean travel times but is not realistic. Consequently, incorporating the possibility of overtaking among users is needed if a more precise knowledge of real traffic is pursued. All these considerations have been the main motivation for this study.

The connection between travel time and flow on a road depends on the possibilities of overtaking (see Svensson (1978) and Buric and Janovsky (2007)). Overtaking can only take place when there is a sufficiently large gap in the oncoming traffic and when the sight distance is large enough. However, under high degrees of congestion overtaking is impossible. All these facts are considered by users, who choose routes that best satisfy their expectations (in our present case overtaking possibilities). Thus, mathematical models must reflect this type of user behavior. In particular, impossibility of overtaking under congestion must be considered in travel time functions of class users. This has been done in the models proposed in this paper.

This chapter is devoted to the overtaking problem at the macro level. Different class of users are considered and overtaking is permitted among users of different classes, but it is assumed forbidden for the same class users, who must respect the FIFO[1] rule. The main original contributions are:

1. The overtaking problem is dealt with at the macro level for static assignment.

2. A new family of link travel time functions specially designed for overtaking is presented based on the BPR functions. This family has a sufficient number of parameters to reproduce a wide range of cases.

3. It is shown that due to the asymmetric character of the problem, a Beckman statement of the Wardrop approach is not possible for the overtaking problem.

[1]The FIFO rule establishes that the *first* user entering a link must be also the *first* exiting that link and, hence, overtaking is not possible.

4. Two variational approaches are formulated to solve the overtaking problem with and without path enumeration, and alternative problems are provided.

5. A discussion of existence and uniqueness of solutions is included.

6. Some examples of applications are given. They include both homogeneous and heterogeneous overtaking classes of users.

The chapter is organized as follows. In Section 8.2, we introduce a new travel time function to consider the particular case of overtaking. In Section 8.3 we introduce two approaches to solve the overtaking problem with and without path enumeration and suggest alternative equivalent problems to solve them. Finally, in Section 8.4 we illustrate the proposed methods and methodology by its application to the Nguyen-Dupuis network for homogeneous and heterogeneous users, and to the Ciudad Real network.

8.2 The proposed link travel time function

An adequate definition of link travel time functions is crucial in the overtaking problem. As already seen, the effect of road capacity on travel times is specified by means of the link travel time functions $t(v/q)$, which express the travel time on a link as a function of the link traffic volume v and its capacity q.

Many different types of volume-delay functions have been proposed and used in practice in the past. One of the most common link travel time formulas is the Bureau of Public Roads (BPR) but other examples are the ones defined by Spiess (1990), Mosher (1963) or Davidson (1966) (see Appendix A in Chapter 3 for a review on these and other link performance functions).

Most existing traffic assignment models consider that there is no overtaking, that is, that the FIFO rule holds. In this paper we assume that we have different classes of users who have different mean velocities and thus, we permit vehicle overtaking. Since congestion affects overtaking, this effect must be considered in the link travel time functions of the different users.

Overtaking is possible and frequent under free flow conditions but becomes difficult or impossible under high congestion. However, the associated difficulties depend on the type of vehicle. For example, motorcycles have less overtaking difficulties than cars, and cars less than trucks. All these features must be included in the mathematical models. From an overtaking point of view, we can consider different classes for bicycles, motorcycles, cars and trucks, but we can also consider different classes for each of these types of vehicles due to the fact that not all users in the same class behave in the same manner. However, congestion usually produces no differences among these classes.

In this section we propose a travel time function family that satisfies an important condition from the overtaking point of view. This property consists of producing different

travel times for the different classes for mild congestion, but the same asymptotic behavior when a high congestion is present. To reproduce this situation, we propose the following link travel time function formula for the α class users:

$$h_\alpha(x) = t(x; t_0, \beta_0, \gamma_0)F(x) + t(x; t_\alpha, \beta_\alpha, \gamma_\alpha)(1 - F(x)), \tag{8.1}$$

where $t(x; t_\alpha, \beta_\alpha, \gamma_\alpha)$ is a BPR link travel time function specific for each α class that reproduces the class user behavior under mild congestion, $t(x; t_0, \beta_0, \gamma_0)$ is a common link travel time function for all classes that represents the common asymptotic behavior under high congestion of all users, $F(x)$ is a cumulative distribution function (cdf), $t_0, t_\alpha, \beta_0, \beta_\alpha, \gamma_0 > 1, \gamma_\alpha > 1$ are constants, and we make the following assumptions

$$t_0 > t_\alpha > 0; \quad \beta_0 \geq \beta_\alpha > 0; \quad \gamma_0 \geq \gamma_\alpha > 0, \tag{8.2}$$

that guarantees that the function

$$g(x) = t(x; t_0, \beta_0, \gamma_0) - t(x; t_\alpha, \beta_\alpha, \gamma_\alpha) \tag{8.3}$$

is nonnegative, strictly increasing and convex, that is,

$$g(x) > 0, \quad g'(x) > 0; \quad \text{and} \quad g''(x) > 0, \quad x > 0, \tag{8.4}$$

three interesting properties to be used later.

Due to the fact that in this case we have different types of users and all of them contribute to congestion, the congestion ratio x must take this into account. So, we assume that there are some weights w_α that allow calculating congestion. In other words, we evaluate the congestion level x as follows:

$$x = \frac{\sum\limits_{\alpha} w_\alpha v_\alpha}{q}, \tag{8.5}$$

where v_α is the traffic volume of class α and q is the link capacity.

For example, we can assume that a car, the reference vehicle with $w_1 = 1$, is equivalent to $w_2 = 1.25$ trucks, to $w_3 = 0.7$ motorbikes or to $w_4 = 2$ lorries, in terms of congestion[2].

Since the BPR functions are particular cases of (8.1), they are generally recognized as physically valid and the cdf $F(x)$ offers a high degree of freedom, we consider that the newly proposed family of link travel time functions are physically valid too. However, since we have no real data or literature based support for this statement, an experimental check in the future would be necessary. Just as BPR functions have been validated and used in practice for decades, we have reasons to believe that this set of newly proposed link time functions can be validated and calibrated. If overtaking is not an important feature

[2]We take the car as the reference vehicle because the BPR function has been designated and calibrated mostly for cars.

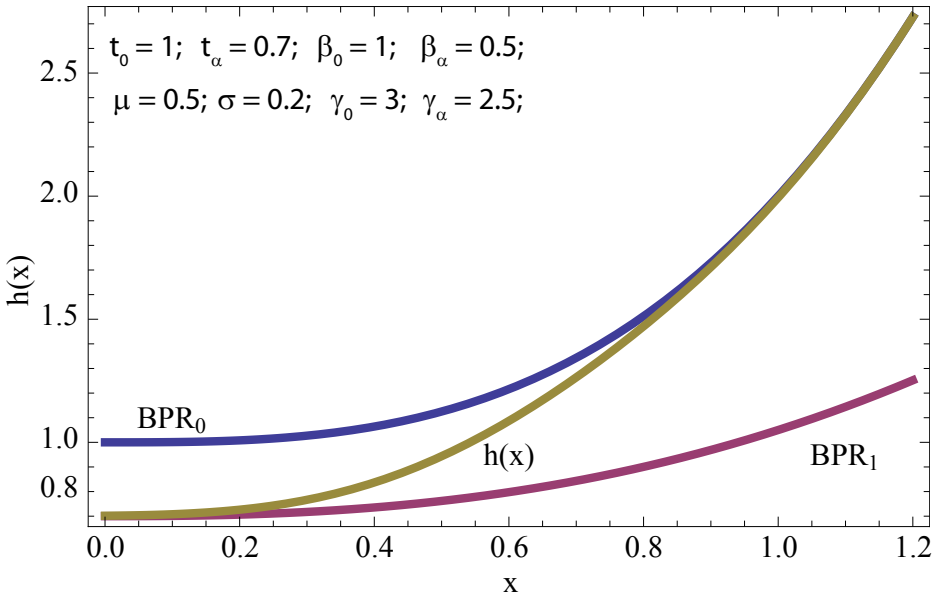


Figure 8.1: Illustration of how the proposed link travel time function is obtained from two BPR functions BPR_0 and BPR_α.

for the scenario, the new link travel time functions will fall back nicely into existing BPR functions.

It is also convenient to calibrate model (8.1) with the above constraints by selecting adequate families of cdfs and fitting the corresponding parameters by running the corresponding experiments. However, this is not the aim of this study.

Figure 8.1 illustrates how the proposed link travel time function is obtained from two BPR functions BPR_0 $(t(x; t_0, \beta_0, \gamma_0))$ and BPR_α $(t(x; t_\alpha, \beta_\alpha, \gamma_\alpha))$.

The rationale behind (8.1) is that for each congestion ratio $x = v/q$, the travel time is a linear convex combination of two travel time functions but the weights change with x going from 0 to 1 as x increases. Using the same t_0, β_0 and γ_0 values and different t_α, β_α and γ_α parameters, we obtain travel time functions that practically coincide for high congestion levels (in fact they converge to the BPR function $t(x; t_0, \beta_0, \gamma_0)$).

One interesting particular case is obtained when $F(x)$ corresponds to the normal distribution, that is,

$$h_\alpha(x) = t(x; t_0, \beta_0, \gamma_0)\Phi\left(\frac{x - \mu_\alpha}{\sigma_\alpha}\right) + t(x; t_\alpha, \beta_\alpha, \gamma_\alpha)\left(1 - \Phi\left(\frac{x - \mu_\alpha}{\sigma_\alpha}\right)\right). \tag{8.6}$$

The rationale behind the use of the normal distribution is that it has an infinite range and it is smooth, so that the linear combinations in (8.6) become smooth if the link travel time functions are smooth too. The normal distribution has some advantages and some

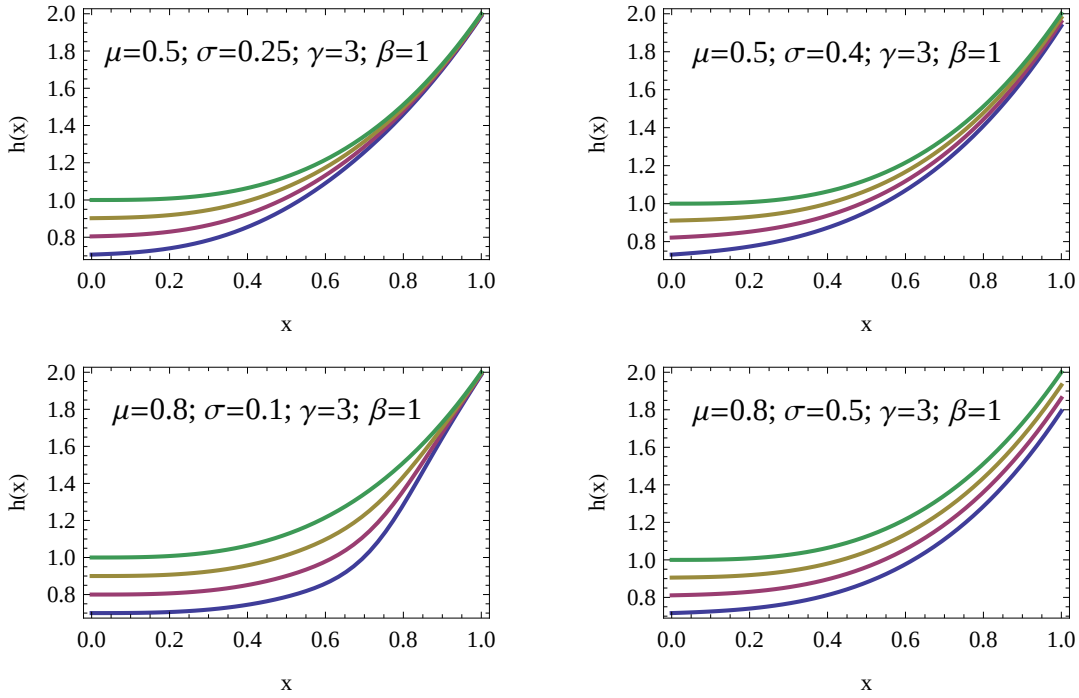


Figure 8.2: Some illustrative examples of the proposed link travel time function based on BPR functions.

disadvantages. The motivation described earlier suggests that an elegant choice of $F(x)$ should be such that at free flow $h(0) = BPR_1(0)$, i.e. that $F(0) = 0$. This is not the exact case for the normal distribution, although values fairly close can be easily obtained as those in the examples of the following sections. Perhaps a non-negative distribution, such as the exponential, gamma or the beta distributions, would fit here more naturally. The same observation could be done for the congested part of the curve, in the sense that all classes should have the same function when the congestion is larger than a certain value and not only asymptotically equal.

Figure 8.2 shows some examples of these curves for different parameter values and $t_0 = 1; t_\alpha = 1, 0.9, 0.8$ and 0.7 based on BPR functions. The examples have been obtained for normal $F(x)$ (Expression (8.6)) functions with mean and standard deviations shown in the corresponding figures.

The link travel time function models some type of interaction between vehicles that can be quantified by the cdf $F(x)$ function (8.1) or the cdf $\Phi(x)$ in (8.6). This permits locating the start of the interaction, its level and the behavior in the congestion zone by playing with the mean and the standard deviation of $F(x)$ or μ and σ in the case of the normal distribution case.

Other type of interaction is included in the link travel time function due to congestion, since we add the link volumes of all class users. A different weight could be given to the different users, but we have left this for a future study.

8.2.1 Some convenient properties of link travel time functions

According to Spiess (1990) the requirements for a well behaved congestion function are:

1. $t(x)$ must be strictly increasing. This is a necessary condition for the assignment to converge to a unique solution.

2. $t(0) = t_0$ and $t(1) = (1+\beta)t_0$. These conditions are convenient in order to facilitate a comparison with the BPR functions. The travel time for a congestion ratio $x = v/b = 1$ must be $(1+\beta)$ times the free travel time t_0.

3. $t'(x)$ is strictly increasing. This ensures the more than desirable property of convexity of the congestion function, because congestion causes not only an increase of the link travel time but an increase of its derivative too. This is an engineering condition, which is not required from a mathematical point of view.

4. The evaluation of $t(x)$ should not take much more computing time than the evaluations of the corresponding BPR functions take.

5. $t'(x) < M\alpha$, where M is a positive constant. The steepness of the congestion curve is limited. This limits the travel time functions not to get too high when congestion ratios $x = v/q$ become much higher than 1[3]. It has no physical sense to have congestion ratios larger than a given value, say 5, for example. In such a cases, the traffic becomes blocked.

6. $t'(0) > 0$. This condition guarantees uniqueness of the link volumes. It also renders the assignment stable. We note that the BPR does not comply with this condition, but still the optimization is strictly convex, so the solution is unique.

These conditions are satisfied by the Spiess conical volume-delay functions (Equation 3.148), and the BPR functions (Equation 3.147) satisfy conditions 1 to 4.

8.2.1.1 Properties of the proposed link travel time functions

Next, we provide conditions under which our proposed functions in (8.1) satisfy these requirements or are close to it. However, we note that some requirements though important are not relevant to the models proposed.

[3]The BPR function derivative is not bounded.

1. $h_\alpha(x); x > 0$ is strictly increasing. Since the two components $t(x; t_0, \beta_0, \gamma_0)$ and $t(x; t_\alpha, \beta_\alpha, \gamma_\alpha)$ are strictly increasing, then any linear convex combination of increasing functions is increasing:

$$\begin{aligned} h'_\alpha(x) &= t'(x;\theta_0)F(x) + t(x;\theta_0)F'(x) + t'(x;\theta_\alpha)(1-F(x)) - t(x;\theta_\alpha)F'(x), \\ &= (t'(x;\theta_0) - t'(x;\theta_\alpha))F(x) + (t(x;\theta_0) - t(x;\theta_\alpha))F'(x) + t'(x;\theta_\alpha) \\ &= g'(x)F(x) + g(x)F'(x) + t'(x;\theta_\alpha), \end{aligned} \tag{8.7}$$

   where for the sake of notation simplicity we have denoted $\theta_\alpha = (t_\alpha, \beta_\alpha, \gamma_\alpha)$, from which we conclude that $h'_\alpha(x) > 0$, because according to (8.4) all its terms and factors are non-negative.

2. $h_\alpha(0) = t_0$ and $h_\alpha(1) = (1+\beta)t_0$. This condition has no meaning in our case. However, we have

$$h_\alpha(0) = t_0F(0) + t_\alpha(1-F(0)) \tag{8.8}$$

$$h_\alpha(1) = t_0(1+\beta_0)F(1) + t_\alpha(1+\beta_\alpha)(1-F(1)), \tag{8.9}$$

   that for $F(0) \approx 0$ and $F(1) \approx 1$ and $\beta_0 = 1$ leads to

$$h_\alpha(0) \approx t_\alpha; \quad h_\alpha(1) \approx 2t_0. \tag{8.10}$$

3. $h_\alpha(x)$ is convex, that is, $h''_\alpha(x) > 0; x > 0$. For the case of normal distributions ($F(x) = \Phi((x-\mu)/\sigma)$), it is demonstrated in Appendix A that this property holds for $x \le x_{max}$ if

$$\sigma^2 \ge \frac{g(x_{max})}{t''(x_{max};\theta_\alpha)\sqrt{2e\pi}}.$$

   However, a better bound for σ can be immediately obtained by solving the problem:

$$\underset{\sigma,x}{\text{Minimize}}\, \sigma \tag{8.11}$$

   subject to

$$\begin{aligned} g''(x)\Phi((x-\mu)/\sigma) + 2g'(x)\Phi'((x-\mu)/\sigma) & \\ + g(x)\Phi''((x-\mu)/\sigma) + t''(x;\theta_\alpha) &\ge 0 \qquad (8.12) \\ x &\le x_{max}. \qquad (8.13) \end{aligned}$$

   If σ^* is the optimum of this problem, then for any $\sigma > \sigma^*$, $h_\alpha(x)$ satisfies convexity.

4. The evaluation of $h_\alpha(x)$ should not take more computing time than the evaluations of the corresponding BPR functions take. In our case we need to evaluate two BPR functions and the normal cdf.

5. $h'_\alpha(x) < M\gamma$, where M is a positive constant. This property holds for $x < x_{max}$, where x_{max} is a finite bound of x[4].

6. $h'_\alpha(x) > 0$ if $x > 0$. According to (8.1) we have

$$h'_\alpha(x) = (t'(x;\theta_0) - t'(x;\theta_\alpha))F(x) + (t(x;\theta_0) - t(x;\theta_\alpha))F'(x) + t'(x;\theta_\alpha), \tag{8.14}$$

where, according to (8.2) and (8.3), all terms and factors are non-negative and $t'(0;\theta_\alpha)$ is positive, which guarantees that $h'_\alpha(x) > 0$. However, $h'_\alpha(0)$ can be zero.

8.2.2 Estimation of the model parameters from data

Since our proposed travel time function $h_\alpha(x)$ for class α is a local convex combination of two BPR functions, one for low congestion and one for high congestion, we can use this property in the estimation process. Our proposal consists of:

1. Estimate the parameters t_α, β_α and γ_α for small congestion data using only class α users.

2. Estimate the parameters t_0, β_0 and γ_0 for large congestion data using all classes.

3. Estimate μ_α and σ_α using intermediate congestion data of class α users.

To estimate the model parameters it is necessary to have some real data. For that purpose, the methods proposed in Soriguera et al. (2007), Soriguera et al. (2010) and Soriguera and Robusté (2011) are very useful tools that permits obtaining travel time measurements. This important problem of estimating and validating the models requires a detailed analysis and deserves a whole study (see, for example, Kalaee (2010), Suh and Kim (1990)).

8.3 The proposed models

In this section we propose two equivalent models for solving the traffic assignment problem, that is, determining the used routes and the link flows for the different overtaking class users.

8.3.1 A model with path enumeration

Consider a network $(\mathcal{N}, \mathcal{A})$, where $\mathcal{N}$ is the set of nodes and $\mathcal{A}$ is the set of links. Given the origin-destination OD traffic flows τ^α_{ks}, where k is the origin node, s is the destination

[4]The bound x_{max} is easy to choose because it is the maximum congestion ratio with a physical meaning (say 5, for example).

node and α is the user class[5], we can obtain the link flows associated with the different OD-pairs and class users solving the following nonlinear complementary problem (NCP):

$$\sum_{\ell_{ij}\in A}\left[c_{ij}^{\alpha}\left(\sum_{\alpha_1,r_1} f_{r_1}^{\alpha_1}\delta_{ijr_1}\right)\delta_{ijr}\right]-\sum_{ks}\rho_{ks}^{\alpha}\xi_{rks}-\mu_r^{\alpha} = 0; \quad r\in\mathcal{R},\alpha\in\mathcal{H} \tag{8.15}$$

$$\sum_{r} f_r^{\alpha}\xi_{rks} = \tau_{ks}^{\alpha}; \quad k,s\in\mathcal{OD},\alpha\in\mathcal{H} \tag{8.16}$$

$$f_r^{\alpha} \geq 0; \quad r\in\mathcal{R},\alpha\in\mathcal{H} \tag{8.17}$$

$$\mu_r^{\alpha} f_r^{\alpha} = 0; \quad r\in\mathcal{R},\alpha\in\mathcal{H} \tag{8.18}$$

$$\mu_r^{\alpha} \geq 0; \quad r\in\mathcal{R},\alpha\in\mathcal{H}, \tag{8.19}$$

where ℓ_{ij} is the link joining nodes i and j, $c_{ij}^{\alpha}(x)\equiv t(x)$ is the travel time function for class α associated with a traffic volume x, f_r^{α} is the route flow r of class α, δ_{ijr} is the link-route incidence matrix ($\delta_{ijr}=1$ if link ℓ_{ij} belongs to path r, and 0 otherwise), ρ_{ks}^{α} are the travel times associated with all used routes r of OD rs, ξ_{rks} is the OD-route incidence matrix, μ_r^{α} are travel time excesses over the minimum travel times ρ_{ks} of route r of OD rs and class α, and τ_{ks}^{α} is the demand of the OD pair ks and class α. We note that δ_{ijr} and ξ_{rks} do not need α subindex or superindex because the same set of candidate routes have been assumed for all classes no matter whether they are used or not for the different class users.

Since the first term in Equation (8.15) is the route r travel time of a user of class α, this equation together with equation (8.18) expresses that this travel time is the same for all users in the same class and all used routes in the same OD if $\mu_r^{\alpha}=0$.

Equation (8.16) is the route flow balance considering all classes and Equation (8.17) forces the non-negativity of route flows for all classes.

Equation (8.18) forces μ_r^{α} to be null when $f_r^{\alpha}>0$ and f_r^{α} to be null when $\mu_r^{\alpha}>0$ for all routes and classes. Finally, Equation (8.19) forces the non-negativity of route travel time excesses for all routes and classes, i.e., all route travel times must be larger or equal that the corresponding OD travel time (ρ_{ks}^{α}).

According to Ran and Boyce (1996), the system (8.15)-(8.19) is equivalent to the following variational inequality problem (VIP) (see Theorem 11 in Chapter 3):

$$(\mathbf{f}-\mathbf{f}^*)^T\cdot\mathbf{n}(\mathbf{f}^*)\geq 0; \quad \forall\mathbf{f}\in\Omega_f, \tag{8.20}$$

where $\mathbf{f}$ is the vector of route flows f_r^{α}, $\mathbf{n}(\mathbf{f}^*)=\sum_{\ell_{ij}\in A}\left[c_{ij}^{\alpha}\left(\sum_{\alpha_1,r_1} f_{r_1}^{\alpha_1}\delta_{ijr_1}\right)\delta_{ijr}\right]$ is vector of route costs, and

$$\Omega_f=\left\{\mathbf{f}\,\middle|\,\sum_r f_r^{\alpha}\xi_{rks}=\tau_{ks}^{\alpha}; \quad \forall k,s,\alpha; \quad f_r^{\alpha}\geq 0; \quad \forall r,\alpha\right\}. \tag{8.21}$$

[5]The set of all OD pairs and α classes are $\mathcal{OD}$ and $\mathcal{H}$, respectively.

The VIP (8.20)-(8.21) for the proposed travel time functions has the following properties:

1. The function $\mathbf{n}(\mathbf{f})$ is continuous and monotone in the f_r^α variables.

2. The feasible set Ω_f is convex, bounded and closed (compact).

Unfortunately, since $\mathbf{n}(\mathbf{f})$ is not strictly monotone[6], it normally has infinitely many solutions. However, the solution is unique in the $\boldsymbol{\rho}$ variables (see Lo and Chen (2000) and theorems 5 and 6 in Chapter 3).

The problem (8.15)-(8.19) cannot be stated as a Beckmann et al. (1956) formulation of a Wardrop equilibrium problem because the lack of symmetry of the Jacobian of the function $\mathbf{n}(\mathbf{f})$ (see Wardrop (1952), Beckmann et al. (1956) and Nagurney (1999)).

Once the VIP (8.15)-(8.19) is solved, that is, the values of f_r^α have been obtained, the OD α-class link flows can be obtained as:

$$v_{ijks}^\alpha = \sum_r f_r^\alpha \delta_{ijr} \xi_{rks} \quad \ell_{ij} \in \mathcal{A}, k, s \in \mathcal{OD}, \alpha \in \mathcal{H} \tag{8.22}$$

which is a class-specific link flow related to a single OD pair.
Expressions (8.15) and (8.18) show that:

1. Used routes of the same OD and class share the same travel time.

2. Unused routes have associated travel times larger than used routes.

8.3.1.1 Alternative model

The NCP (8.15)-(8.19) has the practical inconvenience of having multiplicity of solutions, which is a serious inconvenience in practice. Thus, we suggest solving the alternative problem (this is what we have done in all the examples presented later):

$$\begin{aligned} \sum_{\ell_{ij} \in A} \left[c_{ij}^\alpha \left(\sum_{\alpha_1, r_1} f_{r_1}^{\alpha_1} \delta_{ijr_1} \right) \delta_{ijr} \right] + \eta \sum_{r_1, \alpha_1} f_{r_1}^{\alpha_1} \log(f_{r_1}^{\alpha_1} + \epsilon) & \\ - \sum_{ks} \rho_{ks}^\alpha \xi_{rks} - \mu_r^\alpha &= 0; \quad r \in \mathcal{R}, \alpha \in \mathcal{H} \quad (8.23) \\ \sum_r f_r^\alpha \xi_{rks} &= \tau_{ks}^\alpha; \quad \begin{matrix} k, s \in \mathcal{OD}, \\ \alpha \in \mathcal{H} \end{matrix} \quad (8.24) \\ f_r^\alpha &\geq 0; \quad r \in \mathcal{R}, \alpha \in \mathcal{H} \quad (8.25) \\ \mu_r^\alpha f_r^\alpha &= 0; \quad r \in \mathcal{R}, \alpha \in \mathcal{H} \quad (8.26) \\ \mu_r^\alpha &\geq 0; \quad r \in \mathcal{R}, \alpha \in \mathcal{H}, \quad (8.27) \end{aligned}$$

[6] If two routes have no common links, it is possible that a positive change in one of of the route flows produces no change in the other route travel time.

where η is a very small, with respect to the f_r^α values, positive number such that constraint (8.23) is practically no modified, and ϵ is a small value to guarantee $\log f_r^\alpha$ to be bounded.

The entropy function $\sum_{r,\alpha} f_r^\alpha \log(f_r^\alpha + \epsilon)$ has been added to constraint (8.23) in order to get strict monotonicity. Note that because η has been assumed very small the resulting link flows in Problems (8.15)-(8.19) and (8.23)-(8.27) are identical, that is, the entropy term does not imply a change in link flows.

The VIP (8.23)-(8.27) has the following properties:

1. The function $\sum_{\ell_{ij}\in A}\left[c_{ij}^\alpha\left(\sum_{\alpha_1,r_1} f_{r_1}^{\alpha_1}\delta_{ijr_1}\right)\delta_{ijr}\right] + \eta\sum_{r_1,\alpha_1} f_{r_1}^{\alpha_1}\log(f_{r_1}^{\alpha_1}+\epsilon)$ is continuous and strictly monotone in the f_r^α variables.

2. The feasible set defined by (8.24) and (8.25) is convex, bounded and closed (compact).

Thus, according to Theorem 10 in Chapter 3 the VIP (8.23)-(8.27) has a unique solution.

The Problem (8.23)-(8.27) is equivalent to the optimization problem (see Lo and Chen (2000)):

$$\underset{f_r^\alpha,\mu_r^\alpha,\rho_{ks}^\alpha}{\text{Minimize}} \sum_{r,\alpha}\mu_r^\alpha f_r^\alpha \tag{8.28}$$

subject to (8.23)-(8.25) and (8.27), where (8.28) is convex if the route travel time function is convex and monotone with respect to route flows but not always the constraint (8.27) is convex (see Lo and Chen (2000)).

Note that the objective function (8.28) replaces the constraint (8.26) and has a zero optimal value of the objective function if the problem (8.23)-(8.27) has a solution. This can be seen as a gap function (see Lo and Chen (2000) or the section devoted to their model in Chapter 3).

Proposition 10 *Path flows of two classes that share two or more routes are not unique.*

Proof. Suppose that the path flows $f_{r_1}^{\alpha_1}, f_{r_1}^{\alpha_2}, f_{r_2}^{\alpha_1}, f_{r_2}^{\alpha_2}$ corresponding to two different classes of users α_1, α_2 and two different paths r_1, r_2 are not null, in other words, that classes α_1 and α_2 share paths r_1 and r_2.

In that case, path flows $f_{r_1}^{\alpha_1}, f_{r_1}^{\alpha_2}, f_{r_2}^{\alpha_1}, f_{r_2}^{\alpha_2}$ are not unique as it is possible to find new path flows $f'^{\alpha_1}_{r_1}, f'^{\alpha_2}_{r_1}, f'^{\alpha_1}_{r_2}, f'^{\alpha_2}_{r_2}$ that satisfy constraints (8.23)-(8.25) and (8.27) and lead to the same value of the objective function (8.28). Those new path flows must be of the form

$$f'^{\alpha_1}_{r_1} = f_{r_1}^{\alpha_1} + \epsilon, \quad f'^{\alpha_1}_{r_2} = f_{r_2}^{\alpha_1} - \epsilon, \quad f'^{\alpha_2}_{r_1} = f_{r_2}^{\alpha_1} - \epsilon, \quad f'^{\alpha_2}_{r_2} = f_{r_2}^{\alpha_2} + \epsilon,$$

where $|\epsilon| < min(f_{r_1}^{\alpha_1}, f_{r_1}^{\alpha_2}, f_{r_2}^{\alpha_1}, f_{r_2}^{\alpha_2})$. ∎

8.3.2 A model without path enumeration

Since path enumeration is complicated in practice, in this section we state a problem for which the path enumeration is not required. This has evident practical advantages.

The problem (8.15)-(8.19) can be stated without the need of path enumeration as follows:

$$0 = c_{ij}^{\alpha}\left(\sum_{k_1,s_1,\alpha_1} v_{ijk_1s_1}^{\alpha_1}\right) + \lambda^{iks\alpha} - \lambda^{jks\alpha} - \mu_{ijks\alpha}; \quad \begin{array}{l}\ell_{ij}\in\mathcal{A};\ \ \alpha\in\mathcal{H};\\ k,s\in\mathcal{OD}\end{array} \tag{8.29}$$

$$\tau_{ks}^{\alpha}(\delta_{ik}^{*}-\delta_{is}^{*}) = \sum_{\ell_{ij}\in\mathcal{A}} v_{ijks}^{\alpha} - \sum_{j|\ell_{ji}\in\mathcal{A}} v_{jiks}^{\alpha}; \quad i\in\mathcal{N}; \quad k,s\in OD; \quad \alpha\in\mathcal{H} \tag{8.30}$$

$$0 \le v_{ijks}^{\alpha} \le \tau_{ks}^{\alpha}; \quad \ell_{ij}\in\mathcal{A}; \ \ k,s\in OD; \quad \alpha\in\mathcal{H} \tag{8.31}$$

$$\mu_{ijks\alpha}v_{ijks}^{\alpha} = 0; \quad \ell_{ij}\in\mathcal{A}; \quad k,s\in OD; \quad \alpha\in\mathcal{H} \tag{8.32}$$

$$\mu_{ijks\alpha} \ge 0; \quad \ell_{ij}\in\mathcal{A}; \quad k,s\in OD; \quad \alpha\in\mathcal{H}, \tag{8.33}$$

where v_{ijks}^{α} is the α-class flow through link ℓ_{ij} with origin node k and destination node s, $\lambda^{iks\alpha}$ are dual variables which allow obtaining the link ℓ_{ij} travel time as $\lambda^{jks\alpha}-\lambda^{iks\alpha}$, $\mu_{ijks\alpha}$ are the link travel time excesses over the minimum, and δ_{ij}^{*} are the Kronecker deltas. We note that though constraint (8.31) is not necessary it is convenient to accelerate the numerical solution of the problem.

Equations (8.30) represent the flow balance associated with the OD-pair k,s, for any node, that includes input node k, output node s, and all intermediate nodes i (i different from nodes k and s). It is a compact form of writing the balance equations for three different types of nodes: input, output and intermediate nodes.

Note that if the link travel time function $c_{ij}(\cdot)$ can be expressed as $\lambda^{iks\alpha}-\lambda^{jks\alpha}$, then the route travel times, which are the sums of the corresponding link travel times, are dependent only on their origins and destinations. Consequently, Equation (8.29) for $\mu_{ijks\alpha}=0$ together with (8.32) imply that all used routes of the same OD share the same travel time.

According to Ran and Boyce (1996), the system (8.29)-(8.33) is equivalent to the following variational inequality problem (VIP):

$$(\mathbf{v}-\mathbf{v}^{*})^{T}\cdot\mathbf{m}(\mathbf{v}^{*}) \ge 0; \quad \forall\mathbf{v}\in\Omega_v, \tag{8.34}$$

where $\mathbf{v}$ is the vector of disaggregated class link flows v_{ijks}^{α}, $\mathbf{m}(\mathbf{v})$ is the vector of class link costs $m_{ij}^{\alpha} = c_{ij}^{\alpha}\left(\sum_{k,s,\alpha_1} v_{ijks}^{\alpha_1}\right)$, and

$$\Omega_v = \left\{ \mathbf{v} \;\middle|\; \begin{array}{l} \tau_{ks}^{\alpha}(\delta_{ik}-\delta_{is}) = \sum_{\ell_{ij}\in\mathcal{A}} v_{ijks}^{\alpha} - \sum_{j\in\ell_{ji}} v_{jiks}^{\alpha}; \ \ i\in\mathcal{N}; \ \ k,s\in\mathcal{OD}; \\ 0\le v_{ijks}^{\alpha}\le\tau_{ks}^{\alpha}; \ell_{ij}\in\mathcal{A}; \ \ k,s\in\mathcal{OD}; \ \ \alpha\in\mathcal{H} \end{array} \right\}. \tag{8.35}$$

The VIP (8.34)-(8.35) has the following properties:

1. The function $\mathbf{m}(\mathbf{v})$ is continuous and monotone in the v^{α}_{ijks} variables.

2. The feasible set Ω_v is convex, bounded and closed (compact).

Unfortunately, since the function $\mathbf{m}(\mathbf{v})$ is not strictly monotone, it normally has infinitely many solutions. However, it has unique solutions for the link flows.

Again, the problem (8.34)-(8.35) cannot be stated as a Beckmann et al. formulation of a Wardrop equilibrium problem because the lack of symmetry of the Jacobian of the function $\mathbf{m}(\mathbf{v})$.

Equations (8.29)-(8.33) lead to

$$c^{\alpha}_{ij}\left(\sum_{k_1,s_1,\alpha_1} v^{\alpha_1}_{ijk_1s_1}\right) \geq \lambda^{jks\alpha} - \lambda^{iks\alpha} \perp v^{\alpha}_{ijks} \geq 0; \quad \ell_{ij} \in \mathcal{A};\ k,s \in \mathcal{OD};\ \alpha \in \mathcal{H} \tag{8.36}$$

$$\tau^{\alpha}_{ks}(\delta^{*}_{ik} - \delta^{*}_{is}) = \sum_{\ell_{ij}\in\mathcal{A}} v^{\alpha}_{ijks} - \sum_{j|\ell_{ji}\in\mathcal{A}} v^{\alpha}_{jiks}; \quad i \in \mathcal{N};\ k,s \in OD;\ \alpha \in \mathcal{H} \tag{8.37}$$

From (8.32) and (8.36) for used links we have:

$$c^{\alpha}_{ij}\left(\sum_{k_1,s_1,\alpha_1} v^{\alpha_1}_{ijk_1s_1}\right) = \lambda^{jks\alpha} - \lambda^{iks\alpha}; \quad \ell_{ij} \in \mathcal{A}; \quad \alpha \in \mathcal{H} \tag{8.38}$$

and this permits obtaining the minimum cost (travel time) q^{α}_{ij} from a node i to a node j based on the $\lambda^{iks\alpha}$ variables, when the corresponding path exists, as follows:

$$q^{\alpha}_{ij} = \lambda^{jks\alpha} - \lambda^{iks\alpha}; \quad (i,j) \in \mathcal{N}\times\mathcal{N}; \quad \forall\alpha. \tag{8.39}$$

The existence of these dual variables is based on the fact that the costs to reach a destination node s from another origin node k is independent on the chosen used path. This means that all paths from k to s have the same associated costs for a given α-class.

Once the values of the OD link flows v^{α}_{ijks} have been estimated solving the VIP problem (8.29)-(8.33), we can easily calculate important flow information. Thus, the use of this statement of the flow problem using the set of variables v^{α}_{ijks} has the following important advantages:

1. It avoids path enumeration.

2. One can easily calculate the flows x^{α}_{ijk}, y^{α}_{ijs}, w^{α}_{ij}, z^{α}_{iks}, already defined in Chapter 3.

3. One can identify and/or enumerate the used flow paths very easily assuming that the used paths do not include directed loops. To enumerate paths of an OD-pair k,s for α-class users, one can simply build the tree with non-null flow ($v^{\alpha}_{ijks} = 0$)

branches (links) starting from the origin node k and ending with the destination node s (see Figures 8.4, 8.6, 8.7 and 8.8, for example, to see how simple is to obtain all used paths associated with a given OD pair by combining the used links in all possible forms).

4. The problem can be solved as a standard non-linear programming problem using software packages as GAMS, for example, without the need of a bi-level technique. We have used GAMS and the results were very precise. However, the methods must be checked with large networks before concluding that they will have the same properties for these networks. In particular, a comparison with other existing algorithms in efficiency and/or accuracy, as the established methods of Frank-Wolfe or the more recent origin-based methods of Bar-Gera (2002) must be done in future work.

8.3.2.1 Alternative model

Since the VIP (8.29)-(8.33) has the practical inconvenience of having multiplicity of solutions, we suggest solving the NCP:

$$\begin{aligned} 0 &= c_{ij}^{\alpha}\left(\sum_{k_1,s_1,\alpha_1} v_{ijk_1s_1}^{\alpha_1}\right)+\eta\sum_{i_1j_1k_1s_1\alpha_1} v_{i_1j_1k_1s_1}^{\alpha_1}\log(v_{i_1j_1k_1s_1}^{\alpha_1}+\epsilon) \\ &\quad +\lambda^{iks\alpha}-\lambda^{jks\alpha}-\mu_{ijks\alpha};\quad \ell_{ij}\in\mathcal{A};\quad k,s\in\mathcal{OD};\quad \alpha\in\mathcal{H} \quad (8.40)\\ \tau_{ks}^{\alpha}(\delta_{ik}-\delta_{is}) &= \sum_{\ell_{ij}\in\mathcal{A}} v_{ijks}^{\alpha}-\sum_{j\in\ell_{ji}} v_{jiks}^{\alpha};\quad i\in\mathcal{N};\quad k,s\in\mathcal{OD};\quad \alpha\in\mathcal{H} \quad (8.41)\end{aligned}$$

$$\begin{aligned} 0 &\le v_{ijks}^{\alpha}\le\tau_{ks}^{\alpha};\quad \ell_{ij}\in\mathcal{A};\ \ k,s\in\mathcal{OD};\quad \alpha\in\mathcal{H} \quad (8.42)\\ \mu_{ijks\alpha}v_{ijks}^{\alpha} &= 0;\quad \ell_{ij}\in\mathcal{A};\ \ k,s\in\mathcal{OD};\quad \alpha\in\mathcal{H} \quad (8.43)\\ \mu_{ijks\alpha} &\ge 0;\quad \ell_{ij}\in\mathcal{A};\quad k,s\in\mathcal{OD};\quad \alpha\in\mathcal{H}, \quad (8.44)\end{aligned}$$

where the entropy term $\sum_{ijks\alpha} v_{ijks}^{\alpha}\log(v_{ijks}^{\alpha}+\epsilon)$ has been added to constraint (8.40). After this, the new function

$$c_{ij}^{\alpha}\left(\sum_{k_1,s_1,\alpha_1} v_{ijk_1s_1}^{\alpha_1}\right)+\eta\sum_{i_1j_1k_1s_1\alpha_1} v_{i_1j_1k_1s_1}^{\alpha_1}\log(v_{i_1j_1k_1s_1}^{\alpha_1}+\epsilon)$$

is strictly monotone, and then, similarly to the previous cases, we have uniqueness of solution for the Problem (8.40)-(8.44).

We note that the problem (8.15)-(8.19) is not equivalent to the problem (8.40)-(8.44) due to the use of a different entropy term in their objective function. However, an equivalent entropy link formulation could be formulated based on Akamatsu (1997).

Similarly to the case with path enumeration, we can replace Problem (8.40)-(8.44) by the following equivalent optimization problem:

$$\underset{\mathbf{v},\boldsymbol{\mu},\boldsymbol{\lambda}}{\text{Minimize}}\, Z = \sum_{\ell_{ij}\in A;k,s\in OD;\alpha} \mu_{ijks\alpha} v^{\alpha}_{ijks} \tag{8.45}$$

subject to (8.40)-(8.42) and (8.44).

The equivalence of Problems (8.29)-(8.33) and (8.45) subject to (8.40)-(8.42) and (8.44) relies on the fact that the objective function (8.45) must be non-negative and at the same time null because of (8.41).

8.3.3 Measuring relative accuracy

To examine the convergence of the solution procedure we use the aggregate "relative duality gap" (see Murchland (1969), Rose et al. (1988), Janson (1991)):

$$rel_{gap} = \frac{\sum\limits_{r,\alpha} f^{\alpha}_{r} c^{\alpha}_{r} - \sum\limits_{k,s,\alpha} \tau^{\alpha}_{ks} \rho^{\alpha}_{ks}}{\sum\limits_{k,s,\alpha} \tau^{\alpha}_{ks} \rho_{ks} \alpha}, \tag{8.46}$$

where c^{α}_{r} is the travel time of a user of class α traveling route r, τ^{α}_{ks} is the OD flow associated with OD ks and user of class α, and ρ^{α}_{ks} is the minimum travel time of OD ks and class α.

Some authors have suggested that this relative gap needs to be smaller than 10^{-5} in order to validate the results. This measure has been widely used to test the convergence of iterative UE procedures. In our case, it permits knowing if the solutions follow the equilibrium conditions. In the examples we have computed this measure in order to check the accuracy of the results.

8.4 Example of applications

In this section we present two examples to illustrate the proposed methods and show their power and suitability for practical applications. We start with a simple example and end with a real case network example.

8.4.1 The Nguyen-Dupuis network

With the purpose of illustrating the proposed methods, they are applied to the well known Nguyen-Dupuis network previously used, which is shown in Figure 8.3. In this case, we consider unidirectional links and, hence, the network has 13 nodes and 19 links. We have selected this simple example in order to be able to show the results in form of tables and figures of reasonable size.

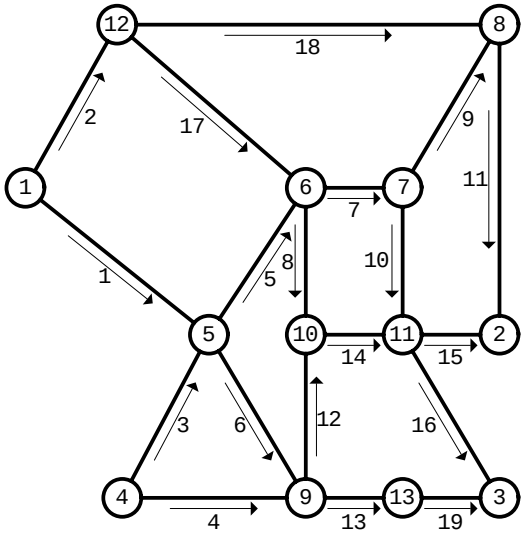


Figure 8.3: The Nguyen-Dupuis network with unidirectional links.

Link	ν_0	c	Link	ν_0	c
1 -5	7.00	700.00	8 -2	9.00	700.00
1 -12	9.00	560.00	9 -10	10.00	280.00
4 -5	9.00	560.00	9 -13	9.00	280.00
4 -9	12.00	280.00	10-11	6.00	700.00
5 -6	3.00	420.00	11-2	9.00	280.00
5 -9	9.00	420.00	11-3	8.00	560.00
6 -7	5.00	700.00	12-6	7.00	140.00
6 -10	5.00	280.00	12-8	14.00	560.00
7 -8	5.00	700.00	13-3	11.00	560.00
7 -11	9.00	700.00			

Table 8.1: Parameters of the Nguyen-Dupuis network.

8.4.1.1 Homogeneous users

The data used in this example are shown in Table 8.1, where ν_0 (to be defined) and capacity c constants used for each link ℓ_{ij} are shown.

Initially, for illustration purposes, we have selected the following OD and OD flows:

$$\tau_{12} = 200; \quad \tau_{13} = 256; \quad \tau_{42} = 192; \quad \tau_{43} = 120,$$

which have been distributed equally among all classes.

We have used the link travel time functions (8.1) with $\mu_\alpha = 0.5$, $\sigma_\alpha = 0.25$, $\beta_0 = \beta_\alpha = 1$, $\gamma_0 = \gamma_\alpha = 3$, $t_0 = 1$ and the q_α values in the heading of Table 8.3, where we have

With path enumeration		
Problem I		
	overtaking	
Problem	no	yes
(8.28) subject to (8.23)-(8.25) and (8.27); $\eta = 10^{-10}$	0.016	0.035
Problem II		
	overtaking	
Problem	no	yes
(8.28) subject to (8.23)-(8.25) and (8.27); $\eta = 0$	0.062	0.078
Without path enumeration		
Problem I		
	overtaking	
Problem	no	yes
(8.45) subject to (8.40)-(8.42) and (8.44); $\eta = 10^{-10}$	0.218	3.791
Problem II		
	overtaking	
Problem	no	yes
(8.45) subject to (8.40)-(8.42) and (8.44); $\eta = 0$	0.094	0.234

Table 8.2: Homogeneous users example. Cpu times required to solve the problems indicated by their objective function equation number for the overtaking problems and the classical approaches (no overtaking).

assumed that the free link travel times t_α are given by $t_\alpha = \nu_0 q_\alpha$. This implies that q_α is a factor that provides the free link travel times of a class with respect to the reference class α_0, for which $q_{\alpha_0} = 1$.

In Table 8.2 we provide a list of the solved problems using GAMS with the CONOPT solver and the corresponding cpu required times on a HP Z200 Workstation, Intel Core i7-870 2.93 8MB/1333 QC, RAM: 8GB (2x4GB). We have solved this example using two different problems. Instead of solving the Problem (8.23)-(8.27), we have solved the Problem I: (8.28) subject to (8.23)-(8.25) and (8.27), and instead of solving the Problem (8.40)-(8.44), we have solved the Problem II: (8.45) subject to (8.40)-(8.42) and (8.44).

In order to compare the times required for several classes (with overtaking) with the times required for a single class (no overtaking), we have run the problem with only one class too. As expected the cpu times for the overtaking cases are larger or equal

to the models without overtaking. Similarly, the cpu times for the models without path enumeration are larger or equal to the models with path enumeration.

We note that both problems I and II led to exactly the same solution for all cases, indicating that for this example they are equivalent.

Table 8.3 shows the total link flows (column 2), the link flows disaggregated by OD and α-classes (columns third to eighteenth)[7] and the travel times of each OD and user class (last row), where we can observe the following facts:

1. Different class users of the same OD pair not necessarily use the same routes. For example, in OD $4-3$, all class users utilize the same route, while in the remaining OD pairs, some users choose different routes.

2. The link flows appear disaggregated by OD and classes, which is an important information.

3. The used routes are identified from the set of all possible routes without the need of a previous enumeration. A very reduced number of routes is normally obtained, unless we have a high congestion.

4. The OD travel times for all classes, shown in the last row, are different and they increase with increasing t_α.

Figure 8.4 shows the used routes corresponding to different OD pairs and class users. The interesting result is that emphasizing the links with non-null flow, we can identify all used routes of each OD pair and overtaking user class.

Table 8.4 shows the route travel times classified by OD and α-classes for the Nguyen-Dupuis network. The used routes have been boldfaced. Note that they correspond to minimum values of travel times in the same class and OD and that they coincide with the OD travel times given in Table 8.3 (last row).

Note also that when a class uses more than one route, the associated travel times are identical.

8.4.1.2 Cars and Motorcycles

In this section we illustrate the case of two types of users (cars and motorcycles), such that congestion does not take place at the same congestion ratio. This is for example the case of cars and motorcycles, because motorcycle users normally use the road shoulders to travel when congestion is very high.

The parameters used in the example for the selected 6 classes (three for cars and three for motorcycles) are given in Table 8.5. We assume that when congestion is low cars

[7] Only one decimal place is shown because of lack of space, but the precision is larger than seven decimal digits.

		OD: 1 -2				OD: 1 -3				OD: 4 -2				OD: 4 -3			
		q_α															
Link	Flow	1.00	0.90	0.80	0.70	1.00	0.90	0.80	0.70	1.00	0.90	0.80	0.70	1.00	0.90	0.80	0.70
1 -5	275.0	19.0	-	-	-	64.0	64.0	64.0	64.0	-	-	-	-	-	-	-	-
1 -12	181.0	31.0	50.0	50.0	50.0	-	-	-	-	-	-	-	-	-	-	-	-
4 -5	169.8	-	-	-	-	-	-	-	-	48.0	48.0	48.0	25.8	-	-	-	-
4 -9	142.2	-	-	-	-	-	-	-	-	-	-	-	22.2	30.0	30.0	30.0	30.0
5 -6	423.4	19.0	-	-	-	64.0	64.0	64.0	42.5	48.0	48.0	48.0	25.8	-	-	-	-
5 -9	21.5	-	-	-	-	-	-	-	21.5	-	-	-	-	-	-	-	-
6 -7	231.4	19.0	-	-	-	-	-	-	42.5	48.0	48.0	48.0	25.8	-	-	-	-
6 -10	192.0	-	-	-	-	64.0	64.0	64.0	-	-	-	-	-	-	-	-	-
7 -8	188.9	19.0	-	-	-	-	-	-	-	48.0	48.0	48.0	25.8	-	-	-	-
7 -11	42.5	-	-	-	-	-	-	-	42.5	-	-	-	-	-	-	-	-
8 -2	369.8	50.0	50.0	50.0	50.0	-	-	-	-	48.0	48.0	48.0	25.8	-	-	-	-
9 -10	22.2	-	-	-	-	-	-	-	-	-	-	-	22.2	-	-	-	-
9 -13	141.5	-	-	-	-	-	-	-	21.5	-	-	-	-	30.0	30.0	30.0	30.0
10-11	214.2	-	-	-	-	64.0	64.0	64.0	-	-	-	-	22.2	-	-	-	-
11-2	22.2	-	-	-	-	-	-	-	-	-	-	-	22.2	-	-	-	-
11-3	234.5	-	-	-	-	64.0	64.0	64.0	42.5	-	-	-	-	-	-	-	-
12-6	0.0	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
12-8	181.0	31.0	50.0	50.0	50.0	-	-	-	-	-	-	-	-	-	-	-	-
13-3	141.5	-	-	-	-	-	-	-	21.5	-	-	-	-	30.0	30.0	30.0	30.0
Travel time		34.1	31.8	29.5	27.3	34.9	33.2	31.5	29.4	35.9	33.9	31.9	29.9	34.9	32.8	30.7	28.6

Table 8.3: Mixed BPR model. Link flows disaggregated by OD and α-classes and OD travel times for the Nguyen-Dupuis network.

OD	Routes	path links					Classes			
							1	2	3	4
1- 2	1	1	5	7	9	11	**34.103**	32.318	30.534	28.749
1- 2	2	2	18	11			**34.103**	**31.826**	**29.548**	**27.270**
1- 3	3	1	5	7	10	16	36.267	33.968	31.668	**29.368**
1- 3	4	1	5	8	14	16	**34.869**	**33.190**	**31.510**	29.831
1- 3	5	1	6	13	19		37.765	34.966	32.167	**29.368**
4- 2	6	3	5	7	9	11	**35.930**	**33.914**	**31.898**	**29.881**
4- 2	7	4	12	14	15		38.752	35.795	32.838	**29.881**
4- 3	8	4	13	19			**34.910**	**32.811**	**30.713**	**28.614**

Table 8.4: Route travel times classified by OD and α-classes for the Nguyen-Dupuis network. Used routes are boldfaced.

travel at higher speeds than motorcycles, and that in the presence of high congestion, cars get blocked and motorcycles still can continue traveling until a larger congestion ratio is reached.

Figure 8.4: The Nguyen-Dupuis example. Used routes by different OD pairs and class users.

Figure 8.5 shows the link travel times for the six class users, three for cars and three for motorcycles with different associated saturation ratios. It is interesting to see that for a congestion ratio 0.80, cars have serious difficulties in traveling, while motorcycles still can run.

We have analyzed two different situations: (a) an uncongested case, and (b) a congested one.

For the uncongested case we have selected the same OD pairs and OD flows as in the homogeneous example above. For the congested case, we have assumed the following OD

	class	q	β	γ	μ	σ	t_0	w_α
Cars	1	1	3	5	0.5	0.25	1	1
	2	0.9	3	5	0.5	0.25	1	1
	3	0.8	3	5	0.5	0.25	1	1
Motos	4	1.4	1	3	0.5	0.5	1.4	0.5
	5	1.3	1	3	0.5	0.5	1.4	0.5
	6	1.2	1	3	0.5	0.5	1.4	0.5

Table 8.5: Parameters used in the cars-motorcycles example.

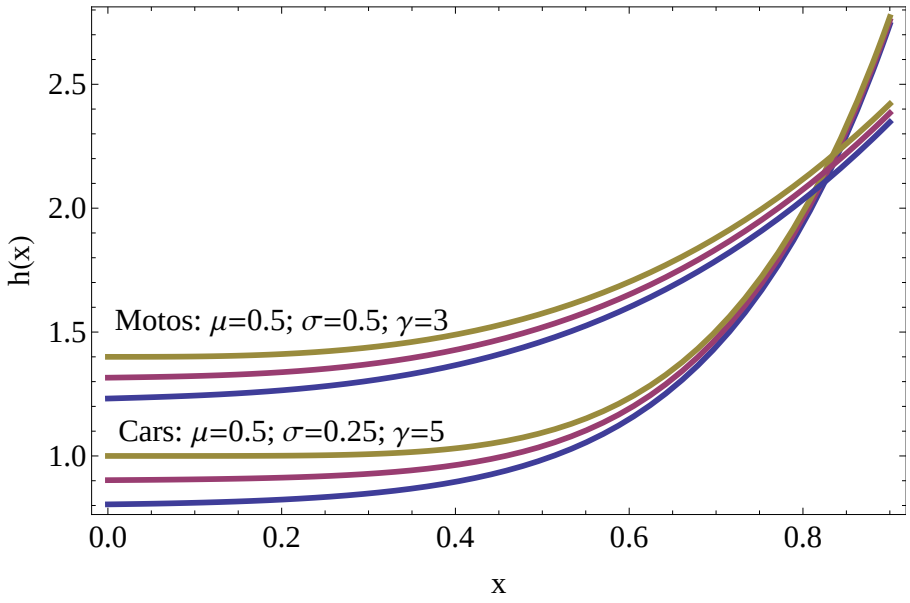


Figure 8.5: Cars and motorcycles example. Link travel times for the six class users, three for cars and three for motorcycles with different associated saturation ratios.

and OD flows:

$$\tau_{12} = 575; \quad \tau_{13} = 736; \quad \tau_{42} = 552; \quad \tau_{43} = 345,$$

which have been distributed equally among all classes.

In Table 8.6 we provide a list of the solved problems using GAMS with the CONOPT solver and the corresponding cpu required times on a HP Z200 Workstation, Intel Core i7-870 2.93 8MB/1333 QC, RAM: 8GB (2x4GB).

In order to compare the times required for the congested and uncongested cases, we have run the problems for both cases. As expected the cpu times for the congested cases

With path enumeration		
Problem I		
	Congestion	
Problem	no	yes
(8.28) subject to (8.23)-(8.25) and (8.27); $\eta = 10^{-10}$	0.188	0.219
Problem II		
	Congestion	
Problem	no	yes
(8.28) subject to (8.23)-(8.25) and (8.27); $\eta = 0$	0.031	0.093

Without path enumeration		
Problem I		
	Congestion	
Problem	no	yes
(8.45) subject to (8.40)-(8.42) and (8.44); $\eta = 10^{-10}$	14.259	22.511
Problem II		
	Congestion	
Problem	no	yes
(8.45) subject to (8.40)-(8.42) and (8.44); $\eta = 0$	0.047	0.733

Table 8.6: Cars and motorcycles example. Cpu times required to solve the problems indicated by their objective function equation number for the congested and uncongested problems.

are larger or equal to the models for uncongested cases. Similarly, the cpu times for the models without path enumeration are larger or equal to the models with path enumeration.

We note that both Problems I and II led to exactly the same solution for all cases, indicating that for this example they are equivalent.

Table 8.7 shows the total link flows (column 2), the link flows disaggregated by OD and α-classes (columns third to twentysixth) and the travel times of each OD and user class (last row) for the uncongested case. We can see that, as expected, cars travel times are smaller than motorcycles travel times because we have no congestion. We also see that different classes of cars and motorcycles use not necessarily the same routes but the travel times are different.

Table 8.8 shows the route travel times classified by OD and α-classes. The used routes have been boldfaced. Note that they correspond to minimum values of travel times in the

		OD: 1 -2						OD: 1 -3						OD: 4 -2						OD: 4 -3					
Link	Flow	1.00	0.90	0.80	1.40	1.30	1.20	1.00	0.90	0.80	1.40	1.30	1.20	1.00	0.90	0.80	1.40	1.30	1.20	1.00	0.90	0.80	1.40	1.30	1.20
1-5	356.0	-	-	-	33.3	33.3	33.3	42.7	42.7	42.7	42.7	42.7	42.7	-	-	-	-	-	-	-	-	-	-	-	-
1-12	100.0	33.3	33.3	33.3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
4-5	183.5	-	-	-	-	-	-	-	-	-	-	-	-	32.0	32.0	23.5	32.0	32.0	32.0	-	-	-	-	-	-
4-9	128.5	-	-	-	-	-	-	-	-	-	-	-	-	-	-	8.5	-	-	-	20.0	20.0	20.0	20.0	20.0	20.0
5-6	517.2	-	-	-	33.3	33.3	33.3	42.7	42.7	20.4	42.7	42.7	42.7	32.0	32.0	23.5	32.0	32.0	32.0	-	-	-	-	-	-
5-9	22.3	-	-	-	-	-	-	-	-	22.3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
6-7	283.5	-	-	-	33.3	33.3	33.3	-	-	-	-	-	-	32.0	32.0	23.5	32.0	32.0	32.0	-	-	-	-	-	-
6-10	233.7	-	-	-	-	-	-	42.7	42.7	20.4	42.7	42.7	42.7	-	-	-	-	-	-	-	-	-	-	-	-
7-8	283.5	-	-	-	33.3	33.3	33.3	-	-	-	-	-	-	32.0	32.0	23.5	32.0	32.0	32.0	-	-	-	-	-	-
7-11	0.0	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
8-2	383.5	33.3	33.3	33.3	33.3	33.3	33.3	-	-	-	-	-	-	32.0	32.0	23.5	32.0	32.0	32.0	-	-	-	-	-	-
9-10	8.5	-	-	-	-	-	-	-	-	-	-	-	-	-	-	8.5	-	-	-	-	-	-	-	-	-
9-13	142.3	-	-	-	-	-	-	-	-	22.3	-	-	-	-	-	-	-	-	-	20.0	20.0	20.0	20.0	20.0	20.0
10-11	242.2	-	-	-	-	-	-	42.7	42.7	20.4	42.7	42.7	42.7	-	-	8.5	-	-	-	-	-	-	-	-	-
11-2	8.5	-	-	-	-	-	-	-	-	-	-	-	-	-	-	8.5	-	-	-	-	-	-	-	-	-
11-3	233.7	-	-	-	-	-	-	42.7	42.7	20.4	42.7	42.7	42.7	-	-	-	-	-	-	-	-	-	-	-	-
12-6	0.0	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
12-8	100.0	33.3	33.3	33.3	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
13-3	142.3	-	-	-	-	-	-	-	-	22.3	-	-	-	-	-	-	-	-	-	20.0	20.0	20.0	20.0	20.0	20.0
		Cars			Motos			Cars			Motos			Cars			Motos			Cars			Motos		
u_α		32.3	29.6	27.0	44.7	42.8	41.0	34.3	32.4	30.5	45.6	43.8	42.0	35.3	33.0	30.8	47.2	45.2	43.2	32.5	30.0	27.6	46.5	44.3	42.2

Table 8.7: Cars and motorcycles example. Mixed BPR model. Link flows disaggregated by OD and α-classes for the Nguyen-Dupuis network (uncongested case).

OD	Routes	path links	Classes					
			1	2	3	4	5	6
1- 2	1	1 5 7 9 11	33.342	31.336	29.330	**44.661**	**42.846**	**41.031**
1- 2	2	2 18 11	**32.317**	**29.645**	**26.974**	45.838	43.576	41.314
1- 3	3	1 5 8 14 16	**34.305**	**32.383**	**30.460**	**45.554**	**43.773**	**41.991**
1- 3	4	1 6 13 19	36.395	33.428	**30.460**	51.744	49.200	46.655
1- 3	5	2 17 7 10 16	38.086	34.664	31.242	53.714	50.814	47.914
1- 3	6	2 17 8 14 16	36.312	33.472	30.631	51.080	48.611	46.141
4- 2	7	3 5 7 9 11	**35.260**	**33.008**	**30.756**	**47.235**	**45.236**	**43.238**
4- 2	8	4 12 14 15	37.213	33.985	**30.756**	52.671	49.909	47.147
4- 3	9	3 5 8 14 16	36.224	34.055	31.886	48.128	46.163	44.197
4- 3	10	4 13 19	**32.485**	**30.022**	**27.559**	**46.468**	**44.338**	**42.207**

Table 8.8: Cars and motorcycles example. Route travel times classified by OD and α-classes for the Nguyen-Dupuis network (uncongested case). Used routes are boldfaced.

same class and OD and that they coincide with the OD travel times given in Table 8.7.

Figure 8.6 shows the used routes corresponding to different OD pairs and class users for the uncongested case. The upper three graphs of each quadrant correspond to cars and the lower three to motorcycles.

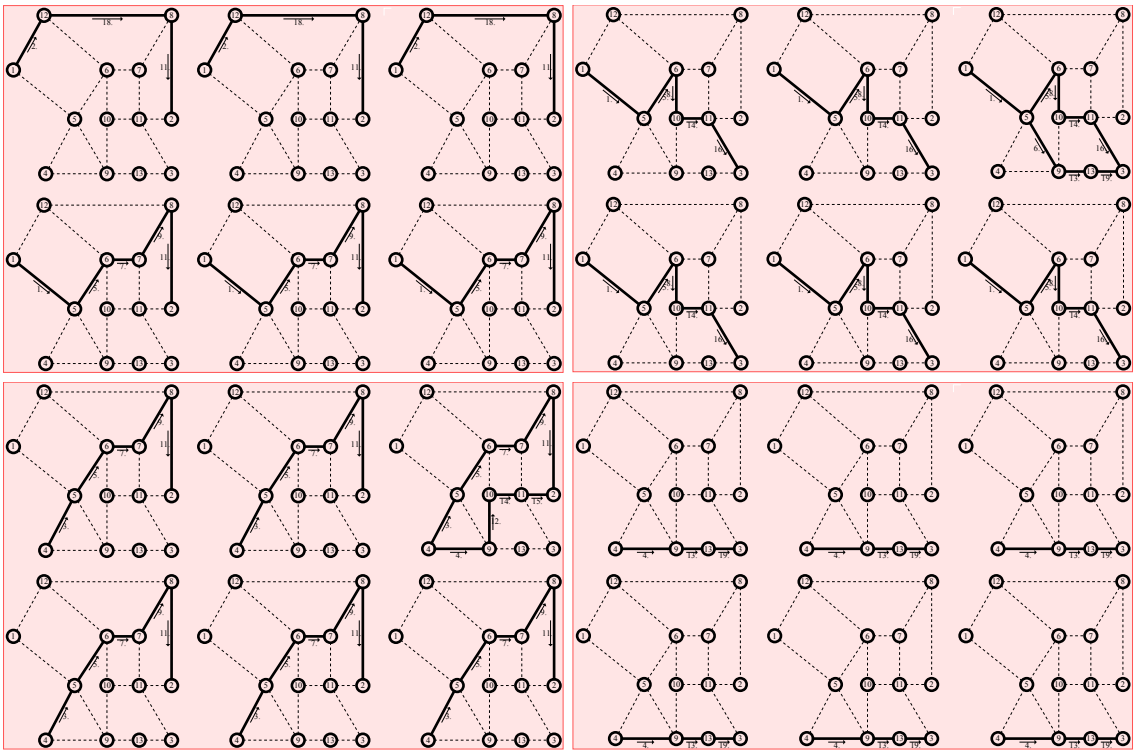


Figure 8.6: The cars and motorcycles example of the Nguyen-Dupuis example (uncongested case). The upper three graphs of each quadrant correspond to cars and the lower three to motorcycles.

Table 8.9 shows the total link flows (column 2), the link flows disaggregated by OD and α-classes (columns third to twentysixth) and the travel times of each OD and user class (last row) for the congested case. We can see that, as expected, cars travel times are bigger than motorcycles travel time because we have high congestion. Furthermore, travel times of both types of users are closer than in the uncongested case. We also see that different classes of cars and motorcycles use not necessarily the same routes but the travel times are different.

Table 8.10 shows the route travel times classified by OD and α-classes. The used routes have been boldfaced. Note that they correspond to minimum values of travel times in the same class and OD and that they coincide with the OD travel times given in Table 8.9.

Though in Tables 8.4, 8.8 and 8.10 we show three decimal digits, the coincidence of the minimum travel times is in at least seven digits. The precision accuracies for these cases are $rel_{gap} = 2.34 \times 10^{-10}$ and $rel_{gap} = 2.17 \times 10^{-10}$ for the saturated and non-saturated cases, respectively. These values, that are smaller than the 10^{-5} recommended, show that the results follow the UE conditions.

		OD: 1 -2						OD: 1 -3						OD: 4 -2						OD: 4 -3					
Link	Flow	1.00	0.90	0.80	1.40	1.30	1.20	1.00	0.90	0.80	1.40	1.30	1.20	1.00	0.90	0.80	1.40	1.30	1.20	1.00	0.90	0.80	1.40	1.30	1.20
1-5	759.8	-	-	-	95.8	75.3	-	122.7	98.0	-	122.7	122.7	122.7	-	-	-	-	-	-	-	-	-	-	-	-
1-12	551.2	95.8	95.8	95.8	-	20.5	95.8	-	24.7	122.7	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
4-5	645.9	-	-	-	-	-	-	-	-	-	-	-	-	-	45.8	92.0	92.0	92.0	92.0	-	2.1	57.5	57.5	57.5	57.5
4-9	251.1	-	-	-	-	-	-	-	-	-	-	-	-	92.0	46.2	-	-	-	-	57.5	55.4	-	-	-	-
5-6	1071.7	-	-	-	95.8	75.3	-	-	-	-	122.7	122.7	122.7	-	43.7	40.3	92.0	92.0	92.0	-	-	-	57.5	57.5	57.5
5-9	334.0	-	-	-	-	-	-	122.7	98.0	-	-	-	-	-	2.1	51.7	-	-	-	-	2.1	57.5	-	-	-
6-7	771.8	-	-	-	95.8	75.3	-	-	24.7	122.7	-	-	93.2	-	43.7	40.3	92.0	92.0	92.0	-	-	-	-	-	-
6-10	447.3	-	-	-	-	-	-	-	-	-	122.7	122.7	29.5	-	-	-	-	-	-	-	-	-	57.5	57.5	57.5
7-8	490.9	-	-	-	95.8	75.3	-	-	-	-	-	-	-	-	43.7	-	92.0	92.0	92.0	-	-	-	-	-	-
7-11	280.9	-	-	-	-	-	-	-	24.7	122.7	-	-	93.2	-	-	40.3	-	-	-	-	-	-	-	-	-
8-2	894.7	95.8	95.8	95.8	95.8	95.8	95.8	-	-	-	-	-	-	-	43.7	-	92.0	92.0	92.0	-	-	-	-	-	-
9-10	273.8	-	-	-	-	-	-	81.8	-	-	-	-	-	92.0	48.3	51.7	-	-	-	-	-	-	-	-	-
9-13	311.3	-	-	-	-	-	-	40.8	98.0	-	-	-	-	-	-	-	-	-	-	57.5	57.5	57.5	-	-	-
10-11	721.1	-	-	-	-	-	-	81.8	-	-	122.7	122.7	29.5	92.0	48.3	51.7	-	-	-	-	-	-	57.5	57.5	57.5
11-2	232.3	-	-	-	-	-	-	-	-	-	-	-	-	92.0	48.3	92.0	-	-	-	-	-	-	-	-	-
11-3	769.7	-	-	-	-	-	-	81.8	24.7	122.7	122.7	122.7	122.7	-	-	-	-	-	-	-	-	-	57.5	57.5	57.5
12-6	147.4	-	-	-	-	-	-	-	24.7	122.7	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
12-8	403.8	95.8	95.8	95.8	-	20.5	95.8	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
13-3	311.3	-	-	-	-	-	-	40.8	98.0	-	-	-	-	-	-	-	-	-	-	57.5	57.5	57.5	-	-	-
		Cars			Motos			Cars			Motos			Cars			Motos			Cars			Motos		
u_α		63.9	63.1	62.2	66.3	65.1	63.7	95.7	94.7	93.6	69.4	68.3	67.1	98.5	97.9	97.0	71.1	69.8	68.5	100.5	99.7	98.7	74.2	73.0	71.8

Table 8.9: Cars and motorcycles example. Mixed BPR model. Link flows disaggregated by OD and α-classes for the Nguyen-Dupuis network (congested case).

8.4.2 The Ciudad Real network

To test the proposed models in a real network, we have used the Ciudad Real network used in Section 7.4 in Chapter 7 and shown in Figure 7.2, which consists of 218 links, 105 nodes, 218 OD pairs, and 321 routes. The results are very similar to those described for the Nguyen-Dupuis network, but we cannot show all of them because of its size and the lack of space. The cpu time required to reach the solution with four α classes were 104.56 sec for the problem (8.28) subject to (8.23)-(8.25) and (8.27) with $\eta = 10^{-10}$ on a HP Z200 Workstation, Intel Core i7-870 2.93 8MB/1333 QC, RAM: 8GB (2x4GB).

Table 8.11 shows the route travel times of some selected OD pairs and all α-classes. The used routes have been boldfaced. Note that they correspond to the minimum travel times of all routes in the same OD pair and class.

Figure 8.8 shows the used routes corresponding to two selected OD pairs for the four class users in the Ciudad Real example.

The resulting precision accuracy was smaller than $rel_{gap} = 1 \times 10^{-10}$.

Appendix

A Convexity of the travel time function $h_\alpha(x)$

In this appendix we demonstrate the following theorem that permits obtaining convex travel time functions of the form (8.1).

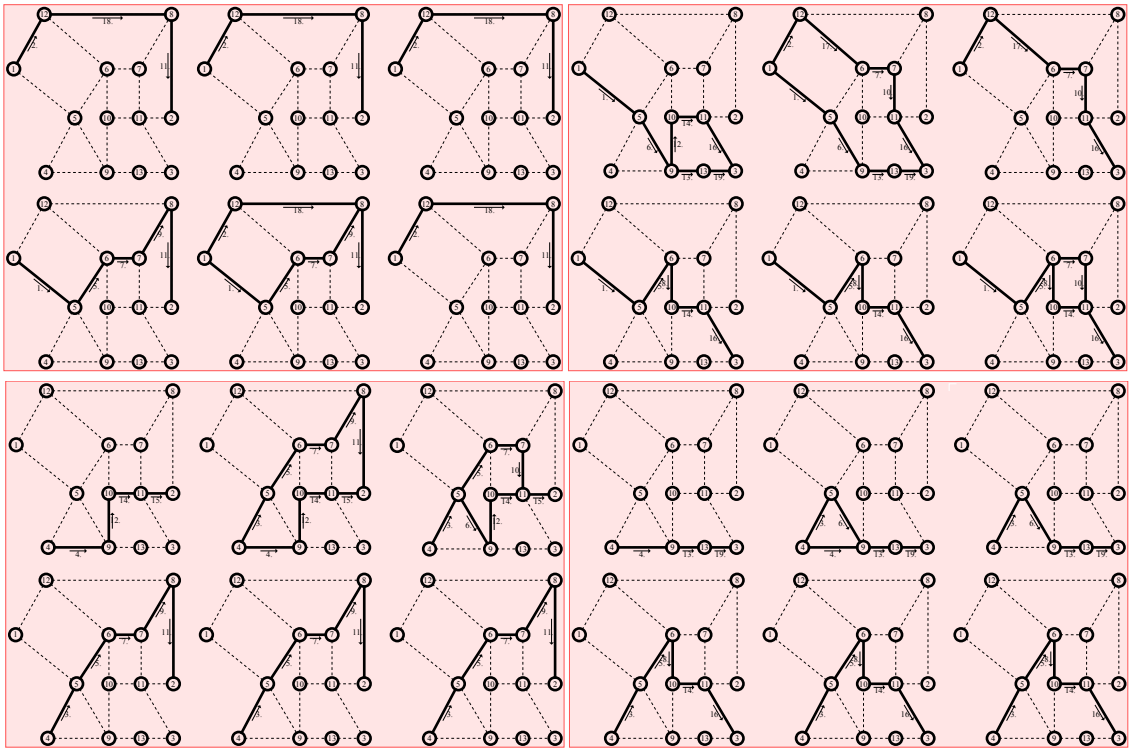


Figure 8.7: The cars and motorcycles example of the Nguyen-Dupuis example (congested case). The upper three graphs of each quadrant correspond to cars and the lower three to motorcycles.

Theorem 15 (Convexity of the travel time functions $h_\alpha(x)$) *The travel time function*

$$h_\alpha(x) = t(x; t_0, \beta_0, \gamma_0)\Phi\left(\frac{x-\mu}{\sigma}\right) + t(x; t_\alpha, \beta_\alpha, \gamma_\alpha)\left(1 - \Phi\left(\frac{x-\mu}{\sigma}\right)\right), \tag{8.47}$$

is convex in the interval $[0, x_{max}]$*, where* $x_{max} < \infty$*, if* $\sigma^2 \geq \dfrac{g(x_{max})}{t''(x_{max}; \theta_\alpha)\sqrt{2e\pi}}$.

Proof.

First, we remind the reader that

$$-\Phi''((x-\mu)/\sigma) \quad \leq \quad \begin{cases} 0 & \text{if} \quad x < \mu \\ \dfrac{1}{\sigma^2\sqrt{2e\pi}} & \forall x \end{cases} \tag{8.48}$$

and since the function $g(x)$ given in (8.3) is bounded in the finite interval $[0, x_{max}]$, let G be an upper bound of function $g(x)$ in (8.3), that is,

$$g(x) < G; \quad x \in [0, x_{max}] \tag{8.49}$$

OD	Routes	path links	Classes					
			1	2	3	4	5	6
1- 2	1	1 5 7 9 11	93.782	92.906	92.029	**66.333**	**65.085**	63.837
1- 2	2	2 18 11	**63.946**	**63.077**	**62.208**	66.468	**65.085**	**63.702**
1- 3	3	1 5 7 10 16	96.415	95.234	94.053	70.101	68.622	**67.144**
1- 3	4	1 5 8 14 16	98.608	97.958	97.309	**69.435**	**68.289**	**67.144**
1- 3	5	1 6 12 14 16	**95.745**	94.899	94.053	89.752	88.153	86.555
1- 3	6	1 6 13 19	**95.745**	**94.749**	93.753	80.066	78.543	77.021
1- 3	7	2 17 7 10 16	95.901	**94.749**	**93.597**	84.220	82.566	80.912
4- 2	8	3 5 7 9 11	98.782	**97.887**	96.991	**71.147**	**69.830**	**68.512**
4- 2	9	3 5 7 10 15	99.479	98.222	**96.964**	75.562	73.950	72.338
4- 2	10	3 6 12 14 15	98.809	97.887	**96.964**	95.213	93.481	91.749
4- 2	11	4 12 14 15	**98.543**	**97.887**	97.230	87.205	85.800	84.395
4- 3	12	3 5 8 14 16	103.608	102.939	102.270	**74.249**	**73.034**	**71.819**
4- 3	13	3 6 13 19	100.745	**99.730**	**98.715**	84.880	83.288	81.695
4- 3	14	4 13 19	**100.479**	**99.730**	98.980	76.871	75.606	74.341

Table 8.10: Cars and motorcycles example. Route travel times classified by OD and α-classes for the Nguyen-Dupuis network (congested case). Used routes are boldfaced.

and then

$$-g(x)\Phi''((x-\mu)/\sigma) < \frac{G}{\sigma^2\sqrt{2e\pi}}; \quad x \in [0, x_{max}]. \tag{8.50}$$

From (8.7) we get

$$h''_\alpha(x) \;=\; g''(x)\Phi((x-\mu)/\sigma) + 2g'(x)\Phi'((x-\mu)/\sigma) + g(x)\Phi''((x-\mu)/\sigma) + t''(x;\theta_\alpha), \tag{8.51}$$

where $g(x)$ is given in (8.3).

Since, according to our assumptions in (8.4),

$$g''(x)\Phi((x-\mu)/\sigma) + 2g'(x)\Phi'((x-\mu)/\sigma) + t''(x;\theta_\alpha) > 0$$

there exists a lower bound $H > 0$, such that

$$g''(x)\Phi((x-\mu)/\sigma) + 2g'(x)\Phi'((x-\mu)/\sigma) + t''(x;\theta_\alpha) > H, \tag{8.52}$$

and then, choosing a sufficiently large value of σ, we have

$$g''(x)\Phi((x-\mu)/\sigma)+2g'(x)\Phi'((x-\mu)/\sigma)+t''(x;\theta_\alpha) > H > \frac{G}{\sigma^2\sqrt{2e\pi}} > -g(x)\Phi''((x-\mu)/\sigma), \tag{8.53}$$

which proves that $h''_\alpha(x) > 0$ if $\sigma^2 \geq \dfrac{g(x_{max})}{t''(x_{max};\theta_\alpha)\sqrt{2e\pi}}$ and $x \leq x_{max}$, where we have taken into account that $G = g(x_{max})$ and $H = t''(x_{max};\theta_\alpha)$.

■

OD	Routes	path links	Classes			
			1	2	3	4
3- 9	1	6 203 35 184 12 17 197 194 191 38 140 142	6.87	**6.56**	**6.25**	**5.94**
3- 9	2	6 204 205 207 208 211 60 58 19 105 129 131 134 135	7.06	6.79	6.52	6.24
3- 9	3	6 204 205 207 209 213 58 19 105 129 131 134 135	7.13	6.83	6.52	6.21
3- 9	4	6 204 205 207 210 216 47 45 54 172 162 134 135	**6.85**	6.64	6.43	6.21
4-14	5	7 65 64 201 86 78 76 81	4.50	4.26	4.03	3.79
4-14	6	7 65 64 201 86 78 77 118 94 95	4.61	4.32	4.04	3.75
4-14	7	7 65 64 201 86 80 179 180 88 92 95	4.62	4.32	4.03	**3.73**
4-14	8	7 66 190 74 75 85 76 81	**4.21**	**4.06**	**3.92**	3.77
4-14	9	7 66 190 74 75 85 77 118 94 95	4.32	4.12	3.93	**3.73**
10- 3	10	19 105 129 132 169 170 56 217 214 212 166 7 65 63 71 42	8.61	8.34	8.07	**7.80**
10- 3	11	19 105 129 132 169 170 56 218 215 2 206 67 70 71 42	8.51	8.28	8.05	7.82
10- 3	12	19 105 130 165 57 56 217 214 212 166 7 65 63 71 42	8.57	8.31	8.06	7.80
10- 3	13	19 105 130 165 57 56 218 215 2 206 67 70 71 42	**8.46**	**8.25**	**8.04**	7.83
10-20	14	19 105 129 132 169 170 56 217 214 212 166 7 65 63 71	8.00	7.76	7.53	**7.29**
10-20	15	19 105 129 132 169 170 56 218 215 2 206 67 70 71	7.89	7.70	7.51	7.31
10-20	16	19 105 130 165 57 56 217 214 212 166 7 65 63 71	7.95	7.73	7.52	7.30
10-20	17	19 105 130 165 57 56 218 215 2 206 67 70 71	**7.85**	**7.67**	**7.49**	7.32
11- 2	18	20 126 129 131 134 137 161 174 52	3.34	3.18	3.03	**2.88**
11- 2	19	20 126 129 132 169 170 55 53	3.30	3.16	3.02	2.89
11- 2	20	20 126 130 165 57 55 53	**3.25**	**3.13**	3.01	2.89
11- 2	21	20 127 131 134 137 161 174 52	3.29	3.15	3.02	2.88
11- 2	22	20 127 132 169 170 55 53	3.26	**3.13**	**3.01**	2.89
11- 3	23	20 126 130 165 57 56 217 214 212 166 7 65 63 71 42	7.84	7.54	7.23	6.92
11- 3	24	20 126 130 165 57 56 218 215 2 206 67 70 71 42	**7.74**	**7.47**	7.21	6.94
11- 3	25	20 127 132 169 170 56 217 214 212 166 7 65 63 71 42	7.85	7.54	7.23	**6.91**
11- 3	26	20 127 132 169 170 56 218 215 2 206 67 70 71 42	7.74	**7.47**	**7.20**	6.94
11- 8	27	20 126 129 131 134 136 139 192 195 196	4.49	4.25	4.01	**3.78**
11- 8	28	20 126 129 131 134 137 161 175 177 196	4.45	4.23	4.01	3.80
11- 8	29	20 126 129 132 169 171 173 175 177 196	4.46	4.23	4.01	3.79
11- 8	30	20 127 131 134 136 139 192 195 196	4.45	4.23	4.00	**3.78**
11- 8	31	20 127 131 134 137 161 175 177 196	**4.41**	**4.20**	4.00	3.80
11- 8	32	20 127 132 169 171 173 175 177 196	4.41	**4.20**	**4.00**	3.79
11-17	33	20 126 130 165 57 55 53 4 50 14 185	6.22	5.98	5.73	5.49
11-17	34	20 126 130 165 57 56 218 215 1	**6.01**	**5.81**	5.60	5.39
11-17	35	20 127 132 169 170 55 53 4 50 14 185	6.22	5.98	5.73	5.48
11-17	36	20 127 132 169 170 56 218 215 1	6.01	**5.81**	**5.60**	**5.39**
11-20	37	20 126 130 165 57 56 217 214 212 166 7 65 63 71	7.23	6.96	6.69	6.41
11-20	38	20 126 130 165 57 56 218 215 2 206 67 70 71	**7.13**	**6.90**	6.66	6.43
11-20	39	20 127 132 169 170 56 217 214 212 166 7 65 63 71	7.23	6.96	6.68	**6.41**
11-20	40	20 127 132 169 170 56 218 215 2 206 67 70 71	7.13	**6.90**	**6.66**	6.43

Table 8.11: Route travel times classified by OD and α-classes for the Ciudad Real network. Used routes are boldfaced.

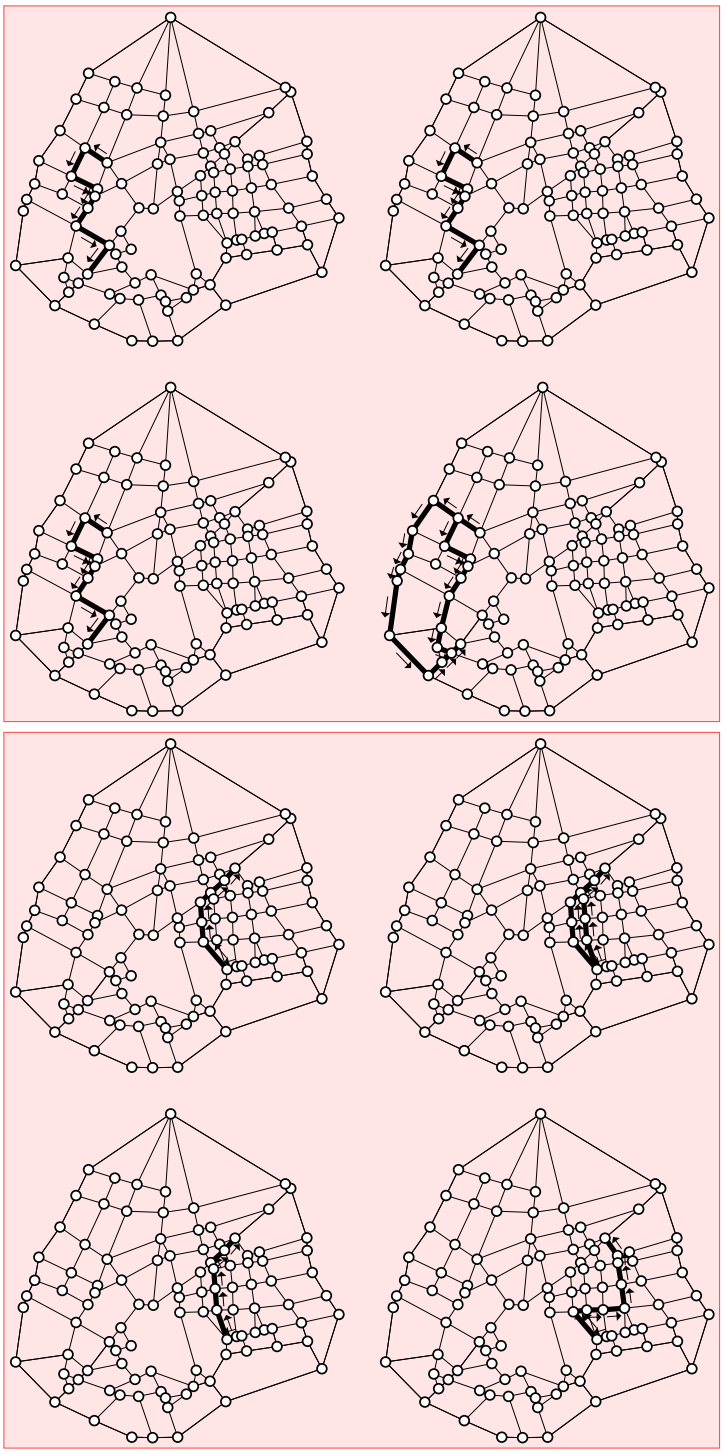


Figure 8.8: The Ciudad Real example. Used routes corresponding to two selected OD pairs ($4-14$ (upper plots) and $11-2$ (lower plots)) and the four classes.

B Notation

α	class.
β	parameter.
β_0	BPR parameter of the reference class α_0.
β_α	parameter associated with class α.
γ	exponent parameter.
γ_0	BPR parameter of the reference class α_0.
γ_α	exponent parameter associated with class α.
δ^*_{ij}	Kronecker delta.
δ_{ijr}	element of the link-route incidence matrix.
ϵ	small scalar.
η	a very small scalar.
θ_α	vector parameter $(t_\alpha, \beta_\alpha, \gamma_\alpha)$.
$\lambda^{iks\alpha}$	dual variables.
μ	mean of the normal distribution.
μ_α	mean of all links flows in class α.
μ^α_r	dual variable.
$\mu_{ijks\alpha}$	dual variable.
ν_0	free travel time of the reference class α_0.
ξ_{rks}	element of the OD-route incidence matrix.
$\boldsymbol{\rho}$	vector of ρ^α_{ks} variables.
ρ^α_{ks}	minimum travel time of class α and OD ks.
σ	standard deviation of the normal distribution.
σ^*	optimum value of σ.
σ_α	standard deviation associated with class α.
τ^α_{ks}	α-class OD flow.
$\Phi(\cdot)$	cumulative distribution function of the normal distribution.
Ω_f	set of all feasible route flows.
Ω_v	set of all feasible link flows.
$\mathcal{A}$	set of links.
c^α_{ij}	link travel time function for class α.
c^α_r	travel time of a user of class α traveling route r.
$F(\cdot)$	cumulative distribution function.
$\mathbf{f}$	vector of route flows f^α_r.
f^α_r	r route flow of class α.
$g(x)$	difference of two link travel time functions.
$g'(\cdot)$	first derivative of function $g(\cdot)$.
$g''(\cdot)$	second derivative of function $g(\cdot)$.
$\mathcal{H}$	set of all α classes.

$h_\alpha(x)$	proposed link travel time function.
i	link begin node.
j	link end node.
k	origin node.
ℓ_{ij}	link joining nodes i and j.
M	large scalar.
m_{ij}^α	cost associated with a user of class α traveling through link ℓ_{ij}.
$\mathbf{m}(\mathbf{v})$	vector of class link costs.
$\mathcal{N}$	set of nodes.
$\mathbf{n}(\mathbf{f})$	vector of route costs.
$\mathcal{OD}$	set of all OD-pairs.
q	link capacity.
q_α	factor that provides the free link travel times of a class with respect to the reference class α_0.
r	route.
s	destination node.
t_0	link free travel time.
$t(x)$	link travel time function.
t_α	link free travel time for class α.
v	traffic volume.
$\mathbf{v}$	vector of disaggregated class link flows v_{ijks}^α.
v_α	traffic volume associated with class use α.
v_{ijks}^α	α-class flow through link i, j with origin node k and destination node s.
w_α	weight associated with class user α.
x	congestion ratio.
x_{max}	number of users producing a unit link congestion ratio.
x_{ijk}^α	flow coming from a given origin node k and using link ℓ_{ij} associated with class α.
y_{ijs}^α	flow going to a given destination node s and using link ℓ_{ij} associated with class α.
z_{iks}^α	flow going from origin node k to destination node s and passing through node i associated with class α.

Chapter 9

A Bayesian Matrix Estimation Model

Contents

9.1 Introduction

In this chapter we deal with the problem of estimating origin-destination trip matrices (ME) based on link flows, which is stated from a hierarchical optimization point of view.

Hierarchical optimization was introduced by Bracken et al. (1973, 1974) and refers to a class of optimization problems characterized by constraints that themselves contain optimization problems.

Hierarchical optimization of a sequence of objective functions (indexed in order of priority) involves a sequential optimization procedure in which nonuniqueness arising from

the optimization of some functions is exploited to optimize other functions. It may be successfully applied provided: 1) the design objectives can be ranked in order of importance and 2) nonunique optimizers exist for the high-priority objectives, and the corresponding solution sets are parameterized in a simple way.

The problem to be dealt with in this work can be stated in several different forms. In fact, we can consider it as a system of equations in which some of them are optimization problems. However, we can choose any of these constraints and use them as objective functions. It is well known that some constraints can be replaced by the objective function obtaining an equivalent problem (see Conejo et al. (2005)).

As already seen in Chapter 4, the ME problem consists of estimating origin-destinations (OD) flow matrices based on some observed link flows. Since the number of OD pairs is normally much larger than the number of links, this problem is under-specified, i.e., it has infinitely many solutions Since we look for OD flow estimates close to the real ones, more information is needed. To this end, we normally use a prior OD matrix and contemplate, as a reasonable set of solutions, the set of matrices close to it. To obtain the OD flow estimates there exist a wide range of possibilities (see Chapter 4 for a summary of these methods).

Under a classical point of view, in the ME problem the route choice proportions, or equivalently the link flows in the road network, are assumed to be the input data and the trip flow matrix, the output. Contrary, in the traffic assignment problem, the trip flow matrix is the input and the route choice proportions, the output. However, since some inconsistencies in the flow solutions of both problems may appear, they have been coupled together, in which both the trip flow matrix and the route choice proportions or link flows become the output, and they have been solved using bi-level approaches (see Section 4.3).

This kind of techniques can in fact be considered as hierarchical methods where in the upper-level the ME problem is solved and the traffic assignment problem is dealt with in the lower-level.

In this chapter we present a hierarchical optimization problem generated by a Bayesian method, for estimating origin-destination matrices, based on Gamma models. This model certainly facilitates the process of updating parameter estimates when new information becomes available, as it is based on conjugate families.

The chapter is organized as follows. Section 9.2 describes the Gamma distribution, and some of their properties and the associated Bayesian learning problem, including the corresponding conjugate distributions. In Section 9.3 we show how Gamma models can be used for trip matrix estimation. The proposed hierarchical approach of the problem is formulated and an efficient algorithm to solve it is proposed in Section 9.4. Finally, in Section 9.5 two examples of application are used to illustrate the proposed methods and methodology.

9.2 Some statistical background on the Gamma models

In this section we introduce the reader to Gamma models, which is the statistical material required to understand the following sections.

Let $(X_1, \ldots, X_{k+1})$ be independent random variables having Gamma distributions $G(\theta_1, 1)$, $\ldots$, $G(\theta_{k+1}, 1)$, with probability density function (pdf)):

$$g(x;\theta) = \frac{x^{\theta-1}e^{-x}}{\Gamma(\theta)}, x > 0, \tag{9.1}$$

where $\Gamma(\theta)$ is the Gamma function, which for integer values of θ coincides with $(\theta-1)!$. The mean and variance of the X_i are $E[X_i] = \theta_i$ and $Var[X_i] = \theta_i$, respectively.

9.2.1 Learning Gamma models by Bayesian methods

In this section we address the problem of learning Gamma models, i.e., learning their parameter values. To this end, we use the Bayes' rule

$$f(\theta;\eta|\mathbf{p}) = \frac{f(\mathbf{p}|\theta)f(\theta;\eta)}{\int\limits_{\theta} f(\mathbf{p}|\theta)f(\theta;\eta)d\theta} \propto f(\mathbf{p}|\theta)f(\theta;\eta), \tag{9.2}$$

where $f(\theta;\eta)$ is the *prior* distribution, $f(\theta;\eta|\mathbf{p})$ is the *posterior* distribution given the data $\mathbf{p}$, η are the hyperparameters, and $f(\mathbf{p}|\theta)$ is the *likelihood* of the data.

As it is well known, the Bayesian approach consists of the following steps:

1. Select the likelihood family.
2. Select the family of priors (normally a conjugate family).
3. Assess the prior distribution on the parameters.
4. Obtain the sample data.
5. Calculate the posterior distribution.
6. Estimate the parameters by the posterior mean or mode and their variabilities using the posterior distributions.

9.2.2 Conjugate of a Gamma Distribution

Bayesian statisticians often work with conjugate priors, which are parametric families of distributions such that their associated posteriors belong to the same families. The parameters $\boldsymbol{\eta}$ of the conjugate family are referred to as *hyperparameters*.

Arnold et al. (1993) and Arnold et al. (1996) justify and suggest the following conjugate family as the most general conjugate for the Gamma family with active parameters:

$$q(\theta|\eta) \doteq \exp\left[\eta_1\theta - \eta_2 \log\Gamma\left(\theta\right)\right], \tag{9.3}$$

where the posterior hyperparameters $\hat{\eta}$ become

$$(\hat{\eta}_1, \hat{\eta}_2) = \left(\eta_1 + \sum_{\ell=1}^{n} \log x_\ell, \eta_2 + n \right), \tag{9.4}$$

n is the sample size, and $\{x_\ell : \ell = 1, 2, \ldots, n\}$ is the sample. Note that Expression (9.4) is the updating formula for this model, that is, it allows us to obtain the posterior hyperparameter values in terms of the corresponding prior values and the sample.

Since $\sum_{\ell=1}^{n} \log x_\ell$ and n tend to ∞, as $n \to \infty$, the effect of prior information (η_1, η_2) vanishes as $n \to \infty$. In fact, for an infinitely large sample, the model based upon random costs leads to a deterministic flow pattern.

Due to its complexity (exponential and Gamma functions appear in it), the mean of the Gamma conjugate distribution (9.3) cannot be obtained in closed form. Thus, as an alternative to the means, we use the mode of (9.3) to estimate the Gamma parameters, i.e., we maximize (9.3), with respect to the θ's, to estimate the θ-parameters.

Note that using the mean as the Bayesian estimate is due only to the use of quadratic utility functions. So, using the mode (the most probable) could not be considered as a worst criteria, mainly when the maximum likelihood criterion is generally accepted in Statistics.

To avoid precision problems we do the following:

- Maximize the logarithm of (9.3) instead of (9.3) itself (note that the Gamma function can take very large values).
- Use a numerical procedure for the direct evaluation of the logarithm of the Gamma function instead of evaluating the Gamma function and taking the logarithm.
- Use parameters $\lambda_i^2 = \theta_i$ to guarantee non-negativity of the parameters.
- For the initial θ-estimates, which are required by any non-linear maximization procedure, we can use moment estimators or the sample itself.

The prior assessment is a very important step of the method, because the results of the proposed method for small samples depend strongly on it. Some practical methods for the prior assessment are given in Section 9.3.2.

9.3 Gamma models for estimating OD matrices

In this section we explain how the Gamma model described in Section 9.2 can be applied to solve the problem of OD matrix estimation. First, we start discussing the assumptions.

9.3.1 Assumptions and derived properties

1. The random number of users f_{ksr} choosing route r of OD pair ks (i.e. those travelers who go from node k to node s using route r) follows a Gamma distribution in the family[1]:

   $$\mathcal{F} \equiv \{G(\theta, 1) | \theta > 0\},$$

   i.e.,

   $$f_{ksr} \sim G(\theta_{ksr}, 1).$$

   We note that this family of Gamma distributions has good properties, because the associated random variables are positive (as the number of users), it is reproductive (the sum of Gamma distributions in $\mathcal{F}$ belongs to $\mathcal{F}$), and it is infinitely divisible.

2. All components of the multivariate Gamma random variable $U_{ks} = (f_{ks1}, f_{ks2}, \ldots, f_{ksR_{ks}})$, where R_{ks} is the number of routes of OD ks, are independent. The independence of path flows means that users decide to use different paths independently of each other. This means that they have preferences but that they are not affected by others' opinions. This assumption does not imply a fixed number of travelers in each path, nor a total number of travelers in all paths. Thus, it is reasonable.

3. The multivariate random variables $U_1, U_2, \ldots, U_I$, where I is the number of OD pairs, are independent. This implies that the users of different OD-pairs act independently.

The first assumption is original, but the other two are common in the existing literature. These assumptions lead to the following derived properties.

1. The random number of users $t_{ks} = \sum_r f_{ksr}$ traveling through the OD pair ks follows a Gamma distribution $G(\nu_{ks}, 1)$, where $\nu_{ks} = \sum_r \theta_{ksr}$, which belongs to family $\mathcal{F}$ too.

2. The number of users v_{ijks} using link ℓ_{ij} and traveling the OD ks is a Gamma distribution $G(\theta_{ijks}, 1)$, where $\theta_{ijks} = \sum_r \theta_{ksr}\delta_{ijr}^{ks}$, which belongs to family $\mathcal{F}$ and δ_{ijr}^{ks} is 1 if link ℓ_{ij} belongs to a route r of the pair ks and 0 otherwise.

3. The number of users $w_{ij} = \sum_{ks} v_{ijks}$ using link ℓ_{ij} is a Gamma distribution $G\left(\sum_{ks} \theta_{ijks}, 1\right)$, which belongs to family $\mathcal{F}$[2].

[1] We approximate the integer variable number of users f_{ksr} by a real variable Gamma distributed.

[2] The reason for using a Gamma $G(\theta, 1)$ is that it leads to the same flow family for OD, links and v_{ijks} flows. If the second parameter is not the same for all routes, this property does not hold anymore.

9.3.2 Practical implementation of the proposed method

In this section we introduce the Gamma model based on the t_{ks} Gamma random variables. Due to the fact that observations are usually in terms of link flows, an additional procedure is proposed in order to get some estimates of the OD flows. Gamma models based on w_{ij} link flows or the v_{ijks} flows could also be used, but because of uniqueness problems in the OD flows calculation, the Gamma model below seems to be more adequate. In addition, Gamma models are convenient, because they reproduce the positive skewness of traffic data, that other models, such as the normal model, cannot reproduce.

For the proposed method to be valid, we need informative priors in order to obtain good results. This is a consequence of the under-specification of the ME problem. Thus, uninformative priors are discarded.

The prior assessment can be done by the following methods:

1. *Assessment of priors for the Gamma model based on an imaginary sample.* Following Klieter (1992), the prior can be assessed by means of an imaginary sample, i.e., we can ask a human expert to provide a virtual sample $\bar{t}_{ks\ell}$ of size m as the most representative of his/her knowledge. Once this sample is known, the prior hyperparameters can be obtained using (9.4), that is:

$$\eta_{ks1} = \sum_{\ell=1}^{m} \log \bar{t}_{ks\ell} \tag{9.5}$$

$$\eta_{ks2} = m. \tag{9.6}$$

   Note that according to (9.4), a sample modifies the η-parameters by adding $\sum_{\ell=1}^{n} \log t_{ks\ell}$ and m to the previous values.

2. *Prior assessment based on an out-of-date trip matrix.* Another possibility consists of using a prior or out-of-date trip matrix. Then hyperparameters are given by

$$\eta_{ks1} = m \log \bar{t}_{ks} \tag{9.7}$$

$$\eta_{ks2} = m, \tag{9.8}$$

   where $\bar{t}_{ks}$ is the out-of-date sample.

9.3.3 Obtaining posteriors for the Gamma model based on the t_{ks} Gamma random variables.

Because normally we have no observations of the OD-flows t_{ks}, but we have some link observations $\{\bar{w}_{ij} | \ell_{ij} \in \mathcal{A}\}$, we must do something to get some estimates $\hat{t}_{ks}$ of t_{ks}. To get

these estimates we solve the optimization problem

$$\underset{\hat{\mathbf{t}},\mathbf{w}}{\text{Minimize}}\, Z = \rho \sum_{ks} \left(\frac{\hat{t}_{ks} - \bar{t}_{ks}}{\bar{t}_{ks}} \right)^2 + \sum_{\ell_{ij} \in \mathcal{A}} \left(\frac{w_{ij} - \bar{w}_{ij}}{\bar{w}_{ij}} \right)^2 \tag{9.9}$$

subject to

$$w_{ij} = \sum_{ks} \beta_{ijks} \hat{t}_{ks}; \quad \ell_{ij} \in \mathcal{A} \tag{9.10}$$

$$\nu_{ks} \geq 0; \quad k \in \mathcal{O}, s \in \mathcal{D}, \tag{9.11}$$

where β_{ijks} is a prior estimate of the proportion of OD flows using link ℓ_{ij}, ρ is a weight factor and $\hat{t}_{ks}$ are the OD flows associated with the observed sample. Then, the posterior $\hat{\eta}$ hyperparameters can be calculated in terms of the prior η parameters as

$$\hat{\eta}_{ks1} = \eta_{ks1} + n \log \nu_{ks} \tag{9.12}$$

$$\hat{\eta}_{ks2} = \eta_{ks2} + n. \tag{9.13}$$

It should be noted that the objective function in (9.9) involves two terms, one related to OD flows and one related to link flows. In addition, the $\rho \geq 0$ coefficient permits us to provide higher or lower hierarchy to any one of them. If $\rho \to \infty$ the first occupies the higher level and the prior becomes the start, and if $\rho \to 0$, the second (the observed flows) becomes the most relevant one.

Note that, m/n measures the relative weight of the human expert or prior information with respect to the information contained in a real sample of size n. For example, if $m = n$, they have the same associated information.

Related to this latter issue is the link between the precision of the estimates arising and the quality of (i) the prior and (ii) the count data. The prior information can have a large weight compared with the observed sample data or vice versa. As indicated a small weight of the prior is not recommendable.

The updating formulas (9.4) provide a clear information about this relative weight. In fact, a comparison of η_1, η_2 with $\hat{\eta}_1, \hat{\eta}_2$ provides this relative weight.

9.3.4 Bayesian estimates of OD flows

The proposed model for OD flow estimation uses the family of conjugate distributions for the Gamma family $\mathcal{F}$ associated with t_{ks}, taking into account that we select the posterior mode for estimating the parameters, and the ν_{ks} parameters are estimated by solving the following optimization problem, one per each OD:

$$\underset{\boldsymbol{\nu_{ks}}}{\text{Maximize}} \quad \hat{\eta}_{ks1} \nu_{ks} - \hat{\eta}_{ks2} \log \Gamma(\nu_{ks}); \quad \forall k, s, \tag{9.14}$$

where we have assumed a prior with independent components for the OD flow parameters.

As it has been pointed out above, unfortunately, we have no observations of the OD-flows, and therefore the proposed alternative in subsection 9.3.3 is used.

The congestion problem is dealt with by solving the Wardrop-Minimum Variance equilibrium model (WMV)(see Section 3.2.1), which together with the iterative process corresponding to the bi-level approach takes into account the interaction of traffic flows. To this end, we use the following bi-level algorithm, that takes into account the congestion problem by means of the BPR function and the iterative process.

9.4 Hierarchical approaches to solve the problem

Based on the above discussion, our traffic problem can be stated as solving the following system of equations:

$$\mathbf{v} = \arg\min \left[\sum_{\ell_{ij} \in A} \int_0^{\left(\sum_{k,s} v_{ijks} \right)} C_{ij}(v) dv + \frac{\kappa}{m} \sum_{\ell_{ij} \in A} \sum_{k,s} (v_{ijks} - \mu)^2 \right], \quad (9.15)$$

$$0 = \nu_{ks}(\delta'_{ik} - \delta'_{is}) - \sum_{\ell_{ij} \in \mathcal{A}} v_{ijks} + \sum_{\ell_{ji} \in \mathcal{A}} v_{jiks}, \quad \forall i, \ \forall k \neq s, \quad (9.16)$$

$$\mu = \frac{1}{m} \sum_{\ell_{ij} \in A} \sum_{k,s} v_{ijks}, \quad (9.17)$$

$$v_{ijks} \geq 0, \quad \ell_{i,j} \in \mathcal{A}, ks \in \mathcal{OD}, \quad (9.18)$$

$$(\hat{\mathbf{t}}, \mathbf{w}) = \arg\min \left[\rho \sum_{ks} \left(\frac{\hat{t}_{ks} - \bar{t}_{ks}}{\bar{t}_{ks}} \right)^2 + \sum_{\ell_{ij} \in \mathcal{A}} \left(\frac{w_{ij} - \bar{w}_{ij}}{\bar{w}_{ij}} \right)^2 \right], \quad (9.19)$$

$$w_{ij} = \sum_{ks} \frac{v_{ijks} \hat{t}_{ks}}{t_{ks}}, \quad \ell_{ij} \in \mathcal{A} \quad (9.20)$$

$$\hat{t}_{ks} \geq 0, \quad ks \in \mathcal{OD}, \quad (9.21)$$

$$\hat{\eta}_{ks1} = \eta_{ks1} + n \log \hat{t}_{ks}, \quad ks \in \mathcal{OD}, \quad (9.22)$$

$$\hat{\eta}_{ks2} = \eta_{ks2} + n, \quad ks \in \mathcal{OD}, \quad (9.23)$$

$$\nu_{ks} = \arg\max \left(\hat{\eta}_{ks1} \nu_{ks} - \hat{\eta}_{ks2} \log \Gamma(\nu_{ks}) \right), \quad ks \in \mathcal{OD}. \quad (9.24)$$

Since the system of equations (9.15)-(9.24) contains optimization problems as constraints, it can be considered as a hierarchical optimization problem. On the other hand, due to the presence of several objective functions, it can also be considered as a multi-objective optimization problem.

In addition to see this problem as a system of equations, we can see it from a different point of view if we convert any of the constraints (not only those involving optimization

problems) in objective functions (see Conejo et al. (2005)). For example, minimize the following objective function in (9.15)

$$\sum_{\ell_{ij}\in A}\int_0^{\left(\sum_{k,s} v_{ijks}\right)} C_{ij}(v)dv + \frac{\kappa}{m}\sum_{\ell_{ij}\in A}\sum_{k,s}(v_{ijks}-\mu)^2 \tag{9.25}$$

subject to (9.16)-(9.24) or minimize

$$\rho\sum_{ks}\left(\frac{\hat{t}_{ks}-\bar{t}_{ks}}{\bar{t}_{ks}}\right)^2 + \sum_{\ell_{ij}\in\mathcal{A}_0}\left(\frac{w_{ij}-\bar{w}_{ij}}{\bar{w}_{ij}}\right)^2, \tag{9.26}$$

subject to (9.15)-(9.18) and (9.20)-(9.24).

Note that in both objective functions (9.25) and (9.26) we have a hierarchy associated with the values of $\frac{\kappa}{m}$ and ρ, respectively.

Since we have many possible alternatives to state the problem, many solution approaches can be used to solve it. The following efficient algorithm provides one of these alternatives.

Algorithm (Gamma model multi-level algorithm)

INPUT. The data of the algorithm consists of the network topology, an out-of-date or virtual OD matrix, and the set of link flow observations.

OUTPUT. The predictions of the OD and link flows given the observed link flows.

1. For each OD pair of the network assess a prior distribution of the form

$$q_{ks}(\theta|\eta)\doteq \exp\left[\eta_{1ks}\theta - \eta_{2ks}\log\Gamma(\theta)\right],$$

using the out-of-date OD matrix $\bar{\mathbf{t}}$ to determine

$$\begin{aligned}\eta_{ks1} &= m\log\bar{t}_{ks}\\ \eta_{ks2} &= m.\end{aligned}$$

2. Solve the WMV assignment problem

$$\underset{\mathbf{v}}{\text{Minimize}}\, Z = \sum_{\ell_{ij}\in A}\int_0^{\left(\sum_{k,s} v_{ijks}\right)} C_{ij}(v)dv + \frac{\kappa}{m}\sum_{\ell_{ij}\in A}\sum_{k,s}(v_{ijks}-\mu)^2$$

subject to

$$\begin{aligned}\bar{t}_{ks}(\delta_{ik}-\delta_{is}) &= \sum_{\ell_{ij}\in\mathcal{A}} v_{ijks} - \sum_{\ell_{ji}\in\mathcal{A}} v_{jiks}, \forall i; k\neq s,\\ \mu &= \frac{1}{m}\sum_{\ell_{ij}\in A}\sum_{k,s} v_{ijks},\\ v_{ijks} &\geq 0\ \forall i,j,k,s,\end{aligned}$$

and calculate the β_{ijks} proportions as:

$$\beta_{ijks} = \frac{v_{ijks}}{\bar{t}_{ks}}.$$

3. Solve the optimization problem

$$\underset{\hat{\mathbf{t}}_{\mathbf{ks}},\mathbf{w}}{\text{Minimize}}\, Z = \rho \sum_{ks} \left(\frac{\hat{t}_{ks} - \bar{t}_{ks}}{\bar{t}_{ks}} \right)^2 + \sum_{\ell_{ij} \in \mathcal{A}_0} \left(\frac{w_{ij} - \bar{w}_{ij}}{\bar{w}_{ij}} \right)^2$$

subject to

$$\begin{aligned} w_{ij} &= \sum_{ks} \beta_{ijks} \hat{t}_{ks}; \quad \ell_{ij} \in \mathcal{A} \\ \hat{t}_{ks} &\geq 0; \quad k \in \mathcal{O}, s \in \mathcal{D}, \end{aligned}$$

to obtain the $\hat{t}_{ks}$ OD flows associated with the observed sample.

4. Obtain the posterior hyperparameters:

$$\begin{aligned} \hat{\eta}_{ks1} &= \eta_{ks1} + n \log \hat{t}_{ks} \\ \hat{\eta}_{ks2} &= \eta_{ks2} + n. \end{aligned}$$

5. For each OD solve the optimization problem

$$\underset{\boldsymbol{\nu}}{\text{Maximize}} \;\; \hat{\eta}_{ks1} \nu_{ks} - \hat{\eta}_{ks2} \log \Gamma(\nu_{ks}); \quad \forall k, s,$$

to estimate the parameters ν_{ks}.

6. Compute the actual error:

$$\text{error} = \sum_{k \in \mathcal{O}, s \in \mathcal{D}} (\bar{t}_{ks} - \nu_{ks})^2,$$

and if the error is less than the tolerance, return the values of ν_{ks} and w_{ij}, and stop. Otherwise, let $\bar{t}_{ks} = \nu_{ks}$ and continue with Step 1.

9.5 Example of application

To illustrate the above proposed models and methods, we have selected two examples: the Nguyen-Dupuis example and the real network of Ciudad Real in Spain.

9.5.1 The Nguyen-Dupuis network

We start with a simple example, so that all the results have a small size to be shown.

Consider the Nguyen-Dupuis network previously used and shown in Figure 8.3. Again, in this case we consider unidirectional links and, hence, the network has 13 nodes, 19 links and four OD pairs[3]. Suppose that we are interested in making inference about the OD and link flows. So, we decide to apply the Gamma model described in Section 9.3 and aim at estimating its parameters and predicting the OD flows. To this end, we apply the Gamma model and the proposed algorithm indicated in section 9.4.

link	t_{0ij}	q_{ij}	β_{ij}	γ_{ij}	link	t_{0ij}	q_{ij}	β_{ij}	γ_{ij}
1 -5	7	70	1	4	8 -2	9	70	1	4
1 -12	9	56	1	4	9 -10	10	56	1	4
4 -5	9	56	1	4	9 -13	9	56	1	4
4 -9	12	70	1	4	10-11	6	70	1	4
5 -6	3	42	1	4	11-2	9	56	1	4
5 -9	9	42	1	4	11-3	8	56	1	4
6 -7	5	70	1	4	12-6	7	14	1	4
6 -10	5	28	1	4	12-8	14	56	1	4
7 -8	5	70	1	4	13-3	11	56	1	4
7 -11	9	70	1	4					

Table 9.1: Parameters of the Nguyen-Dupuis network in Figure 8.3

In this example we have used the BPR cost function explained in the Appendix of Chapter 3. The values used for the parameters in the Nguyen-Dupuis network example are shown in Table 9.1 and $\kappa = 0.000001$ has been used.

To illustrate, we have used the prior OD flows shown in Table 9.3 and solved the WMV problem (3.43)-(3.46) to get the prior link flows, which appear in Table 9.2 column 3.

Keeping the prior values constant, we have assumed as true flows those resulting from the WMV model (3.43)-(3.46) using as OD the prior $\bar{t}_{ks}$ values multiplied by independent random uniform numbers $U(1.0, 1.3)$. In other words, we have assumed different true values in each simulation. This assumption can be observed by comparing columns 2 and 3 in Tables 9.2 and 9.3, and corresponds to having an old OD pair flow matrix and an increasing flow up to a 30%.

For a given simulation, in Table 9.2 we show: the links, the true link simulated values, the link priors (identical for all simulations), and the link flows resulting from our algorithm for four different combinations of sizes (m and n) of virtual prior and real samples.

The set of link observations is the one boldfaced in column two of Table 9.2, associated

[3]For the sake of simplicity and in order to reduce the size of the resulting tables, we have considered only four OD pairs, but there is no problem in considering more OD pairs.

with a subset of four observed links.

LINK	True	Prior	Case 1 $m = 100$ $n = 1$	Case 2 $m = 10$ $n = 10$	Case 3 $m = 1$ $n = 10$	Case 4 $m = 1$ $n = 100$	LS
1 -5	**71.15**	67.27	67.71	69.76	71.59	71.96	72.51
1 -12	57.74	52.73	53.37	55.34	57.02	57.35	58.99
4 -5	25.28	21.74	21.86	22.62	23.19	23.29	25.67
4 -9	66.15	58.26	59.21	61.86	63.96	64.37	66.92
5 -6	64.00	59.25	59.71	61.35	62.72	62.99	64.77
5 -9	32.43	29.76	29.86	31.03	32.05	32.26	33.42
6 -7	57.34	50.95	51.53	53.92	55.89	56.27	58.38
6 -10	20.89	21.03	21.02	20.87	20.78	20.77	20.82
7 -8	23.43	21.74	21.86	22.62	23.19	23.29	23.35
7 -11	33.92	29.21	29.67	31.30	32.70	32.98	35.03
8 -2	66.93	61.74	62.39	64.51	66.26	66.60	67.90
9 -10	**43.14**	38.26	38.70	40.67	42.26	42.56	44.18
9 -13	**55.44**	49.76	50.37	52.22	53.76	54.06	56.16
10-11	64.03	59.28	59.71	61.54	63.04	63.33	65.00
11-2	42.87	38.26	38.70	40.67	42.26	42.56	44.14
11-3	55.08	50.24	50.69	52.18	53.48	53.75	55.89
12-6	14.23	12.73	12.84	13.45	13.95	14.05	14.44
12-8	**43.51**	40.00	40.53	41.89	43.07	43.31	44.55
13-3	55.44	49.76	50.37	52.22	53.76	54.06	56.16

Table 9.2: True, prior and resulting link flows when the proposed algorithm is used for four different cases of relative weight of the prior and the sample with respect to information. The boldfaced values corresponds to the observed link flows

To illustrate the role played by the prior and the observed sample in terms of information, four different cases have been studied (see Tables 9.2 and 9.3):

Case 1. $m = 100$ and $n = 1$: A strongly informative prior and a weak sample. As expected, the mean flow estimates are very close to the prior mean flow estimates, due to the high value of m with respect to n. The observed flows are a little far from their estimated values (the estimates are on the prior side).

Case 2. $m = 10$ and $n = 10$: Same information in the prior and the sample. As expected, the estimated mean values are intermediate to prior and observed flows.

Case 3. $m = 1$ and $n = 10$: Weak informative prior and a informative sample. The mean estimated flows are closer to the observed ones than in the previous case.

OD	True	Prior	Case 1 $m = 100$ $n = 1$	Case 2 $m = 10$ $n = 10$	Case 3 $m = 1$ $n = 10$	Case 4 $m = 1$ $n = 100$	LS
1 -2	43.51	40.00	40.53	41.89	43.07	43.31	44.55
1 -3	85.38	80.00	80.55	83.21	85.54	86.01	86.95
4 -2	66.30	60.00	60.56	63.29	65.45	65.86	67.45
4 -3	25.14	20.00	20.51	21.19	21.70	21.80	25.11

Table 9.3: True, prior and resulting ν_{ks} values (OD mean flow estimates) when the proposed algorithm for four different cases of relative weight of the prior and the sample with respect to information. A comparison with an standard Least Squares (LS) method is provided

Case 4. $m = 1$ and $n = 100$: Weak informative prior and strongly informative sample. The mean estimated flows are much closer to their observed values. In this case, the sample is given a high relative weight.

The algorithm converged in $5, 7, 8$ and 8 iterations for Cases 1 to 4, respectively, which implies a fast convergence. The computer programs were run on a Dell Optiplex 755 computer with 4 Gb of memory and a processor Intel Core 2 Quad Q6700 (2.66 GHz, 8 MB de cache L2, 1066 MHz FSB). The corresponding cpu times were 1.39, 2.11, 2.37 and 2.56 sec., respectively.

To illustrate the influence of an erroneous prior, we have modified the fictitious sample by multiplying the flows by a constant coefficient 1.3, and analyzed how the results are corrected when one observes the sample. The results appear in Table 9.4, where we have also shown the original prior. Note that the posterior estimates are very close to the real values, showing the power of the method to correct erroneous priors by means of informative samples.

OD	Prior	Mod. prior	Posterior $(m = 1; n = 100)$
1 -2	40.00	52.00	45.12
1 -3	80.00	104.00	87.69
4 -2	60.00	78.00	67.63
4 -3	20.00	26.00	25.73

Table 9.4: Modified prior and resulting ν_{ks} values (OD mean flow estimates) when solving the gamma model.

9.5.1.1 A comparison with other models

In order to compare our method with a standard one, we have used a standard bi-level procedure based on the Least Squares (LS) method (9.9)-(9.11) and the WMV problem (3.43)-(3.46)[4], and the results appear in the last column of Tables 9.2 and 9.3. As it can be seen, the point estimates are comparable with those of the proposed method, which in some cases outperforms the existing methods. Since the estimated link and OD flows are very similar to those given by other recognized methods, the proposed method can be validated. However, our Bayesian proposed method permits including prior information, which is important when sufficient data is not available, as it usually happen when estimating OD flows.

Since we have assumed the true flow values, we can evaluate the quality of the proposed method by providing the relative Root Mean Square Errors (RMSEs) of the OD estimates. A simulation study (100 trials) has been done and the relative RMSEs evaluated. The results indicate that the Bayesian method is good and equivalent in terms of quality to that of the LS method (see Table 9.5).

	Relative RMSE	
OD	Bayesian method	LS method
1-2	0.02458	0.02034
1-3	0.01812	0.01146
4-2	0.02541	0.01765
4-3	0.08192	0.10471

Table 9.5: Relative RMSE of the OD estimates for the proposed and LS methods.

9.5.2 The Ciudad Real network

In this section we use the Ciudad Real network to illustrate the proposed methods. It consists of 555 OD pairs, 412 links and 183 nodes, which means the representation of 90% of the city streets. This network is shown in Figure 9.1, where the subset of observed link flows appears outlined.

The proposed Gamma model and methods have also been used to predict the Ciudad Real OD flows and estimate the parameters, using the proposed algorithm described in section 9.3.4.

As in the previous example, the prior is generated with the sample resulting from applying the method described in subsection 9.3.3 with $\kappa = 0.000001$.

The flows resulting from the WMV problem are also used to calculate the flow proportions matrix β_{ijks}, of OD flows t_{ks} using link ℓ_{ij}.

[4]This model can be seen as a simple extension of Yang et al. (1992)'s model explained in Section 4.3.

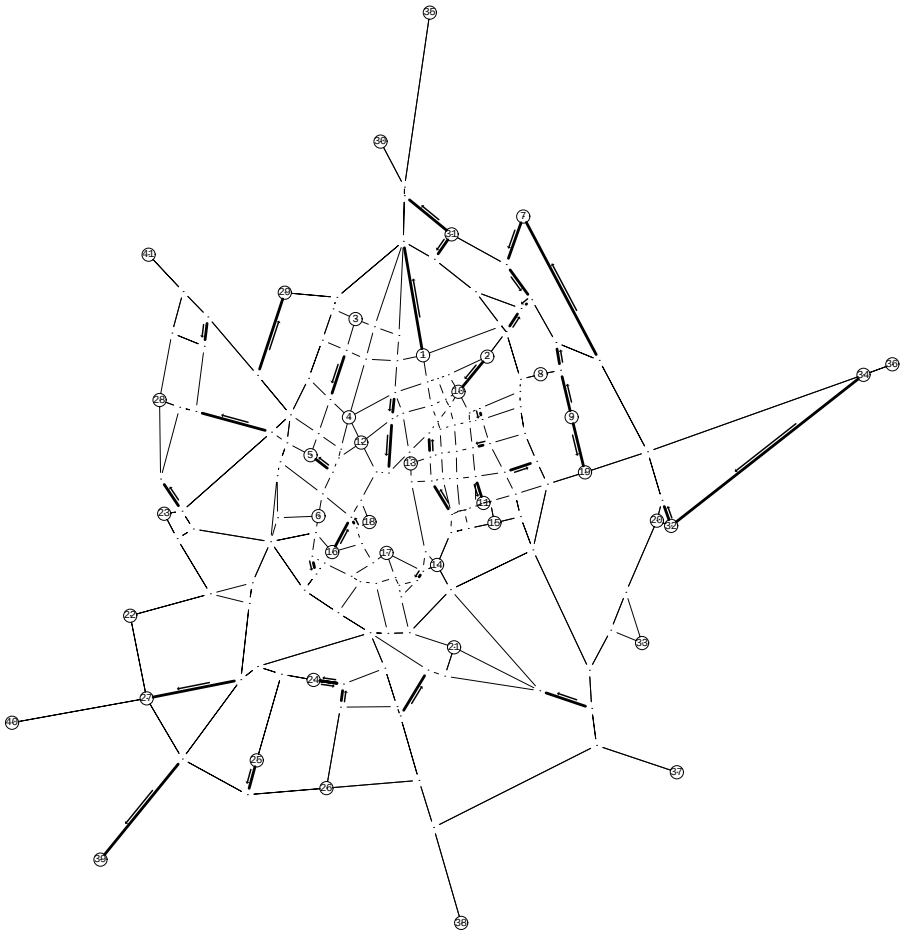


Figure 9.1: The Ciudad Real network showing the link count location and the origin-destination nodes.

For calculating the posterior hyperparameters and the OD and link flows we have used the proposed algorithm. Figure 9.2 shows some of the prior and resulting $\hat{\nu}_{ks}$ values (OD mean flow estimates) when solving the algorithm for four different cases of relative weight of the prior and the sample with respect to information. In addition, the results are compared with the Bayesian Network BN-WMV method proposed in Castillo et al. (2008g).

An analysis of the results (see Table 9.6 and Figure 9.2) leads to the fact that they are not very far from each other, but the prior OD matrix seems to have more weight in the Gamma models.

As expected, when a strong informative prior is supposed ($m = 100$ and $n = 1$), the resulting flows are very similar to the prior ones.

When a weak informative prior is supposed ($m = 1$ and $n = 100$ and also $m = 1$

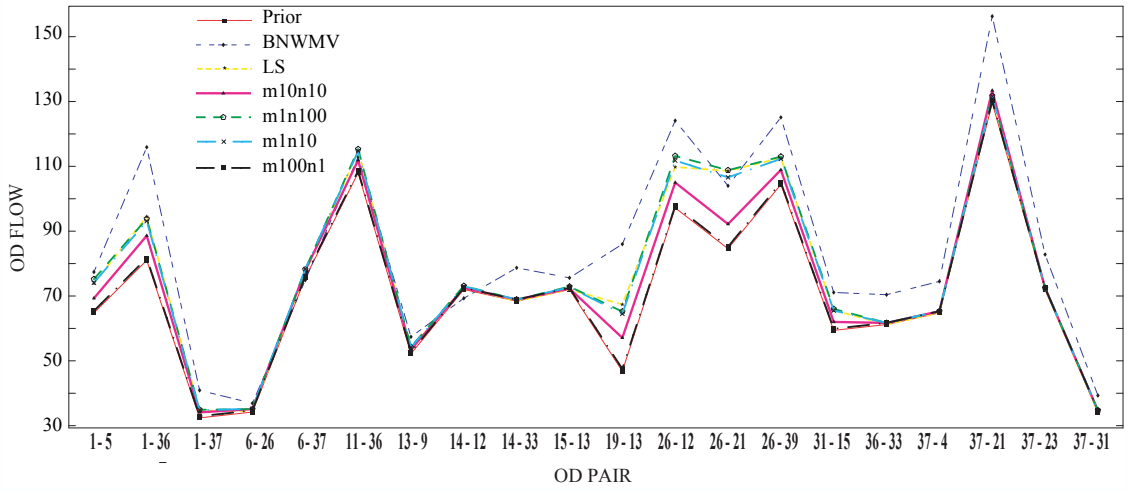


Figure 9.2: Some OD flows calculated using the Bayesian network and the Gamma Bayesian methods.

and $n = 10$), the resulting flows are usually closer to those predicted using the BN-WMV method. Note that in this last case, the predictions, using both weight coefficients, are very similar. This suggests that for large networks, the relation m/n need not be bigger than $1/10$. Note also that the estimated flows using $m = 10$ and $n = 10$, as expected, are values which are intermediate to the prior and those predicted using $m = 1$ and $n = 100$.

In summary, an analysis of Figure 9.2 and Table 9.6 leads to the same conclusions as those in the simple example of the previous section. Even though, in this figure only a fraction of the results are shown, due to space limitations, the conclusions are valid for all of them. In addition, since the predicted flows are similar to those corresponding to the LS method, they seem to be a satisfactory estimation of the traffic flows.

The computation times required to solve Problems (3.43)-(3.46) and (9.9)-(9.11) on the indicated computer (see the Nguyen-Dupuis example) were around 400 seconds and 0.03 seconds for this network, while the computational times to solve Problem (9.14) was negligible (GAMS reports zero time, which means a time smaller than 0.01 sec.).

LINK	True	Prior	$m = 100$ $n = 1$	$m = 10$ $n = 10$	$m = 1$ $n = 10$	$m = 1$ $n = 100$	LS	BN-WMV
1 -65	239.70	176.33	182.58	203.03	217.35	219.97	215.98	236.62
2 -10	63.21	64.90	64.00	62.08	63.86	64.21	61.79	62.84
7 -106	21.73	19.80	21.28	21.92	22.42	22.53	21.07	21.66
9 -19	41.49	37.80	39.31	40.35	41.24	41.42	39.95	41.34
9 -116	39.51	37.80	39.31	41.15	43.34	43.85	42.44	39.44
16 -96	257.79	223.80	222.85	231.01	238.06	239.55	240.05	254.47
24 -168	11.34	10.80	11.30	11.54	11.76	11.81	11.32	11.32
25 -42	12.20	10.80	11.79	12.24	12.63	12.70	11.73	12.15
31 -73	104.03	97.20	99.71	94.23	103.37	103.71	101.28	106.30
31 -153	197.67	194.25	191.54	183.11	183.57	185.30	184.02	193.17
32 -157	81.64	54.33	60.85	65.09	68.42	69.06	62.90	82.15
34 -32	27.55	27.00	27.50	27.73	27.91	27.95	27.46	27.53
45 -27	584.36	559.32	563.79	567.15	570.00	570.86	566.17	575.12
46 -39	624.47	516.60	522.04	545.38	566.00	570.21	565.75	620.11
54 -55	248.49	208.69	212.37	220.73	227.59	229.86	228.36	243.18
56 -177	71.54	68.51	67.57	71.33	73.89	73.67	72.77	75.13
58 -29	230.60	212.47	211.08	214.20	219.23	220.30	217.94	229.43
66 -67	253.99	222.15	226.43	236.72	246.69	248.78	245.31	253.42
67 -82	42.86	9.75	10.93	15.92	22.76	24.39	23.25	42.24
90 -87	154.13	136.78	138.47	142.54	144.77	145.40	143.86	152.33
98 -5	112.42	87.16	87.10	91.67	98.06	99.60	98.29	114.04
106 -172	68.04	146.83	134.40	116.73	90.44	86.50	82.62	66.75
107 -102	193.78	183.60	187.13	189.72	193.72	194.56	191.82	193.24
112 -96	148.90	153.76	155.03	151.89	151.30	151.36	148.39	146.19
115 -118	67.34	54.00	55.08	59.23	62.97	63.76	62.79	66.80
116 -84	31.99	16.20	16.76	20.21	23.65	24.40	23.99	32.46
126 -7	149.54	80.26	93.70	108.40	128.91	132.28	128.80	151.62
130 -11	64.74	55.80	56.85	59.82	62.42	62.95	62.02	64.38
131 -104	299.07	277.20	279.81	285.31	290.00	290.94	288.56	297.66
134 -173	184.63	129.35	135.65	147.31	153.88	155.54	151.19	191.56
142 -127	340.36	342.85	344.44	342.58	340.71	340.39	340.16	335.23
143 -144	126.49	68.12	73.87	82.00	97.10	100.86	96.37	130.22
146 -80	110.37	97.20	98.81	104.23	108.82	109.70	108.58	109.83
158 -85	322.37	290.31	294.32	312.59	323.93	324.18	319.83	315.91
161 -170	131.46	113.77	115.44	117.12	118.40	120.61	119.89	132.42
168 -24	252.87	228.60	231.78	241.37	249.77	251.29	250.11	251.89
169 -168	30.25	14.73	17.92	23.43	27.20	28.13	25.55	30.49
172 -173	86.49	58.61	63.67	69.73	73.16	74.31	70.46	86.89
178 -180	27.16	21.60	22.14	24.12	25.91	26.28	25.83	26.94

Table 9.6: Observed, prior and resulting link flows when the proposed algorithm is used for four different cases of relative weight of the prior and the sample with respect to information. The two last columns correspond with the results using the standard LS bi-level and BN-WMV approaches, respectively.

Appendix

A Notation

β_{ij}	parameter of the BPR function.
β_{ijks}	proportion of trips in the OD pair ks using link ℓ_{ij}.
γ_{ij}	parameter of the BPR function.
$\Gamma(\cdot)$	Gamma function.
δ'_{ij}	Kronecker delta.
δ^{ks}_{ijr}	element of the link-route-OD incidence matrix.
η	prior hyperparameters.
$\hat{\eta}$	posterior hyperparameters.
η_1	first component of the prior hyperparameters η.
η_2	second component of the prior hyperparameters η.
$\hat{\eta}_1$	first component of the posterior hyperparameters $\hat{\eta}$.
$\hat{\eta}_2$	second component of the posterior hyperparameters $\hat{\eta}$.
ℓ_{ij}	link joining nodes i and j.
θ	scale parameters of the Gamma function.
θ_{ijks}	scale parameter of the Gamma distribution associated with variable v_{ijks}.
θ_{ksr}	scale parameter of the Gamma distribution associated with variable f_{ksr}.
κ	weighting factor.
λ_i^2	auxiliary parameter.
μ	mean of the v_{ijks} variables.
ν_{ks}	scale parameter of the Gamma distribution associated with variable t_{ks}.
ρ	weighting factor.
$\mathcal{A}$	set of links.
c_{ij}	cost associated with traversing link ℓ_{ij}.
$C_{ij}(\cdot)$	integral of the travel time function associated with link ℓ_{ij}.
$\mathcal{D}$	set of destination nodes.
$\mathcal{F}$	family of Gamma distributions.
$f(\theta;\eta)$	prior distribution.
$f(\theta;\eta\|\mathbf{p})$	posterior distribution.
f_{ksr}	random number of users choosing route r of OD pair ks.
$f(\mathbf{p}\|\theta)$	likelihood of the data.
$g(x;\theta)$	probability density function.
$G(\theta_\iota, 1)$	Gamma distribution with parameters θ_ι and 1.
i	link begin node.
I	number of OD pairs.
j	link end node.
k	origin node.

ks	origin-destination pair from nod k to node$s$.
m	size of the virtual sample and weight of the prior information.
n	sample size.
$\mathcal{O}$	set of origin nodes.
$\mathcal{OD}$	set of origin-destination pairs.
$\mathbf{p}$	generic vector containing the data.
$q(\theta\|\eta)$	general conjugate for the Gamma family.
q_{ij}	parameter of the BPR function.
r	route.
$\mathcal{R}$	set of routes.
R_{ks}	number of routes of OD ks.
s	destination node.
t_{0ij}	parameter of the BPR function.
t_{ks}	random number of users traveling through OD pair ks.
$\bar{t}_{ks}$	prior OD flow associated with OD pair ks.
$\hat{t}_{ks}$	OD flows associated with the observed sample.
$\bar{t}_{ks\ell}$	element of the virtual sample.
U_{ks}	multivariate Gamma random variable associated with OD ks.
$\mathbf{v}$	vector of link flows v_{ijks}.
v_{ijks}	random number of users using link ℓ_{ij} and traveling the OD ks.
$\mathbf{W}$	vector containing the link flows to be estimated.
w_{ij}	random number of users using link ℓ_{ij}.
$\bar{w}_{ij}$	observed flow through link ℓ_{ij}.
X_ι	random variable.
x_ℓ	element of the sample.

Chapter 10

Upper bound of the number of sensors required for total observability

Contents

10.1 Introduction

In this chapter we will focus on the observability problem in traffic models. As already explained in Chapter 5, the observability problem consists of determining if a given subset of available flow measurements is sufficient to estimate another subset of traffic flows. In particular, we deal with the problem of link flow estimation based on link flow observations. In Hu et al. (2009), Castillo et al. (2010), Castillo et al. (2011) and Ng (2012) the problem of determining the smallest number of counting sensors to be installed to infer the flows of all other non-equipped links in a traffic network is discussed (see Chapter 5 for more details on these works). Obtaining the exact number of links required to be equipped with sensors is relevant because it is well known that redundant information can lead to incompatibility problems (in the presence of error measurements or model inadequacy).

Ng (2012) proposes a node-based approach to solve the problem whose main advantage is that path enumeration is not required. Furthermore, the author proves that the conjecture made in Hu et al. (2009) that "there may be an upper bound on the number of basis links that is governed by the network topology irrespective of the total number of links in the network" does not hold. Moreover, the upper bound is demonstrated to be $m-n$, where m and n are the number of links and non-centroid nodes, respectively, which is dependent on the total number of links and nodes but not on the network topology.

Therefore, the problem raised by Ng (2012) is:

> *"Given that only link flow information is permitted, what is the minimum number of links to be equipped with sensors in order to have full link observability (knowledge of all link flows)?"*

Contrary, in this chapter we answer the following question:

> *"If we relax the limitation of link flow information, does a better bound for the minimum number of links to be equipped with sensors in order to get full observability of link flows exist?"*

In this chapter we demonstrate that:

1. The upper bound of the number of counting sensors to be installed to infer the flows of all other non-equipped links proposed by Ng (2012) can be improved if partial path information (a subset of linearly independent paths) is used.

2. The approaches of Hu et al. (2009), Castillo et al. (2010) and Castillo et al. (2011) can be improved to work without enumeration of all paths. In fact, the number of required paths is low and bounded by the upper bound given by Ng (2012).

3. The exact number of links to be equipped with sensors for the inference of all link flows can be obtained only from path information and this number, which is known to be the rank of the link-path incidence matrix (see Hu et al. (2009)), can be more than 16% less than the $m-n$ bound.

The chapter is structured as follows. In Section 10.2 we discuss the node-based approach and comment on its main disadvantages. In Section 10.3 we introduce some important concepts of linear algebra, such as cones and dual cones, that are needed to understand the following sections. In Section 10.4 we deal with the path-based approach. In Section 10.5 the problem of obtaining non-basis link flows in terms of basis link flows is solved for the path-based approach. In Section 10.6 we provide an algorithm to obtain a minimum set of linearly independent paths. Finally, in Section 10.7 we give some recommendations for the observability problem.

10.2 The node-based approach

Ng (2012) in his node-based approach considers as the only constraints to be satisfied by the link flows, the flow conservation equations at the non-centroid nodes

$$\mathbf{A}\mathbf{w} = \mathbf{0}, \tag{10.1}$$

where $\mathbf{A}$ is the modified node-link incidence matrix of dimension $n \times m$, n is the number of non-centroid nodes (ordinary nodes), m is the number of links and $\mathbf{w}$ is the matrix of all link flows.

We note that the set of solutions of (10.1) is a linear space:

$$\mathbf{w} = \mathbf{K}\boldsymbol{\rho}, \tag{10.2}$$

where $\mathbf{K}$ is a matrix of column vectors (the basis of the linear space) and $\boldsymbol{\rho}$ is a column vector of real numbers (the coefficients of the linear combinations). Unfortunately, this set of flow solutions includes negative flows. Thus, we need to consider the non-negativity of link flows, which is crucial[1]. In fact, Equation (10.1) must be replaced by the system of equalities and inequalities

$$\mathbf{A}\mathbf{w} = \mathbf{0}, \quad \mathbf{w} \geq \mathbf{0}. \tag{10.3}$$

The set of solutions of (10.3) is a polyhedral convex cone, which contains only non-negative link flows.

As it will be seen in Section 10.3, the cone definition differs from the definition of linear space only in the non-negativity of the coefficients involved in the linear combinations.

Example 8 (The parallel network) *Consider the parallel network used in Hu et al. (2009) and Ng (2012) and reproduced in Figure 5.1. We recall that this network has four centroid nodes* $\{1, 2, 8, 9\}$ *and five non-centroid nodes* $\{3, 4, 5, 6, 7\}$.

In this case considering the flow balance at each of the regular nodes, the system (10.1) becomes

$$\begin{pmatrix} 1 & 0 & 1 & 0 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & -1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & -1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & -1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & -1 & -1 & 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \\ w_5 \\ w_6 \\ w_7 \\ w_8 \\ w_9 \\ w_{10} \\ w_{11} \\ w_{12} \\ w_{13} \\ w_{14} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \tag{10.4}$$

[1]This condition needs to be included explicitly in the mathematical flow model not only because link flows must be non-negative, but in order to avoid negative flows that can arise if some errors in data or model imprecisions exist.

with solution (see Castillo et al. (2000)) including negative flows, which have no sense:

$$\mathbf{w} = \mathbf{K}\boldsymbol{\rho} = \begin{pmatrix} -1 & 0 & -1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & -1 & 1 & 0 & -1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \rho_1 \\ \rho_2 \\ \vdots \\ \rho_9 \end{pmatrix}, \tag{10.5}$$

where $\mathbf{K}$ *is a matrix and* $\boldsymbol{\rho}$ *is a column vector of arbitrary real numbers (coefficients of the linear combinations).*

10.3 Some algebraic required background

In order to understand the mathematical derivations in this paper some mathematical background is needed. In particular the concepts of cone, edges of a cone and dual cone are relevant. Important applications of cones and dual cones can be seen in Castillo and Jubete (2004) and Castillo et al. (2002).

Similar to linear spaces, which are sets of vectors generated by linear combinations (with real coefficients) of basis vectors, we can define cones, which are non-negative linear combinations of basis vectors. The reason for limiting the coefficients to be non-negative is that the linear coefficients here are link or route flows and they cannot be negative. This explains that the set of feasible link flows is given by (10.3) and not by (10.1).

The role played by a basis, as a minimal subset of vectors able to generate all vectors in a linear space is played by the set of edges in a cone, which is a minimal subset of vectors able to generate all vectors in a cone (using non-negative linear combinations). Thus, knowing the edges of a cone is as important as knowing a basis in a linear space.

Definition 7 (Polyhedral convex cone) *Given a set of vectors* $\{\mathbf{m}_1, \mathbf{m}_2, \ldots, \mathbf{m}_r\}$ *the set of all vectors that can be obtained from this set by non-negative linear combinations of them is called the* polyhedral convex cone *generated by this set of vectors. We denote as* $\mathbf{M}_\pi$ *to the cone generated by the columns of matrix* $\mathbf{M}$*. We will use* π *to refer to non-negative real numbers used in the non-negative linear combinations to differentiate them from the real numbers* ρ *used in linear spaces.*

It is interesting to know that any linear subspace is a cone, but not every cone is a linear subspace (see Castillo et al. (1999) or Castillo et al. (2000)).

Definition 8 (Dual or polar cone of a given cone) *Given a cone* $\mathbf{M}_\pi$*, the set of vectors defined as*

$$\{\mathbf{u} \in E^n | \mathbf{u} \cdot \mathbf{v} \leq 0; \forall \mathbf{v} \in \mathbf{M}_\pi\}, \tag{10.6}$$

that is, the dual cone is the set of vectors of the Euclidean space E^n *of dimension* n *such that their dot products with all vectors of the cone* $\mathbf{M}_\pi$ *are non-positive.*

We indicate that the dual cone is an extension of the concept of orthogonal linear subspace to a given linear subspace, because if $\mathbf{M}_\pi$ is a linear subspace, its dual cone is its orthogonal set (linear subspace).

To find the set of all solutions of the system of Equations (10.3) it is sufficient to realize that the system of these two equations can be written as[2]

$$\mathbf{A}\mathbf{w} = (\mathbf{A}| - \mathbf{A}| - \mathcal{I})^T \mathbf{w} \quad \leq \quad \mathbf{0}, \tag{10.7}$$

which expresses that $\mathbf{w}$ is the dual cone of the cone generated by the rows of the partitioned matrix $\mathbf{A} = (\mathbf{A}| - \mathbf{A}| - \mathcal{I})^T$, where $\mathcal{I}$ is the identity matrix.

To find this dual cone defined in (10.3), which is the solution of the system (10.3), we can use the techniques described in Castillo and Jubete (2004) (the Γ-algorithm) to obtain

$$\mathbf{w} = \sum_{i=1}^{r} r_i \mathbf{h}_i = \mathbf{H}\mathbf{r}, \tag{10.8}$$

where $\{\mathbf{h}_i; i = 1, 2, \ldots, r\}$ is the set of cone generators, that are associated with the set of all paths, $\{r_i; i = 1, 2, \ldots, r\}$ is the set of all path (non-negative) flows and $\mathbf{r}$ is the column matrix of all non-negative path flows.

Definition 9 (Path) *A path is a set of links that satisfy the following conditions*[3]*:*

1. *One of the links starts at one centroid node.*
2. *One of the links ends at one centroid node.*
3. *The set of links is connected (with the exception of the first and last links all other links share only one node with other links in the path).*

Definition 10 (Path vector) *A path vector is a vector of* m *components which has all zero elements but those associated with the links of a given path, that are ones.*

The reason to make these elements equal to one is that we are interested in the coefficients r_i *in (10.8) to be the path flows.*

[2]The $\mathbf{A}$ and $-\mathbf{A}$ matrices arise because we have replaced the equation $\mathbf{A}\mathbf{w} = \mathbf{0}$ by the two inequations $\mathbf{A}\mathbf{w} \leq \mathbf{0}$ and $\mathbf{A}\mathbf{w} \geq \mathbf{0}$.

[3]We exclude paths with cycles, because they are not very common and distort the problem. However, this is not a limitation of the proposed methods and has not a practical relevance.

This means that one link must start at a centroid, one link must end at a centroid and all other links must be connected (the origin node of a link must coincide with the end node of another link).

It is clear that a path vector is a solution of the system (10.3), because all its components are non-negative ((10.3b) is satisfied) and each node either has no links in the associated path or exactly two links (one entering the node and one exiting the node), which implies that (10.3a) holds.

It is initially surprising to see that the set of generators of the cone (cone edges) is the set of all paths, but a deep reflection lead us to the conclusion that it must be that way. Thus, we can obtain the set of all paths by obtaining the dual cone of the cone generated by the rows of matrix **A**. The following theorem proves this important property.

Theorem 16 (Edges of the cone (10.3)) *The path vectors are the only vectors in the cone (10.3) that cannot be obtained by linear convex combinations of any other two vectors, that is, they are the edges of the cone (10.3).*

Proof. Assume that a path **p** vector is a convex linear combination of other two vectors **u** and **v** in cone (10.3), then we have

$$\mathbf{p} = \alpha\mathbf{u} + (1-\alpha)\mathbf{v}; \quad 0 < \alpha < 1; \quad \alpha \in \mathbb{R}.$$

If one component of **p** is null (one), the corresponding components of **u** and **v** must be null (one). This means that **u**, **v** and **p** are the same vectors. Thus, **p**, **u** and **v** must be edges of the cone. ∎

Example 9 (The parallel network) *If we consider the parallel network in Figure 5.1 and the associated system of equations (10.3), we conclude that its general solution for the link flows is:*

$$\mathbf{w} \;=\; \mathbf{Hr} = \begin{pmatrix} 1000100010001010000001000 \\ 0100001000100000010100010 \\ 0010010001000101000001000 \\ 0001000100010000010100001 \\ 0000110011000011000011000 \\ 1010000000000110000000000 \\ 0000001100110000110000011 \\ 0101000000000000000110000 \\ 0000000011110011110000000 \\ 0000111100000000000001111 \\ 0000000000000111111000000 \\ 1010000011110000000000000 \\ 0000000000000000000111111 \\ 0101111100000000000000000 \end{pmatrix} \begin{pmatrix} r_1 \\ r_2 \\ r_3 \\ r_4 \\ r_5 \\ r_6 \\ r_7 \\ r_8 \\ r_9 \\ r_{10} \\ r_{11} \\ r_{12} \\ r_{13} \\ r_{14} \\ r_{15} \\ r_{16} \\ r_{17} \\ r_{18} \\ r_{19} \\ r_{20} \\ r_{21} \\ r_{22} \\ r_{23} \\ r_{24} \end{pmatrix}, \qquad (10.9)$$

where each column of matrix $\mathbf{H}$ *corresponds to one path starting and ending at the centroid nodes, the row numbers refer to the link numbers in Figure 5.1, and the* $r_i; i = 1, 2, \ldots, 24$ *values of the column matrix* $\mathbf{r}$ *contains the (non-negative) path flows.*

Note that the matrix in (10.9) is in fact the link-path incidence matrix considering all topologically feasible paths. However, it is clear that the subset of link flows generated in (10.9) is not the same subset of link flows generated in (10.5). Nevertheless, any subset of link flows of the form (10.9) can be written in the form in (10.5).

10.4 The path-based approach

First of all we indicate that, as in the node approach, we assume static flow, that is, we assume that the network conditions considered are static.

Equation (10.9) is the model to be used if all topologically possible paths are considered. However, since not all paths are always considered in real cases, we need to give a precise indication of which paths are considered and which ones are not.

Once the list of considered paths (a submatrix $\mathbf{H}_0$ of $\mathbf{H}$) and the corresponding path flows (submatrix $\mathbf{r}_0$ of $\mathbf{r}$) have been given, the traffic network problem is completely defined. This means that our set of link flows will be given by the cone

$$\mathbf{w} = \mathbf{H}_0\mathbf{r}_0, \tag{10.10}$$

where $\mathbf{H}_0$ and $\mathbf{r}_0$ are submatrices of matrices $\mathbf{H}$ and $\mathbf{r}$, respectively, and matrix $\mathbf{H}_0$ contains as columns only the actual path vectors (not all feasible paths).

Since Equations (10.3) do not define completely our traffic problem because it does not necessarily include all paths, we can ask: what are the set of equations defining it? or in other words, what are the equations of the cone $\mathbf{w} = \mathbf{H}_0\mathbf{r}_0$? This set of equations can be obtained by calculating the dual cone of the cone $\mathbf{H}_0$.

Equation (10.9) expresses the link flows in terms of the path flows, which are the relations used by Hu et al. (2009) and Castillo et al. (2010). However, in order to determine the minimum set of links to be equipped with sensors, we do not need the whole $\mathbf{H}_0$ matrix but a submatrix providing its rank, that by definition must satisfy the constraint $r_h = rank(\mathbf{H}_0) \leq \min(m, r)$, where m is the number of links and r is the number of paths, and after the contribution of Ng (2012) we know that $r_w = rank(\mathbf{H}_0) \leq \min(m - n, r^*)$, where r^* is the number of linearly independent columns of matrix $\mathbf{H}_0$, which proves that $r^* \leq m - n$ independent routes are required to work with the methods proposed by Hu et al. (2009) and Castillo et al. (2010), but the number of linearly independent routes can be even less if $r_h < m - n$.

It is important to observe linearly independent link flows, because if we observe linearly dependent link flows we can have inconsistencies if some errors are present. In other words, linearly dependent links can be a problem. Thus, it is important to obtain the right upper bound of links to be observed.

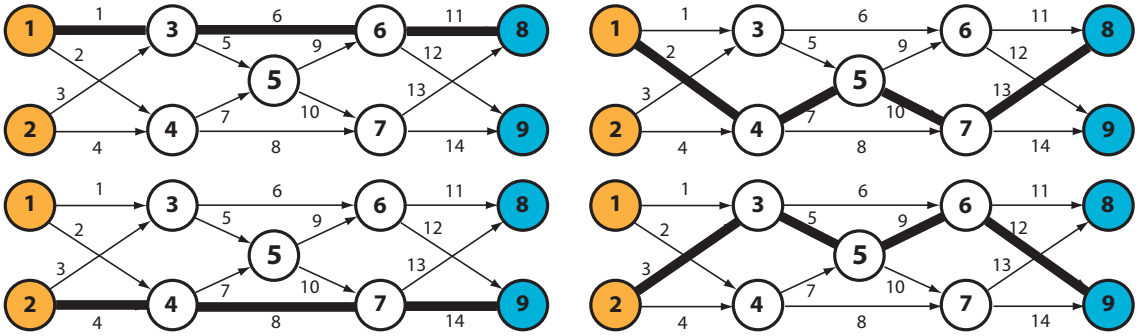


Figure 10.1: Illustration of the four paths used in the example.

Example 10 (The parallel network revisited) *If in the parallel network we consider only the four paths shown in Figure 10.1 and given by:*

$$\{1,6,11\}, \quad \{2,7,10,13\}, \quad \{4,8,14\}, \quad \{3,5,9,12\}$$

we can write

$$\mathbf{w} = \mathbf{H}_0\mathbf{r}_0 = \begin{pmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \\ w_5 \\ w_6 \\ w_7 \\ w_8 \\ w_9 \\ w_{10} \\ w_{11} \\ w_{12} \\ w_{13} \\ w_{14} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} r_1 \\ r_2 \\ r_3 \\ r_4 \end{pmatrix}, \tag{10.11}$$

which shows that we need to install only 4 *sensors, for example, in links* $1, 2, 3$ *and* 4*, even though* $m = 14$*,* $n = 5$ *and* $m - n = 9$*. Thus, the bound rank* $r_h = 4$ *of matrix* $\mathbf{H}$ *is a better bound (the exact value) than* $m - n = 9$ *and does not depend directly on the network topology nor on the number of links or nodes of the network. This new bound implies a saving of* 55.56% *in the number of required links to be equipped with sensors.*

The equations defining this set can be obtained, as indicated above, by obtaining the dual of the cone (10.11), using the Γ*-algorithm described in Castillo and Jubete (2004)*

and Castillo et al. (2002). These equations are:

$$\begin{pmatrix} 0 & 0 & -1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \mathbf{w} = \mathbf{0} \tag{10.12}$$

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix} \mathbf{w} \geq \mathbf{0}. \tag{10.13}$$

Thus, (10.11) and (10.12)-(10.13) are equivalent and both define the set of feasible link flows, the first by an explicit expression and the last two by its equations.

Equation (10.12) simply expresses that the flows in all links of the same four paths (see Figure 10.1) must be the same, and condition (10.13) forces the flows of links 1 *to* 4 *to be non-negative. Finally, note that (10.12)-(10.13) together force all link flows to be non-negative. Of course, these very restrictive conditions refer only to the very special case of this example.*

The case in this example has been selected on purpose to show that the number of required links can be much smaller (in percentage) than the one resulting from the $m - n$ bound. However, it is clear that in real large networks the difference will not be so large.

We have selected a case in which the number of paths is smaller than the number of links, because the example needs to be simple to be illustrated. However, finding an example in a real case with many more paths than links is very easy, because the number of linearly independent paths is very large to choose from.

The following example shows that the bound obtained when using the proposed model is far from the bound obtained from the node-based approach.

Example 11 (The Ciudad Real and Cuenca networks) *In order to see the relevance of path-based upper bound compared with the node-based bound, in Table 10.1 we show the characteristics of the Ciudad Real and Cuenca networks (see Figures 7.2 and 10.2, respectively) and the resulting bounds.*

As you can see, the savings can reach important values such as more than 16% *in the number of required links to be equipped with sensors, and more than* 80% *in the number of paths to be enumerated. For very large networks this savings can increase further.*

Note that even though the number of paths selected is large[4] *(2190 for Cuenca and*

[4]The Ciudad Real and Cuenca routes were selected using the Castillo et al. model explained in Section 3.2.1 which does not require path enumeration.

Network	links (m)	nodes	non-centroid nodes (n)	paths	$rank(H)$	$m-n$	% savings in	
							link counts	path enumeration
Cuenca	672	232	164	2190	422	508	16.92%	80.73%
Ciudad Real	423	183	133	1141	248	290	14.48%	78.26%

Table 10.1: Illustration of the improvement of the path-based with respect to the node-based bounds and associated savings.

1141 for Ciudad Real), the number of linearly independent paths is small (422 and 248, respectively).

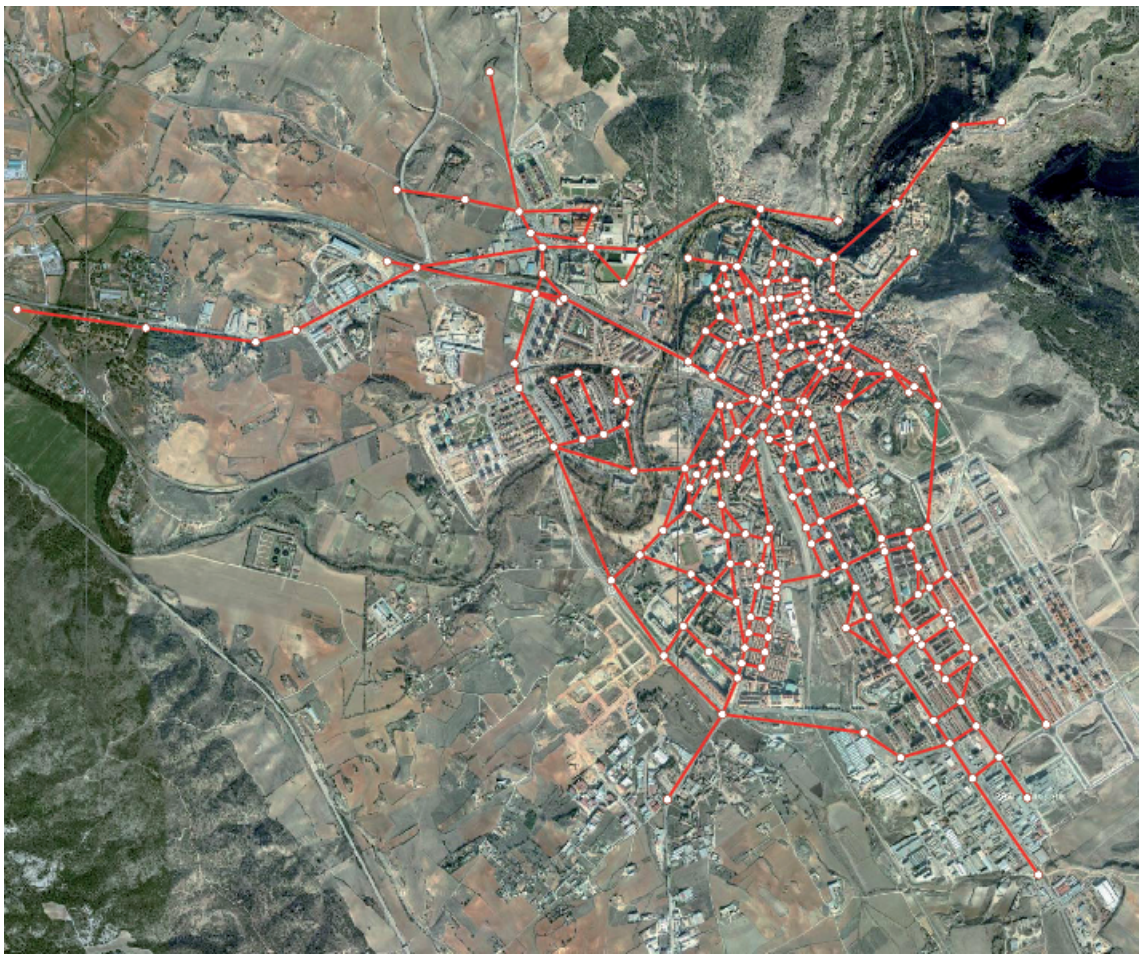


Figure 10.2: Cuenca traffic network used in the Example of application.

Fuente documento base: Google maps

10.5 Obtaining non-basis in terms of basis link flows

Ng (2012) provides a formula to obtain non-basis in terms of basis link flows, as follows. After proving that the rank of matrix $\mathbf{A}$ is n, he partitions matrix $\mathbf{A}$ and writes (10.1) as

$$(\mathbf{B}|\mathbf{N})\begin{pmatrix}\mathbf{w}_B\\ \mathbf{w}_N\end{pmatrix} = \mathbf{0}, \tag{10.14}$$

where $\mathbf{B}$ is a regular matrix of dimension $n\times n$ and $\mathbf{N}$ is a matrix of dimensions $n\times(m-n)$. Next, from (10.14) he gets

$$\mathbf{w}_B = -\mathbf{B}^{-1}\mathbf{N}\mathbf{w}_N, \tag{10.15}$$

which is the expression relating non-basis $\mathbf{w}_B$ and basis $\mathbf{w}_N$ link flows.

However, there is another alternative to obtain this relation based on partitioning the link-path incidence matrix $\mathbf{H}_0$ in (10.10), that can be written as

$$\mathbf{w} = \begin{pmatrix}\mathbf{w}'_B\\ \mathbf{w}'_N\end{pmatrix} = \left(\begin{array}{c|c}\mathbf{C} & \mathbf{D}\\ \hline \mathbf{E} & \mathbf{F}\end{array}\right)\begin{pmatrix}\mathbf{r}_1\\ \mathbf{r}_2\end{pmatrix} = \mathbf{H}_0\mathbf{r}_0, \tag{10.16}$$

where $\mathbf{w}'_B$ and $\mathbf{w}'_N$ are the non-basis and basis link flow vectors with dimensions $n'\times 1$ and $(m-n')\times 1$, respectively, $\mathbf{H}_0$ is the matrix in (10.10), $\mathbf{E}$ is an invertible matrix of dimension $(m-n')\times(m-n')$, $\mathbf{C}$, $\mathbf{D}$ and $\mathbf{F}$ are matrices of dimensions $n'\times(m-n')$, $n'\times(r-m+n')$ and $(m-n')\times(r-m+n')$, respectively, and r is the number of routes.

We note that in principle, the basis flows $\mathbf{w}_N$ and $\mathbf{w}'_N$ need not be equal, and consequently, non-basis flows $\mathbf{w}_B$ and $\mathbf{w}'_B$ need not be equal either.

Since $\mathbf{E}$ is invertible, we can use a result from Castillo et al. (2010) (see Theorem 13 in Chapter 5) to obtain

$$\mathbf{C}\mathbf{E}^{-1}\mathbf{F} = \mathbf{D}, \tag{10.17}$$

and using (10.16) and (10.17) we get

$$\begin{aligned}\mathbf{w}'_B &= \mathbf{C}\mathbf{r}_1 + \mathbf{D}\mathbf{r}_2\\ &= \mathbf{C}\mathbf{r}_1 + \mathbf{C}\mathbf{E}^{-1}\mathbf{F}\mathbf{r}_2\\ &= \mathbf{C}\left(\mathbf{E}^{-1}\mathbf{E}\mathbf{r}_1 + \mathbf{E}^{-1}\mathbf{F}\mathbf{r}_2\right)\\ &= \mathbf{C}\mathbf{E}^{-1}\left(\mathbf{E}\mathbf{r}_1 + \mathbf{F}\mathbf{r}_2\right)\\ &= \mathbf{C}\mathbf{E}^{-1}\mathbf{w}'_N,\end{aligned} \tag{10.18}$$

which is the new relation between the non-basis $\mathbf{w}'_B$ and the basis flows $\mathbf{w}'_N$.

The question we can raise is: can the same relations be obtained from both methods? If not, under what conditions does this equivalence hold? The following theorem answers this question.

Theorem 17 (Equivalence of node and path-based approaches) *If* $\mathbf{w}_B = \mathbf{w}'_B$, *then (10.15) and (10.18) are equivalent.*

Proof. If $\mathbf{w}_B = \mathbf{w}'_B$, then $\mathbf{w}_N = \mathbf{w}'_N$ and $n = n'$. As both $\mathbf{w}_N$ and $\mathbf{w}'_N$ form a basis, $\mathbf{w}_B$ and $\mathbf{w}'_B$ can be determined uniquely from them through equations (10.15) and (10.18), respectively. In other words, matrices $\mathbf{B}^{-1}\mathbf{N}$ and $\mathbf{C}\mathbf{E}^{-1}$ are unique. Therefore, taking into account that $\mathbf{w}_B = \mathbf{w}'_B$, $\mathbf{w}_N = \mathbf{w}'_N$ it follows that

$$\mathbf{B}^{-1}\mathbf{N} = -\mathbf{C}\mathbf{E}^{-1}, \tag{10.19}$$

where (10.19) proves that the solutions in (10.15) and (10.18) are the same. ∎

Corollary 2 (Reconstructing information) *Matrices* $-\mathbf{C}\mathbf{E}^{-1}$, $\mathbf{E}$ *and* $\mathbf{F}$ *permit reconstructing matrices* $\mathbf{C}$ *and* $\mathbf{D}$.

Proof.

From Equations (10.17) and (10.19) we get

$$\mathbf{C} = -\mathbf{B}^{-1}\mathbf{N}\mathbf{E} = \mathbf{C}\mathbf{E}^{-1}\mathbf{E} \tag{10.20}$$

$$\mathbf{D} = \mathbf{C}\mathbf{E}^{-1}\mathbf{F}. \tag{10.21}$$

∎

Corollary 2 is an interesting result with immediate practical applications because it allows saving memory when storing all paths. This means that matrices $\mathbf{C}$ and $\mathbf{D}$ need not be stored if $-\mathbf{C}\mathbf{E}^{-1}$, $\mathbf{E}$ and $\mathbf{F}$ are saved.

The following example illustrates the equivalence of the node and the path approaches to obtain non-basis link flows in terms of basis link flows.

Example 12 (The parallel network with all paths) *If we partition matrix* $\mathbf{A}$ *in (10.4) and* $\mathbf{w}$ *as follows*

$$\mathbf{A}\mathbf{w} = \left(\mathbf{B} \,\middle|\, \mathbf{N} \right) \left(\frac{\mathbf{w}_B}{\mathbf{w}_N} \right), \tag{10.22}$$

where

$$\mathbf{w}_B = (w_6, w_8, w_{10}, w_{12}, w_{14})'$$

$$\mathbf{w}_N = (w_1, w_2, w_3, w_4, w_5, w_7, w_9, w_{11}, w_{13})'$$

$$\mathbf{B} = \begin{pmatrix} -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & -1 & 0 \\ 0 & 1 & 1 & 0 & -1 \end{pmatrix}; \quad \mathbf{N} = \begin{pmatrix} 1 & 0 & 1 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \end{pmatrix}$$

we get the transformation $\mathbf{w}_B = -\mathbf{B}^{-1}\mathbf{N}\mathbf{w}_N$*, that is,*

$$\begin{pmatrix} w_6 \\ w_8 \\ w_{10} \\ w_{12} \\ w_{14} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & -1 & 0 & 0 \\ 1 & 0 & 1 & 0 & -1 & 0 & 1 & -1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & -1 & 0 & -1 \end{pmatrix} \begin{pmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \\ w_5 \\ w_7 \\ w_9 \\ w_{11} \\ w_{13} \end{pmatrix}. \tag{10.23}$$

In this case, the following set of nine link flows $\{w_1, w_2, w_3, w_4, w_5, w_7, w_9, w_{11}, w_{13}\}$ *are sufficient to calculate all other links* $\{w_6, w_8, w_{10}, w_{12}, w_{14}\}$*.*

If, alternatively, we partition matrix $\mathbf{H}$ *in (10.9) as in (10.16) with*

$$\mathbf{H} = \left(\begin{array}{c|c} \mathbf{C} & \mathbf{D} \\ \hline \mathbf{E} & \mathbf{F} \end{array}\right) = \left(\begin{array}{ccccccccc|ccccccccccccccccc} 1&0&1&0&0&0&0&1&0 & 0&0&0&0&0&1&0&0&0&0&0&0&0&0&0 \\ 0&1&0&1&0&0&0&0&1 & 0&0&0&0&0&0&0&0&0&0&1&0&0&0&0 \\ 0&0&0&0&1&1&0&0&0 & 1&1&0&0&0&0&0&0&0&0&0&1&1&1&1 \\ 1&0&1&0&0&0&1&0&0 & 0&0&1&1&1&0&0&0&0&0&0&0&0&0&0 \\ 0&1&0&1&1&1&0&0&0 & 1&1&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ \hline 1&0&0&0&1&0&1&1&0 & 0&0&0&0&0&0&1&0&0&0&0&1&0&0&0 \\ 0&1&0&0&0&1&0&0&1 & 0&0&0&1&0&0&0&0&1&0&0&0&0&1&0 \\ 0&0&1&0&0&0&0&0&0 & 1&0&1&0&0&1&0&1&0&0&0&0&1&0&0 \\ 0&0&0&1&0&0&0&0&0 & 0&1&0&0&1&0&0&0&0&1&1&0&0&0&1 \\ 0&0&0&0&1&0&1&0&0 & 1&0&1&0&0&0&1&1&0&0&0&1&1&0&0 \\ 0&0&0&0&0&1&0&0&0 & 0&1&0&1&1&0&0&0&1&1&0&0&0&1&1 \\ 0&0&0&0&0&0&1&0&0 & 0&0&1&1&1&0&1&1&1&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&1&0 & 0&0&0&0&0&1&1&1&1&1&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&1 & 0&0&0&0&0&0&0&0&0&0&1&1&1&1&1 \end{array}\right); \tag{10.24}$$

$$\begin{aligned} \mathbf{r}_1 &= (r_1, r_2, r_3, r_4, r_5, r_7, r_9, r_{13}, r_{19})'; \\ \mathbf{r}_2 &= (r_6, r_8, r_{10}, r_{11}, r_{12}, r_{14}, r_{15}, r_{16}, r_{17}, r_{18}, r_{20}, r_{21}, r_{22}, r_{23}, r_{24})'; \end{aligned}$$

we get

$$\mathbf{C}\mathbf{E}^{-1} = \begin{pmatrix} 1 & 0 & 1 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & -1 & 0 & 0 \\ 1 & 0 & 1 & 0 & -1 & 0 & 1 & -1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & -1 & 0 & -1 \end{pmatrix}$$

which is exactly the matrix in (10.23).

This means that we do not need the 24 *paths. In fact, it is sufficient with the nine paths* $\{1, 2, 3, 4, 5, 7, 9, 13, 19\}$*.*

We note that we can obtain matrices $\mathbf{C}$ *and* $\mathbf{D}$ *by multiplying the matrix* $-\mathbf{B}^{-1}\mathbf{N}$ *by* $\mathbf{E}$ *and* $\mathbf{F}$*, respectively, and then we have not to store those matrices.*

Example 13 (The parallel network with only four paths) *However, if we use (10.11) and partition the matrix in this expression as in (10.16) with*

$$\mathbf{E} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}; \quad \mathbf{C} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

where $\mathbf{D} = \mathbf{F} = \emptyset$ *and* $\emptyset$ *is the empty matrix, we get the transformation*

$$\begin{pmatrix} w_5 \\ w_6 \\ w_7 \\ w_8 \\ w_9 \\ w_{10} \\ w_{11} \\ w_{12} \\ w_{13} \\ w_{14} \end{pmatrix} = \mathbf{C}\mathbf{E}^{-1} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} w_1 \\ w_2 \\ w_3 \\ w_4 \end{pmatrix},$$

which is a transformation different from (10.23). In this case four link flows $\{w_1, w_2, w_3, w_4\}$ *are sufficient to generate all other link flows* $\{w_5, w_6, w_7, w_8, w_9, w_{10}, w_{11}, w_{12}, w_{13}, w_{14}\}$.

10.6 Obtaining a set of independent paths

Since a set of linearly independent paths is all we need to obtain the minimum number of links to be equipped with sensors to infer all link flows, it is convenient to have a tool to identify these subsets of linearly independent paths. The following algorithm (see Castillo et al. (2000)) is such a tool.

The practical relevance of this algorithm is high because it can save a lot of work due to the fact that the number of linearly independent paths is smaller or equal (normally far) than the number of links, which is normally much smaller than the number of paths. In other words, only a very small fraction of paths need to be enumerated, what is very important for very large networks. In addition, it facilitates the obtention of linearly independent paths.

Algorithm 2 (Obtaining a set of linearly independent paths) *The algorithm is as follows:*

INITIALIZATION STEP. *Let I be the identity matrix of dimension m (the number of links). Let $j = 1$ and choose a path vector $\mathbf{r}_j$.*

REGULAR PROCESS.

1. **Step 1. Obtain dot products.** *Multiply matrix $\mathbf{r}_j^T$ and $\mathbf{I}$ to obtain the dot products $\mathbf{t} = \mathbf{r}_j^T\mathbf{I}$ of $\mathbf{r}_j$ by all column vectors of $\mathbf{I}$.*

2. **Step 2. Choose a pivot.** *Choose one $t_p \neq 0$. This means that the path vector $\mathbf{r}_j$ is linearly independent of the previously selected path vectors.*

3. **Step 3. Update matrix I.** *First, divide the pivot column $\mathbf{I}_p$ by the pivot value t_p and next, transform the non pivot columns $\mathbf{I}_k; k \neq p$ of $\mathbf{I}$ by*

$$\mathbf{I}_k^* = \mathbf{I}_k - t_k\mathbf{I}_p^*; \quad k \neq p,$$

   where $\mathbf{I}_k^*$ and $\mathbf{I}_p^*$ refer to the transformed columns of $\mathbf{I}_k$ and $\mathbf{I}_p$, respectively. Note that if $t_p = 0$ then we have $\mathbf{I}_k^* = \mathbf{I}_k$.

4. **Step 4. Remove pivot column.** *Remove the pivot column $\mathbf{I}_p$ from matrix $\mathbf{I}$.*

5. **Step 5. Choose a path vector linearly independent of the previously selected paths.** *To choose a path vector $\mathbf{r}_{j+1}$ linearly independent of the previous ones, that is, such that $\mathbf{r}_{j+1} \cdot \mathbf{I_1} \neq 0$ we analyze the non-null elements of column matrix $\mathbf{I}_1$ and select path vectors with some unit elements (corresponding to the path links) in the same positions[5]. If this path vector does not exist, choose the following column of $\mathbf{I}$ until a path vector $\mathbf{r}_{j+1}$ and one index p such that $t_p = \mathbf{r}_{j+1} \cdot \mathbf{I_1} \neq 0$ is found. If there is no $t_p \neq 0$, this means that the path $\mathbf{r}_j$ is a linear combination of all previously selected path vectors. Otherwise, it is linearly independent of them. If no $\mathbf{r}_{j+1}$ can be found after using all columns of matrix $\mathbf{I}$, stop the process because your list of j already selected path vectors is the maximum number of paths satisfying this linear independence property. Otherwise, let j=j+1 and continue with Step 1.*

The complexity and the number and type of operations of this algorithm are exactly the same as in the Gaussian elimination, which has been studied for very long. This implies that the method can be applied to very large networks. Even more, we have no numerical problems because we have integer numbers as coefficients and almost all times ones, zeroes and minus ones (see Table 10.2).

[5]The ones (links in the path) must be selected such that the dot product be different from zero.

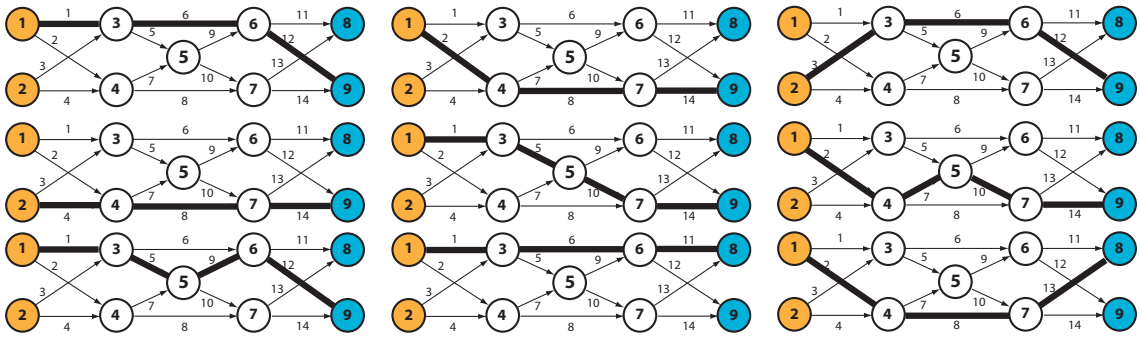


Figure 10.3: One set of 9 linearly independent path vectors of the parallel highway network of Hu et al. (2009).

Example 14 (Linearly independent paths (parallel example)) *To illustrate our algorithm, we apply it to the parallel network. We have chosen the set of path vectors*

$$\{1, 2, 3, 4, 5, 7, 9, 13, 19\},$$

which is shown in Figure 10.3. In Table 10.2 we show the evolution of matrix $\mathbf{I}$ *and the dot products as the algorithm advances. In the first column of Iteration 1 we have indicated the order used for the links.*

In Figure 10.4 we show what happens at Iteration 9*, where the last path (path vector* 9*) is selected based on its pivot column (boldfaced). The links associated with the non-null elements of* $\mathbf{I}_5$ *(fifth column of* $\mathbf{I}$*) have been emphasized in Figure 10.4(a). To facilitate the selection of the last linearly independent path vector* 9*, the links associated with value* 1 *appear in red and with a* $+$ *sign, while the links associated with a value* -1 *appear in green and a* $-$ *sign. In order to have a path vector which dot product with the column vector* $\mathbf{I}_i$ *is non-null, we must have a path passing through links having the* $+$ *or* $-$ *signs but not both. One such a path is the path vector* $\{2, 8, 13\}$*, but other path vectors are also valid, such as* $\{4, 8, 13\}$*,* $\{4, 7, 10, 13\}$ *and* $\{2, 7, 10, 13\}$*. This illustrates how the algorithm facilitates the linearly independent path selection.*

Finally, the graphs (b) to (f) in Figure 10.4 illustrate columns one to five of matrix $\mathbf{I}$ *in the final iteration. It is easy to see that it is impossible to find new paths providing a non-null dot product. If we focus on graphs (b) and (c) we can see that any path must contain links* $1, 2, 3$ *or* 4*. In addition, if it contains* 1 *or* 3*, it must contain links* 5 *or* 6*, and if it contains* 2 *or* 4 *must contain links* 7 *or* 8*. Note that under these constraints, it is impossible to satisfy the condition of a non-null dot product with those columns. A similar analysis can be made with graphs (d) to (f), concluding that no path linearly independent with the previously selected ones can be found.*

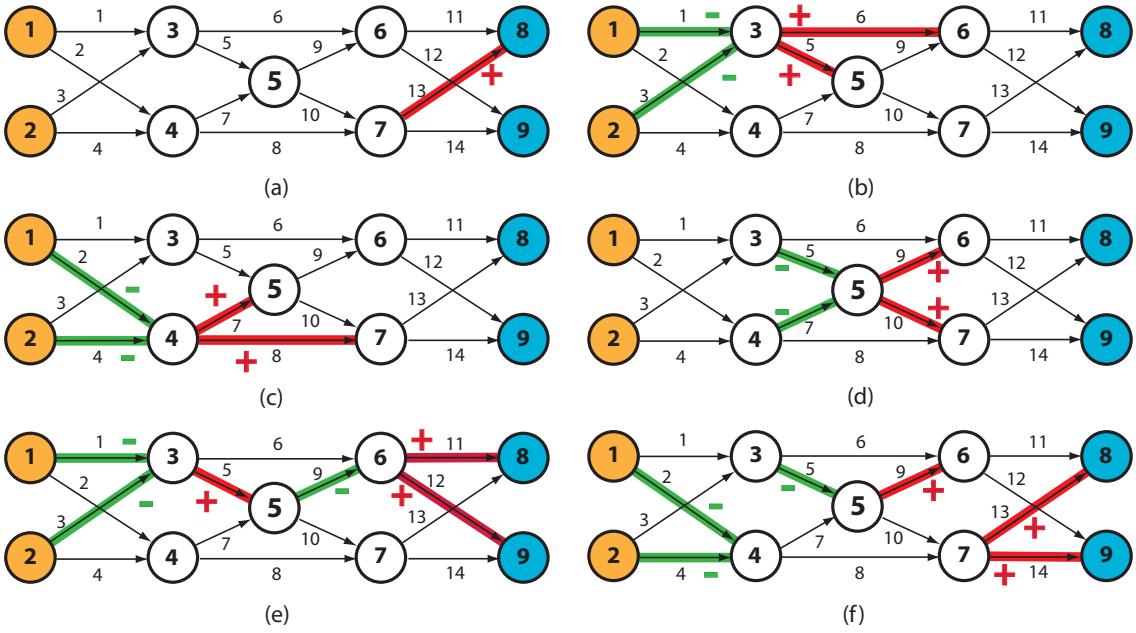


Figure 10.4: Illustration of how to select a new path vector in iteration 9 based on its pivot column (boldfaced).

10.7 Some recommendations

Based on the analysis of this chapter, we can make the following recommendations:

1. To determine the minimum number of counting sensors to be installed to infer the flows of all other non-equipped links in a traffic network we must look not for all paths but for linearly independent paths, considering that the number of linearly independent paths is at most $m - n$.

2. To obtain the transformation providing the dependent link flows in terms of the independent link flows we can use two alternatives: (a) the node-based approach or (b) the path-based approach. The first is more conservative but simpler, but only the second provides a minimum sensor solution if $r_W < m - n$ and most times it is smaller.

Iteration 1

$\mathbf{r}_1$	$\mathbf{v}_1^1$	$\mathbf{v}_2^1$	$\mathbf{v}_3^1$	$\mathbf{v}_4^1$	$\mathbf{v}_5^1$	$\mathbf{v}_6^1$	$\mathbf{v}_7^1$	$\mathbf{v}_8^1$	$\mathbf{v}_9^1$	$\mathbf{v}_{10}^1$	$\mathbf{v}_{11}^1$	$\mathbf{v}_{12}^1$	$\mathbf{v}_{13}^1$	$\mathbf{v}_{14}^1$
1	**1**	0	0	0	0	0	0	0	0	0	0	0	0	0
0	**0**	1	0	0	0	0	0	0	0	0	0	0	0	0
0	**0**	0	1	0	0	0	0	0	0	0	0	0	0	0
0	**0**	0	0	1	0	0	0	0	0	0	0	0	0	0
0	**0**	0	0	0	1	0	0	0	0	0	0	0	0	0
1	**0**	0	0	0	0	1	0	0	0	0	0	0	0	0
0	**0**	0	0	0	0	0	1	0	0	0	0	0	0	0
0	**0**	0	0	0	0	0	0	1	0	0	0	0	0	0
0	**0**	0	0	0	0	0	0	0	1	0	0	0	0	0
0	**0**	0	0	0	0	0	0	0	0	1	0	0	0	0
0	**0**	0	0	0	0	0	0	0	0	0	1	0	0	0
1	**0**	0	0	0	0	0	0	0	0	0	0	1	0	0
0	**0**	0	0	0	0	0	0	0	0	0	0	0	1	0
0	**0**	0	0	0	0	0	0	0	0	0	0	0	0	1
$\mathbf{t}^1$	**1**	0	0	0	0	1	0	0	0	0	0	1	0	0

Iteration 2

$\mathbf{r}_2$	$\mathbf{v}_1^2$	$\mathbf{v}_2^2$	$\mathbf{v}_3^2$	$\mathbf{v}_4^2$	$\mathbf{v}_5^2$	$\mathbf{v}_6^2$	$\mathbf{v}_7^2$	$\mathbf{v}_8^2$	$\mathbf{v}_9^2$	$\mathbf{v}_{10}^2$	$\mathbf{v}_{11}^2$	$\mathbf{v}_{12}^2$	$\mathbf{v}_{13}^2$
0	**0**	0	0	0	-1	0	0	0	0	0	-1	0	0
1	**1**	0	0	0	0	0	0	0	0	0	0	0	0
0	**0**	1	0	0	0	0	0	0	0	0	0	0	0
0	**0**	0	1	0	0	0	0	0	0	0	0	0	0
0	**0**	0	0	1	0	0	0	0	0	0	0	0	0
0	**0**	0	0	0	1	0	0	0	0	0	0	0	0
0	**0**	0	0	0	0	1	0	0	0	0	0	0	0
0	**0**	0	0	0	0	0	1	0	0	0	0	0	0
0	**0**	0	0	0	0	0	0	1	0	0	0	0	0
1	**0**	0	0	0	0	0	0	0	1	0	0	0	0
0	**0**	0	0	0	0	0	0	0	0	1	0	0	0
0	**0**	0	0	0	0	0	0	0	0	0	1	0	0
0	**0**	0	0	0	0	0	0	0	0	0	0	1	0
0	**0**	0	0	0	0	0	0	0	0	0	0	0	1
1	**0**	0	0	0	0	0	0	0	0	0	0	0	1
$\mathbf{t}^2$	**1**	0	0	0	0	0	1	0	0	0	0	0	1

Iteration 3

$\mathbf{r}_3$	$\mathbf{v}_1^3$	$\mathbf{v}_2^3$	$\mathbf{v}_3^3$	$\mathbf{v}_4^3$	$\mathbf{v}_5^3$	$\mathbf{v}_6^3$	$\mathbf{v}_7^3$	$\mathbf{v}_8^3$	$\mathbf{v}_9^3$	$\mathbf{v}_{10}^3$	$\mathbf{v}_{11}^3$	$\mathbf{v}_{12}^3$
0	**0**	0	0	-1	0	0	0	0	0	-1	0	0
0	**0**	0	0	0	0	-1	0	0	0	0	0	-1
1	**1**	0	0	0	0	0	0	0	0	0	0	0
0	**0**	1	0	0	0	0	0	0	0	0	0	0
0	**0**	0	1	0	0	0	0	0	0	0	0	0
1	**0**	0	0	1	0	0	0	0	0	0	0	0
0	**0**	0	0	0	1	0	0	0	0	0	0	0
0	**0**	0	0	0	0	1	0	0	0	0	0	0
0	**0**	0	0	0	0	0	1	0	0	0	0	0
0	**0**	0	0	0	0	0	0	1	0	0	0	0
0	**0**	0	0	0	0	0	0	0	1	0	0	0
0	**0**	0	0	0	0	0	0	0	0	1	0	0
1	**0**	0	0	0	0	0	0	0	0	0	1	0
0	**0**	0	0	0	0	0	0	0	0	0	0	1
$\mathbf{t}^3$	**1**	0	0	1	0	0	0	0	0	1	0	0

Iteration 4

$\mathbf{r}_4$	$\mathbf{v}_1^4$	$\mathbf{v}_2^4$	$\mathbf{v}_3^4$	$\mathbf{v}_4^4$	$\mathbf{v}_5^4$	$\mathbf{v}_6^4$	$\mathbf{v}_7^4$	$\mathbf{v}_8^4$	$\mathbf{v}_9^4$	$\mathbf{v}_{10}^4$	$\mathbf{v}_{11}^4$
0	**0**	0	-1	0	0	0	0	0	-1	0	0
0	**0**	0	0	0	-1	0	0	0	0	0	-1
0	**0**	0	-1	0	0	0	0	0	-1	0	0
1	**1**	0	0	0	0	0	0	0	0	0	0
0	**0**	1	0	0	0	0	0	0	0	0	0
0	**0**	0	1	0	0	0	0	0	0	0	0
0	**0**	0	0	1	0	0	0	0	0	0	0
1	**0**	0	0	0	1	0	0	0	0	0	0
0	**0**	0	0	0	0	1	0	0	0	0	0
0	**0**	0	0	0	0	0	1	0	0	0	0
0	**0**	0	0	0	0	0	0	1	0	0	0
0	**0**	0	0	0	0	0	0	0	1	0	0
0	**0**	0	0	0	0	0	0	0	0	1	0
1	**0**	0	0	0	0	0	0	0	0	0	1
$\mathbf{t}^4$	**1**	0	0	0	1	0	0	0	0	0	1

Iteration 5

$\mathbf{r}_5$	$\mathbf{v}_1^5$	$\mathbf{v}_2^5$	$\mathbf{v}_3^5$	$\mathbf{v}_4^5$	$\mathbf{v}_5^5$	$\mathbf{v}_6^5$	$\mathbf{v}_7^5$	$\mathbf{v}_8^5$	$\mathbf{v}_9^5$	$\mathbf{v}_{10}^5$
1	**0**	-1	0	0	0	0	0	-1	0	0
0	**0**	0	0	-1	0	0	0	0	0	-1
0	**0**	-1	0	0	0	0	0	-1	0	0
0	**0**	0	0	-1	0	0	0	0	0	-1
1	**1**	0	0	0	0	0	0	0	0	0
0	**0**	1	0	0	0	0	0	0	0	0
0	**0**	0	1	0	0	0	0	0	0	0
0	**0**	0	0	1	0	0	0	0	0	0
0	**0**	0	0	0	1	0	0	0	0	0
0	**0**	0	0	0	0	1	0	0	0	0
1	**0**	0	0	0	0	0	1	0	0	0
0	**0**	0	0	0	0	0	0	1	0	0
0	**0**	0	0	0	0	0	0	0	1	0
1	**0**	0	0	0	0	0	0	0	0	1
$\mathbf{t}^5$	**1**	-1	0	0	0	1	0	-1	0	1

Iteration 6

$\mathbf{r}_6$	$\mathbf{v}_1^6$	$\mathbf{v}_2^6$	$\mathbf{v}_3^6$	$\mathbf{v}_4^6$	$\mathbf{v}_5^6$	$\mathbf{v}_6^6$	$\mathbf{v}_7^6$	$\mathbf{v}_8^6$	$\mathbf{v}_9^6$
0	-1	**0**	0	0	0	-1	0	0	
1	0	**0**	-1	0	0	0	0	0	-1
0	-1	**0**	0	0	0	-1	0	0	
0	0	**0**	-1	0	0	0	0	0	-1
0	1	**0**	0	0	-1	0	1	0	-1
0	1	**0**	0	0	0	0	0	0	0
1	0	**1**	0	0	0	0	0	0	0
0	0	**0**	1	0	0	0	0	0	0
0	0	**0**	0	1	0	0	0	0	0
1	0	**0**	0	0	1	0	0	0	0
0	0	**0**	0	0	0	1	0	0	0
0	0	**0**	0	0	0	0	1	0	0
0	0	**0**	0	0	0	0	0	1	0
1	0	**0**	0	0	0	0	0	0	1
$\mathbf{t}^6$	0	**1**	-1	0	1	0	0	0	0

Iteration 7

$\mathbf{r}_7$	$\mathbf{v}_1^7$	$\mathbf{v}_2^7$	$\mathbf{v}_3^7$	$\mathbf{v}_4^7$	$\mathbf{v}_5^7$	$\mathbf{v}_6^7$	$\mathbf{v}_7^7$	$\mathbf{v}_8^7$
1	-1	0	**0**	0	0	-1	0	0
0	0	-1	**0**	0	0	0	0	-1
0	-1	0	**0**	0	0	-1	0	0
0	0	-1	**0**	0	0	0	0	-1
1	1	0	**0**	-1	0	1	0	-1
0	1	0	**0**	0	0	0	0	0
0	0	1	**0**	-1	0	0	0	0
0	0	1	**0**	0	0	0	0	0
1	0	0	**1**	0	0	0	0	0
0	0	0	**0**	1	0	0	0	0
0	0	0	**0**	0	1	0	0	0
1	0	0	**0**	0	0	1	0	0
0	0	0	**0**	0	0	0	1	0
0	0	0	**0**	0	0	0	0	1
$\mathbf{t}^7$	0	0	**1**	-1	0	1	0	-1

Iteration 8

$\mathbf{r}_8$	$\mathbf{v}_1^8$	$\mathbf{v}_2^8$	$\mathbf{v}_3^8$	$\mathbf{v}_4^8$	$\mathbf{v}_5^8$	$\mathbf{v}_6^8$	$\mathbf{v}_7^8$
1	-1	0	0	**0**	-1	0	0
0	0	-1	0	**0**	0	0	-1
0	-1	0	0	**0**	-1	0	0
0	0	-1	0	**0**	0	0	-1
0	1	0	-1	**0**	1	0	-1
1	1	0	0	**0**	0	0	0
0	0	1	-1	**0**	0	0	0
0	0	1	0	**0**	0	0	0
0	0	1	0	**0**	0	0	0
0	0	0	1	**0**	-1	0	1
0	0	0	1	**0**	0	0	0
0	0	0	0	**1**	0	0	0
0	0	0	0	**0**	1	0	0
0	0	0	0	**0**	0	1	0
0	0	0	0	**0**	0	0	1
$\mathbf{t}^8$	0	0	0	**1**	-1	0	0

Iteration 9

$\mathbf{r}_9$	$\mathbf{v}_1^9$	$\mathbf{v}_2^9$	$\mathbf{v}_3^9$	$\mathbf{v}_4^9$	$\mathbf{v}_5^9$	$\mathbf{v}_6^9$
0	-1	0	0	-1	**0**	0
1	0	-1	0	0	**0**	-1
0	-1	0	0	-1	**0**	0
0	0	-1	0	0	**0**	-1
0	1	0	-1	1	**0**	-1
0	1	0	0	0	**0**	0
0	0	1	-1	0	**0**	0
1	0	1	0	0	**0**	0
0	0	0	1	-1	**0**	1
0	0	0	1	0	**0**	0
0	0	0	0	1	**0**	0
0	0	0	0	1	**0**	0
1	0	0	0	0	**1**	0
0	0	0	0	0	**1**	-1

Final

$\mathbf{v}_1$	$\mathbf{v}_2$	$\mathbf{v}_3$	$\mathbf{v}_4$	$\mathbf{v}_5$
-1	0	0	-1	0
0	-1	0	0	-1
-1	0	0	-1	0
0	-1	0	0	-1
1	0	-1	1	-1
1	0	0	0	0
0	1	-1	0	0
0	1	0	0	0
0	0	1	-1	1
0	0	1	0	0
0	0	0	1	0
0	0	0	1	0
0	0	0	0	1
0	0	0	0	1

Table 10.2: Illustration of the different steps of the algorithm showing the different **I** matrices, the dot products and the pivot columns (boldfaced).

Appendix

A Notation

α	auxiliary parameter.
$\boldsymbol{\rho}$	column vector of real numbers.
$\mathbf{A}$	modified node-link incidence matrix.
$\mathbf{B}$	regular matrix of dimension $n \times n$.
$\mathbf{C}$	matrix of dimensions $n' \times (m - n')$.
$\mathbf{D}$	matrix of dimensions $n' \times (r - m + n')$.
$\mathbf{E}$	invertible matrix of dimension $(m - n') \times (m - n')$.
E^n	Euclidean space of dimension n.
$\mathbf{F}$	matrix of dimensions $(m - n') \times (r - m + n')$.
$\mathbf{K}$	generic matrix of column vectors.
$\mathbf{H}$	matrix of cone generators.
$\mathbf{H}_0$	submatrix of matrix $\mathbf{H}$.
$\mathbf{h}_i$	cone generator vectors.
$\mathcal{I}$	identity matrix.
$\mathbf{I}$	matrix used in the algorithm.
$\mathbf{I}_k^*$	transformed column of matrix $\mathbf{I}$.
m	number of links.
$\mathbf{M}$	generic matrix of column vectors.
$\mathbf{m}_i$	columns of matrix $\mathbf{M}$.
$\mathbf{M}_\pi$	cone generated by the columns of matrix $\mathbf{M}$.
n	number of non-centroid nodes.
$\mathbf{N}$	matrix of dimensions $n \times (m - n)$.
n'	number of non-basis link flows.
$\mathbf{p}$	path vector.
r	number of paths.
$\mathbf{r}$	column matrix of path flows.
$\mathbb{R}$	set of real numbers.
$\mathbf{r}_0$	submatrix of matrix $\mathbf{r}$.
$\mathbf{r_1}$	partition of route flows.
$\mathbf{r_2}$	partition of route flows.
r_h	rank of matrix $\mathbf{H}_0$.
r_i	flow on path i.
r^*	number of linearly independent columns of matrix $\mathbf{H}_0$.
$\mathbf{t}$	vector of dot products.
t_p	component of vector $\mathbf{t}$.
$\mathbf{u}$	path vector.

$\mathbf{v}$	path vector.
$\mathbf{v}_i^j$	column i of iteration j of matrix $\mathbf{I}$ in the algorithm.
$\mathbf{w}$	vector of link flows.
$\mathbf{w}_B$	set of non-basis link flows.
$\mathbf{w}'_B$	set of non-basis link flows.
$\mathbf{w}_N$	set of basis link flows.
$\mathbf{w}'_N$	set of basis link flows.

Chapter 11

A Model for Continuous Dynamic Network Loading Problem with Different Overtaking Class Users

Contents

11.1 Introduction

Most traffic models used in practice are static in nature (see Chapter 3), that is, they are concerned on total link, OD-pair or path flows during given periods of time (hour, day, week, etc.).

However, for solving most real traffic problems total flows are not sufficient so that knowledge of the time evolution of traffic flows is required. Thus, dynamic traffic models are needed to reproduce the real traffic flow behavior. These models (see, for example, Zhou and Mahmassani (2006)) aim at predicting how the traffic intensities evolve with time, given the traffic intensities at the origins. These problems present two components: (i) the assignment strategy, i.e., how the travelers select their nodes, destinations and

routes (see, for example, Janson (1991), Smith (1979), Friesz et al. (1993), Lo and Szeto (2002), Ban et al. (2008) or Ran et al. (1996)) and (ii) the network loading, i.e., how traffic propagates on a transport network and hence governs the network performance in terms of travel time (see, for example, Hopf (1950) or Friesz et al. (2011)).

The network loading is a process that is used to calculate how flows distribute over a network with a given route inflow profile for each origin-destination pair. In general, we can distinguish between two categories: the simulation-based and the analytical-based approaches. The simulation-based approach emphasizes each individual driver's behavior (see Chandler et al. (1958), Gazis et al. (1961) or Wagner et al. (1996)) and thus it implies a great computational effort. The analytical-based approach concerns the average driver's behavior, and is essentially macroscopic. In this group we can mention the following models: the hydrodynamic, the Merchant-Nemhauser, the cell transmission and the point and physical queue models (see Chapter 6 for a review on these models).

11.1.1 Overtaking

In the traffic assignment problem travelers compete for space in a network and choose paths depending on congestion. This problem has been treated in the existing literature for many years assuming homogeneous users in terms of speed, travel time, etc. However, recently, the travel time reliability problem has given place to heterogeneous behaviors, in which different users perceive punctual arrivals from a different perspective. This means that we can classify users in different class users who choose different routes.

It has been recognized in the existing literature (see Section 3.4) that travelers' decisions (route choices) are largely influenced by travel time variability and reliability, that is, they are the two main criteria for route choice.

Another interesting case of heterogeneity that we consider in this paper arises when we consider overtaking and include classes of users which are prone to overtake, classes of users who avoid overtaking and intermediate classes (see Svensson (1978), Buric and Janovsky (2007) and Castillo et al. (submitted)).

Overtaking can only take place when there is a large enough gap in the oncoming traffic. However, overtaking is not possible under high congestion. Users choose routes that best satisfy their expectations (overtaking possibilities). Consequently, traffic models must reflect real situations and, in particular, the impossibility of overtaking under congestion.

Though most of the existing traffic assignment methods consider that the link travel time for all users is the same and assume the FIFO rule (see Daganzo (1994), Nie and Zhang (2005) or Castillo et al. (2012)), we introduce the possibility of overtaking. In Chapter 8, we have also included the possibility of overtaking in the static traffic assignment, analyzed the convenient properties of link travel time functions from the point of view of overtaking and derived Wardrop type models for traffic assessment with and without traffic enumeration.

The main original contribution of this chapter consists of providing for the first time a dynamic loading network model that includes different classes of users from the point of view of overtaking. Furthermore, the model used can be considered as a physical-queue model and, therefore, it takes into account the queue spill back and junction blockage effects. To illustrate, some examples of applications are given.

The chapter is organized as follows. In Section 11.2, some existing link travel time functions are discussed and a new function to consider the particular case of overtaking is selected. In Section 11.3 the proposed model is described by enumerating all used assumptions. In Section 11.4 the model assumptions are discussed in some detail. Finally, in Section 11.5 the proposed methods and methodology is illustrated by its application to a simple network and the Sioux-Falls and the Cuenca network.

11.2 Link travel time functions

In most traffic assignment models, the effect of road capacity on travel times is specified by means of the so called volume-delay or link travel time functions $t(v/c)$ which express the travel time of a link as a function of the traffic volume v and its related link capacity c, as it has been already explained (see Appendix A in Chapter 3 for a review on some link performance functions).

Most existing traffic models consider that there is no overtaking, that is, that the FIFO rule holds. In this paper we assume that we have different classes of users who have different mean velocities and thus, we permit vehicle overtaking.

Since congestion affects overtaking, this effect must be considered in the link travel time functions of the different users. Overtaking is possible and frequent under free flow conditions but becomes difficult or impossible under high congestion. However, the associated difficulties depend on the type of vehicle. For example, motorcycles have less overtaking difficulties than cars, and cars less than trucks. All these features must be considered in the mathematical models. From an overtaking point of view, we can consider different classes for bicycles, motorcycles, cars and trucks, but we can also consider different classes for each of these types of vehicles due to the fact that not all users in the same class behave in the same manner. However, a high congestion usually produces no differences among these classes.

In this chapter we will use the travel time function family proposed in Chapter 8. This volume-delay functions permit producing different travel times for the different classes for mild congestion, but the same asymptotic behavior when a high congestion is present. The proposed link travel time function for the α class users is given by:

$$w^{\alpha}(x) = t(x; t_0^0, \beta^0, \gamma^0)F(x) + t(x; t_0^{\alpha}, \beta^{\alpha}, \gamma^{\alpha})(1 - F(x)), \tag{11.1}$$

where $t(x; t_0^{\alpha}, \beta^{\alpha}, \gamma^{\alpha})$ is a link travel time function specific for each α class that reproduces the class user behavior under mild congestion, $t(x; t_0^0, \beta^0, \gamma^0)$ is a common link travel

time function for all classes that represents the common asymptotic behavior under high congestion of all users, $F(x)$ is a cumulative distribution function (cdf), $t_0^0, t_0^\alpha, \beta^0, \beta^\alpha, \gamma^0 > 2, \gamma^\alpha > 2$ are constants, and we make the following assumptions

$$t_0^0 > t_0^\alpha > 0; \quad \beta^0 \geq \beta^\alpha > 0; \quad \gamma^0 \geq \gamma^\alpha > 0. \tag{11.2}$$

The well-known BPR function has been used in this study because of its simplicity and its wide spread use. However, as noted by Spiess (1990), it has some inherent drawbacks. In particular, the BPR function can be improved under congested traffic flow conditions, i.e., when the ratio v/c is higher than one. Spiess (1990) proposed the following conical link travel time function to overcome some of the BPR shortcomings:

$$f_{Spiess}(x; \rho, \beta) = 2 + \sqrt{\rho^2(1-x)^2 + \beta^2} - \rho(1-x) - \beta, \tag{11.3}$$

where β is given as

$$\beta = \frac{2\rho - 1}{2\rho - 2}$$

and ρ is any number larger than 1.

The versatility of the proposed function in (11.1) permits working with any desired link travel time function and does not limit to BPR functions. Therefore, the link travel time function can be chosen depending on the nature of the problem under study. In the remaining of the paper the BPR function will be utilized with the exception of one example in which the Spiess function has been used (section 11.5.2).

The rationale behind (11.1) is that for each congestion ratio $x = v/c$, the travel time is a linear convex combination of two travel time functions but the weights change with x from 0 to 1 as x increases. Using the same t_0^0, β^0 and γ^0 values and different t_0^α, β^α and γ^α parameters, we obtain travel time functions that practically coincide for high congestion levels (in fact they converge to the BPR_0 function $t(x; t_0^0, \beta^0, \gamma^0)$).

To illustrate the different concepts and methods introduced in this paper, we consider a traffic network $(\mathcal{N}, \mathcal{A})$ where $\mathcal{A}$ is a set of links and $\mathcal{N}$ is a set of nodes, from which one can distinguish two subsets $\mathcal{O}$ and $\mathcal{D}$, of origins and destinations, respectively, and we use the simple illustrative example shown in Figure 11.1. It consists of 6 nodes, 9 links and 9 paths as indicated in Table 11.1, where its three columns contain the OD-pairs defined by their origin and destination nodes, the path number and the path links, respectively.

In the example, we have used three class users: cars, trucks and motorbikes. Figure 11.2 shows the link travel time functions for three selected links: 6, 7 and 8. The examples have been obtained for normal $F(x)$. Note that cars approach the truck congestion asymptote before than motorbikes[1].

[1]For illustrative purposes, we have assumed 30% of truck with respect to the cars. This high percentage has been selected in order to be able to show the effect of congestion and the class differences in the figures.

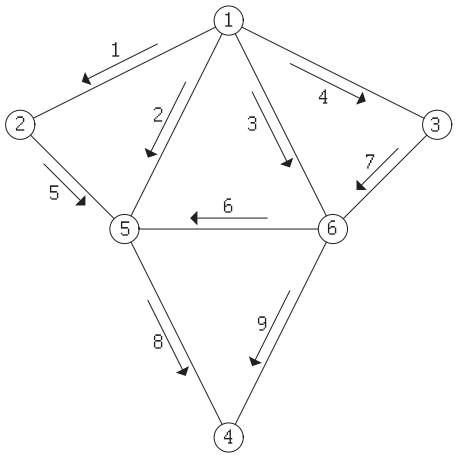


Figure 11.1: The elementary example network used for illustrative purposes, showing the nodes and links.

OD	Path number (r)	Arcs
1-4	1	1 5 8
1-4	2	2 8
1-4	3	3 9
1-4	4	3 6 8
1-4	5	4 7 9
1-4	6	4 7 6 8
2-4	7	5 8
3-4	8	7 6 8
3-4	9	7 9

Table 11.1: Set of 3 OD–pairs and 9 paths (defined by its end nodes and links) in the elementary example.

11.3 Proposed model

The proposed model is based on the following assumptions:

1. α **classes.** We consider different α classes of users with a different overtaking behavior.

2. **Path origin flow intensity functions.** Since we face a dynamic loading problem, we assume that routes have been already selected by users, that is, we assume that the route flows at their origins are given. In our model we propose path origin flow

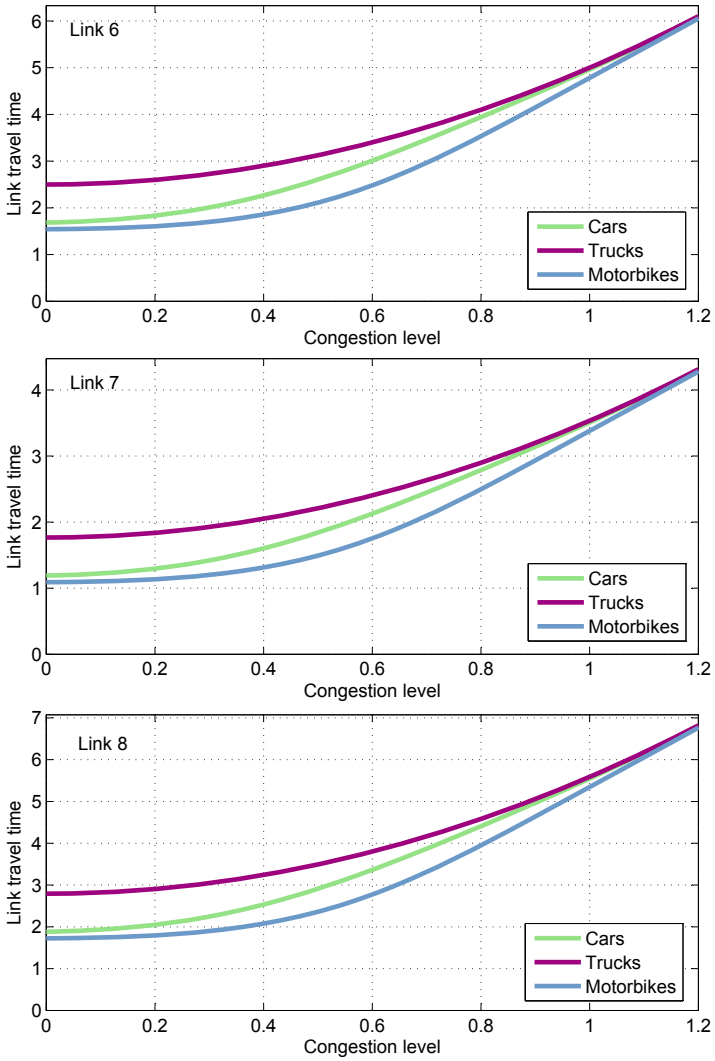


Figure 11.2: Some illustrative examples of the proposed link travel time function based on BPR functions used in our example to follow.

intensity functions of the form

$$h_r^{\alpha}(t) = \sum_{l=1}^{n_r} h_{rl}^{\alpha} q^{\alpha}(t; \bar{\eta}_{rl}^{\alpha}), \tag{11.4}$$

where $h_r^{\alpha}(t)$ is the inflow traffic intensity (veh/hour) associated with path r and class user α at its origin and time t, n_r is the number of function components, $q^{\alpha}(t; \bar{\eta}_{rl}^{\alpha})$

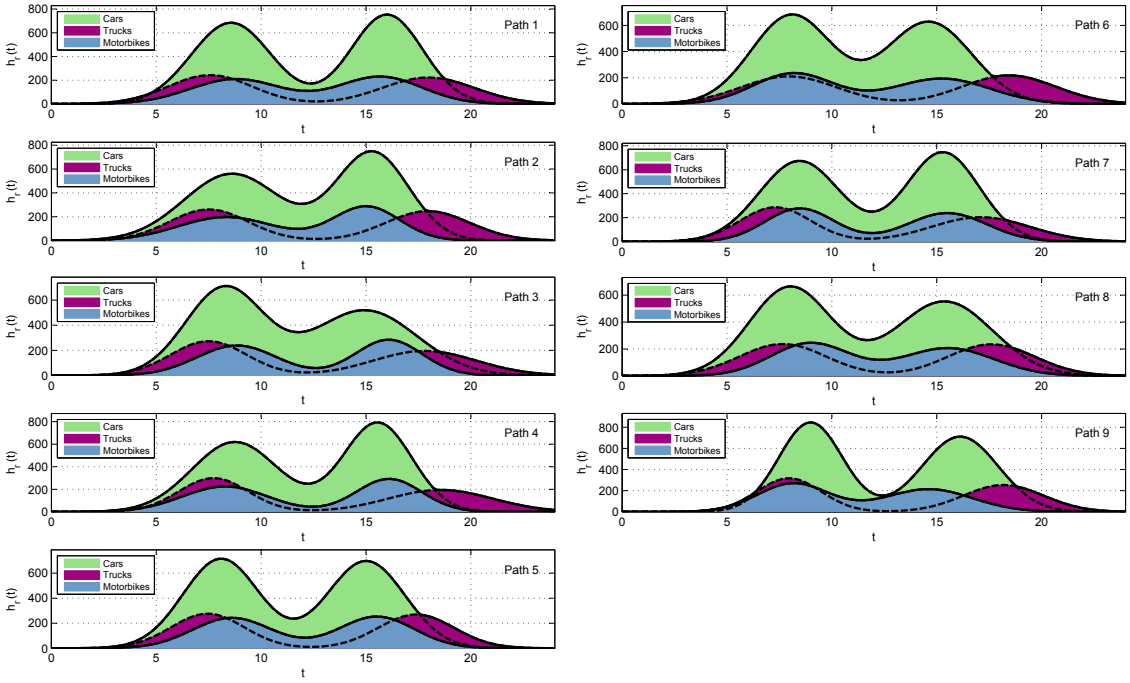


Figure 11.3: Illustrative example. Path flow intensities at the path origins as a function of time for the illustrative example, showing that trucks start trips earlier than cars and motorbikes in the morning, and later in the afternoon.

is a parametric family of probability density functions[2], with vector parameter $\bar{\eta}_{rl}^{\alpha}$, and $h_{rl}^{\alpha} > 0$ (component total flow) are the coefficients of the linear combination.

A simple example results if two normal densities are considered for each path (see Figure 11.3).

We define the cumulative inflow rate $H_r^{\alpha}(t)$ of class α at the origin of path r at time t as

$$\begin{aligned} H_r^{\alpha}(t) &= \int_0^t h_r^{\alpha}(t)dt = \sum_{l=1}^{n_r} h_{rl}^{\alpha} \int_0^t q^{\alpha}(t;\bar{\eta}_{rl}^{\alpha})dt \\ &= \sum_{l=1}^{n_r} h_{rl}^{\alpha} Q^{\alpha}(t;\bar{\eta}_{rl}^{\alpha}), \end{aligned} \tag{11.5}$$

where $Q^{\alpha}(t;\bar{\eta}_{rl}^{\alpha})$ are cumulative distribution functions (cdf).

[2]Note that we have used probability density functions in order to have a total flow h_{rl}^{α} associated with each component.

3. **FIFO rule.** Users of the same class α satisfy the FIFO rule, that is, there is no overtaking among users of the same class.

4. **Congestion effect.** We consider the congestion effect at the adequate time and location, i.e. our model evaluates the congestion effect taking into account the interaction of flows of all paths and their coincidence at different times and locations. However, since we have several class users, the total link congestion ratio $s_{ij}(t)$ is obtained by adding the contributions to total congestion of all class users, that is,

$$s_v(t) = \sum_{\alpha} \frac{x_v^{\alpha}(t)}{x_{ij}^{\alpha max}}, \tag{11.6}$$

where $x_{ij}^{\alpha}(t)$ is the number of vehicles of class α on link ℓ_{ij} at time t and $x_{ij}^{\alpha max}$ is the number of vehicles on link ℓ_{ij} leading to a travel time $t_{0ij}^{\alpha}(1+\beta_{ij})$ and, as indicated, $s_{ij}(t)$ is a sum of dimensionless ratios measuring the degree of congestion for all class users at link ℓ_{ij}.

The link travel time function for the α class of users is assumed to be of the form:

$$D_{ij}^{\alpha}(t) = w^{\alpha}(s_v(t)) + t_{0ij}^{\alpha}\delta_{ij}^{\alpha} \max_{\ell_{ji'} \in \mathcal{S}(\ell_{ij})} (s_{ji'}(t))^{\gamma}, \tag{11.7}$$

where $w^{\alpha}(\cdot)$ is given in (11.1), δ_{ij}^{α} is a parameter used to take into account the congestion ahead of the link ℓ_{ij} being considered, t_{0ij}^{α} and γ are the free travel time, and saturation parameters, respectively, associated with link ℓ_{ij} and class α, $\mathcal{S}(\ell_{ij})$ is the set of all links downstream link ℓ_{ij} in all its routes. All these parameters must be calibrated based on real data.

We note that $\delta_{ij}^{\alpha} << \beta_{ij}^{\alpha}$ (see (11.1)) because the influence of the degrees of saturation of downstream links on the ℓ_{ij} link travel time must be smaller than the influence of its own degree of saturation. Note that the term $\delta_{ij}^{\alpha} \max_{\ell_{ji'} \in \mathcal{S}(\ell_{ij})} (s_{ji'}(t))^{\gamma}$ measures the saturation effect in link ℓ_{ij} due to downstream congestion for class α users.

The number of vehicles on link ℓ_{ij} at time t that appears in (11.6) and used in (11.7) can be calculated by:

$$x_{ij}^{\alpha}(t) = E_{ij}^{\alpha}(t_{ij}^{\alpha out}(t)) - E_{ij}^{\alpha}(t) \quad \forall \ell_{ij}, \forall t, \alpha, \tag{11.8}$$

where $t_{ij}^{\alpha out}(t)$ is the link exit time of a user of class α who enters link ℓ_{ij} at time t, and the accumulated number of vehicles that has left the link ℓ_{ij} at time t is given by

$$E_{ij}^{\alpha}(t) = \sum_{r} H_r^{\alpha}\left(\theta_{ij}^{r\alpha}(t)\right), \tag{11.9}$$

where $\theta_{ij}^{r\alpha}(t)$ is the departure time from the origin of path r of a user of class α who exits link ℓ_{ij} at time t.

5. **Continuous model.** We deal with a continuous model for real traffic networks, which seems more convenient than a discretized version of the real continuous problem. To this end, we approximate these functions by monotone cubic splines whose number of parameters remains finite and small.

6. **Path link exit time functions.** Path link exit time functions $\theta_{ij}^{r\alpha}(t)$ are evaluated based on link exit-entry time functions $\tau_{ij}^{\alpha}(t)$.

7. **Approximation of the $\tau_{ij}^{\alpha}(t)$ function by monotone cubic splines.** We consider a set of discrete times $t_k; k = 1, 2, \ldots, m$, which are used to approximate the continuous link entry-exit time functions $\tau_{ij}^{\alpha}(t)$ by monotone cubic splines. There are several reasons to use monotone spline functions to approximate the link travel time function:

   (a) To obtain a continuous link travel time approximation instead of a discretized function.

   (b) To guarantee the FIFO rule for users of the same α class. The monotone spline functions guarantee that if the basic points reveal an increasing trend, the whole spline is increasing, that is, the FIFO rule holds for all interpolated points (see the Fritsch and Carlson reference Fritsch and Carlson (1980) or the Matlab manual for pchip function).

8. **Conservation law.** We consider the conservation law for each class α users as:

$$\int_{t_1}^{t_2} g_{ij}^{r\alpha}(t)dt = \int_{\theta_{ij}^{r\alpha}(t_1)}^{\theta_{ij}^{r\alpha}(t_2)} h_r^{\alpha}(\theta_{ij}^{r\alpha}(t))d\theta_{ij}^{r\alpha}(t), \tag{11.10}$$

   where $g_{ij}^{r\alpha}(t)$ is the outflow rate of link ℓ_{ij} and class α at time t due to path r, which implies

$$g_{ij}^{r\alpha}(t) = h_r^{\alpha}(\theta_{ij}^{r\alpha}(t))\frac{d\theta_{ij}^{r\alpha}(t)}{dt}, \tag{11.11}$$

9. **Arc and node flow intensities.** The traffic flow intensity of a user of class α at the exit of link ℓ_{ij} is given by adding the flow intensities of the corresponding paths (flow conservation constraints), that is,

$$g_{ij}^{\alpha}(t) = \sum_{r\in\mathcal{R}_{ij}} g_{ij}^{r\alpha}(t) = \sum_{r\in\mathcal{R}_{ij}} h_r^{\alpha}\left(\theta_{ij}^{r\alpha}(t)\right)\frac{d\theta_{ij}^{r\alpha}(t)}{dt}, \tag{11.12}$$

   where $\mathcal{R}_{ij}$ is the set of paths containing link ℓ_{ij}. Similarly, the node flow intensities excluding origin path intensities can be calculated by adding the flow intensities of

the corresponding paths (flow conservation constraints), that is, by means of the formula

$$r_i^{\alpha}(t) = \sum_{\ell_{ij} \in \mathcal{A}(i)} g_{ij}^{\alpha}(t), \tag{11.13}$$

where $r_i^{\alpha}(t)$ is the flow intensity of a class α user at node i and time t and $\mathcal{A}(i)$ is the set of links entering node i.

10. **The link physical-queues.** The link exit time $t_{ijk}^{\alpha out}$ associated with an entering time t_k is obtained as follows:

$$t_{ijk}^{\alpha out} = \max\left\{t_k + D_{ij}^{\alpha}(t_k), t_{ak-1}^{\alpha out} + Q_{ijk}^{\alpha}\right\}, \tag{11.14}$$

where Q_{ijk}^{α}, is the physical-queue dissipation time at link ℓ_{ij} for class α at time t added in order to include the traffic jam effect, as follows,

$$Q_{ijk}^{\alpha} = \left[E_{ij}^{\alpha}(t_{ijk}^{\alpha out}) - E_v^{\alpha}(t_{ijk-1}^{\alpha out})\right] \frac{t_{0ij}^{\alpha}(1+\beta_{ij}^{\alpha})}{k_{cong}^{\alpha} x_{ij}^{\alpha max}}, \tag{11.15}$$

where k_{cong}^{α} is a speed dissipation physical-queue factor for class α that takes values smaller than one.

The rationale behind the k_{cong}^{α} factor is as follows. According to (11.7) and neglecting the downstream effect, if link ℓ_{ij} holds $x_{ij}^{\alpha max}$ users, the link travel time becomes $t_{0ij}^{\alpha}(1+\beta_{ij}^{\alpha})$, which implies a link evacuation speed of

$$u_{ij}^{\alpha evac} = \frac{x_{ij}^{\alpha max}}{t_{0ij}^{\alpha}(1+\beta_{ij}^{\alpha})}, \tag{11.16}$$

which corresponds to a congestion level associated with a certain service level previous to queue generation. Since the queue condition can be assumed worse than this service level, the queue dissipation speed will be $k_{cong}^{\alpha} u_{ij}^{\alpha evac}$, where $k_{cong}^{\alpha} < 1$. Consequently, the time required to dissipate a queue of $\left[E_{ij}^{\alpha}(t_{ijk}^{\alpha out}) - E_{ij}^{\alpha}(t_{ijk-1}^{\alpha out})\right]$ users becomes (11.15).

11. **Iteration scheme.** We use an iterative scheme. At a given iteration, we first determine the link travel times associated with a carefully selected discrete set of users based on a previous iteration cubic-spline approximation of link travel time functions and later we update the monotone cubic splines, fitting them to the updated travel times. The process is iterated until convergence.

It is important to remark that the proposed model is an extension of the Dynamic Network Loading model proposed by Castillo et al. (2012) and explained in detail in Chapter 7 in Nogal (2011). For that reason, some details may be skipped, but can be found in those published works.

11.4 A more detailed discussion of model assumptions

Next, the above assumptions are discussed in more detail. Each subsection correspond to one assumption.

α classes.

This is the key difference between the proposed model and previous existing models, that allows results to be more realistic. The introduction of overtaking classes is made by means of the overtaking travel time function in (11.1). To illustrate we assume that in our example we have three classes: motorbikes, cars and trucks.

Path origin flow intensity functions.

An important item of information for dynamic models of traffic flow consists of a function that gives the corresponding path flow intensities at their origin as a function of time, i.e., the time evolution of the users entering the network. As indicated, these functions are assumed to be known.

Since a function has infinite degrees of freedom and then arbitrary functions cannot be dealt with, a good way of defining path flow functions is by means of parametric families of functions that are defined by a finite number of parameters. In this way, a small set of real numbers provides the required information for each path function. In this paper we propose path origin flow intensity functions of the form (11.4).

In summary, route intensity functions are assumed linear combinations of a basic set of probability density functions (pdf).

Example 15 (Normal densities) *Since in most cases the path flows normally present two relative maxima and decay at night hours, we can reproduce with sufficient precision the traffic intensity of each path by a linear combination of normal densities (flow waves), which total area is the total path flow h_{rl}^{α}, that is:*

$$h_r^{\alpha}(t) = \sum_{l=1}^{n_r} h_{rl}^{\alpha} f_{N(\mu_{rl}^{\alpha}, \sigma_{rl}^{\alpha})}(t), \tag{11.17}$$

where n_r is the number of components (normally two), $h_r^{\alpha}(t)$ is the traffic intensity (veh/hour) associated with path r and class α at time t, $f_{N(\mu_{rl}^{\alpha}, \sigma_{rl}^{\alpha})}(t)$ is the probability density function (pdf) of the normal distribution associated with class α, h_{rl}^{α} is the daily flow associated with the normal density $N(\mu_{rl}^{\alpha}, \sigma_{rl}^{\alpha})$, and μ_{rl}^{α} and σ_{rl}^{α} are the time associated with the mean and the corresponding standard deviation, that measures the traffic spread of flow wave l of path r, respectively.

The time dependent path flow functions $h_r^{\alpha}(t)$ are basic to build the functions giving the evolution of flows in all links and nodes of the network.

For illustrative purposes, in Table 11.2 and Figure 11.3 we present one example of the assumed path flow functions for the illustrative example in Figure 11.1. Note that cars and motorbikes exhibit two relative maxima around 9.00 *and* 16.00 *hours, and that trucks exhibit maxima at* 8.00 *and* 18.00 *hours, which are assumed to be the corresponding peak hours, though different shapes modeled by using different standard deviation parameters* σ_{rl}^{α}*. Note that the number of car volumes are larger than the motorbike volumes, and these larger than the track volumes.*

The values of μ_{rl}^{α}*,* σ_{rl}^{α} *and total volumes have been simulated randomly.*

FIFO rule.

We assume that overtaking is not permitted among users of the same class, but it is among users of different classes. This provides a flexibility not available in other existing models.

Congestion effect.

To consider the congestion effect, we must take into account that we have several class users traveling at different speeds and that overtaking among different classes is permitted. We analyze the congestion in order to determine how the link volume varies with time. To this end, we define $D_{ij}^{\alpha}(t)$ as the link travel time for vehicles of class α entering link ℓ_{ij} at time t when no physical-queue exists. To evaluate the $D_{ij}^{\alpha}(t)$ function we must take into account the network congestion. Traffic models normally assume a relation between the traffic intensity and the travel time, taking into account that the larger the congestion, the smaller the velocity or larger the travel time.

In this paper, the travel time $D_{ij}^{\alpha}(t)$ of a vehicle of class α that enters link ℓ_{ij} at time t when no physical-queue exists is assumed to be a function of the number of vehicles $x_{ij}^{\alpha}(t)$ (link volume) of all possible classes α at link ℓ_{ij} and time t and in order to reproduce the upstream shock wave due to congestion, we also assume that it depends on the immediate downstream route link volumes. More precisely we use the extension to the BPR function in Equation (11.7).

The travel time function (11.7) is an important part of the overall dynamic network loading (DNL) model. The first term penalizes the number of cars inside link ℓ_{ij} at entry time t, and the second term penalizes high congestions on any route immediately ahead of link ℓ_{ij} and permits propagating congestion upstream of link ℓ_{ij}.

Table 11.3 provides the parameters used in the congestion function (11.7) for the network in Figure 11.1. Note that the capacity $x_6^{\alpha max}$ is much smaller than the values for other links to analyze the effect on the traffic flow, and that the capacity of link 8 is larger because most routes use it. We have assumed a free flow speed of 120 Km/hour for class 1 (cars), 80 Km/hour for class 2 (trucks) and 130 Km/hour for class 3 (motorbikes). Finally, we have assumed that the β^3 and δ^3 parameters of class 3 (motorbikes) are much smaller, because they behave in a different way under congestion.

Class 1: Cars							
route	μ^1_{r1}	σ^1_{r1}	h^1_{r1}	μ^1_{r2}	σ^1_{r2}	h^1_{r2}	Route flow
1	8.56	1.90	3263	16.00	1.71	3234	6497
2	8.61	2.30	3228	15.30	1.72	3206	6434
3	8.25	1.85	3244	14.94	2.49	3233	6477
4	8.75	2.07	3207	15.56	1.63	3238	6445
5	8.09	1.81	3245	15.01	1.85	3237	6481
6	8.08	1.92	3264	14.62	2.06	3234	6498
7	8.46	1.93	3251	15.29	1.75	3278	6529
8	8.01	1.94	3220	15.37	2.31	3201	6421
9	8.98	1.51	3212	16.12	1.81	3235	6447
Class 2: Trucks							
route	μ^2_{r1}	σ^2_{r1}	h^2_{r1}	μ^2_{r2}	σ^2_{r2}	h^2_{r2}	Route flow
1	7.57	2.00	1220	17.96	2.20	1229	2449
2	7.50	1.88	1232	17.95	2.00	1239	2472
3	7.49	1.80	1234	17.80	2.45	1201	2435
4	7.73	1.65	1240	18.55	2.47	1207	2446
5	7.45	1.75	1213	17.39	1.81	1223	2435
6	7.94	2.28	1201	18.44	2.23	1220	2422
7	7.31	1.69	1217	17.09	2.35	1203	2420
8	7.66	2.07	1228	17.60	2.08	1220	2448
9	7.95	1.53	1219	18.18	1.91	1215	2435
Class 3: Motorbikes							
route	μ^3_{r1}	σ^3_{r1}	h^3_{r1}	μ^3_{r2}	σ^3_{r2}	h^3_{r2}	Route flow
1	8.78	2.18	1146	15.68	1.98	1144	2290
2	8.32	2.29	1134	15.02	1.58	1134	2268
3	8.84	1.90	1138	16.08	1.58	1129	2266
4	8.27	2.03	1147	16.10	1.55	1137	2284
5	8.60	1.88	1147	15.52	1.80	1140	2287
6	8.17	1.91	1124	15.23	2.31	1124	2249
7	8.47	1.65	1147	15.50	1.90	1138	2286
8	8.96	1.88	1147	15.54	2.19	1133	2280
9	8.15	1.70	1146	14.61	2.10	1127	2273

Table 11.2: Parameters used in route flows of the illustrative example.

Continuous model.

Though link travel time functions involve infinite degrees of freedom, we approximate these functions by monotone cubic splines whose number of parameters remains finite and

Class 1: Cars					
Arc	t^1_{0ij}	β^1_{ij}	δ^1_{ij}	γ	x^{1max}_{ij}
1	1.86	1.00	0.40	2.00	100.00
2	1.86	1.00	0.40	2.00	100.00
3	1.86	1.00	0.40	2.00	100.00
4	1.86	1.00	0.40	2.00	100.00
5	1.18	1.00	0.40	2.00	100.00
6	1.67	1.00	0.40	2.00	50.00
7	1.18	1.00	0.40	2.00	100.00
8	1.86	1.00	0.40	2.00	200.00
9	1.86	1.00	0.40	2.00	100.00

Class 2: Trucks					
Arc	t^2_{0ij}	β^2_{ij}	δ^2_{ij}	γ	x^{2max}_{ij}
1	2.80	1.00	0.40	2.00	40.00
2	2.80	1.00	0.40	2.00	40.00
3	2.80	1.00	0.40	2.00	40.00
4	2.80	1.00	0.40	2.00	40.00
5	1.77	1.00	0.40	2.00	40.00
6	2.50	1.00	0.40	2.00	20.00
7	1.77	1.00	0.40	2.00	40.00
8	2.80	1.00	0.40	2.00	80.00
9	2.80	1.00	0.40	2.00	40.00

Class 3: Motorbikes					
Arc	t^3_{0ij}	β^3_{ij}	δ^3_{ij}	γ	x^{3max}_{ij}
1	1.72	0.50	0.05	2.00	250.00
2	1.72	0.50	0.05	2.00	250.00
3	1.72	0.50	0.05	2.00	250.00
4	1.72	0.50	0.05	2.00	250.00
5	1.09	0.50	0.05	2.00	250.00
6	1.54	0.50	0.05	2.00	125.00
7	1.09	0.50	0.05	2.00	250.00
8	1.72	0.50	0.05	2.00	500.00
9	1.72	0.50	0.05	2.00	250.00

Table 11.3: Parameters used in the congestion function. The values $\mu^1 = \mu^2 = 0.5$, $\mu^3 = 0.65$, $\sigma^1 = \sigma^2 = \sigma^3 = 0.25$ were also used in the example.

small. To this end we determine the link travel times of all class users at a given discrete set of fixed times, and use these data to directly approximate the link travel time functions by monotone cubic splines (continuous functions).

Path link exit time functions.

Path link exit time functions, which are a very large number of functions (number of classes×number of paths×mean number of links per path), are evaluated based on link

exit-entry time functions, which are a small number (the number of links), by means of

$$\theta_{ij}^{r\alpha}(t) = \tau_{ijn}^{\alpha}(\tau_{ij(n-1)}^{\alpha}(\ldots \tau_{i}^{\alpha} j(t))), \tag{11.18}$$

where $ij1, ij2, \ldots, ijn$ are the sorted links of path r and $\tau_{ij}^{\alpha}(t)$ is the entry time of a user that exits link ℓ_{ij} at time t.

We note that the above process requires the knowledge of the functions $\theta_{ij}^{r\alpha}(t)$. There are two possible options to deal with these functions:

1. Use monotone cubic splines to fit these functions. This option requires fitting a high number of splines (number of routes $\times$ mean number of links per route), which is memory and time consuming.

2. Build these functions based on the $\tau_{ij}^{\alpha}(t)$ functions. This process requires only to fit as many functions as links, but has the inconvenience that each evaluation is more costly.

Nevertheless, with respect to required computer time and resources, the second alternative is more convenient. For this reason, we recommend the second option, which is the one used in the examples in this chapter and in the MatLab computer implementation that has been developed for the examples.

We note that spline functions have been used before by other authors to approximate travel time functions, but not all spline functions are necessarily monotone (see Rubio-Ardanaz et al. (2003)).

Note that the times required to cross nodes are assumed to be null. This implies that the link exit time functions can be built directly from the link travel functions alone.

Example 16 (Normal densities) *Figure 11.4 shows the $\theta_{ij}^{r\alpha}(t)$ functions obtained after using a cubic-spline interpolation between the selected discretized times with the units being hours. The number of colored bands coincides with the number of links per path. The link exit-entry time functions have been obtained by adding the departure time function $f(t) = t$ and the corresponding link travel time functions for all routes with the corresponding link.*

Note that the fact that motorbikes travel faster than cars and cars faster then trucks is revealed in the corresponding band widths in Figure 11.4.

The plots show that there is a high congestion level only in link 6 *(shown by its irregular band width). The lack of congestion in other links is revealed by the almost constant width of the corresponding bands in the figure. Consequently, only paths* 4 *and* 6*, which include link* 6*, show congestion.*

Approximation of the $\tau_{ij}^{\alpha}(t)$ function by monotone cubic splines.

To approximate the $\tau_{ij}^{\alpha}(t)$ functions, we partition an extended day (of 32 hours say) into $m-1$ disjoint and exhaustive intervals $(t_k, t_{k+1}); k = 1, 2, \ldots, m-1$, where $t_1 = 0$ and

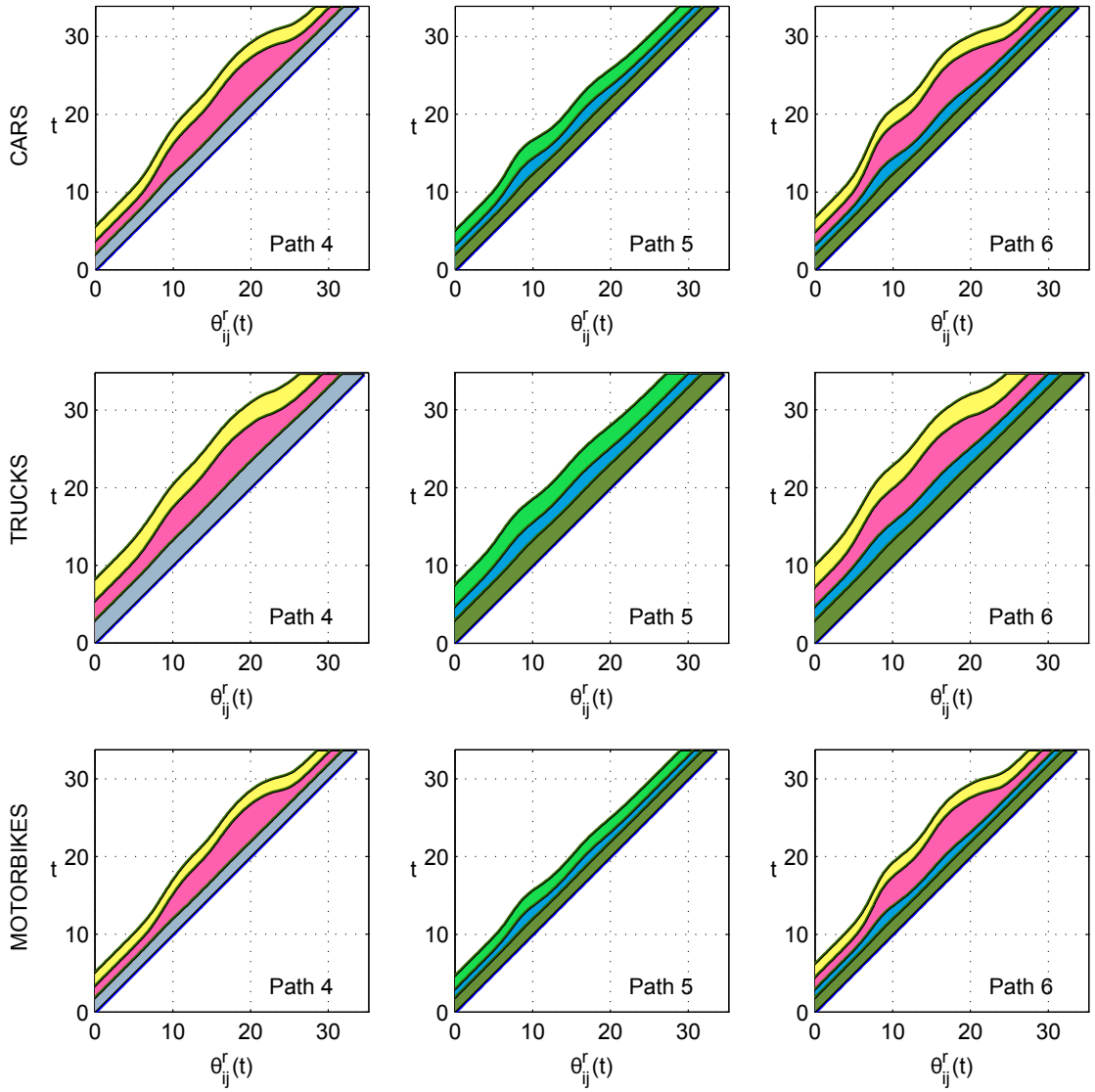


Figure 11.4: Illustrative example. Link exit time functions obtained by adding the departure time function $f(t) = t$ and the corresponding link travel time functions for all routes with the corresponding link. $\theta_{ij}^{r\alpha}(t)$ is the departure time from the origin of path r of a user of class α who exits link ℓ_{ij} at time t.

$t_m = 32$, and we evaluate the travel times of all links when the link travels are started at times $t_k; k = 1, 2, \ldots, m$. In other words, we analyze users who enter the links at discrete times $t_1, t_2, \ldots, t_m$ and calculate the number of link users $x_{ij}^{\alpha}(t)$ at the time of arrival to the link. With this information, we use the conservation law and the congestion equation (11.7) to obtain the times $D_{ij}^{\alpha}(t_k)$ required for these users to travel each path link. Finally,

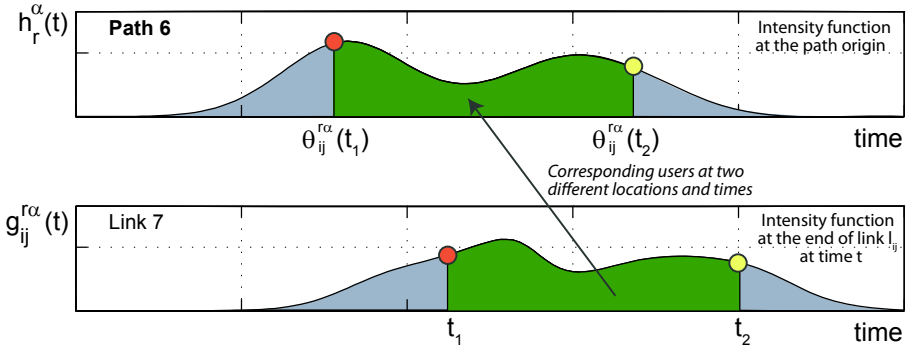


Figure 11.5: Illustrative example. Illustration of how the traffic flow wave satisfies the conservation law and the function $\theta_{ij}^{r\alpha}(t)$.

the set of points $\{(t_{ijk}^{\alpha out}, t_k); \forall k\}$ is used to fit the monotone cubic splines to approximate the link travel time functions for each link ℓ_{ij} and class α.

In order to satisfy the FIFO rule, we work with monotone functions $\tau_{ij}^{\alpha}(t)$, i.e., the necessary and sufficient condition for the FIFO condition to hold. These monotone functions are monotone cubic splines, the monotonicity of which is guaranteed by means of two conditions: (a) a set of monotone increasing basic points $\{(t_{ijk}^{\alpha out}, t_k) | k = 1, 2, \ldots, m\}$ used to fit the splines, and (b) the use of interpolating monotone cubic splines preserving its increasing character at all points.

In other words, we approximate the $\tau_v^{\alpha}(t)$ function by means of

$$\tau_{ij}^{\alpha}(t) \approx spline_{ij}^{\alpha}(t; (\mathbf{t_{ij}^{\alpha out}}, \mathbf{t})), \tag{11.19}$$

where $\mathbf{t_{ij}^{\alpha out}} = (\mathbf{t_{ij1}^{\alpha out}}, \mathbf{t_{ij2}^{\alpha out}}, \ldots, \mathbf{t_{ijm}^{\alpha out}})$ and $t = (t_1, t_2, \ldots, t_m)$ are a set of discrete entry times where the exit times to link ℓ_{ij} are evaluated. Note that identical entry times $\mathbf{t}$ are selected for all links.

To be more precise, within each iteration we fit new splines to the new set of basic points $(\mathbf{t_{ij}^{\alpha out}}, \mathbf{t})$, where only $\mathbf{t_{ij}^{\alpha out}}$ is recalculated from iteration to iteration.

Conservation law.

In order to see how the individual path flow waves progress throughout the path, we need to apply the conservation law. In Figure 11.5 we show the traffic flow wave at two given locations: (a) the path origin and (b) the end of link ℓ_{ij} of path r, and we show the associated time intervals corresponding to the same users. We also illustrate the $\theta_{ij}^{r\alpha}(t)$ function, which provides the departure time from the origin of path r of a user who exits link ℓ_{ij} of path r at time t.

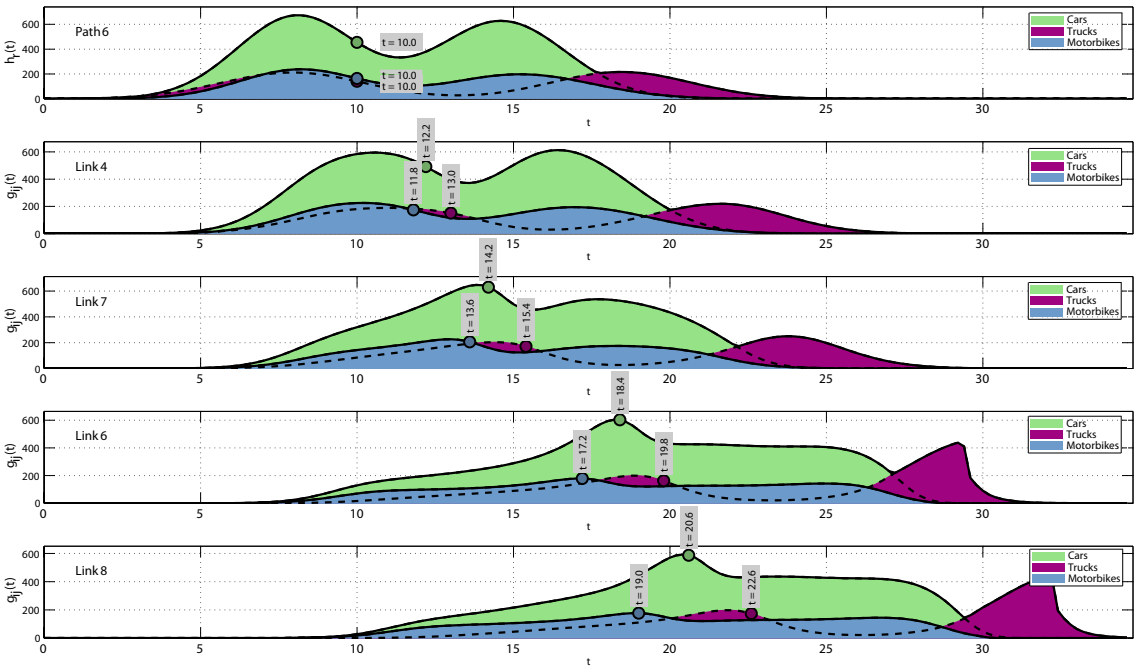


Figure 11.6: Illustrative example. Evolution of the flow wave at the origin node and at the ends of all links of path 6 for cars, trucks and motorbikes together with the time evolution of three users one from each class with the same path departure time.

If $g_{ij}^{r\alpha}(t)$ is the outflow rate of link ℓ_{ij} and class α at time t due to path r, since the area associated with a time interval and below the intensity function is the number of users that pass through that location, the mass balance constraint leads to the following equation that expresses the coincidence of the number of users in the time intervals (t_1, t_2) and $(\theta_{ij}^{r\alpha}(t_1), \theta_{ij}^{r\alpha}(t_2))$, (see the graph in Figure 11.5 and Friesz et al. (2001)). This leads to Equation (11.11).

Note that the argument $\theta_{ij}^{r\alpha}(t)$ of function $h_{ij}^{\alpha}(\cdot)$ models the local stretching or enlarging of the flow wave, and that $\dfrac{d\theta_{ij}^{r\alpha}(t)}{dt}$ modifies the traffic intensity (flow wave height) accordingly.

We point out that there exists a different $\theta_{ij}^{r\alpha}(t)$ function for each link ℓ_{ij} of each path r and class α. Consequently, the progression of the individual path flow wave throughout their paths can be known as soon as the function $\theta_{ij}^{r\alpha}(t)$ is known and the resulting flow wave form is given by Equation (11.11).

Equation (11.11) guarantees that the global mass balance constraint holds, that is, Equation (11.10), which can be used to derive $\theta_{ij}^{r\alpha}(t)$ when $h_r^{\alpha}(t)$ is given.

In Figure 11.6 we show how the path flow waves for the different class users progress

through path 6 and evolve with time. The upper plot corresponds to the origin node traffic intensities and the other four plots indicate how these flow waves move due to time and deforms due to increasing congestion. This implies that the traffic intensities decrease and the travel times increase with increasing congestion. More precisely, once each path flow enters the network, it progresses as a flow wave that stretches or enlarges, depending on the degree of congestion of the traversed links, as shown in Figure 11.6. Note that at earlier times in the day the route congestion is nonexistent or low (upper plot in Figure 11.6), but later it increases due to the presence of users of other routes, producing a decrease in the traffic intensities, that is, an enlargement of the traffic flow wave (intermediate and lower plots in Figure 11.6). It is easy to see that trucks traveling at night time have a larger mean speed.

The areas below these curves are the number of users of the corresponding path, so that these areas must be identical at all times for all individual path flows.

Note that the flow wave trend generated at the congested link 6 is transmitted and propagated throughout link 8 with almost no deformation.

In Figure 11.6 we can also observe the time evolution of three users one from each class with the same path departure time (10.00 hours), as indicated in the upper plot. Since motorbikes are faster than cars, and these faster than trucks, the travel times increase in this order and the differences increase with time. We point out that the travel times associated with cars, trucks and motorbikes are 10.6, 12.6 and 9.0 hours, respectively, which have been calculated by subtracting the times in the lower and upper plots.

Arc and node flow intensities.

Since the flows corresponding to each path mix with other path flows and all together generate a mixed flow function, we can obtain the link and node flows by combining the path flows adequately, that is, as indicated by Equations (11.8) and (11.9). Note that in Expression (11.8) we use the $\theta_{ij}^{r\alpha}(t)$ function, which must be known in order Equation (11.8) to be applicable. In our iterative approach, to be explained later, we use an actual version of this function which is updated until convergence.

Illustrative examples of the link and node flow intensities is given in the following examples.

Example 17 (Normal densities) *Figure 11.7 illustrates Expression (11.8) showing the flow intensity curves for all links and class users at the end of the iterative process, that is, after convergence, where the different colors refer to the different path components. We can easily see that they correspond to the sum of the flows of all paths, taking into account the delays associated with each path. Note that the appearance of peaks corresponds to peaks of the different paths. In some cases, as in link* 6*, some peaks disappear due to congestion.*

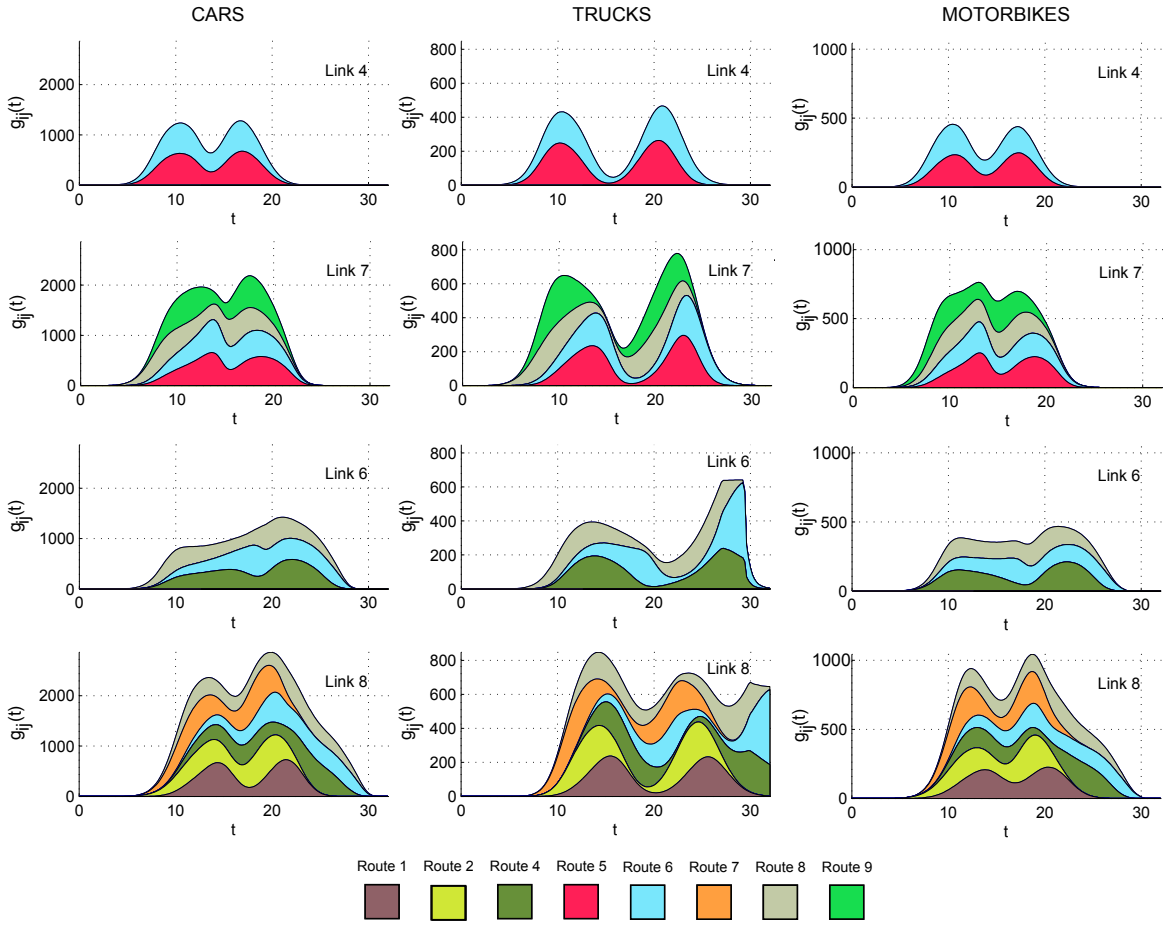


Figure 11.7: Illustrative example. Arc flow intensity curves showing the corresponding path components for the case of the illustrative example.

It is interesting to point out the following facts (see Figure 11.7): (a) we have similar plot trends for cars and motorbikes and different trends for trucks, (b) a physical-queue is generated in link 6 *(see the constant (horizontal) flow densities in this link after t=27.00), and (c) the flow intensity curve shapes generated at link* 6 *propagate with practically no deformation throughout link* 8*.*

Figure 11.8 shows the traffic intensity entering the different nodes at the end of the iterative process, that is, after convergence, where the different colors refer to the different path components. A detailed analysis of these curves allows us to identify the route interactions and their effect on congestion. Note that the plots are similar to those in Figure 11.7, so that similar conclusions can be drawn.

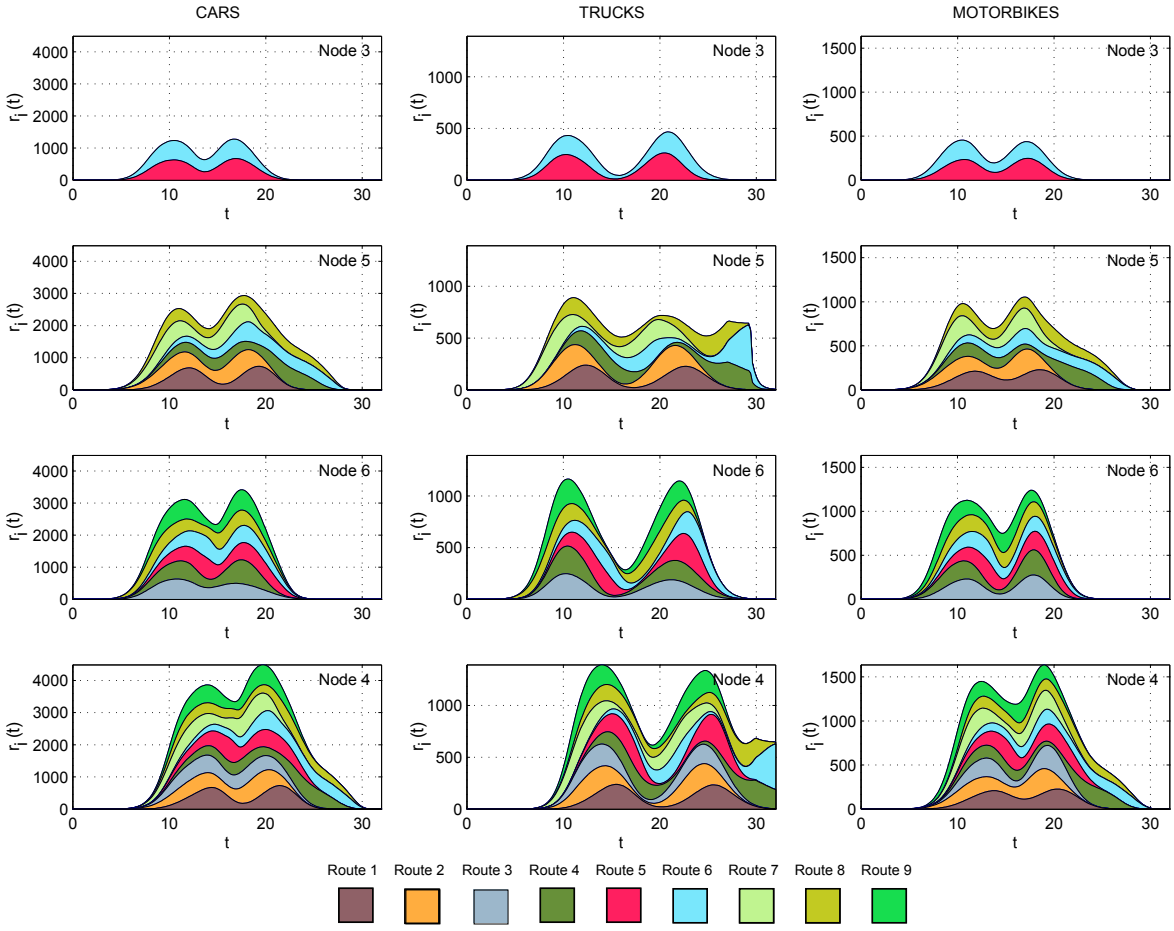


Figure 11.8: Illustrative example. Node flow evolution and the corresponding path flow contributions.

The link physical-queues.

Note that in fact (11.14) can imply an increase of the link travel time due to physical-queues if the second term (a queuing term) $t^{\alpha out}_{ij(k-1)} + Q^{\alpha}_{ijk}$ is greater than the first term $t_k + D^{\alpha}_{ij}(t_k)$.

Note also that preventing vehicles from overtaking means queueing the corresponding vehicles behind other vehicles, and this is what expression (11.14) does. A detailed explanation of Expression (11.14) can be seen in Castillo et al. (2012).

Note the high congestion in link 6 (see Figure 11.9). An associated light upstream congestion propagation can be observed at link 7.

To approximate the link travel time curves by monotone cubic splines, we select a

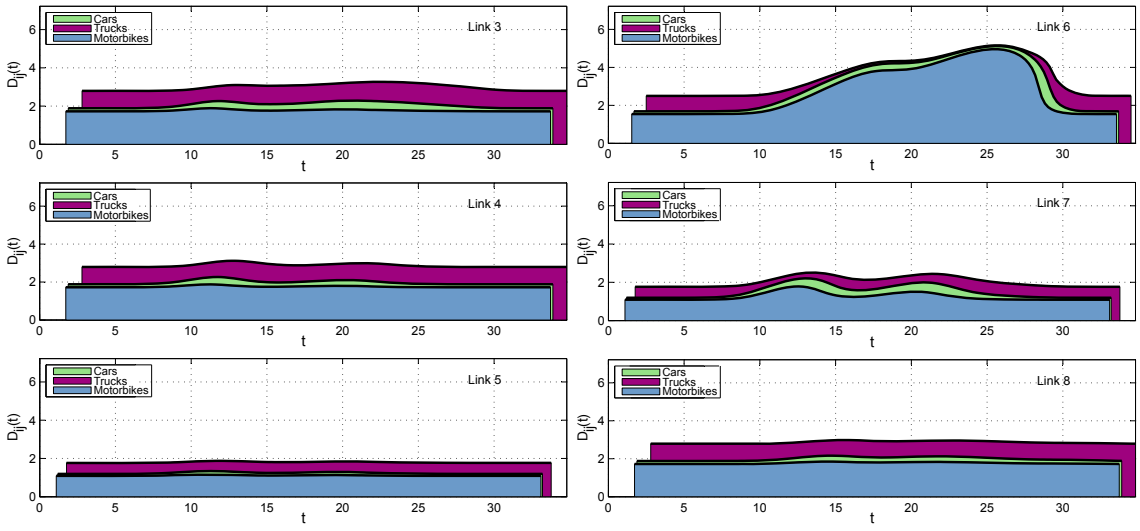


Figure 11.9: Illustrative example. Link travel time evolution of the different class users.

discrete set of points $\{(t_{ijk}^{\alpha out}, t_{k\alpha}); k = 1, 2, \ldots, m\}$, where $t_k = (k-1)\delta$ with $m = 32/\delta + 1$ and δ can take different values, for example $\delta = 0.1, 0.2, 0.5$ or 1 if the network congestion wants to be evaluated every 6 minutes, 12 minutes, 30 minutes or one hour.

Iteration scheme.

Since, as indicated previously, the $x_{ij}^{\alpha}(t)$ and $D_{ij}^{\alpha}(t)$ functions are interrelated, and they are dependent on congestion which is not known initially, we need to iterate the process until convergence. In other words, we need to assume an initial degree of congestion at the different links, and according to it we evaluate the link travel times, and with this information we update the congestion degrees, and repeat the process until convergence of the process.

The main problem is that initially we do not know the number of vehicles $x_{ij}^{\alpha}(t)$ inside link ℓ_{ij}, and we need to make an initial guess to later iterate until convergence.

In other words, $\tau_{ij}^{\alpha}(t)$ is a function of $x_{ij}^{\alpha}(t)$, and $x_{ij}^{\alpha}(t)$ is a function of $\tau_{ij}^{\alpha}(t)$, so that we need an iterative process until convergence.

Eleven iterations were required for convergence in our illustrative example.

11.5 Examples of application

In this section the illustrative example is completed and the proposed method is used to solve more complicated examples to show that it is applicable in real practice.

11.5.1 Illustrative example

We start by mentioning that the cpu time required to solve the illustrative problem on a HP Z200 Workstation, Intel Core i7-870 2.93 8MB/1333 QC, RAM: 8GB (2x4GB), was 73.00 sec.

In Figure 11.4 we saw the evolution of the different $\theta_{ij}^{r\alpha}(t)$ time functions for every path. Each of them permits us to identify the instant at which a traveler reaches the end of the path and the end of every link during the journey. Note that it is easy to identify when is the worst time to start the path travel. At the same time it is interesting to underline that in all graphs the time needed for covering the journey is very similar for people who start at the first and the last hours of the day, because of the lack of congestion.

In this case, the different graphs in Figure 11.6 show an important distortion due to the congestion produced at link 6 when the link intensity is higher than $x_{ij}^{\alpha max}$. More precisely, it maintains its shape (flow wave length and height) at links 4 and 7 due to lack of congestion. At link 6 the flow wave suffers two different processes: (a) an elongation, due to congestion, roughly between 22 and 26 hours, and (b) a stretching (flow wave length reduction and height increase) after 26 hours on, due to the end of congestion produced by arrival to destinations of users from other paths. Finally, since link 8 is not congested it does not deform further the flow wave shape.

Figure 11.7 shows the flow intensity curves at the different links. We have accumulated the flow intensities corresponding to each path so that each color corresponds to a unique path. In this way it is possible to see the combined effect of all of them at each link, and the contribution of each path to the total link flow intensity, but it is a bit more difficult to analyze the evolution of each path independently, because both limits are curved lines with the exception of the first path with flow.

Figure 11.9 shows that for high congestion levels all class users have the same link travel time (see the link 6 plot around $t = 26.00$), but different for low level congestions (see, for example, the link 6 plot around $t = 5.00$).

11.5.2 The Sioux-Falls example

In this section we use the well known Sioux Falls network in Figure 11.10 to illustrate the proposed method. It consists of 24 nodes and 76 links. A total of 161 paths and three class users (Cars1, Cars2 and Motorbikes sorted by increasing speed) have been considered.

Moreover, the Spiess function in (11.3) has been considered in this example. Due to this assumption, Equation (11.15) becomes

$$Q_{ijk}^{\alpha} = \left[E_{ij}^{\alpha}(t_{ijk}^{\alpha out}) - E_{ij}^{\alpha}(t_{ij(k-1)}^{\alpha out})\right] \frac{2\, t_{0ij}^{\alpha}}{k_{cong}^{\alpha} x_{ij}^{\alpha max}}. \tag{11.20}$$

The rationale behind this modification is as follows: if link ℓ_{ij} holds $x_{ij}^{\alpha max}$ users, the link travel time becomes $2\, t_{0ij}^{\alpha}$.

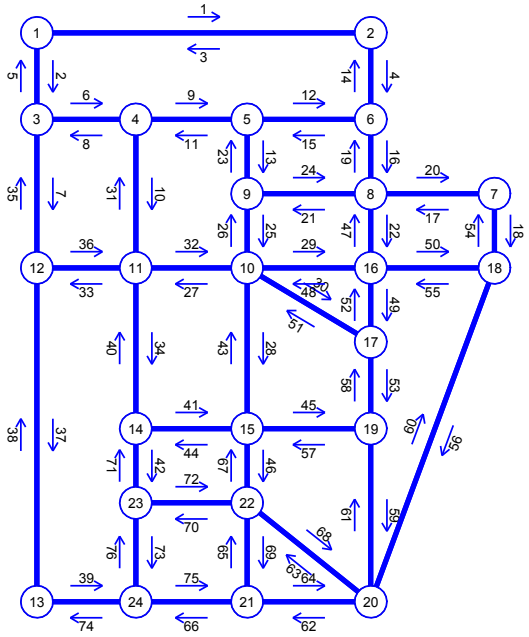


Figure 11.10: Sioux-Falls network.

Class α	ρ_{ij}^{α}	δ_{ij}^{α}	γ	$x_{ij}^{\alpha max}$	μ^{α}	σ^{α}	μ_{r1}^{α}	μ_{r2}^{α}
1	3.00	0.33	1.00	100.00	0.5	0.25	20	95
2	3.00	0.33	1.00	100.00	0.5	0.25	25	95
3	2.00	0.10	1.00	250.00	0.6	0.25	3 5	105

Table 11.4: Sioux-Falls. Parameters used in the congestion function and the path origin flow intensity curves.

Table 11.4 shows the parameters used in the congestion function and the mean values of the peak instants used to generate the random path origin flow intensity curves. Since all data cannot be displayed, a small selected part of them are shown. Some of the assumed path flow intensities at their origins are shown in Figure 11.11 and a study period of two hours have been selected (from 0 to 120 minutes in the figures). A value of $m = 120$ has been used, which is equivalent to a discretization every minute. The cpu time required was 20 minutes.

Figure 11.12 shows the evolution of the flow wave throughout path 15. The upper plot corresponds to the flow intensity at the origin node and the other plots show the flow intensity at the ends of all links of path 15 for Cars1, Cars2 and Motorbikes users. In addition, the time evolutions of three users one from each class with the same path

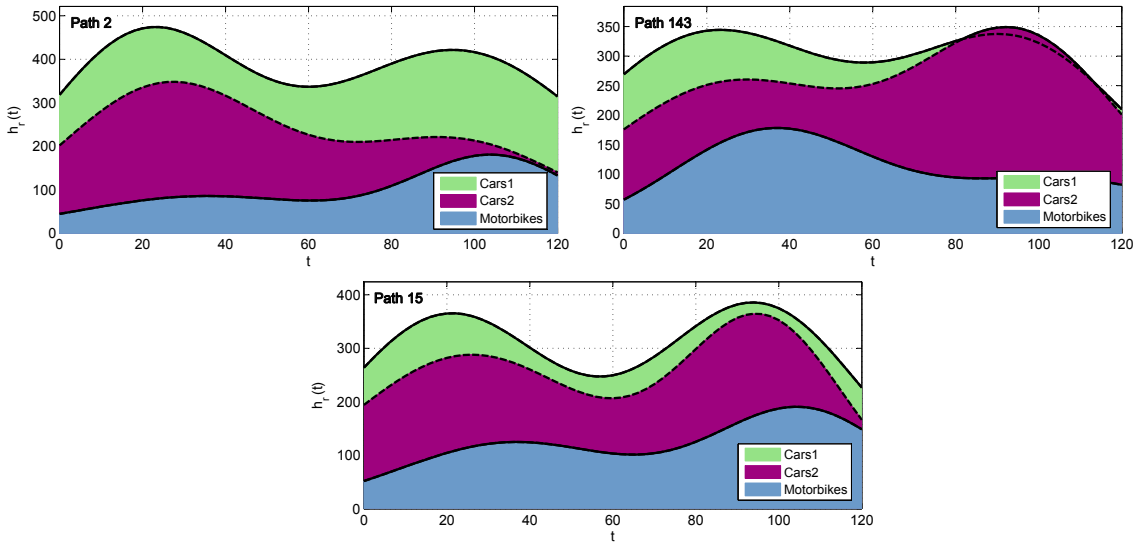


Figure 11.11: Sioux-Falls example. Path flow intensities at the path origins as a function of time for the Sioux-Falls example and the corresponding three class users.

departure time are shown. As expected motorbikes show smaller link travel times than Cars2, and Cars2 smaller than Cars1. This is revealed in all link plots, where it is shown that the time difference among the three users increases with time.

Figure 11.13 shows the link travel time (in minutes) evolution of the different class users for the links in path 15. It can be seen that links 4, 15 and 25 have small congestion, but link 13 reaches a high degree of congestion, which is revealed by the almost coincidence of the link travel times of the Cars1 and Cars2 users from time 35 minutes and for the three classes from time 115 minutes. Contrary, in links 4, 15 and 25 the travel times are different for the three users.

11.5.3 The Cuenca example

Since the two examples dealt with in the previous sections are small in size and were selected with the aim of illustrating the proposed methods, in this section the proposed method is applied to the Cuenca (Spain) network, with 127 nodes, 672 links, and 219 paths (see Figure 10.2). In this case two class users and a period of 20 minutes have been considered. The resulting CPU time was 39 minutes.

The observed flow trends corresponding to this example were very similar to those presented before for the small networks. Due to space limitations, we show only a few graphs.

Figure 11.14 shows the evolution of the flow wave at the origin node and at the ends of all links of path 90 for the Cars1 and Cars2 users together with the time evolution of

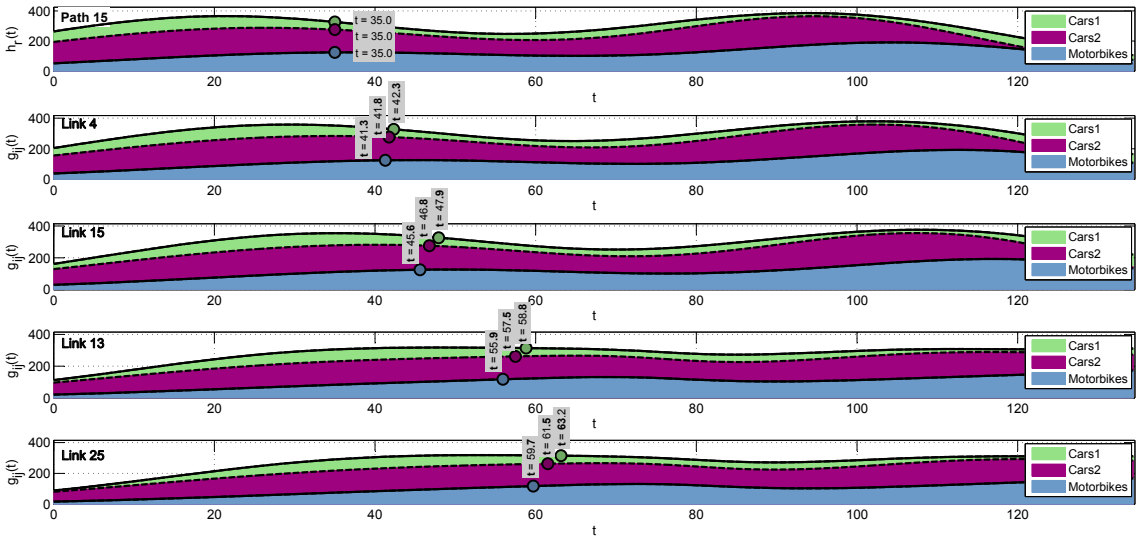


Figure 11.12: Sioux-Falls example. Evolution of the flow wave at the origin node and at the ends of all links of path 15 for the Cars1, Cars2 and Motorbikes users together with the time evolution of three users one from each class with the same path departure time.

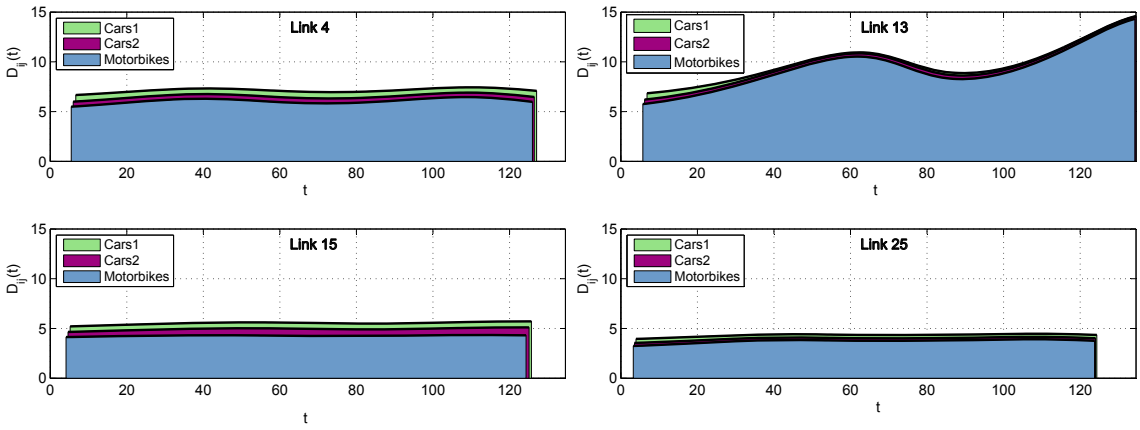


Figure 11.13: Sioux-Falls example. Link travel time (in minutes) evolution of the different class users in path 15.

two users one from each class with the same path departure time. Since link 622 is highly congested, the link travel time difference between Cars1 and Cars2 users is very small (4.78 and 4.79 minutes, respectively).

Figure 11.15 shows the link travel time (in minutes) evolution of the different class users in path 90, where it can be seen that links 622 and 352 are highly congested as

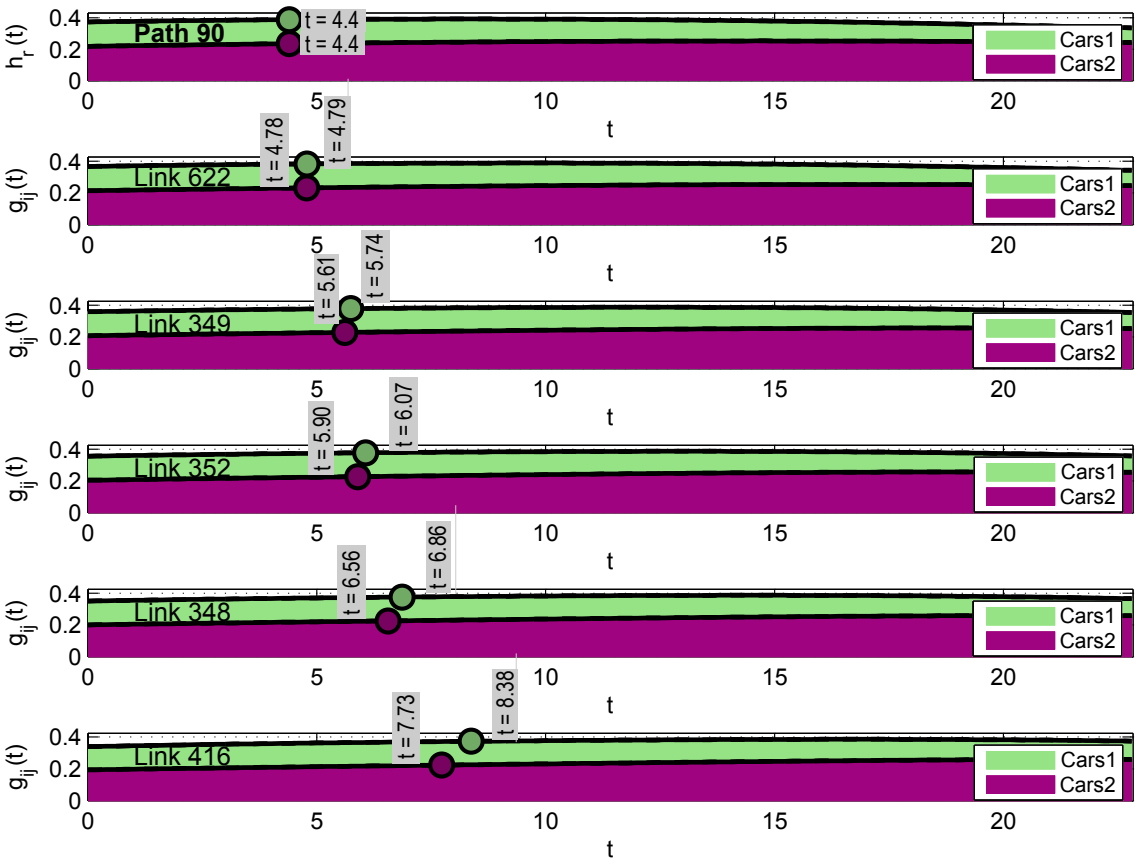


Figure 11.14: The Cuenca network. Evolution of the flow wave at the origin node and at the ends of all links of path 90 for the Cars1 and Cars2 users together with the time evolution of two users one from each class with the same path departure time.

indicated by the almost coincidence of the link travel times for both users.

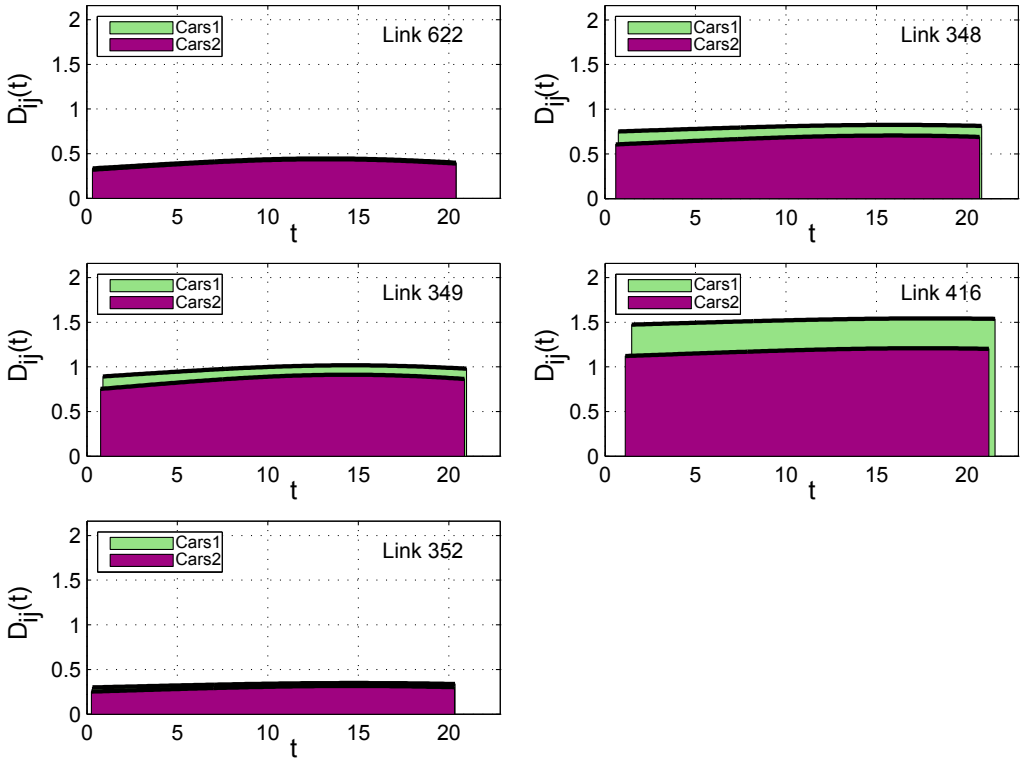


Figure 11.15: The Cuenca network. Link travel time (in minutes) evolution of the different class users in path 90.

Appendix

A Notation

α	class.
β	parameter of the travel time function.
β_{ij}^{0}.	link travel time function parameter associated with asymptotic class α and link ℓ_{ij}.
β_{ij}^{α}	link travel time function parameter associated with class α and link ℓ_{ij}.
γ	exponent link travel time function parameter.
γ^{0}	exponent link travel time function parameter for asymptotic class.
δ_{ij}^{α}	saturation parameter of the travel time function of link ℓ_{ij} and class α.
$\bar{\eta}_{rl}^{\alpha}$	vector of parameters.
ℓ_{ij}	link joining nodes i and j.
μ^{α}	mean of the normal distribution for class α.
μ_{rl}^{α}	mean parameter of the l-th component of path r and class α.
ρ	parameter of the travel time function.
σ^{α}	standard deviation of the normal distribution for class α.
σ_{rl}^{α}	standard deviation parameter of the l-th component of path r and class α.
$\tau_{ij}^{\alpha}(t)$	entry time of a user that exits link ℓ_{ij} at time t and class α.
$\theta_{\tau}^{r\alpha}(t)$	departure time from the origin of path r of a user of class α who exits link ℓ_{τ} of path r at time t.
$\mathcal{A}$	set of links.
$\mathcal{A}(i)$	set of links whose tail node is node i.
c	link capacity.
$\mathcal{D}$	set of node destinations.
$D_{ij}^{\alpha}(t)$	link ℓ_{ij} travel time at time t for class α when no physical-queues exists.
$E_{ij}^{\alpha}(t)$	cumulative flow associated with the exit of link ℓ_{ij} of class α by time t.
$f(x)$	normalized congestion function.
$f_{N(\mu,\sigma)}(t)$	probability density function (pdf) of the normal distribution with mean μ and standard deviation σ.
$F(x)$	cumulative distribution function.
$g_{ij}^{\alpha}(t)$	traffic flow intensity of class α at the exit of link ℓ_{ij} at time t.
$g_{ij}^{r\alpha}(t)$	outflow rate of link ℓ_{ij} at time t due to path r and class α.
$h_{r}^{\alpha}(t)$	inflow rate at the origin of path r at time t and class α.
h_{rl}^{α}	coefficients of the linear combination to generate $h_{r}^{\alpha}(t)$.
$H_{r}^{\alpha}(t)$	cumulative inflow rate at the origin of path r and class α at time t.
i	link begin node.
j	link end node.
k_{cong}^{α}	speed dissipation physical queue factor for class α.

m	number of discrete times considered.
n_r	number of function components.
$\mathcal{N}$	set of nodes.
$\mathcal{O}$	set of node origins.
$q^\alpha(t;\bar{\eta}^\alpha_{rl})$	parametric ($\bar{\eta}^\alpha_{rl}$) family of probability density functions of class α.
Q^α_{ijk}	physical-queue dissipation time at link ℓ_{ij} and class α at time t_k.
r	path.
$r^\alpha_i(t)$	flow intensity of class α users at node i and time t.
$\mathcal{R}_{ij}$	set of paths containing link ℓ_{ij}.
$\mathcal{S}(\ell_{ij})$	set of all links downstream link ℓ_{ij} in all its routes.
$s_{ij}(t)$	link ℓ_{ij} congestion ratio.
t	time.
t^0_0	free travel time for the asymptotic class.
$t(\cdot)$	link travel time function.
t^α_{0ij}	free-flow travel time of link ℓ_{ij} for class α.
t^α_0	link free travel time for class α.
t_k	set of discrete times to be considered for $k = 1,\ldots,m$.
$t^{\alpha out}_{ijk}$	link exit time of a user of class α who enters link ℓ_{ij} at time t_k.
$u^{\alpha evac}_{ij}$	link evacuation speed of link ℓ_{ij} and class α.
v	traffic volume.
$w^\alpha(x)$	proposed link travel time function for class α users and congestion ratio x.
x	congestion ratio v/c.
x_{max}	maximum expected congestion ratio.
$x^\alpha_{ij}(t)$	number of vehicles of class α on link ℓ_{ij} at time t (link traffic volume).
$x^{\alpha max}_{ij}$	number of vehicles on link ℓ_{ij} leading to a travel time $t^\alpha_{0ij}(1+\beta_{ij})$.

Chapter 12

Graphical Methods to Analyze Traffic Trajectories with and without Overtaking

Contents

12.1 Introduction

With the appearance of dynamic traffic models (see Chapter 6 for a review on these models), we need to deal with data that incorporate spatial and time coordinates at the same time (see Caceres et al. (2012)). Thus, sometimes it is difficult to understand such complex data. Based on the adage "A picture is worth a thousand words", we want to show that complex relations can be easily understood with the help of simple but very informative graphics. As it is well known, one of the main goals of visualization consists in making the understanding of large amounts of data possible and quickly.

One simple way to represent the traffic behavior consists in plotting the trajectories

of vehicles, that is, the two-dimensional diagrams (x, t), where x is the location and t is a reference time (local time). However, in many occasions we need to compare users who have departed from the origin at different times. In those cases another variable t_0, which represents the departure time, is needed. Thus, three variables $\{x, t, t_0\}$ must be represented. However, since three dimensional plots are difficult to analyze, we resort to two-dimensional trajectory plots, and we plot a family of curves that represent trajectories of users corresponding to a given set of different departure times. One decision to be made is how this set of trajectories is selected because the interpretation of the corresponding plots depends on how this selection has been made.

When different class vehicles are dealt with, we also need to superimpose trajectories and make the corresponding interpretation. In this paper and for the sake of illustration we have considered several class users whose speeds are different, and then we have considered the overtaking problem.

The interpretation of families of trajectories in terms of traffic behavior is not trivial and needs some study. If these trajectories correspond to different users, the problem complicates even more.

In this chapter the problems of plotting trajectories and how to interpret them for the different criteria used are analyzed. To our knowledge, the main original contributions are:

1. We introduce new concepts and measures to analyze the traffic behavior, such as "slowness", "promptness distance gain", "promptness time gain", "distance-gain promptness", "speed promptness rate", "slowness promptness rate", etc. They provide different means to identify interesting and useful properties of traffic flow. In particular, these measures permit, among others, to analyze what we gain or lose in terms of travel time if we advance or delay our departure time.

2. We distinguish between equally delayed plots and equal flow trajectory plots and indicate their practical relevance.

3. We provide rules to identify in trajectory plots how these measures or properties change with time and location.

4. We suggest the superposition of trajectories of different class users to characterize the overtaking frequencies and its spatial and time changes.

5. We provide some trajectory plots based rules to choose departure times in order to reduce travel times and congestions.

To illustrate the problem we have used the simple network[1] and data simulated with the model described in Chapter 11.

[1]The simple network is shown in Figure 11.1. It consists of 6 nodes, 9 links and 9 paths as indicated in Table 11.1.

The chapter is organized as follows. In Section 12.2 the information that can be extracted from trajectories is studied. In Section 12.3 the equally delayed and equal flow single class trajectory plots and how can they be interpreted are discussed. In Section 12.4 double class trajectory plots are introduced and an interpretation to the appearing bands in terms of overlapping is given. In Section 12.5 multiple trajectory plots are analyzed. Finally, in Section 12.6 we give some recommendations.

12.2 Information available from trajectories

In this section we analyze some interesting information that can be obtained from the trajectory profiles.

In this section we use two different plots: *equally delayed plots* and *equal flow trajectory plots*. In both plots we do not represent all user trajectories but only a fraction of them. In the first plot consecutive trajectories correspond to users whose departures are equally delayed, that is, the departure times differences are identical. We note that this does not mean that users depart at equally delayed times. In the second type of plot the number of users between any two consecutive plotted trajectories are the same.

As we will see, new concepts arise that require a name for which some proposals are made. However expert support is needed in order to assign them the most appropriate names[2].

12.2.1 Speed, slowness, acceleration and slowness distance and slowness time rates

Let $x = f(t; t_0)$ be the user location x, referred to its departure location, at time t when its departure time is t_0 and $t = g(x, t_0)$ be the time t to reach location x for a vehicle departed at time t_0. Since the *speed* (traveled length per unit time) at time t of a user whose departure took place at time t_0 is $\dfrac{\partial f(t, t_0)}{\partial t}$, the trajectory slopes are the corresponding speeds and the $\dfrac{\partial g(x, t_0)}{\partial x}$ are the inverses of the speeds. We will use the term *slowness* (time required to travel a unit distance) to refer to this inverse. So, the steeper the trajectory slope, the smaller the speed and the larger the slowness. We have another two first partial derivatives: $\dfrac{\partial f(t, t_0)}{\partial t_0}$, denoted *promptness distance gain* (traveled distance at time t per

[2]To understand the need of new terminology, we use a simple example. The fuel consumption in the US and Europe is measured in two different ways: (a) miles per gallon and (b) liter per Km. These two concepts are one the inverse of the other and consequently they have different units, but both are useful to measure consumption. However, the European term is more appropriate, because an increase of consumption implies an increase in the measure. This is not the case for the American term, that should be called *fuel efficiency*. As we will see, speeds, accelerations, etc. can also be measured by different terms.

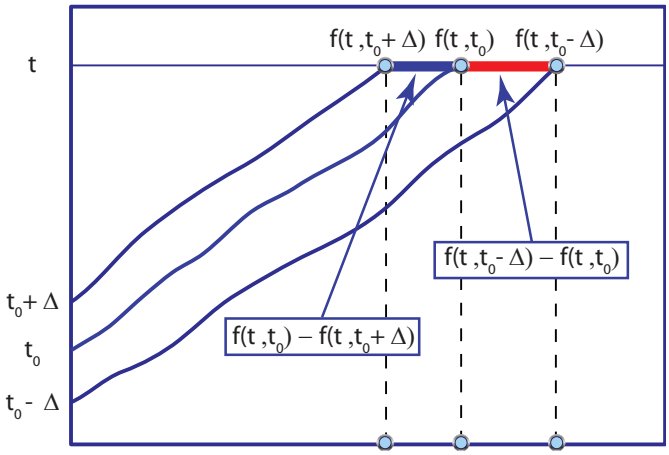


Figure 12.1: Trajectories of three users with gap departures Δ and locations reached at time t.

unit departure time change) and $\dfrac{\partial g(x,t_0)}{\partial t_0}$, denoted *promptness time gain* (time required to reach x per unit departure time change).

Since $\dfrac{\partial^2 f(t,t_0)}{\partial t^2}$ is the *acceleration* (speed rate), the trajectory curvature gives information about the acceleration associated with each time-location pair. A positive curvature means deceleration and a negative curvature means acceleration.

12.2.1.1 Interpretation of $\dfrac{\partial^2 f(t,t_0)}{\partial t_0^2}$

The second derivative $\dfrac{\partial^2 f(t,t_0)}{\partial t_0^2}$ measures the change increment in $\dfrac{\partial f(t,t_0)}{\partial t_0}$ per unit change in departure time and will be called *distance-gain promptness rate*. According to Figure 12.1, $f(t,t_0) - f(t,t_0+\Delta)$ is the distance at time t between two users whose departure times were t_0 and $t_0+\Delta$, respectively, and $f(t,t_0-\Delta) - f(t,t_0)$ is the distance at time t between two users whose departures were $t_0-\Delta$ and t_0, respectively.

Since both distances correspond to the same time departure difference Δ, if

$$f(t,t_0) - f(t,t_0+\Delta) < f(t,t_0-\Delta) - f(t,t_0),$$

the distance to vehicle ahead is larger than distance to vehicle behind (we refer to vehicles equally delayed at departure time).

Since

$$\frac{\partial^2 f(t,t_0)}{\partial t_0^2} = \lim_{\Delta\to 0} \frac{f(t,t_0+\Delta) - 2f(t,t_0) + f(t,t_0-\Delta)}{\Delta^2},$$

the second partial derivative $\frac{\partial^2 f(t,t_0)}{\partial t_0^2}$ provides an indication of how the traffic conditions change with t_0. A positive value of $\frac{\partial^2 f(t,t_0)}{\partial t_0^2}$ implies that the distance to vehicle ahead is larger than distance to vehicle behind. Contrary, a negative value implies that the distance to vehicle ahead is smaller than the distance to vehicle behind.

These interpretations are valid only when equally delayed plots are used.

In the particular case of free flow conditions we have:

$$\frac{\partial^2 f(t,t_0)}{\partial t_0^2} = h_0''(t-t_0),$$

where $h_0(t-t_0)$ gives the position of a user circulating at the free flow speed and whose departure time is t_0 at time t (see Section 12.2).

12.2.1.2 Interpretation of $\frac{\partial^2 f(t,t_0)}{\partial t \partial t_0}$

The second derivative $\frac{\partial^2 f(t,t_0)}{\partial t \partial t_0}$ measures the change in speed due to a unit change in departure time (what we win in speed if we delay our departure a unit time) and will be called *speed lateness rate*.

According to Figure 12.2, the distance traveled by a user whose departure time is $t_0+\alpha$ from t to $t+\Delta$ is $f(t+\Delta,t_0+\alpha)-f(t,t_0+\alpha)$ and the distance traveled by a user whose departure time is t_0 from t to $t+\Delta$ is $f(t+\Delta,t_0)-f(t,t_0)$. Thus, the difference between these two distances is:

$$f(t+\Delta,t_0+\alpha)-f(t,t_0+\alpha)-f(t+\Delta,t_0)+f(t,t_0). \tag{12.1}$$

A different interpretation of (12.1) is possible, as follows. The distance of two users whose departure times are t_0 and $t_0+\alpha$ at time t are $f(t,t_0)-f(t,t_0+\alpha)$ and the distance of two users whose departure times are t_0 and $t_0+\alpha$ at time $t+\Delta$ are $f(t+\Delta,t_0)-f(t+\Delta,t_0+\alpha)$. The difference of both is (12.1) (see Figure 12.2).

Since

$$\frac{\partial^2 f(t,t_0)}{\partial t \partial t_0} = \lim_{\Delta,\alpha\to 0} \frac{f(t+\Delta,t_0+\alpha)-f(t,t_0+\alpha)-f(t+\Delta,t_0)+f(t,t_0)}{\alpha\Delta},$$

the second partial derivative $\frac{\partial^2 f(t,t_0)}{\partial t \partial t_0}$ provides information about the sign and magnitude of this difference with t and t_0. Positive values mean that we win in speed if we delay our departure, and negative values that we lose in speed if we delay our departure.

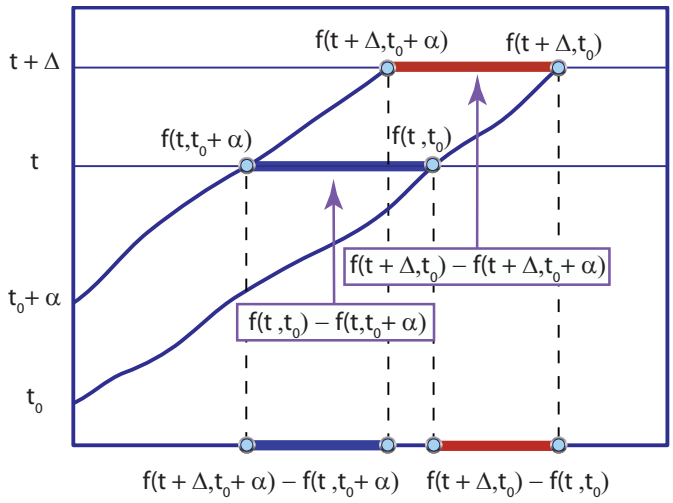


Figure 12.2: Trajectories of two users with gap departure α and locations reached by both at times t and $t + \Delta$.

12.2.1.3 Interpretation of $\dfrac{\partial^2 g(x,t_0)}{\partial x^2}$

The second derivative $\dfrac{\partial^2 g(x,t_0)}{\partial x^2}$ measures the change in slowness $\dfrac{\partial g(x,t_0)}{\partial x}$ per unit change in traveled distance x. We note that $\dfrac{\partial^2 g(x,t_0)}{\partial x^2}$ is not the inverse of the acceleration. We propose to call this second derivative the *slowness distance rate*. According to Figure 12.3, the time required for a user whose departure time is t_0 to travel from x to $x+\delta$ is $g(x+\delta,t_0)-g(x,t_0)$, and the time required to travel from $x-\delta$ to x is $g(x,t_0)-g(x-\delta,t_0)$. Thus, its difference is the increase in time to travel the same distance δ with x, and since

$$\frac{\partial^2 g(x,t_0)}{\partial x^2} = \lim_{\delta\to 0}\frac{g(x+\delta,t_0)-2g(x,t_0)+g(x-\delta,t_0)}{\delta^2},$$

the second partial derivative $\dfrac{\partial^2 g(x,t_0)}{\partial x^2}$ provides information on how the slowness increases with x. A positive value implies a slowness increase (deterioration) and a negative value a slowness decrease (improvement).

12.2.1.4 Interpretation of $\dfrac{\partial^2 g(x,t_0)}{\partial t_0^2}$

The second derivative $\dfrac{\partial^2 g(x,t_0)}{\partial t_0^2}$ measures the change increment in the promptness time gain $\dfrac{\partial g(x,t_0)}{\partial t_0}$ per unit change in departure time t_0 and will be called *time-gain promptness*

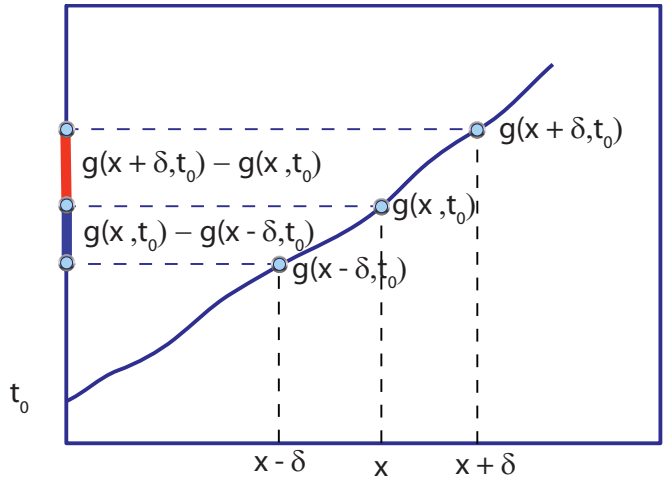


Figure 12.3: Trajectory of a user departuring at t_0 and times required to reach locations $x - \delta$, x and $x + \delta$.

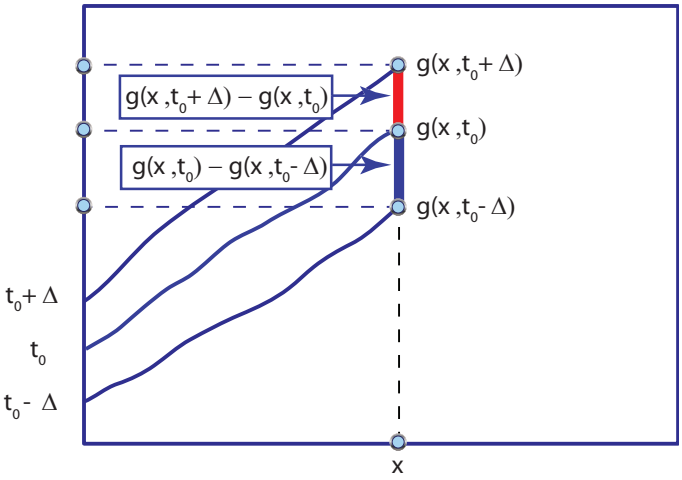


Figure 12.4: Trajectories of three users with gap departures Δ and times required to reach location x.

rate.

According to Figure 12.4, $g(x, t_0 + \Delta) - g(x, t_0)$ and $g(x, t_0) - g(x, t_0 - \Delta)$ are the time differences required to travel a distance x from the origin of three users whose departure times are $t_0 - \Delta$, t_0 and $t_0 + \Delta$, respectively.

Since

$$\frac{\partial^2 g(x,t_0)}{\partial t_0^2} = \lim_{\Delta\to 0}\frac{g(x,t_0+\Delta)-2g(x,t_0)+g(x,t_0-\Delta)}{\Delta^2},$$

the second partial derivative $\frac{\partial^2 g(x,t_0)}{\partial t_0^2}$ is a measure of how these time differences change with t_0. A positive value of $\frac{\partial^2 g(x,t_0)}{\partial t_0^2}$ implies a speed decrease (traffic deterioration) (it is better an earlier departure) and a negative value a speed increase.

These interpretations are valid only when equally delayed plots are used.

In the particular case of free flow conditions we have (see (12.3)):

$$\frac{\partial^2 g(x,t_0)}{\partial t_0^2} = 0,$$

which implies a stationary traffic for all x and t_0.

12.2.1.5 Interpretation of $\frac{\partial^2 g(x,t_0)}{\partial x \partial t_0}$

The second derivative $\frac{\partial^2 g(x,t_0)}{\partial x \partial t_0}$ measures the change increment in slowness $\frac{\partial g(x,t_0)}{\partial x}$ per unit change in departure time t_0 and will be called *slowness promptness rate*.

According to Figure 12.5, $g(x+\beta,t_0+\Delta)-g(x+\beta,t_0)$ and $g(x,t_0+\Delta)-g(x,t_0)$ are the gaps (time differences) corresponding to two users whose departures were t_0 and $t_0+\Delta$, when passing throughout locations $x+\beta$ and x, respectively.

If

$$g(x+\beta,t_0+\Delta)-g(x+\beta,t_0) < g(x,t_0+\Delta)-g(x,t_0)$$

the time gap (delay) between two users decreases with x, and since

$$\frac{\partial^2 g(x,t_0)}{\partial x \partial t_0} = \lim_{\Delta,\beta\to 0}\frac{g(x+\beta,t_0+\Delta)-g(x,t_0+\Delta)-g(x+\beta,t_0)+g(x,t_0)}{\Delta\beta},$$

the second partial derivative $\frac{\partial^2 g(x,t_0)}{\partial x \partial t_0}$ measures the time gap increment between two close locations. A positive value of $\frac{\partial^2 g(x,t_0)}{\partial t_0^2}$ implies a gap increase (traffic deterioration) and a negative value, a gap decrease (traffic improvement) (it is better an earlier departure).

Table 12.1 illustrates the different first and second partial derivatives of functions $f(t,t_0)$ and $g(x,t_0)$ together with what they compare, the physical meaning, the trajectory field feature and what they permit to evaluate for the cases of equally delayed and equal flow trajectories. All these helps interpreting the plots and extracting valuable information on the traffic behavior and how it evolves with time.

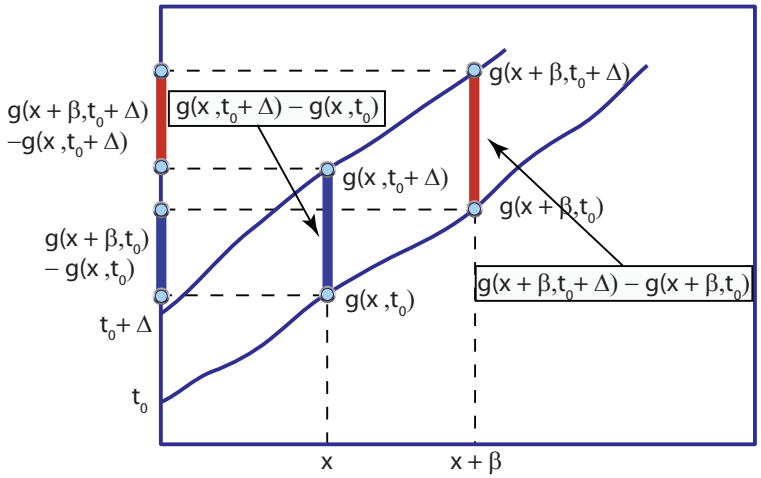


Figure 12.5: Trajectories of two users with a gap departure Δ and time differences to reach locations x and $x + \beta$.

12.3 Single class trajectory plots

This type of plots consists of trajectories (traveled distance x from the origin versus time t) of individual users all of the same class. Since it has been assumed that there is no overtaking among users of the same class, this set of trajectory profiles does not intersect.

Let x be the distance traveled from the origin at time t by a user who has departed from the origin at time t_0. We are interested in knowing the relation $x = f(t, t_0)$ that provides the distance x traveled by a user as a function of t and t_0.

In this paper two different types of single class trajectory plots are considered:

1. *Free flow plots.*They correspond to the hypothetical case of a user traveling alone throughout the network.

2. *Equally delayed plots.* They reproduce the trajectories of users whose departure times are equally delayed. This produces bands (sets of users between two trajectory lines) corresponding to the same departure time increment.

3. *Equal flow plots.* They reproduce the trajectories of users such that the accumulated number of users with previous departure defer in a constant. This produces bands (regions between two consecutive trajectories) associated with the same number of users.

Partial derivative	Compares	Physical meaning	Trajectory field feature	Equally delayed flow trajectories	Equal flow trajectories
$\frac{\partial f(t,t_0)}{\partial t}$	Traveled distances of a single user at two different very close times	Speed (distance rate)	Slope	The steeper the slope, the slower the speed.	
$\frac{\partial f(t,t_0)}{\partial t_0}$	Traveled distances of two users whose departure times are very close	Promptness distance gain	Horizontal distance between consecutive equally delayed trajectories.	The larger the horizontal distance with the following trajectory, the larger the promptness distance gain.	
$\frac{\partial g(x,t_0)}{\partial x}$	Passing times of a single user by two different very close locations	Slowness (time distance rate)	Inverse slope	The steeper the slope, the larger the slowness	
$\frac{\partial g(x,t_0)}{\partial t_0}$	Passing times of two users by the same location whose departure times are very close	Promptness time gain	Vertical distance between consecutive equally delayed trajectories.	The larger the vertical distance with the following trajectory, the larger promptness time gain.	
$\frac{\partial^2 f(t,t_0)}{\partial t^2}$	Traveled distances of a single user at three different very close times	Acceleration (speed rate)	Curvature (changed sign)	A positive curvature means deceleration.	
$\frac{\partial^2 f(t,t_0)}{\partial t_0^2}$ (see Figure 12.1)	Traveled distances at time t of three users whose departures times where equally delayed (Δ delay) and very close to t_0	Distance-gain promptness rate	Horizontal distance between consecutive trajectories.	The larger the horizontal distance to the preceding trajectory is with respect the horizontal distance with the following trajectory, the larger the distance-gain promptness rate is.	—
$\frac{\partial^2 f(t,t_0)}{\partial t \partial t_0}$ (see Figure 12.2)	Distances between two users with departure time very close to t_0 (α delay) at two times very close to t (Δ delay)	Speed lateness rate	Horizontal band width	The larger the horizontal band width, the larger the speed promptness rate.	
$\frac{\partial^2 g(x,t_0)}{\partial x^2}$ (see Figure 12.3)	Time gaps associated with three equally spaced (δ distance) locations very close to x for a single user with departure time t_0	Slowness distance rate	Curvature of the inverse function	A positive curvature means positive slowness distance rate.	
$\frac{\partial^2 g(x,t_0)}{\partial t_0^2}$ (see Figure 12.4)	The two time gaps at location x associated with three users whose departure times were equally delayed (Δ delay) and very close to t_0	Time-gain promptness rate	Vertical distances between consecutive trajectories.	The larger the vertical distance to the preceding trajectory is with respect the vertical distance with the following trajectory, the larger the time-gain promptness rate is.	—
$\frac{\partial^2 g(x,t_0)}{\partial x \partial t_0}$ (see Figure 12.5)	Time gaps at two locations (separated by a distance β) very close to x between two users who departed very close to t_0 (delayed Δ)	Slowness promptness rate	Vertical band width	The larger the vertical band width, the larger the slowness promptness rate.	

Table 12.1: Illustration of the different first and second partial derivatives of functions $f(t,t_0)$ and $g(x,t_0)$ together with what they compare, the physical meaning, the trajectory field feature and what they permit to evaluate for the cases of equally delayed and equal flow trajectories.

12.3.1 Free flow single class trajectories

A particular example is the case of free flow trajectories, that can be written in terms of a single argument function, as shown by the following theorem.

Theorem 18 (Free flow trajectories) *The free flow trajectories are given by*

$$x = f(t, t_0) = h_0(t - t_0), \tag{12.2}$$

where $h_0(\cdot)$ is an arbitrary increasing function such that $h_0(t - t_0)$ gives the position of a user circulating at the free flow speed and whose departure time is t_0 at time t, or its inverse

$$t = g(x, t_0) = h_0^{-1}(x) + t_0. \tag{12.3}$$

Proof. The following property must be satisfied by free flow users:

$$x = f(t, t_0) = f(t + \Delta, t_0 + \Delta); \quad \forall t, t_0 \geq 0, \tag{12.4}$$

which expresses that under free flow conditions, the distance traveled at time t by a user whose departure was t_0 must be identical to the distance x traveled at time $t + \Delta$ by another user whose departure was $t_0 + \Delta$, that is, the same travel time $t - t_0$.

Equation (12.4) is a functional equation with one unknown $f(t, t_0)$, which is easy to solve. To this end, we replace first $t_0 = 0$ in (12.4), next t by $t - t_0$ and then Δ by t_0 to get

$$x = f(t - t_0, 0) = f(t, t_0) = h_0(t - t_0), \tag{12.5}$$

which is (12.2).

Note that Equation (12.5) expresses a property of free flow users, that is equivalent to (12.4). This property says that the traveled distance is a function only on the traveled time $t - t_0$.

From (12.5) we can immediately obtain the time t as a function of x and t_0, that is, Equation (12.3). ∎

As indicated, the previous theorem states that the free flow trajectories can be given in terms of $t - t_0$. In other words in terms of a single argument $(t - t_0)$ function $h_0(\cdot)$. This is an important result, because once a trajectory is known, all of them can be immediately obtained.

The previous theorem shows that the free flow trajectories are parallel lines. They correspond to an unrealistic condition, because at least one user must be traveling, but it is the limit of a real case in which the number of traveling users tends to zero. Finally, free flow equally delayed plots lead to parallel and equally delayed trajectories, while free flow equal flow plots lead to parallel but evenly spaced trajectories.

However, in the case of congestion, the trajectories are not parallel anymore.

12.3.2 Single class equally delayed trajectory profiles under congestion

In Figure 12.6 we consider two sets (bands) of users associated with departures in the two intervals of identical time amplitude $(t_1 - t, t_1)$ and $(t_2 - t, t_2)$. These users follow

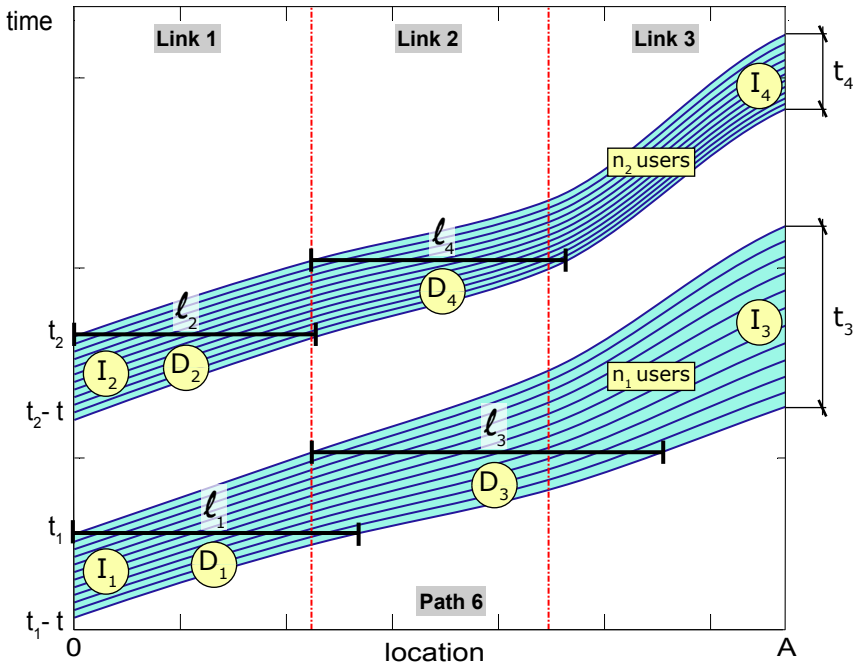


Figure 12.6: Illustration of the meaning of traffic intensity ratios (local flow rates) for two trajectory bands in the case of equally delayed trajectories.

trajectories in the shadowed regions in Figure 12.6 and reach the location A at times within time intervals of amplitudes t_3 and t_4, respectively.

If we assume that the FIFO rule is satisfied, from Figure 12.6 we can see that the traffic intensities at the two locations, the path origin O and location A, associated with the two bands of users are

$$I_1 = \frac{n_1}{t}; \quad I_2 = \frac{n_2}{t}; \quad I_3 = \frac{n_1}{t_3}; \quad I_4 = \frac{n_2}{t_4}, \tag{12.6}$$

where n_1 and n_2 are the number of users traveling in the first and second band, respectively.

This means that the traffic flow intensities for users traveling from O to A change from I_1 to I_3 and from I_2 to I_4 for users of the first and second band, respectively. Thus, the evolution of traffic intensities for the two bands can be measured by the intensity ratios (local flow rates) $r_1 = I_3/I_1$ and $r_2 = I_4/I_2$, respectively, which has the virtue of being dimensionless.

In addition, according to (12.6), we have

$$R = \frac{I_4/I_2}{I_3/I_1} = \frac{r_2}{r_1} = t_3/t_4.$$

Consequently, the time ratio $R = t_3/t_4$ is useful to compare the traffic intensity ratios for different departure intervals of the same width.

For example, if $t_3 > t_4$ ($R > 1$) the intensity ratio of band 2 is larger than the intensity ratio of band 1 ($r_2 > r_1$). In other words, the lighter the zone, the smaller the traffic intensity ratios. All this shows that the plots in Figure 12.6 cannot be interpreted in terms of intensities, but in terms of intensity ratios.

Similarly, if as before we assume that the FIFO rule is satisfied, from Figure 12.6 we can see that the traffic densities at the two locations associated with the two bands of users are

$$D_1 = \frac{n_1}{\ell_1}; \quad D_2 = \frac{n_2}{\ell_2}; \quad D_3 = \frac{n_1}{\ell_3}; \quad D_4 = \frac{n_2}{\ell_4}. \tag{12.7}$$

Then, we get

$$\frac{D_3}{D_1} = \frac{\ell_1}{\ell_3}; \quad \frac{D_4}{D_2} = \frac{\ell_2}{\ell_4}. \tag{12.8}$$

So, the length ratios $\frac{\ell_1}{\ell_3}$ and $\frac{\ell_2}{\ell_4}$ are good to compare traffic densities within the same bands.

However,

$$\frac{D_2}{D_1} = \frac{n_2\ell_1}{n_1\ell_2}; \quad \frac{D_4}{D_3} = \frac{n_2\ell_3}{n_1\ell_4}, \tag{12.9}$$

that is, it is not convenient to compare traffic densities of bands associated with the same width departure time intervals.

In Figure 12.7 we can see the separate trajectories of vehicles of the three classes: cars, trucks and motorbikes (one color for each class) corresponding to equally delayed departure times (user band associated with 12 minutes). The curves shown at the left (departure) and right (arrival) sides of each plot are the travel time excesses (measured as the difference between the trajectory travel time and the free flow travel time) associated with the different trajectories. In addition, the fastest and slowest path trajectories are shown and they correspond to midnight (smallest path travel time) and around 16 hours, where the largest travel time is attained.

It is interesting to see that the congested zones can be identified by an increase of the trajectory slope. This is easy because during the night hours the slopes correspond to free flow. For instance, in links 4, 7 and 8 the trajectories are almost parallel to the free flow trajectories. However, in link 6, the slopes increase with time and later decrease. Consequently, congested time-location pairs correspond to points in the graph with a high relative slope with respect to the free flow slope.

For example, point B in the figure corresponds to congestion because the slope of the trajectory passing throughout it has a slope larger than the slope of point A. On the contrary, point C is a non-congested time-location pair, because its slope is similar to that of point A.

In this case, link 6 is congested and produces a speed reduction, which is practically maintained constant with time in link 8.

If users are forced to travel with departures in the range from 10 to 15 hours, the figures permit to choose the best trajectory in terms of travel time, using the right and left curves, which indicate that the optimal option is between 12 and 13 hours and depends on the user class (see the marked trajectories with departure time between 10 to 15 hours in each of the three plots in Figure 12.7), which correspond to a relative minimum of the travel time excess curves.

Suppose a motorbike user wants to arrive at a given hour, say 30 h, then the figure permits to calculate the travel time (8 h) and the departure time, which in this case is 22 h. However, the figure also allows us to see that a delay of one hour in the arrival time (31 h) implies a departure at 24 h, that is, a saving of one hour in the travel time (see the dashed lines in the lower plot in Figure 12.7).

It is important to remark that congestion corresponds to large slope trajectories (low speed) but that large slope trajectories do not necessarily imply congestion, because the free flow speed can be small. So, what is important is the *relative slope* small slope compare to slope at other times in the same location.

Finally, to illustrate the role and physical meaning of some of the partial derivatives described in Section 12.2, the velocity, acceleration and $\dfrac{\partial^2 g(x,t_0)}{\partial x \partial t_0}$ (slowness promptness rate) contours are shown on the cars, trucks and motorbikes plots, respectively. In the cars and trucks plots we can see the velocities and accelerations (in relative values), respectively, associated with the different location-time pairs, what provides a valuable information about the traffic behavior. In the motorbikes plot we can see the relative changes of $\dfrac{\partial^2 g(x,t_0)}{\partial x \partial t_0}$ which provide information about how the vehicles joint (negative value contours) or separate apart (positive value contours) at the different location-time pairs.

12.3.3 Single class equal flow trajectory profiles under congestion

In this interpretation the number of users between two consecutive trajectories is the same as indicated in Figure 12.8.

If as before we assume that the FIFO (first in first out) rule is satisfied, from Figure 12.8 we can see that the traffic intensities at the two locations, the path origin O and location A, associated with the two bands of users are

$$I_1 = \frac{n}{t_{01}}; \quad I_2 = \frac{n}{t_{02}}; \quad I_3 = \frac{n}{t_3}; \quad I_4 = \frac{n}{t_4}, \tag{12.10}$$

where n is the common number of users of the two bands.

This means that the traffic flow intensities for users traveling from O to A change from I_1 to I_3 and from I_2 to I_4 for users of the first and second band, respectively. Thus, to compare the traffic intensities of the two bands we can use the intensity ratio:

$$r = \frac{I_4}{I_3} = \frac{t_3}{t_4}, \tag{12.11}$$

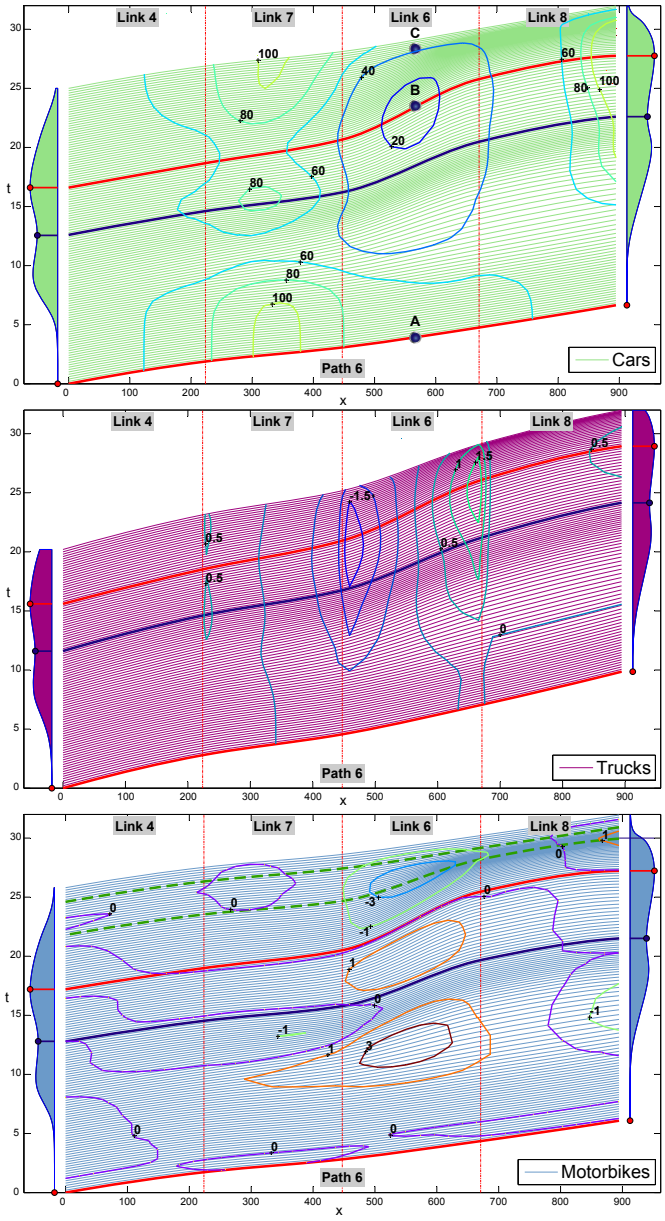


Figure 12.7: Trajectories of vehicles of the three classes: cars, trucks and motorbikes corresponding to equally delayed departure times (12 minutes), and travel times of the different trajectories located at the departure and arrival locations. The fastest and slowest trajectories are shown together with an alternative for a user willing to reach the destination at 31 h. The velocity, acceleration and slowness promptness rate contours are shown on the cars, trucks and motorbikes plots, respectively.

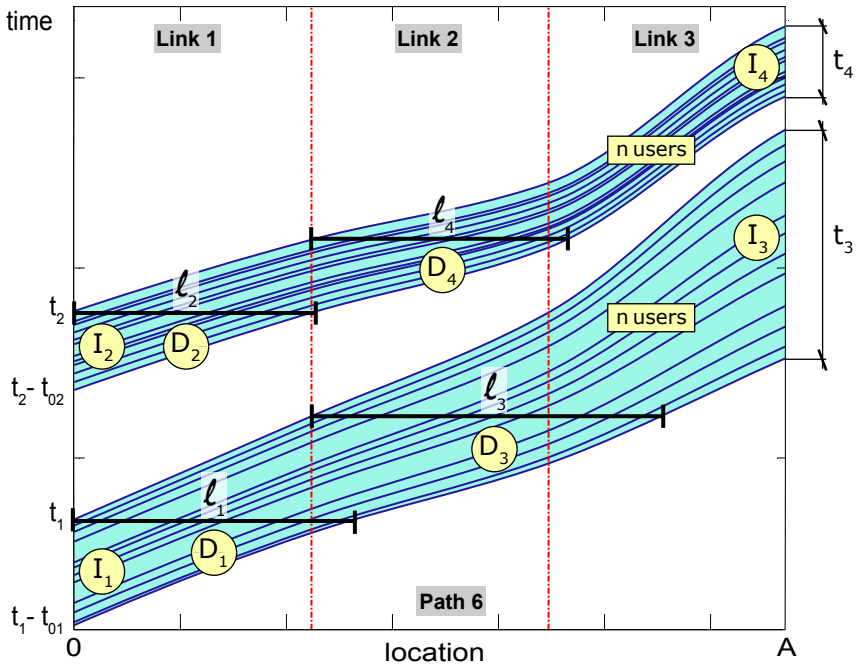


Figure 12.8: Illustration of the meaning of traffic intensity ratios for two trajectory bands in the case of equal flow trajectories.

showing that this type of plots is useful to interpret relative intensities of different bands.

For example, if $t_3 > t_4$ ($r > 1$) the intensity of band 2 is larger than the intensity of band 1 ($I_4 > I_3$). In other words, the lighter the zone, the smaller the traffic intensity ratios.

However, according to (12.10), we have

$$R = \frac{I_4/I_2}{I_3/I_1} = \frac{r_2}{r_1} = \frac{t_{02}t_3}{t_{01}t_4},$$

which shows that the ratio $R = t_3/t_4$ is not useful to compare the traffic intensity ratios for different departure intervals of the same flow.

Similarly, if as before we assume that the FIFO (first in first out) rule is satisfied, from Figure 12.8 we can see that the traffic densities at the two locations associated with the two bands of users are

$$D_1 = \frac{n}{\ell_1}; \quad D_2 = \frac{n}{\ell_2}; \quad D_3 = \frac{n}{\ell_3}; \quad D_4 = \frac{n}{\ell_4}. \tag{12.12}$$

Then, we get

$$\frac{D_3}{D_1} = \frac{\ell_1}{\ell_3}; \quad \frac{D_4}{D_2} = \frac{\ell_2}{\ell_4}; \quad \frac{D_2}{D_1} = \frac{\ell_1}{\ell_2}; \quad \frac{D_4}{D_3} = \frac{\ell_3}{\ell_4}. \tag{12.13}$$

So, the horizontal length ratios $\frac{\ell_i}{\ell_j}$ are good to compare traffic densities within the same or different bands.

Figure 12.9 shows the separate trajectories of vehicles of the three classes: cars, trucks and motorbikes corresponding to equal number of vehicles in each band and the travel times of the different trajectories located at the departure and arrival locations. The trajectories corresponding to the maximum and minimum travel times are indicated by thick lines.

It is important to note the difference between Figures 12.7 and 12.9, which correspond to equally delayed and same flow bands, respectively, because the interpretation of the plots must be done adequately.

As expected, in Figure 12.9 the assumed mixture of normal densities for the flow demand at the origin of the route becomes apparent and is propagated through the path though modulated by the interaction with other flows and mainly because of the congestion.

When looking to the trajectory plots we can perceive several things, such as the trajectory slope, the trajectory curvature, the concentration of trajectories in a given zone, the number of vehicles traveling between consecutive trajectories, the time interval associated with two consecutive trajectories, etc. However, we must be careful when interpreting the different trajectory profiles mentioned in this paper: equally delayed or equal flow trajectories. For example, in the equally delayed departure plots the number of users traveling between consecutive trajectories are not the same, and the light and dark areas (concentration of trajectories in a given zone) cannot be interpreted as areas of low and high density or intensity, respectively.

12.4 Double class trajectory plots

Since interactions among different classes cannot be easily detected by separate class trajectories, in this section we combine several classes. This type of plots consists of trajectories (traveled distance x versus time t) of two different class users. Since it has been assumed that there is no overtaking among users of the same class but overtaking is permitted for different class users, the set of trajectories does not intersect for the same class, but produces intersections for users of different classes.

In order to see how the traffic behaves with respect to the different class users, in Figure 12.10 the trajectories of vehicles of the three classes: cars, trucks and motorbikes are superposed with those of a virtual class of users who are assumed to circulate at the free flow speed of each class and with the same departure times as the regular users. The travel time excesses with respect to free travel time of the different trajectories are shown at the departure and arrival locations, and the fastest and slowest trajectories are shown too. In other words, we compare each class trajectory with its corresponding free flow

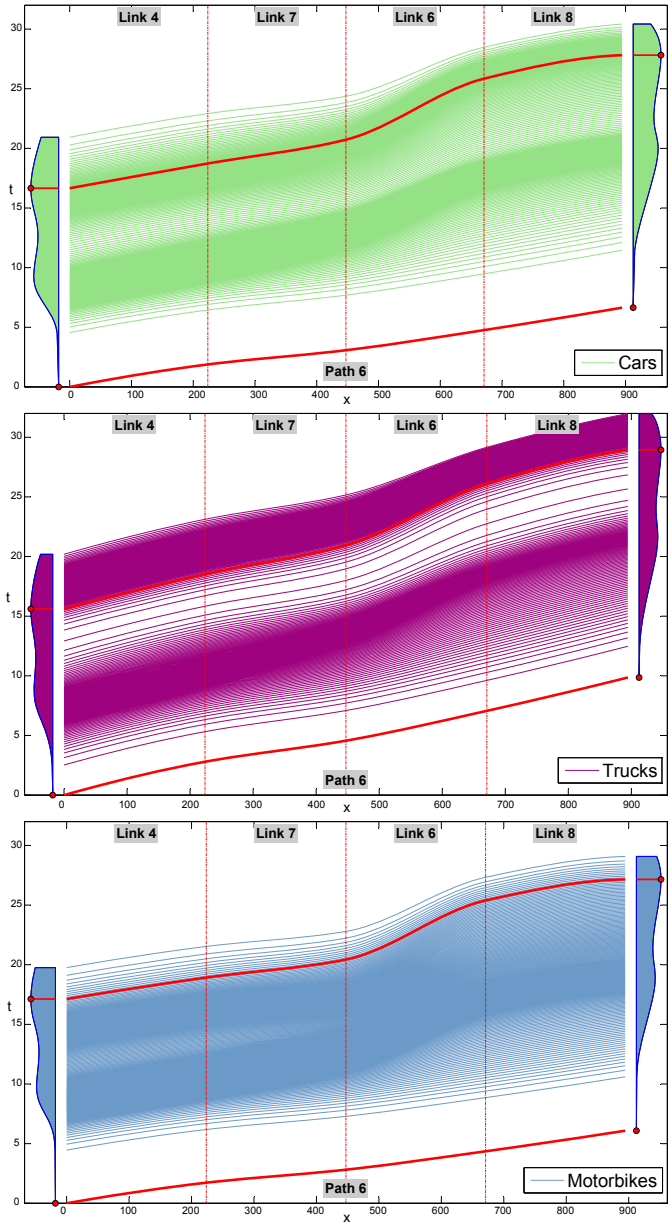


Figure 12.9: Trajectories of vehicles of the three classes: cars, trucks and motorbikes corresponding to equal number of vehicles in each band, and travel times of the different trajectories located at the departure and arrival locations. The fastest and slowest trajectories are shown.

trajectory.

It is worthwhile mentioning that some white bands appear in the pictures, whose

meaning is explained next. Since the two sets of trajectories (the one corresponding to the user and the associated free flow set) are plotted on a white paper, when these trajectories intersect the white background is shown. Thus, each white band corresponds to a virtual overtaking of a vehicle of the corresponding class user by the virtual vehicle circulating at the free flow speed. Consequently, the number p of white bands intersected by a given trajectory indicates that the corresponding vehicle would have been overtaken by all virtual vehicles with a departure produced p times 12 minutes after its departure time (because trajectories correspond to users delayed 12 minutes).

Since motorbikes are affected by congestion less than cars and cars less than trucks, the number of white bands present in the corresponding plots in Figure 12.10 increase from motorbikes to cars and trucks.

Figure 12.11 shows the superposition of trajectories of vehicles of the three classes: cars, trucks and motorbikes and free flow speed truck trajectories corresponding to equally delayed departure times (12 minutes), and travel times of the different trajectories located at the departure and arrival locations. The fastest and slowest trajectories are also shown.

Since motorbikes have been assumed faster than cars and cars faster than trucks, the number of white bands present in the corresponding plots in Figure 12.11 increase from motorbikes to cars and trucks. Thus, in order to optimize, a trajectory must intersect as least as possible white bands (the least times you need to virtually overtake, the best).

Figure 12.12 shows the superposition of the trajectories of vehicles corresponding to all combinations of two classes for equally delayed arrival times and *with the same departure times for both classes*. In addition, the trajectories of five pairs of class vehicles departing at the same time are emphasized in order to illustrate their different travel times all over the path.

Note that forcing the departure times of the different class users to coincide implies forcing a vertical white band at departure (where the different class vehicles coincide at the same time).

Since in this case we do not use the virtual free flow trajectories but the real trajectories of two classes of users, these plots refer to relative behavior of the corresponding classes and must be used with this aim. In particular, the white bands correspond to overtaking between the associated classes.

It is easy to see that some trajectories imply a high number of overtakes and that other produce a reduced number because of the congestion, in which the speeds of all class users coincide or almost coincide (parallel or almost parallel trajectories). One example of this occurs in link 6 (the congested link) where practically no overtake occurs, with the exception of early and late hours.

Note that the number of white bands is related to the different speeds of the two corresponding classes. Since the maximum speed difference is between trucks and motorbikes, the intermediate plot shows the largest number of white bands.

We point out that congestion means that the white bands deform (see link 6), while

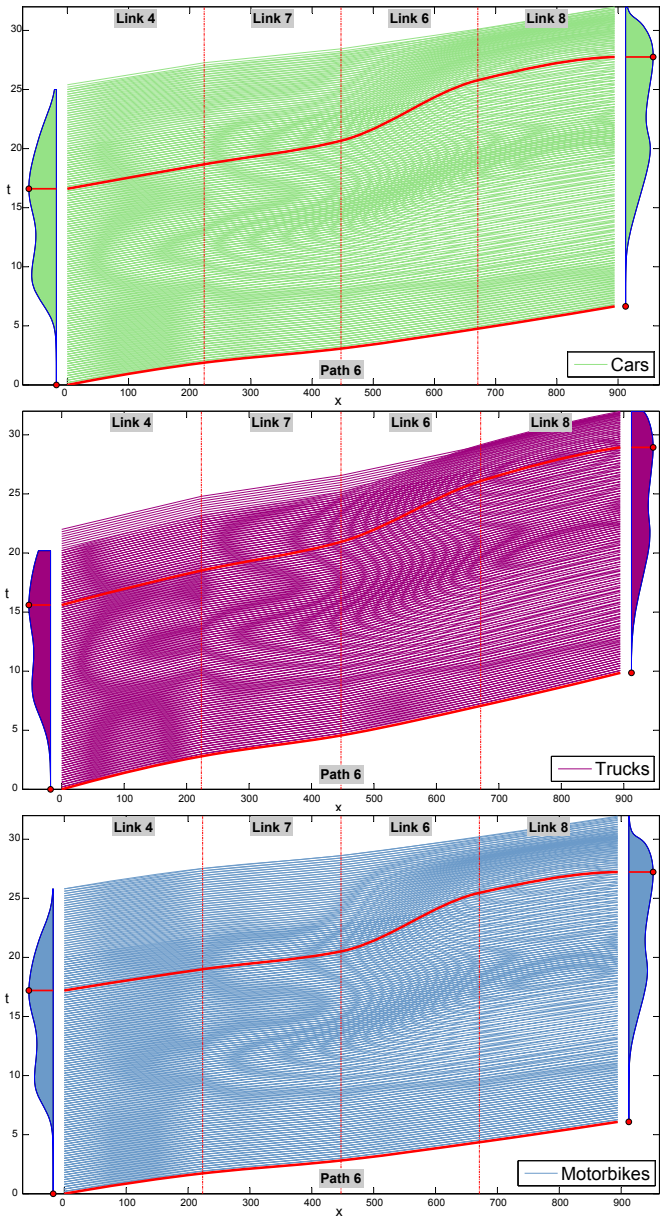


Figure 12.10: Superposition of trajectories of vehicles of the three classes: cars, trucks and motorbikes and their free flow speed trajectories corresponding to equally delayed departure times (12 minutes), and travel times of the different trajectories located at the departure and arrival locations. The fastest and slowest trajectories are shown.

lack of congestion implies maintaining the shapes (see links $4, 7$ and 8).

Figure 12.13 shows the superposition of the trajectories of vehicles corresponding to

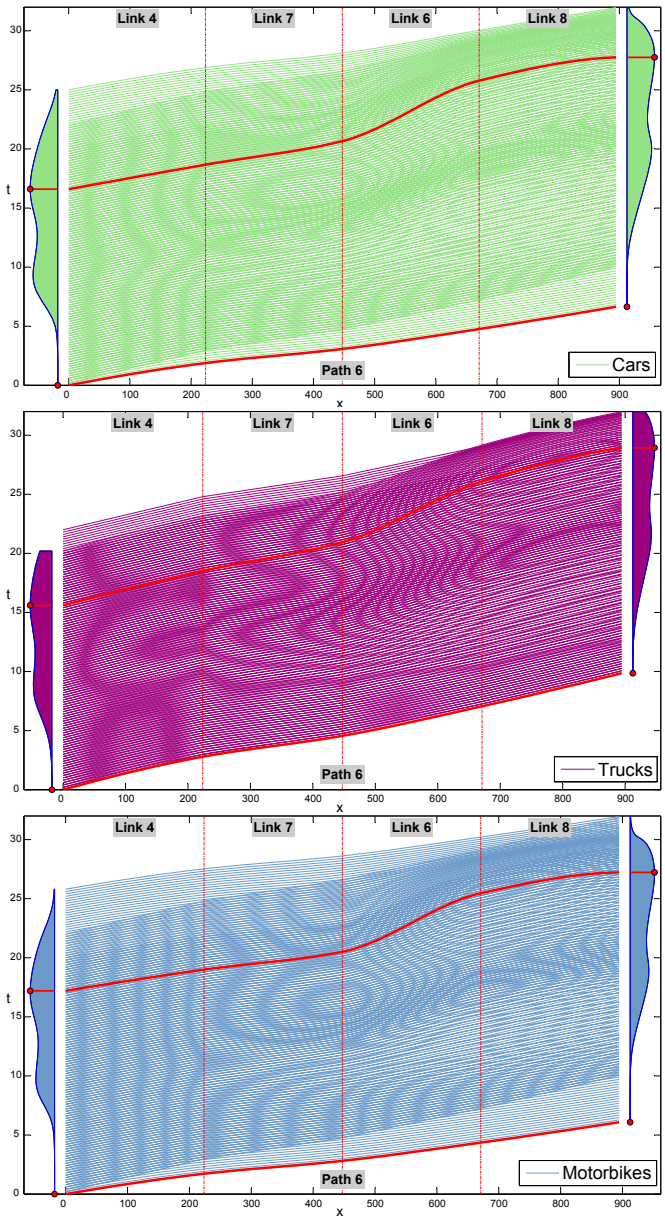


Figure 12.11: Superposition of trajectories of vehicles of the three classes: cars, trucks and motorbikes and free flow speed truck trajectories corresponding to equally delayed departure times (12 minutes), and travel times of the different trajectories located at the departure and arrival locations. The fastest and slowest trajectories are shown.

all combinations of two classes for equally delayed departure times (12 minutes) and *with the same arrival time for both classes*. In addition, the trajectories of five pairs of class

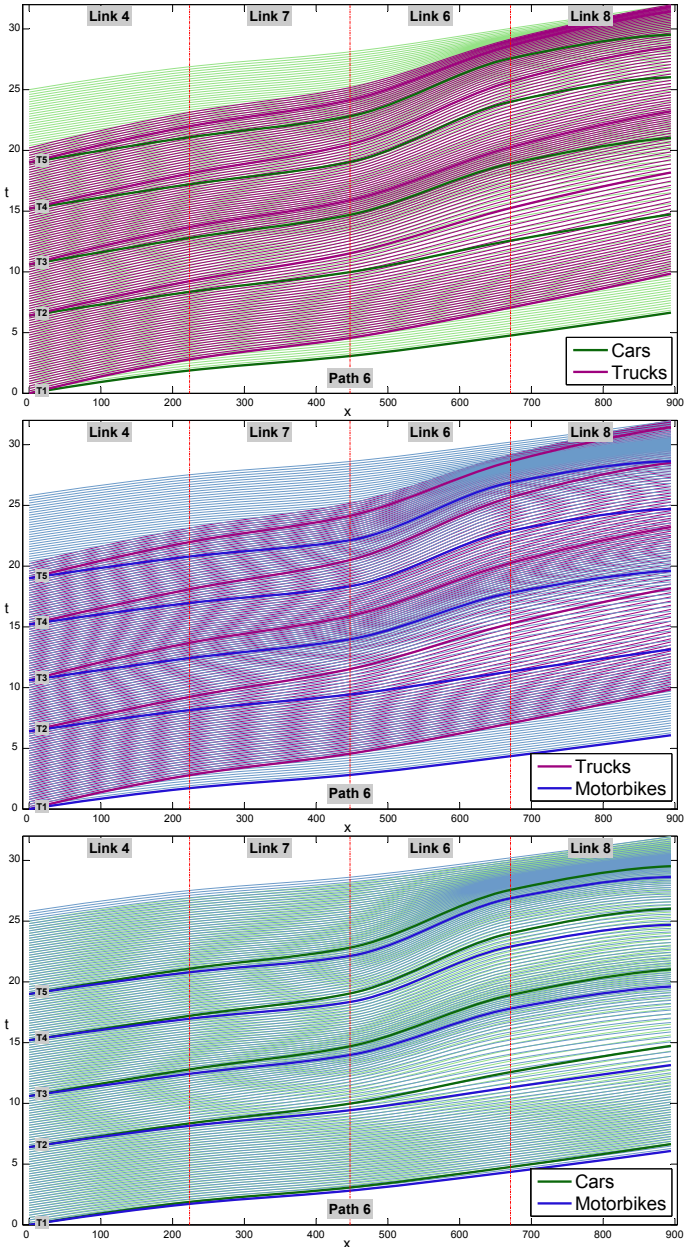


Figure 12.12: Superposed trajectories of vehicles of all combination of two classes corresponding to equally delayed departure times (12 minutes). The trajectories of some pair of class vehicles departing at the same time are emphasized.

vehicles arriving (reaching the path end) at the same time are emphasized in order to illustrate their different travel times all over the path.

The conclusions are similar to those derived from the previous figure but now a white

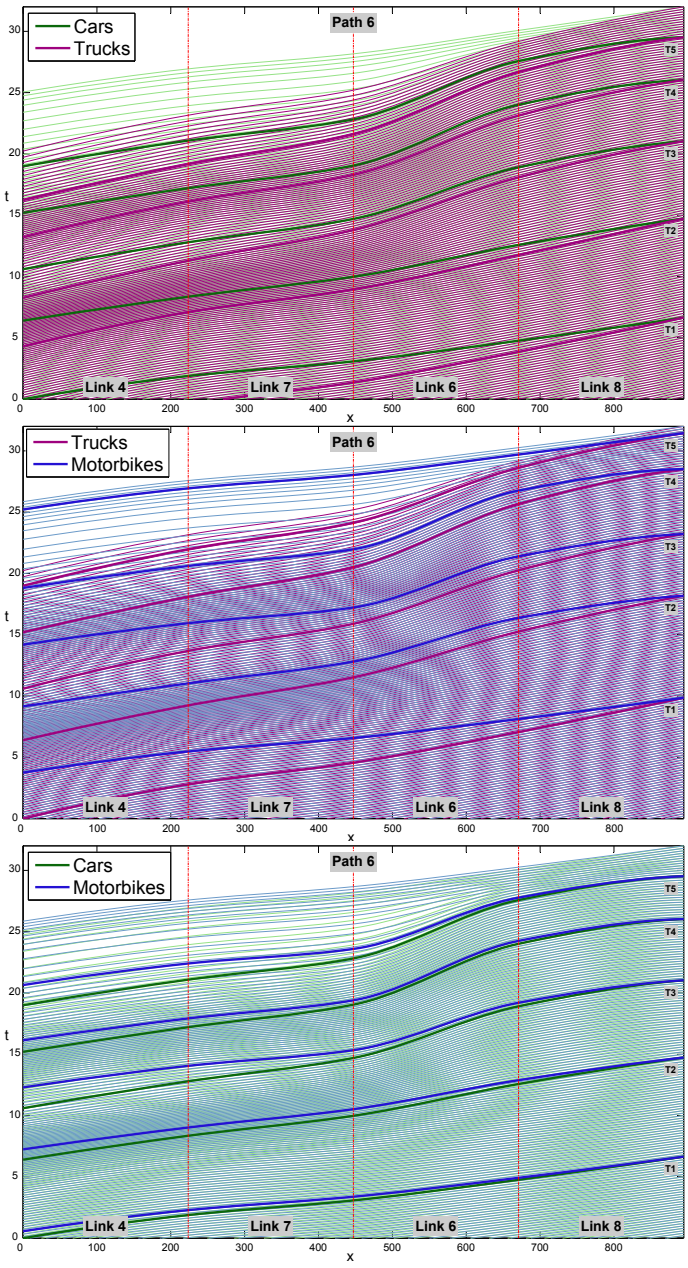


Figure 12.13: Superposed trajectories of vehicles of all combination of two classes corresponding to equally delayed arrival times. The trajectories of five pairs different class vehicles arriving at the same time are emphasized.

band has been forced at the arrival (where the different class vehicle coincide at the same time).

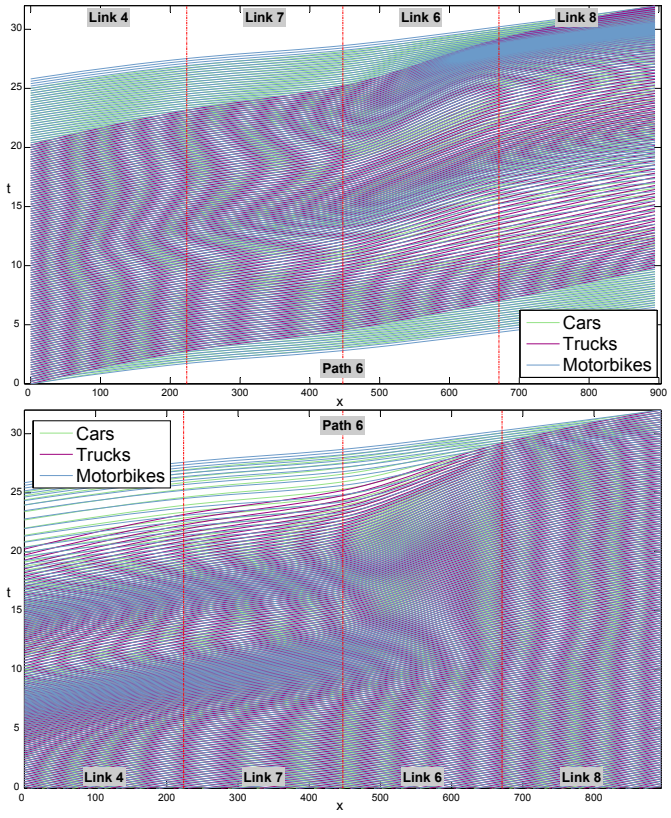


Figure 12.14: Superposed trajectories of vehicles of the three classes corresponding to equally delayed departure (upper plot) and arrival (lower plot) times.

12.5 Multiple class trajectory plots

This type of plots consists of trajectories (traveled distance x versus time t) of all different class users. Since it has been assumed that there is no overtaking among users of the same class but overtaking is permitted for different class users, the set of trajectories does not intersect for the same class, but produces intersections for users of different classes.

Figure 12.14 shows the superposition of the trajectories of the three classes of vehicles corresponding to equally delayed departure (upper plot) and arrival (lower plot) times.

Since three classes of users are considered, three overtake types appear (motorbikes overtake cars, motorbikes overtake trucks and cars overtake trucks). For this reason the light bands have now different colors. A white band indicates that motorbikes and cars simultaneously overtake trucks, a green band indicates that motorbikes overtake trucks, and a magenta band indicates that motorbikes overtake cars.

12.6 Final recommendations

Finally, we provide the following recommendations to be considered when plotting and interpreting traffic graphs:

1. Be careful in distinguishing individual trajectories from group trajectories and how the groups have been defined. For example, the interpretation is different if equally delayed or same number of users are the grouping criteria, as it has been shown in this paper.

2. Plotting many different lines in a single plot causes confusion. Thus, sometimes it is better to represent the trajectories of some representative users or group them.

3. Look at the slopes and curvatures of single trajectories to detect changes in speed, slowness and acceleration.

4. Look at the horizontal and vertical distances and their evolutions between consecutive trajectories when equally delayed plots are used to determine the effect of earlier or later departure (promptness).

5. Compare trajectories, slopes and curvatures of close users to see the time and location evolution of traffic.

6. Look at overtaking bands to decide the best times to travel. The larger the number of bands crossed, the smaller the travel times.

Part IV

Conclusions and Future Work

Chapter 13

Conclusions, future work and publications

Contents

In this chapter we will enumerate the conclusions than can be drawn form the models developed in the thesis. Moreover, some possible lines of future work are included.

It is important to remark that the computer programs have been implemented without a sufficient optimization effort to reduce the cpu times. Therefore, the programs should be

reviewed in depth in order to implement more efficient programs. For instance, they could be written in different computer languages to compare their performances, and parallel computing could be implemented.

13.1 A Percentile Traffic Assignment model

This section provides some concluding remarks and some possible future work related to the topics dealt with in Chapter 7, where the Percentile Traffic Assignment problem is treated.

13.1.1 Conclusions

1. The Nie (2011) conjecture on permutability of partial derivatives of route travel times with respect to route flows and percentiles does not hold, and then it cannot be used as a simplification.

2. A location-scale statistical model which makes assumptions at the path or OD instead of the link level, has been proposed. In particular, the route or OD travel times are assumed random variables of a infinitely divisible location-scale family. However, the path travel time means and variances are evaluated in terms of the mean and variance-covariance matrices of link travel times, allowing the consideration of dependent link travel times. This alternative model requires neither direct assumption of normal distributions for link travel times nor use of the central limit theorem. This assumption could improve the practical applicability of some path based existing models.

3. A Percentile System Optimal (PSO) model has been proposed, where the sum of the α-percentiles of the total travel time for each OD-pair is minimized.

4. Contrary to most existing models that require path enumeration or an iterative method to add paths sequentially, we have presented a PSO alternative in its two versions: (a) with and (b) without path enumeration. In fact, by solving the (b) version we can identify very easily the set of paths used by any class user.

5. The proposed methods have been applied to solve two practical examples: the Nguyen-Dupuis and the Ciudad Real network. These examples show that the method is applicable not only to simple but to complex and real size networks.

6. For not very large networks standard packages, such as GAMS, can be used to solve the problem without the need of developing ad hoc special computer programs or methods.

13.1.2 Future work

The following are two possible lines of future work:

PUE model without path enumeration. Because of its difficulty, a Percentile User Equilibrium model without path enumeration has not been proposed yet in the literature. We have proposed a PSO model but it would be interesting to develop also models based on the ideas of Lo et al. (2006) and Nie (2011) that does not require path enumeration and, hence, their practical applicability can be improved.

Dynamic Percentile Traffic Assignment. It would be convenient to propose models that deal with the problem of Dynamic Percentile traffic assignment extending the work presented in this thesis and in Lo et al. (2006) and Nie (2011).

13.2 A Traffic Assignment problem including overtaking classes

This section provides a concluding overview and some possible future work related to the models developed in Chapter 8, where the traffic assignment problem including overtaking has been dealt with by means of optimization and Variational Inequality (VIP) problems.

13.2.1 Conclusions

1. Vehicle overtaking can be taken into account in traffic models at macro level in a simple form by adequately modifying the link travel time functions.

2. A link travel time family of functions has been proposed that permits reproducing the same asymptotic congestion behavior of several overtaking classes, while reproduces different class link travel times under mild congestion.

3. The link travel time family of functions proposed is based on convex linear combinations of link travel time functions and, thus, any desired link travel time function can be applied. Furthermore, it has a large number of parameters that makes it applicable to a wide range of problems.

4. Two nonlinear complementary (NCP) and variational inequality (VIP) equivalent problems, one with and another without route enumeration have been proposed to solve the traffic assignment problem considering different overtaking classes.

5. Alternative problems have been proposed that can advantageously replace the original NCP and VIP problems.

6. The proposed models (8.40)-(8.44) and (8.45) subject to (8.40)-(8.42) and (8.44) do not require route enumeration and permit identifying the used routes by all class users.

7. The reported examples prove that different class users use different routes in order to fulfill their overtaking tendencies.

13.2.2 Future work

Some possible future works related to this line of research, i.e. to traffic assignment problem including overtaking classes, are:

Estimation and validation of the proposed link travel time function. It is important to validate and test the proposed link travel time function with real data to determine if it really can take into account the overtaking behavior. It would be also interesting to develop a procedure that permits estimating the parameters of this new link travel time function. Some general ideas on this topic has been given in Section 8.2.2 but a more specific procedure should be developed.

Percentile traffic assignment problem including overtaking classes. As already seen, most models in the literature assume that users behave in a homogeneous way. However, this is generally not the case and much more realistic and, hence, complicated models should be developed. One interesting possibility is to take into account the travel time reliability problem and the overtaking classes at the same time, that is, to develop a traffic assignment model including users that choose their paths according to the reliability (i.e., the variability) and overtaking possibilities of those paths.

13.3 A Bayesian Matrix Estimation Model

This section provides a concluding overview some suggestions about future work related to Chapter 9, where the Matrix Estimation (ME) problem has been treated.

13.3.1 Conclusions

1. The problem of traffic flow estimation based on Gamma variables in a Bayesian context is a hierarchical optimization problem in which five different objective functions are used. The hierarchy of four of these functions is clear, but two of the functions can be given different hierarchies depending on the relative weight given to the prior information with respect to the field observations.

2. A simple Gamma $G(\theta, 1)$ model for route flows, which has interesting properties in the sense that the random link, node and OD flows belong to the same family of Gamma distributions, has been proposed.

3. A Bayesian method with its most general conjugate family of prior-posterior distributions has been utilized for learning the Gamma model. This leads to very simple and closed formulas for the exact updating of uncertainty.

4. The difficulties in calculating the posterior means is avoided by considering the posterior modes instead, which can be easily calculated by standard optimization procedures.

5. The Gamma model combined with the Wardrop-minimum variance optimization problem permits assigning traffic flows without the need of enumerating route paths of the different OD pairs using a multi-level technique.

6. A multi-level algorithm has been proposed as an efficient alternative to solve the hierarchical optimization problem discussed in this paper.

7. Since the computational requirements for the examples shown are low, because the GAMS software is very powerful, and the implied functions behave well, we should not expect computational problems for moderately large networks.

8. The methods have been applied to two examples of applications for illustration purposes and compared with a standard bi-level LS method. The results of this analysis give close results, which seem to validate the proposed method.

13.3.2 Future work

The following lines of future research on this topic are suggested:

Bayesian Matrix Estimation Model assuming other distributions. An interesting line of future research is to assume that the path travel times belong to other families of distributions different from the Gamma. For example, the Generalized Gamma distribution would be a good option as it includes the Weibull, Gamma and Log-normal distributions as particular cases and it is then suitable for a large range of scenarios.

Extend the bi-level methods to traffic assignment with heterogeneous users. Bi-level methods were developed in order to take into account the congestion effect in the estimation procedure. It would be interesting to introduce the new traffic assignment equations with heterogeneous users (travel time reliability or overtaking) in the lower level of the bi-level problem. In that case, more realistic estimations could be obtained.

13.4 Upper bound of the number of sensors required for total observability

This section provides a concluding overview and some possible future work related to the topics dealt with in Chapter 10, where the link observability problem is treated.

13.4.1 Conclusions

The main conclusions are:

1. The upper bound given by Ng (2012) is right, but can be improved using partial path information (a subset of linearly independent paths).

2. Path information is needed to know the smallest number of link sensors to be installed to infer the flows of all other non-equipped links in a traffic network.

3. Not all paths are required to determine the upper bound. If a set of paths with rank $m-n$ is already available, no more paths are needed to get this sharpest upper bound, which is $m-n$.

4. The upper bound given by Ng (2012) is the sharpest one only when the rank of the link-path incidence matrix is $r_w = m-n$. In this case, the transformations providing inferred non-basis link flows from basis link flows obtained from the node-link balance matrix (10.1) and from the link-path incidence matrix (10.10) are identical.

5. As indicated by other authors (see, for example Hu et al. (2009)) the minimum number of links to be equipped with sensors in order to ensure observability of all link flows is the rank r_w of the link-path incidence matrix $\mathbf{A}$, which depends on m and n only when it reaches its maximum possible value $r_w = m - n$.

6. We have provided a method to obtain minimum subsets of linearly independent path vectors that facilitates the application of the proposed method to real cases. We note that this set is very limited and that at most it coincides with the number of links.

7. Though the upper bound produced by the node-based approach can be close to the real bound produced by the proposed method, the fact that they are different is relevant because observation of more links than those strictly necessary produce redundancy and this is known to lead to incompatibility problems. In other words, if we observe the number of links indicated by the node-based approach we can have incompatibility problems due to redundant observations. On the contrary, if we observe the number of links indicated by the path-based approach we will not have incompatibility problems.

13.4.2 Future work

Some suggestions for future work are:

Dynamic observability problem. In Chapter 10 we have assumed static flow, that is, we assume that the network conditions considered are static. Future research is needed for the dynamic case, which is much more complicated.

New methods for the observability problem. We have developed an algebraic model to solve this problem. However, it would be interesting if other types of models (such as topological or graphical) are available.

13.5 Continuous Dynamic Network Loading model with Different Overtaking Class Users

This section provides some concluding remarks and some possible future work related to the topics dealt with in Chapter 11, where the a Continuous Dynamic Network Loading model including overtaking classes is proposed.

13.5.1 Conclusions

The main conclusions drown from this chapter are:

1. The proposed method is a continuous method in which different class users from the point of view of travel speed and overtaking are considered.

2. The link exit-entry time functions $\tau_{ij}^{\alpha}(t)$ are approximated by monotone cubic-spline functions and path origin flow functions are assumed to be linear combinations of a basic set of functions.

3. The model permits determining how the flow waves associated with different class users progress throughout the links of the network and stretch or enlarge due to congestion.

4. The model takes into account that overtaking is not permitted for high congestion levels by considering a family of link travel time functions with a common asymptote.

5. The family of link travel time functions has been assumed to be based on the BPR function, but other link travel time functions can be used, depending on the nature of the network.

6. Instead of using different functions for determining the origin departure times of users of different paths, they are evaluated recursively from link exit-entry time functions.

7. Link physical-queues are considered to take into account high congestion levels.

8. Monotone cubic splines are used for reproducing the link exit-entry time functions. This guarantees satisfaction of the FIFO rule for the same class users at the interpolated points if the base points already satisfy this rule.

9. The treatment of traffic congestion can be easily done by means of an iterative process: first, we evaluate traffic congestion based on the actual link travel time functions, and later, we update the link travel time functions based on congestion. The process can be started by assuming no congestion in the first iteration, and correcting this in successive iterations. We have obtained convergence in 3 or 4 iterations for no congestion and no more than 15 iterations when some congestion is present.

10. Because path flow intensities are used as the basic time functions, and the link and node flow intensities are obtained from them based on the network topology, the conservation laws are satisfied over all the network.

13.5.2 Future work

The following lines of future research on this topic are suggested:

Estimation of the parameters. The parameters of the proposed model need to be estimated and the model tested with real data. To this end, the standard counter information is not valid and observability methods (see Castillo et al. (2008a), Castillo et al. (2010)) and more sophisticated ones, such as the plate scanning (Mínguez et al. (2010), Castillo et al. (2008d)), GPS or mobile phones (Caceres et al. (2007), Herrera et al. (2010)) based techniques must be used.

Calibration of the model. Future research is needed to reproduce the link congestion and calibrate the model.

13.6 Graphical Methods to Analyze Traffic Trajectories with and without Overtaking

This section provides a concluding overview and some possible future work related to the topics dealt with in Chapter 12, where some graphical methods to analyze traffic trajectories with and without overtaking are proposed.

13.6.1 Conclusions

The main conclusions are:

1. Several new concepts and measures are required to fully understand the time and spatial traffic behavior. Since users are not familiar with these concepts, an effort is required to fully understand their physical meaning and how these measures can be derived from the different trajectory plots.

2. User trajectories are an important source of information about the dynamic traffic behavior because they permit identifying relevant aspects of traffic flow and how they change with time and location.

3. The magnitude of the first derivatives of the $f(t, t_0)$ and $g(x, t_0)$ functions can be obtained by taking a look at the trajectory slopes and their time evolution. This means that speeds, slowness, slowness distance and slowness time rates can be identified at all location-time pairs from trajectory plots.

4. The signs and orders of magnitude of the second partial derivatives of $f(t, t_0)$ and $g(x, t_0)$ functions with respect to t_0 can be derived only from equally delayed plots.

5. The signs and orders of magnitude of all other second partial derivatives of $f(t, t_0)$ and $g(x, t_0)$ can be derived from both, equally delayed and equal flow plots.

6. Plotting the contours of the different partial derivatives of functions $f(t, t_0)$ and $g(x, t_0)$ facilitates the interpretation of the plots and permits understanding the traffic behavior and how it evolves with time and location.

7. Superposition of the trajectories of different class users provide a useful information about the overtaking possibilities at different times and locations by means of the appearing bands.

13.6.2 Future work

The following are two possible lines of future work:

Continue analyzing trajectory plots. Much more work is needed to fully understand the information reported by trajectory plots.

Selection of appropriate terms. Work is needed in providing appropriate names to the different concepts and measures introduced and to clarify how they can be useful in traffic trajectory interpretation.

13.7 Publications from this thesis

The results of this thesis have been published or accepted for publication in the following International Journals:

- Castillo, E., Calviño, A., Nogal, M., Sánchez-Cambronero, S. and Rivas, A. A probabilistic user equilibrium approach. Solving a conjecture on permutability of partial derivatives of route travel times with respect to route flows and percentiles. *Submitted*.

- Castillo, E., Calviño, A., Sánchez-Cambronero, S. and Lo, H. K. (Revision). A Multi-class User Equilibrium Model Considering Overtaking across Classes. *IEEE Transactions on Intelligent Transportation Systems*.

- Castillo, E., Menéndez, J.M., Sánchez-Cambronero, S., Calviño, A. and Sarabia, J. M. (2012). A hierarchical optimization problem: Estimating traffic flow using Gamma random variables in a Bayesian context. *Computers and Operations Research*, In press.

- Castillo, E., Calviño, A., Menéndez, J. M., Jiménez, P. and Rivas, A. (Accepted) Deriving the upper bound of the number of sensors required to know all link flows in a traffic network. *IEEE Transactions on Intelligent Transportation Systems*.

- Castillo, E., Nogal, M., Calviño, A., Rivas, A. and Lo, H. K. (Revision). A Model for Continuous Dynamic Network Loading Problem with Different Overtaking Class Users. *Journal of Intelligent Transportation Systems*.

- Castillo, E., Nogal, M., Calviño, A., Sánchez-Cambronero, S. and Lo, H. K. Some Graphical Methods to Analyze Traffic Trajectories with and without Overtaking. *Submitted*.

Part V

Appendix

Appendix A
Program codes

Contents

A.1 Gams codes for the Probabilistic System Optimal model

This section deals with the GAMS implementation of the Probabilistic System Optimal model presented in Chapter 7, Section 7.3. The codes for the models with and without overtaking are shown for the Nguyen-Dupuis example.

```
$Title PSONGUYENDUPUIS

*******************************************************************************
* The output files are defined
*******************************************************************************
file out/PSONGUYENDUPUIS.out/;
file out1/PSOMathematicaNGUYENDUPUIS.out/;
file out2/PSOTablasNGUYENDUPUIS.out/;
put out;

*******************************************************************************
* The solver is chosen
*******************************************************************************
OPTION nlp=IPOPT;
```

```
SCALAR
CRITICALFLOW     critical value for a non-null flow /0.001/
iii              auxiliary counter
cbound           bound for the objective function
Qaref            parameter used to make dimensionless capacities
Tref             parameter used to make dimensionless times
nlinks           number of links
ilink            counter for the links
nOD              number od OD pairs
iAA              counter for the alpha-classes
iOD              counter for the OD pairs
aux              auxiliary scalar
mm               auxiliary scalar
indic            auxiliary scalar
gamma            BPR parameter /2/
incre            epsilon value /0.001/
done             binary scalar indicating the convergence
;

SETS
iter             set of iterations /1*20/
PATH             set of paths /1*46/
SS               set of links in a path /1*5/
AA               classes /1*4/
I                set of nodes /1*13/
Origin(I)        set of origin nodes
Destination(I)   set of destination nodes
LINK(I,I)        set of links
LINK1(I,I)       copy of the set of links
OD(I,I)          set of OD pairs
OD1(I,I)         copy of the set of OD pairs
;

ALIAS(I,J,K,S,I1,J1);
ALIAS(AA,AA1);
ALIAS(PATH,PATH1);

TABLE
pathlinks(PATH,SS) paths by its links
       1   2   3   4   5
1      1  11  14  18   0
2      1  11  14  18  20
3      1  11  14  19  30
4      1  11  14  19  31
5      1  11  15  29  30
6      1  11  15  29  31
7      1  12  25  29  30
```

```
8       1  12  25  29  31
9       1  12  26  37   0
10      2  35  14  18   0
11      2  35  14  18  20
12      2  35  14  19  30
13      2  35  14  19  31
14      2  35  15  29  30
15      2  35  15  29  31
16      2  36   0   0   0
17      2  36  20   0   0
18      3  21  17  13  10
19      3  22   0   0   0
20      3  22  34   0   0
21      4  32  17  13  10
22      4  33  27  13  10
23      4  33  28  23   0
24      4  33  28  24  10
25      5  32  17  13   9
26      5  32  17  13  10
27      5  32  17  16   0
28      5  32  17  16  34
29      5  33  27  13   9
30      5  33  27  13  10
31      5  33  27  16   0
32      5  33  27  16  34
33      5  33  28  23   0
34      5  33  28  24  10
35      6  38  23   0   0
36      6  38  24   9   0
37      6  38  24  10   0
38      7  11  14  18  20
39      7  11  14  19  31
40      7  11  15  29  31
41      7  12  25  29  30
42      7  12  25  29  31
43      7  12  26  37   0
44      8  25  29  30   0
45      8  25  29  31   0
46      8  26  37   0   0
;

PARAMETER
XX(I,J)                 flow on a link
pathorigin(PATH)        origin node of a path
pathdestination(PATH)   destination node of a path
order(I,J)              number associated with a link
classalpha(AA)          alpha-percentiles of the standard normal for the classes
/1 -1.64
2 0
```

```
3 1.64
4 1.96/
traveltime(I,I,AA)       traveltime of a OD associated with a class
eta(I,J)                 BPR function  parameter
delta(I,I)               Kronecker delta
nu(I,J)                  free link travel time
/ 1.5         7
  1.12        9
  2.8         9
  2.11        9
  3.11        8
  3.13       11
  4.5         9
  4.9        12
  5.1         7
  5.4         9
  5.6         3
  5.9         9
  6.5         3
  6.7         5
  6.10        5
  6.12        7
  7.6         5
  7.8         5
  7.11        9
  8.2         9
  8.7         5
  8.12        14
  9.4         12
  9.5         9
  9.10        10
  9.13        9
  10.6        5
  10.9        10
  10.11       6
  11.2        9
  11.3        8
  11.7        9
  11.10       6
  12.1        9
  12.6        7
  12.8       14
  13.3       11
  13.9       9
/
Qa(I,J)                  link capacities in BPR function
/
1.5      700
1.12     560
```

```
4.5      560
4.9      280
5.6      420
5.9      420
6.7      700
6.10     280
7.8      700
7.11     700
8.2      700
9.10     280
9.13     280
10.11    700
11.2     280
11.3     560
12.6     140
12.8     560
13.3     560
/
T0(I,J)                  OD flows
/
1.2      210
1.3      430
1.8      320
2.1      210
2.4      320
2.12     50
3.1      430
3.4      110
3.12     40
4.2      320
4.3      110
/;

delta(I,I)=1;
eta(I,J)=1;
Qa(I,J)=max(Qa(J,I),Qa(I,J));

*******************************************************************************************
* Identify arcs using free travel time parameter
*******************************************************************************************
LINK(I,J)=no;
LINK(I,J)$(nu(I,J)>0)=yes;
LINK1(I,J)=LINK(I,J);
nlinks=SUM(LINK,1);

*******************************************************************************************
* Associate a number with each link
*******************************************************************************************
aux=0;
```

```
LOOP(LINK(I,J),
         put I.tl:3:0,"-",J.tl:3:0/;aux=aux+1;order(I,J)=aux
);

*******************************************************************************************
* Check if the routes are well defined and print the errors
*******************************************************************************************
put "CHECKING ROUTES"/;
loop((PATH,PATH1)$(ord(PATH)<ord(PATH1)),
         aux=sum(SS$(pathlinks(PATH,SS)=pathlinks(PATH1,SS)),1);
         if(aux=card(SS),put "ERROR ROUTES ",PATH.tl:3:0," AND ",
                 PATH1.tl:2:0," ARE IDENTICAL"/;);
);
loop(PATH,
         loop(SS$(ord(SS)<card(SS) and pathlinks(PATH,SS)>0 and pathlinks(PATH,SS+1)>0),
                 loop((LINK(I,J),LINK1(I1,J1))$(order(LINK)=pathlinks(PATH,SS)
                 and order(LINK1)=pathlinks(PATH,SS+1) and ord(J)<>ord(I1)),
                         put "ERROR in PATH ",PATH.tl:4:0," LINK ",order(LINK):4:0,
                         " IS NOT CONNECTED WITH LINK ",order(LINK1):4:0/;
                 );
         );
);

*******************************************************************************************
* Definition of the parameters of the chi normal variables
*******************************************************************************************
PARAMETER
mua(I,J)         mean of the chi variables
sigmaa(I,J)      standard deviation of the chi variables
;
Loop(LINK(I,J),
         mua(LINK)=(1/Qa(LINK))**gamma;
         sigmaa(LINK)=0.1*mua(LINK);
);
put "mu_ij and sigma_ij"/;
loop(LINK,
         put mua(LINK):8:3,sigmaa(LINK):8:3/;
);

*******************************************************************************************
* Make parameters dimensionless
*******************************************************************************************
tref=smax(LINK,nu(LINK));
Qaref=smax(LINK,Qa(LINK));
nu(LINK)=nu(LINK)/tref;
Qa(LINK)=Qa(LINK)/Qaref;
mua(LINK)=mua(LINK)*(Qaref**gamma);
sigmaa(LINK)=sigmaa(LINK)*(Qaref**gamma);
```

```
*******************************************************************************************
* Converts the OD flows to dimensionless values
*******************************************************************************************
T0(i,j)=T0(i,j)*1;
T0(i,j)=T0(i,j)/Qaref;

*******************************************************************************************
* Generates ODs based on flows
*******************************************************************************************
OD(I,J)=no;
OD(I,J)$(T0(I,J)>0)=yes;
nOD=SUM(OD,1);

*******************************************************************************************
* Makes copies of ODs
*******************************************************************************************
OD1(I,J)=OD(I,J);

PARAMETER
T00(I,J,AA)              OD flows by alpha class
Frequencies1(I,I,I,I,AA) auxiliary parameter for the mathematica codes;

*******************************************************************************************
* Distribute flows among classes
*******************************************************************************************
T00(OD,AA)=T0(OD)/(card(AA));
T00(OD,'1')=T0(OD)-sum(AA$(ord(AA)>1),T00(OD,AA));

put "T00"/;
loop(OD(K,S),
         put K.tl:3:0,"-",S.tl:3:0;
         loop(AA,
                 put T00(OD,AA):8:3;
         );
         put /;
);

*******************************************************************************************
* Determine origins and destinations and print them
*******************************************************************************************
Origin(I)=no;
loop(OD(K,S),
         Origin(I)$(ord(K)=ord(I))=yes;
);
Destination(I)=no;
loop(OD(K,S),
         Destination(I)$(ord(S)=ord(I))=yes;
);
put "Origins"/;
```

```
loop(Origin(I),
         put I.tl:3:0/;
);
put "Destinations"/;
loop(Destination(I),
         put I.tl:3:0/;
);

put "PATH ORIGINS AND DESTINATIONS"/;
loop(PATH,
         loop(LINK(I,J)$(order(LINK)=pathlinks(PATH,'1')),
                 pathorigin(PATH)=ord(I)
         );
         loop(SS,
                 loop(LINK(I,J)$(pathlinks(PATH,SS)>0 and order(LINK)=pathlinks(PATH,SS)),
                         pathdestination(PATH)=ord(J)
                 );
         );
         put "PATH ",PATH.tl:3:0," ORIGIN ",pathorigin(PATH):3:0," DESTINATION "
                 ,pathdestination(PATH):3:0/;
);

*****************************************************************************************
* Define scalars for the mathematica codes
*****************************************************************************************
SCALAR
nn
umin
umax
vmin
vmax
plotsize/10/;

PARAMETERS
FI22(I,J,AA)     alpha percentile of the mean travel time for an OD pair and class
FI11(PATH,AA)    alpha percentile of the mean travel time for a route and class
XX1(I,J)         flow on a link
QQ1(I,J,K,S,AA)  flow on a link disagreggated by OD and class
FF1(PATH,AA)     flow of a class on a path
QQ(I,J,K,S,AA)   flow on a link disagreggated by OD and class
delta1(I,J,PATH) link-path incidence matrix
U(I)             abscisa of node I
/1 0
2 9.3
3 9.3
4 1.9
5 3.6
6 5.5
7 7.4
```

```
8 9.3
9 5.5
10 5.5
11 7.4
12 1.8
13 7.4/
V(I)             ordinate of node I
/1 3.8
2 2
3 0
4 0
5 2
6 3.8
7 3.8
8 5.8
9 0
10 2
11 2
12 5.8
13 0/;

*******************************************************************************************
* Determine max and min network coordinates
*******************************************************************************************
umin=Smin(I,U(I));
umax=Smax(I,U(I));
vmin=Smin(I,V(I));
vmax=Smax(I,V(I));

*******************************************************************************************
* Normalize coordinates
*******************************************************************************************
U(I)=(U(I)-umin)/(umax-umin)*plotsize;
V(I)=(V(I)-vmin)/(vmax-vmin)*plotsize;

*******************************************************************************************
* Compute the link-path incidence matrix
*******************************************************************************************
delta1(LINK(I,J),PATH)=0;
loop((PATH,SS,LINK),
         if(order(LINK)=pathlinks(PATH,SS),delta1(LINK,PATH)=1;);
);

*******************************************************************************************
* The variables are defined
*******************************************************************************************
VARIABLES
z                objective variable for models
;
```

```
POSITIVE VARIABLES
Q(I,J,K,S,AA)    disagreggated link flow
F(PATH,AA)       path flow
X(I,J)           link flow
;

EQUATIONS
*****************************************************************************************
* Equations for the initial problem
*****************************************************************************************
z3def            objective function
*****************************************************************************************
* Equations for solving the problem with path enumeration
*****************************************************************************************
z1def            objective function
balance1         balance flow equations
Xdef1            link flow definition
bound1           virtual constarint to ensure global optimality
*****************************************************************************************
* Equations for solving the problem without path enumeration
*****************************************************************************************
z2def            objective function
balance2         balance constraints
Xdef2            link flow definition
bound2           virtual constarint to ensure global optimality
;

*****************************************************************************************
* The equations are defined
*****************************************************************************************
z3def..z=e=SUM((OD,AA),SUM(LINK(I,J),Q(LINK,OD,AA)));

z1def..z=e=SUM((OD(K,S),AA),(SUM(LINK(I,J),Sum(PATH$(ord(K)=pathorigin(PATH) and
         ord(S)=pathdestination(PATH)),F(PATH,AA)*delta1(LINK,PATH))*(nu(I,J)*(1+eta(I,J)*
         power(X(LINK),gamma)*mua(LINK))))+classalpha(AA)*sqrt(SUM(LINK(I,J),power(Sum(
         PATH$(ord(K)=pathorigin(PATH) and ord(S)=pathdestination(PATH)),F(PATH,AA)*delta1(
         LINK,PATH))*nu(I,J)*eta(I,J)*power(X(LINK),gamma)*sigmaa(I,J),2)))));
balance1(OD(K,S),AA)..TOO(OD,AA)=e=SUM(PATH$(ord(K)=pathorigin(PATH) and ord(S)=
         pathdestination(PATH)),F(PATH,AA));
Xdef1(LINK)..X(LINK)=e=SUM((PATH,AA),F(PATH,AA)*delta1(LINK,PATH));
bound1..SUM((OD(K,S),AA),(SUM(LINK(I,J),Sum(PATH$(ord(K)=pathorigin(PATH) and
         ord(S)=pathdestination(PATH)),F(PATH,AA)*delta1(LINK,PATH))*(nu(I,J)*(1+eta(I,J)*
         power(X(LINK),gamma)*mua(LINK))))+classalpha(AA)*sqrt(SUM(LINK(I,J),power(Sum(
         PATH$(ord(K)=pathorigin(PATH) and ord(S)=pathdestination(PATH)),F(PATH,AA)*delta1
         (LINK,PATH))*nu(I,J)*eta(I,J)*power(X(LINK),gamma)*sigmaa(I,J),2)))))=l=cbound;

z2def..z=e=SUM((OD,AA),(SUM(LINK(I,J),Q(LINK,OD,AA)*nu(I,J)*(1+eta(I,J)*power(X(LINK),gamma)
         *mua(LINK)))+classalpha(AA)*sqrt(SUM(LINK(I,J),power(Q(LINK,OD,AA)*nu(I,J)*
```

```
         eta(I,J)*power(X(LINK),gamma)*sigmaa(I,J),2)))));
balance2(I,OD(K,S),AA)..TOO(OD,AA)*(DELTA(I,K)-DELTA(I,S))=e=SUM((LINK(I,J)),Q(LINK,OD,AA))
         -SUM((LINK(J,I)),Q(LINK,OD,AA));
Xdef2(LINK)..X(LINK)=e=SUM((OD,AA),Q(LINK,OD,AA));
bound2..SUM((OD,AA),(SUM(LINK(I,J),Q(LINK,OD,AA)*nu(I,J)*(1+eta(I,J)*power(X(LINK),gamma)*
         mua(LINK)))+classalpha(AA)*sqrt(SUM(LINK(I,J),power(Q(LINK,OD,AA)*nu(I,J)*
         eta(I,J)*power(X(LINK),gamma)*sigmaa(I,J),2)))))=l=cbound;

MODEL InitialPSO/z3def,balance2,Xdef2/;
MODEL PSOwith/z1def,balance1,Xdef1,bound1/;
MODEL PSOwithout/z2def,balance2,Xdef2,bound2/;

*****************************************************************************************
* The inicial model is solved to get initial values
*****************************************************************************************
F.l(PATH,AA)=1;
SOLVE InitialPSO USING lp MINIMIZING z;
put "InitialPSO=",z.l:12:3," modelstat=",InitialPSO.modelstat," solvestat=",
         InitialPSO.solvestat," resusd=",InitialPSO.resusd:12:8/;

*****************************************************************************************
* The model without path enumeration is solved
*****************************************************************************************
cbound=INF;
OPTION reslim=55000;
done=0;
loop(iter$(done=0),
         put "PSOwithout: iter=",ord(iter):3:0/;
         PSOwithout.optfile = 1;
         SOLVE PSOwithout USING nlp MINIMIZING z;
         put "PSOwithout=",z.l:12:3," modelstat=",PSOwithout.modelstat," solvestat=",
                 PSOwithout.solvestat," resusd=",PSOwithout.resusd:12:8/;
         if(PSOwithout.modelstat<>2 or PSOwithout.solvestat<>1,
                 done=1;
         );
                 cbound=z.l-incre;
);

QQ1(LINK,OD,AA)=Q.l(LINK,OD,AA);XX1(LINK)=X.l(LINK);
FI22(OD,AA)=(SUM(LINK(I,J),Q.l(LINK,OD,AA)*(nu(I,J)*(1+eta(I,J)*power(X.l(LINK),gamma)*
         mua(LINK))))+classalpha(AA)*sqrt(SUM(LINK(I,J),power(Q.l(LINK,OD,AA)*nu(I,J)*
         eta(I,J)*power(X.l(LINK),gamma)*sigmaa(I,J),2))))/TOO(OD,AA);

Q.l(LINK,OD,AA)=QQ1(LINK,OD,AA);
XX(LINK)=XX1(LINK);

*****************************************************************************************
* The model with path enumeration is solved
*****************************************************************************************
```

```
cbound=INF;

done=0;
loop(iter$(done=0),
         put "PSOwith: iter=",ord(iter):3:0/;
         PSOwith.optfile = 1;
         SOLVE PSOwith USING nlp MINIMIZING z;
         put "PSOwith=",z.l:12:3," modelstat=",PSOwith.modelstat," solvestat=",
                 PSOwith.solvestat," resusd=",PSOwith.resusd:12:8/;
         if(PSOwith.modelstat<>2 or PSOwith.solvestat<>1,
                 done=1;
         else
                 FF1(PATH,AA)=F.l(PATH,AA);XX1(LINK)=X.l(LINK););
         cbound=z.l-incre;
);

*******************************************************************************
* Convert the dimensionless parameter to their initial dimensions
*******************************************************************************
nu(LINK)=nu(LINK)*tref;
Qa(LINK)=Qa(LINK)*Qaref;
mua(LINK)=mua(LINK)/(Qaref**gamma);
sigmaa(LINK)=sigmaa(LINK)/(Qaref**gamma);
FF1(PATH,AA)=FF1(PATH,AA)*Qaref;
FI22(OD,AA)=FI22(OD,AA)*tref;
XX1(LINK)=XX1(LINK)*Qaref;
XX(LINK)=XX(LINK)*Qaref;
Q.l(LINK,OD,AA)=Q.l(LINK,OD,AA)*Qaref;
T00(OD,AA)=T00(OD,AA)*Qaref;


F.l(PATH,AA)=FF1(PATH,AA);
X.l(LINK)=XX1(LINK);

FI11(PATH,AA)=SUM(LINK(I,J),delta1(LINK,PATH)*(nu(I,J)*(1+eta(I,J)*power(X.l(LINK),gamma)*
         mua(LINK))))+classalpha(AA)*sqrt(SUM(LINK(I,J),delta1(LINK,PATH)*power(nu(I,J)*
         eta(I,J)*power(X.l(LINK),gamma)*sigmaa(I,J),2)));

QQ(LINK,OD(K,S),AA)=Sum(PATH$(ord(K)=pathorigin(PATH) and ord(S)=pathdestination(PATH))
         ,F.l(PATH,AA)*delta1(LINK,PATH));

traveltime(OD(K,S),AA)=balance2.m(S,K,S,AA)-balance2.m(K,K,S,AA);

Frequencies1(LINK,OD,AA)=Q.l(LINK,OD,AA);
Frequencies1(LINK,OD,AA)=0;
Frequencies1(LINK,OD,AA)$(Q.l(LINK,OD,AA)>CRITICALFLOW)=order(LINK);

*******************************************************************************
* MATHEMATICA PLOT NETWORK
```

```
*******************************************************************************************
put out1;
put "Arrowpos = 0.4;"/;
put "ArrowSize = 0.15;"/;
put "FontSize1 = 12;"/;
put "Radious = 0.4;"/;
put "Points = {";
loop(I,if(ord(I)<card(I),put "{",U(I):7:3,",",V(I):7:3,"},"/;
         else put "{",U(I):7:3,",",V(I):7:3,"}};"/;););
put /;
nn=0;
nn=0;
put "LinkNodes = {";

loop(LINK(I,J),nn=nn+1;if(nn<nlinks, put "{",ord(I):3:0,",",ord(J):3:0,"},"/;
         else put "{",ord(I):3:0,",",ord(J):3:0,"}}"/; ););
put "bb = NetworkPlot[Points, LinkNodes, Arrowpos, ArrowSize, FontSize1, Radious];"/;
put "Show[bb, ImageSize -> 360, Axes -> False, AxesLabel -> None, AxesStyle -> False];"/;

nlinks=SUM(LINK,1);
put "nlinks=",nlinks:5:0/;
put "Frequencies1={";
ilink=0;
loop(LINK,
         ilink=ilink+1;
         put "{";
         iOD=0;
         loop(OD,
                 iOD=iOD+1;
                 put "{";
                 loop(AA,
                         if(ord(AA)=card(AA),
                                 if(iOD=nOD,
                                         if(ilink=nlinks,
                                                 put Frequencies1(LINK,OD,AA):8:4,"}}};"/;
                                         else
                                                 put Frequencies1(LINK,OD,AA):8:4,"}},"/;
                                         );
                                 else
                                         put Frequencies1(LINK,OD,AA):8:4,"},"/;
                                 );
                         else
                                 put Frequencies1(LINK,OD,AA):8:4,","/;
                         );
                 );
         );
);
mm=0;
iAA=0;
```

```
Loop(AA,
         iAA=iAA+1;
         iOD=0;
         loop(OD(K,S),
                 iOD=iOD+1;
                 put "OriginNode=",K.tl:5:0,";"/;
                 put "DestinationNode=",S.tl:5:0,";"/;
                 mm=mm+1;
                 nn=0;
                 indic=0;
                 nlinks=SUM(LINK$(Frequencies1(LINK,OD,AA)>CRITICALFLOW),1);
                 loop(LINK(I,J)$(Frequencies1(LINK,OD,AA)>CRITICALFLOW),
                         if(nn=0,
                                 indic=1;
                                 put "LinkNodes1= {";
                         );
                         nn=nn+1;
                         if(nn<nlinks,
                                 put "{",ord(I):3:0,",",ord(J):3:0,"},"/;
                         else
                                 put "{",ord(I):3:0,",",ord(J):3:0,"}};"/;
                         );
                 );

                 if(indic=1,
                         put "ff = Table[Frequencies1[[i,",iOD:5:0,",",iAA:5:0,"]],
                                 {i, 1,", SUM(LINK,1):5:0,"}]"/;
                         put "a[",mm:5:0,"]=NetworkPlot2[Points, LinkNodes, LinkNodes1,
                                  Arrowpos, ArrowSize, FontSize1, Radious,ff];"/;
                 );
         );
);
aux=0;
loop(OD, aux=aux+1;
put "Show[GraphicsGrid[{{a[",aux:3:0,"], a[",(aux+nOD):3:0,"]}, {a[",(aux+2*nOD):3:0,
         "], a[",(aux+3*nOD):3:0,"]}}], ImageSize -> 800,Axes -> False, AxesLabel ->
         None, AxesStyle -> False]"/;
);

*******************************************************************************************
* Create latex codes for showing the results in tables
*******************************************************************************************
put out2;
put /;
put "\begin{table}[h]"/;
put "\small\centering"/;
put "\renewcommand{\tabcolsep}{0.2cm}"/;
put "\renewcommand{\arraystretch}{0.9}"/;
put "\begin{tabular}{cc}"/;
```

```
put "\begin{tabular}{|c|l|}"/;
put "\hline"/;
put " Route ($r$)&   \multicolumn{1}{|c|}{Links ($\ell$)} \\"/;
put "\hline"/;
aux=0;
loop(OD(K,S),aux=aux+1;
put "OD pair ",aux:3:0," &  \\"/;
loop(PATH,
if(ord(K)=pathorigin(PATH) and ord(S)=pathdestination(PATH),
         put PATH.tl:3:0," & ";loop(SS$(pathlinks(PATH,SS)>0),
         put pathlinks(PATH,SS):4:0;);put "\\"/;);
);
put "\hline"/;
);
put "\hline"/;
put "\end{tabular}"/;
put "\end{tabular}"/;
put "\caption{Set of ${\cal OD}$-pairs and routes (routes) considered in the
         Nguyen-Dupuis network.}"/;
put "\end{table}"/;

put /;
put "{\small"/;
put "\begin{table}"/;
put "\centering"/;
put "\renewcommand{\tabcolsep}{1mm}"/;
put "\renewcommand{\arraystretch}{0.8}"/;
put "\begin{tabular}{ccrcrcrcrc}"/;
put "\hline"/;
put "O-D   & Route  & \multicolumn{2}{c}{($\alpha=$,     0.10)} & \multicolumn{2}{c}
         {($\alpha=$     0.50)} & \multicolumn{2}{c}{($\alpha=$     0.70)} &
         \multicolumn{2}{c}{($\alpha=$     0.90)}\\"/;
put "& & Flow & Percentile  & Flow & Percentile  & Flow & Percentile  & Flow &
         Percentile \\"/;
put "\hline"/;
loop(OD(K,S),put K.tl:2:0,"-",S.tl:2:0;loop(PATH$(ord(K)=pathorigin(PATH) and
         ord(S)=pathdestination(PATH) and sum(AA,F.l(PATH,AA))>0.001),put " & ",
         PATH.tl:2:0;loop(AA,if(abs(F.l(PATH,AA))>0.001,put " & ",F.l(PATH,AA):8:2,
         " & ",FI11(PATH,AA):8:2; else put " & ","-":8:2,"& ","-":8:2;););put "\\"/;)
         ;put "\hline"/;put  "OD & ";loop(AA,put " & ",T00(OD,AA):8:2," & ",
         FI22(OD,AA):8:2;);put "\\"/;put "\hline"/;);
put "\hline"/;
put "\end{tabular}"/;
put "\caption{\label{}Percentile PSO solution for the Nguyen-Dupuis example.}"/;
put "\end{table}"/;
put "}"/;
```

A.2 Codes for the Traffic Assignment problem including overtaking classes

A.2.1 Gams codes for the Traffic Assignment problem including overtaking classes

This section is devoted to the GAMS implementation of the Traffic Assignment problem including overtaking classes presented in Chapter 8, Section 8.3. The codes for the models with and without overtaking are shown for the Nguyen-Dupuis example in the case of heterogeneous users (cars and motorcycles) with high congestion.

```
$Title DupuisEquilibriumMixed

*****************************************************************************************
*The output files are defined
*****************************************************************************************
file out/DupuisEquilibriumMixed.out/;
file out1/DupuisEquilibriumMixedMathematica.out/;
file out2/DupuisEquilibriumMixedLatex.out/;
file out3/DupuisEquilibriumMixedMathlab.out/;
put out;

OPTION nlp=CONOPT;

SCALAR
CRITICALFLOW     critical value for a non-null flow /0.000001/
epsilon          small scalar added to the logarithm /0.0000001/
eta              weight of the entropy term /0.00000000001/
ilink            counter for the links
nOD              number of OD pairs
iAA              counter for the classes
iOD              counter for the OD pairs
aux              auxiliary scalar
mm               auxiliary scalar
indic            auxiliary scalar
gamma            gamma parameter of the BPR function for the reference class /3/
;

SETS
AA               classes/1*6/
I                set of nodes /1*13/
LINK(I,I)        set of links
LINK1(I,I)       copy of set of links
LINK2(I,I)       copy of set of links
OD(I,I)          set of OD pairs
OD1(I,I)         copy of set of OD pairs
OD2(I,I)         copy of set of OD pairs
Origin(I)        set of origin nodes
Destination(I)   set of destination nodes
```

```
PATH             set of paths /1*14/
SS               set of links in a path /1*5/;

ALIAS(I,J,K,S);
ALIAS(AA,AA1,AA2);
ALIAS(PATH,PATH1,PATH2);

TABLE
pathlinks(PATH,SS)
       1   2   3   4   5
1      1   5   7   9  11
2      1   5   7  10  16
3      1   5   8  14  16
4      1   6  12  14  16
5      1   6  13  19   0
6      2  17   7  10  16
7      2  18  11   0   0
8      3   5   7   9  11
9      3   5   7  10  15
10     3   5   8  14  16
11     3   6  12  14  15
12     3   6  13  19   0
13     4  12  14  15   0
14     4  13  19   0   0
;

set param  arc cost table headers / a, b, k /
table arc_cost(I,I,param)  arc cost data
            a         b         k
1.5         7         7       700
1.12        9         9       560
4.5         9         9       560
4.9        12        12       280
5.6         3         3       420
5.9         9         9       420
6.7         5         5       700
6.10        5         5       280
7.8         5         5       700
7.11        9         9       700
8.2         9         9       700
9.10       10        10       280
9.13        9         9       280
10.11       6         6       700
11.2        9         9       280
11.3        8         8       560
12.6        7         7       140
12.8       14        14       560
13.3       11        11       560
;
```

```
option arc_cost:4;
display arc_cost;

PARAMETER
Weight(AA)       weight given to cars and motos
/1 1
2 1
3 1
4 0.5
5 0.5
6 0.5/
LinkCost(I,J,AA)
classalpha(AA)   value of q_alpha
/1 1
2 0.9
3 0.8
4 1.4
5 1.3
6 1.2/
classalpha0(AA)  free flow time for each class
/1 1
2 1
3 1
4 1.4
5 1.4
6 1.4/
mualpha(AA)      mean of the normal distribution
/1 0.5
2 0.5
3 0.5
4 0.5
5 0.5
6 0.5/
sigmaalpha(AA)   standard deviation of the normal distribution
/1 0.25
2 0.25
3 0.25
4 0.5
5 0.5
6 0.5/
gammaalpha(AA)   parameter of the BPR function
/1 5
2 5
3 5
4 3
5 3
6 3/
betaalpha(AA)    parameter of the BPR function
```

```
/1 3
2 3
3 3
4 1
5 1
6 1/
traveltime(I,I,AA)
routecost(path,aa)
LINKFLOWS(I,I,AA)

Ca(I,J)          uncongested link costs
Qa(I,J)          link capacities
delta(I,I)       kronecker delta
potentials(I,K,S) dual variable
order(I,J)       order given to the links
;
*******************************************************************************************
*Identify arcs using flow cost parameter a
*******************************************************************************************

Ca(i,j) = arc_cost(i,j,"a");
LINK(i,j)=Ca(i,j);
Qa(I,J)= arc_cost(i,j,"k");

aux=0;
LOOP(LINK(I,J),put I.tl:3:0,"-",J.tl:3:0/;aux=aux+1;order(I,J)=aux);

TABLE T0(i,j)  trip matrix from i to j
  1    2    3    4    5    6    7    8    9   10   11   12   13
1 0  500  640
2      0
3           0
4    480 300     0
;
*******************************************************************************************
* Multiply the OD flows for the congested case
*******************************************************************************************
T0(i,j)=T0(i,j)*1.15;
*******************************************************************************************
* Generate ODs based on flows
*******************************************************************************************
OD(I,J)=no;
OD(I,J)$(T0(I,J)>0)=yes;
*******************************************************************************************
* Make copies of LINKS and ODs
*******************************************************************************************
OD1(I,J)=OD(I,J);
OD2(I,J)=OD(I,J);
LINK1(I,J)=LINK(I,J);
```

```
LINK2(I,J)=LINK(I,J);

PARAMETERS
pathorigin(PATH)         origin node of each path
pathdestination(PATH)    destination node of each path
delta1(I,J,PATH)         link-path incidence matrix
delta2(I,J,PATH)         OD-path incidence matrix
F0(PATH,AA)
Q0(I,I,I,I,AA)
T00(I,J,AA)
Frequencies1(I,I,I,I,AA);

*******************************************************************************************
* Compute the incidence matrices and the origin and destination of paths
*******************************************************************************************
loop(PATH,
         loop(LINK(I,J)$(order(LINK)=pathlinks(PATH,'1')),pathorigin(PATH)=ord(I));
         loop((SS,LINK(I,J))$(pathlinks(PATH,SS)>0 and order(LINK)=pathlinks(PATH,SS)),
         pathdestination(PATH)=ord(J));
);
delta(I,I)=1;
loop((PATH,SS,LINK)$(order(LINK)=pathlinks(PATH,SS)),delta1(LINK,PATH)=1;);
loop((PATH,OD(K,S))$(pathorigin(PATH)=ord(K) and pathdestination(PATH)=ord(S)),
        delta2(OD,PATH)=1);

*******************************************************************************************
* Distributes flows among classes
*******************************************************************************************
T00(OD,AA)=T0(OD)/(card(AA));
T00(OD,'1')=T0(OD)-sum(AA$(ord(AA)>1),T00(OD,AA));

*******************************************************************************************
* Determines origins and destinations
*******************************************************************************************
Origin(I)=no;
loop(OD(K,S),Origin(I)$(ord(K)=ord(I))=yes;);
Destination(I)=no;
loop(OD(K,S),Destination(I)$(ord(S)=ord(I))=yes;);
*******************************************************************************************
* Prints origins and destinations
*******************************************************************************************
put "Origins"/;
loop(Origin(I),put I.tl:3:0/;);
put "Destinations"/;
loop(Destination(I),put I.tl:3:0/;);

*******************************************************************************************
* Define scalar and parameters needed to make the figures
*******************************************************************************************
```

```
SCALAR
nn
nlinks number of links
umin
umax
vmin
vmax
plotsize/10/;

PARAMETERS
U(I) abscisa of node I
/1 0
2 9.3
3 9.3
4 1.9
5 3.6
6 5.5
7 7.4
8 9.3
9 5.5
10 5.5
11 7.4
12 1.8
13 7.4/
V(I) ordinate of node I
/1 3.8
2 2
3 0
4 0
5 2
6 3.8
7 3.8
8 5.8
9 0
10 2
11 2
12 5.8
13 0/;
*****************************************************************************
* Determine max and min network coordinates
*****************************************************************************
umin=Smin(I,U(I));
umax=Smax(I,U(I));
vmin=Smin(I,V(I));
vmax=Smax(I,V(I));
*****************************************************************************
* Normalize coordinates
*****************************************************************************
U(I)=(U(I)-umin)/(umax-umin)*plotsize;
```

```
V(I)=(V(I)-vmin)/(vmax-vmin)*plotsize;

*******************************************************************************************
* The variables are defined
*******************************************************************************************
VARIABLES
z                 objective variable for models
lambda(I,I,I,AA) dual variable
rho(I,J,AA)       minumum OD travel time for each class
;

POSITIVE VARIABLES
Q(I,J,K,S,AA)     disgregated link flow
muu(I,I,I,I,AA)   dual variable

mu(PATH,AA)       dual variable
f(PATH,AA)        path flow
;

Q.up(LINK,OD,AA)=TOO(OD,AA);

EQUATIONS
*******************************************************************************************
* Equations for solving the problem without path enumeration
*******************************************************************************************
zdef              objective function
balance           balance constraints
KKT               cost function equation
*******************************************************************************************
* Equations for solving the problem with path enumeration
*******************************************************************************************
zdef_paths        objective function
ODflow            OD flow constraint
KKT_paths         cost function equation
;

*******************************************************************************************
* The equations are defined
*******************************************************************************************
zdef..z=e=SUM((LINK,OD,AA),muu(LINK,OD,AA)*Q(LINK,OD,AA));
balance(I,OD(K,S),AA)..TOO(OD,AA)*(DELTA(I,K)-DELTA(I,S))=e=SUM((LINK(I,J)),Q(LINK,OD,AA))
                                                          -SUM((LINK(J,I)),Q(LINK,OD,AA));
KKT(LINK(I,J),OD(K,S),AA)..Ca(I,J)*(classalpha0(AA)*(1+betaalpha(AA)*(SUM((OD1,AA1),
        Weight(AA1)*Q(LINK,OD1,AA1))/Qa(I,J))**gammaalpha(AA))*errorf((SUM((OD1,AA1),
        Weight(AA1)*Q(LINK,OD1,AA1))/Qa(I,J)-mualpha(AA))/sigmaalpha(AA))+(classalpha(AA)
        +classalpha(AA)*betaalpha(AA)*(SUM((OD1,AA1),Weight(AA1)*Q(LINK,OD1,AA1))/Qa(I,J))
        **gammaalpha(AA))*(1-errorf((SUM((OD1,AA1),Weight(AA1)*Q(LINK,OD1,AA1))/Qa(I,J)-
        mualpha(AA))/sigmaalpha(AA))))+eta*sum((AA2,LINK2,OD2),Q(LINK2,OD2,AA2)*
        log(Q(LINK2,OD2,AA2)+epsilon))+lambda(I,OD,AA)*(1)-lambda(J,OD,AA)*(1)
```

```
        -muu(LINK,OD,AA)=e=0;

zdef_paths..z=e=SUM((PATH,AA),mu(PATH,AA)*f(PATH,AA));
ODflow(OD,AA)..TO0(OD,AA)=e=SUM(PATH,f(PATH,AA)*delta2(OD,PATH));
KKT_paths(PATH,AA)..0=e=SUM(LINK(I,J),Ca(I,J)*(classalpha0(AA)*(1+betaalpha(AA)*(SUM(
        (PATH1,AA1),Weight(AA1)*f(PATH1,AA1)*delta1(LINK,PATH1))/Qa(I,J))**gammaalpha(AA))
        *errorf((SUM((PATH1,AA1),Weight(AA1)*f(PATH1,AA1)*delta1(LINK,PATH1))/Qa(I,J)
        -mualpha(AA))/sigmaalpha(AA))+(classalpha(AA)+classalpha(AA)*betaalpha(AA)*(SUM(
        (PATH1,AA1),Weight(AA1)*f(PATH1,AA1)*delta1(LINK,PATH1))/Qa(I,J))**gammaalpha(AA))
        *(1-errorf((SUM((PATH1,AA1),Weight(AA1)*f(PATH1,AA1)*delta1(LINK,PATH1))/Qa(I,J)
        -mualpha(AA))/sigmaalpha(AA))))*delta1(LINK,PATH))+eta*SUM((AA2,PATH2),f(PATH2,AA2)
        *log(f(PATH2,AA2)+epsilon))+SUM(OD,rho(OD,AA)*delta2(OD,PATH))-mu(PATH,AA);

*****************************************************************************************
* The models are defined
*****************************************************************************************
MODEL WARDROPnoPATH/zdef,balance,KKT/;

MODEL WARDROPpaths/zdef_paths,ODflow,KKT_paths/;

*****************************************************************************************
* Calculate initial values for the path flows and solve the problem
*****************************************************************************************
F0(PATH,AA)=SUM(OD,TO0(OD,AA)*delta2(OD,PATH))/sum(OD$(delta2(OD,PATH)=1),SUM(PATH1,
            delta2(OD,PATH1)));
F.l(PATH,AA)=F0(PATH,AA);
SOLVE WARDROPpaths USING nlp MINIMIZING z;
put "WARDROPpaths=",z.l:12:3," modelstat=",WARDROPpaths.modelstat," solvestat=",
WARDROPpaths.solvestat," resusd=",WARDROPpaths.resusd:12:8/;

*****************************************************************************************
* Calculate the disaggregated link flows and link travel times for each class
*****************************************************************************************
Q.l(LINK,OD,AA)=sum(PATH,F.l(PATH,AA)*delta1(LINK,PATH)*delta2(OD,PATH));

LinkCost(LINK(I,J),AA)=Ca(I,J)*(classalpha0(AA)*(1+betaalpha(AA)*(SUM((OD1,AA1),Weight(AA1)
            *Q.l(LINK,OD1,AA1))/Qa(I,J))**gammaalpha(AA))*errorf((SUM((OD1,AA1),Weight(AA1)
            *Q.l(LINK,OD1,AA1))/Qa(I,J)-mualpha(AA))/sigmaalpha(AA))+(classalpha(AA)+
            classalpha(AA)*betaalpha(AA)*(SUM((OD1,AA1),Weight(AA1)*Q.l(LINK,OD1,AA1))
            /Qa(I,J))**gammaalpha(AA))*(1-errorf((SUM((OD1,AA1),Weight(AA1)
            *Q.l(LINK,OD1,AA1))/Qa(I,J)-mualpha(AA))/sigmaalpha(AA))));

put "LINK COSTS"/;
loop(LINK(I,J),
         put I.tl:3:0,"-",J.tl:3:0;
         loop(AA,
                 put LinkCost(I,J,AA):8:3;
         );
         put /;
```

```
);

put "Q values"/;
loop(LINK(I,J),
         put I.tl:3:0,"-",J.tl:3:0;
         loop(OD,
                 loop(AA,
                         put Q.l(LINK,OD,AA):7:3;
                 );
                 put "  |";
         );
         put /;
);

put "Total Q values"/;
loop(LINK(I,J),
         put I.tl:3:0,"-",J.tl:3:0;
         put (SUM((OD,AA),Q.l(LINK,OD,AA))/Qa(I,J)):7:3;
         put /;
);

put "Route costs"/;
loop(AA,
         put "AA=",AA.tl:3:0/;
         loop(PATH,
                 routecost(path,aa)=SUM(SS$(pathlinks(PATH,SS)>0),SUM(LINK$
                     (order(LINK)=pathlinks(PATH,SS)),linkcost(LINK,AA)));
                 PUT routecost(path,aa):8:3;
         );
         put /;
);
*******************************************************************************
* Compute the relative duality gap
*******************************************************************************
traveltime(OD(K,S),AA)=smin(PATH$(delta2(OD,PATH)=1),routecost(path,aa));

Scalar
DualityGap;

DualityGap=(sum((PATH,AA),routecost(path,aa)*F.l(PATH,AA))-sum((OD,AA),traveltime(OD,AA)
                                *TOO(OD,AA)))/sum((OD,AA),traveltime(OD,AA)*TOO(OD,AA));
put "Duality Gap =", DualityGap:20:18/;

*******************************************************************************
* Create the LATEX tables
*******************************************************************************
put out2;
put /;
put "\begin{table}"/;
```

```
put "{\scriptsize"/;
put "\centering"/;
put "\renewcommand{\tabcolsep}{1mm}"/;
put "\renewcommand{\arraystretch}{1.2}"/;
put "\begin{tabular}{|c|c|";loop(OD,loop(AA,put "c";);put "|";);put "}"/;
put "\hline"/;
put " & ";loop(OD(K,S),put "& \multicolumn{",card(AA):3:0,"}{|c|}{OD:",K.tl:3:0,"-",
        S.tl:3:0,"}";);put "\\"/;
put "\cline{3-",(sum(OD,1)*card(AA)+2):4:0,"}"/;
put "Link & Flow ";loop(OD(K,S),loop(AA,put " & ",classalpha(AA):4:2;););put "\\"/;
put "\hline"/;
loop(LINK(I,J),
         put I.tl:2:0,"-",J.tl:2:0," & ",SUM((OD,AA),Q.l(LINK,OD,AA)):6:1;loop(OD(K,S),
         loop(AA,put " & ";if(Q.l(LINK,OD,AA)<0.01,put "-":6:0; else
         put Q.l(LINK,OD,AA):6:1;);););put "\\"/;
);
put "\hline"/;
put "\multicolumn{2}{|c|}{$u_\alpha$} ";loop(OD(K,S),loop(AA,put " & ",
            traveltime(OD,AA):6:1;););put "\\"/;
put "\hline"/;
put "\end{tabular}"/;
put "\caption{Cars and motorcycles example. Mixed BPR model. .
    Link flows disaggregated by OD and $\alpha$-classes for the Nguyen-Dupuis
    network (congested case).}"/;
put "}"/;
put "\end{table}"/;


put /;
put "\begin{table}"/;
put "\centering"/;
put "\begin{tabular}{|c|c|";loop(SS,put "c";);put "|";loop(AA, put "c";);put "|}"/;
put "\hline"/;
put " OD & Routes & \multicolumn{",card(SS):2:0,"}{|c|}{path links} &
                                \multicolumn{",card(AA):2:0,"}{|c|}{Classes}\\"/;
put " & & \multicolumn{",card(SS):2:0,"}{|c|}{} ",loop(AA,put " & ",ord(AA):2:0;);
put " \\"/;
put "\hline"/;
aux=0;
loop(OD(K,S), loop(PATH$(pathorigin(PATH)=ord(K) and pathdestination(PATH)=ord(S)),
aux=aux+1;put ord(K):2:0,"-",ord(S):2:0 "&";put aux:3:0;
loop(SS, if(pathlinks(PATH,SS)>0,put " & ",pathlinks(PATH,SS):3:0;else put " & ";););
loop(AA,if(abs(traveltime(OD,AA)-routecost(path,aa))<0.00001, PUT " & {\bf ",
routecost(path,aa):12:3,"}"; else PUT " & ",routecost(path,aa):12:3;););PUT "\\"/;);
put "\hline"/;
);
put "\end{tabular}"/;
put "\caption{Cars and motorcycles example. Route travel times classified by OD and
$\alpha$-classes for the Nguyen-Dupuis network (uncongested case).
```

```
Used routes are boldfaced.}"/;
put "\end{table}"/;

put out;

*******************************************************************************************
* Compute initial values for the disaggregated link flows
*******************************************************************************************
Q0(LINK,OD,AA)=T00(OD,AA)/8;
Q.l(LINK,OD,AA)=Q0(LINK,OD,AA);

*******************************************************************************************
* Solve the problem without path enumeration
*******************************************************************************************
SOLVE WARDROPnoPATH USING nlp MINIMIZING z;
put "WARDROPnoPATH=",z.l:12:3," modelstat=",WARDROPnoPATH.modelstat," solvestat=",
WARDROPnoPATH.solvestat," resusd=",WARDROPnoPATH.resusd:12:8/;

Frequencies1(LINK,OD,AA)=Q.l(LINK,OD,AA);
Frequencies1(LINK,OD,AA)=order(LINK)*sign(Q.l(LINK,OD,AA));

*******************************************************************************************
* Compute and print some interesting variables
*******************************************************************************************
put "LINK COSTS"/;
loop(LINK(I,J),
         put I.tl:3:0,"-",J.tl:3:0;
         loop(AA,
                 LinkCost(I,J,AA)=Ca(I,J)*(classalpha0(AA)*(1+betaalpha(AA)*(SUM((OD1,AA1),
                                 Weight(AA1)*Q.l(LINK,OD1,AA1))/Qa(I,J))**gammaalpha(AA))*
                                 errorf((SUM((OD1,AA1),Weight(AA1)*Q.l(LINK,OD1,AA1))
                                 /Qa(I,J)-mualpha(AA))/sigmaalpha(AA))+(classalpha(AA)
                                 +classalpha(AA)*betaalpha(AA)*(SUM((OD1,AA1),Weight(AA1)
                                 *Q.l(LINK,OD1,AA1))/Qa(I,J))**gammaalpha(AA))*(1-errorf(
                                 (SUM((OD1,AA1),Weight(AA1)*Q.l(LINK,OD1,AA1))/Qa(I,J)
                                 -mualpha(AA))/sigmaalpha(AA))));
                 put LinkCost(I,J,AA):8:3;
         );
         put /;
);

put "Q values"/;
loop(LINK(I,J),
         put I.tl:3:0,"-",J.tl:3:0;
         loop(OD,
                 loop(AA,
                         put Q.l(LINK,OD,AA):7:3;
                 );
                 put "  |";
```

```
        );
        put /;
);

put "mu values"/;
loop(LINK(I,J),
        put I.tl:3:0,"-",J.tl:3:0;
        loop(OD,
                loop(AA,
                        put muu.l(LINK,OD,AA):7:3;
                );
                put "  |";
        );
        put /;
);

put "mu*Q values"/;
loop(LINK(I,J),
        put I.tl:3:0,"-",J.tl:3:0;
        loop(OD,
                loop(AA,
                        put (muu.l(LINK,OD,AA)*Q.l(LINK,OD,AA)):7:3;
                );
                put "  |";
        );
        put /;
);

put "Total Q values"/;
loop(LINK(I,J),
        put I.tl:3:0,"-",J.tl:3:0;
        put (SUM((OD,AA),Q.l(LINK,OD,AA))/Qa(I,J)):7:3;
        put /;
);


put "Route costs"/;
loop(AA,
        put "AA=",AA.tl:3:0/;
        loop(PATH,
                routecost(path,aa)=SUM(SS$(pathlinks(PATH,SS)>0),SUM(LINK$(order(LINK)=
                                                pathlinks(PATH,SS)),linkcost(LINK,AA)));
                PUT routecost(path,aa):8:3;
        );
        put /;
);

traveltime(OD(K,S),AA)=smin(PATH$(delta2(OD,PATH)=1),routecost(path,aa));
```

```
*******************************************************************************************
* Create the LATEX tables
*******************************************************************************************
put out2;
put /;
put "\begin{table}"/;
put "{\scriptsize"/;
put "\centering"/;
put "\renewcommand{\tabcolsep}{1mm}"/;
put "\renewcommand{\arraystretch}{1.2}"/;
put "\begin{tabular}{|c|c|";loop(OD,loop(AA,put "c";);put "|";);put "}"/;
put "\hline"/;
put " & ";loop(OD(K,S),put "& \multicolumn{",card(AA):3:0,"}{|c|}{OD:",K.tl:3:0,"-",
    S.tl:3:0,"}";);put "\\"/;
put "\cline{3-",(sum(OD,1)*card(AA)+2):4:0,"}"/;
put "Link & Flow ";loop(OD(K,S),loop(AA,put " & ",classalpha(AA):4:2;););put "\\"/;
put "\hline"/;
loop(LINK(I,J),
         put I.tl:2:0,"-",J.tl:2:0," & ",SUM((OD,AA),Q.l(LINK,OD,AA)):6:1;loop(OD(K,S),
         loop(AA,put " & ";if(Q.l(LINK,OD,AA)<0.01,put "-":6:0;
         else put Q.l(LINK,OD,AA):6:1;);););put "\\"/;
);
put "\hline"/;
put "\multicolumn{2}{|c|}{$u_\alpha$} ";loop(OD(K,S),loop(AA,put " & ",
    traveltime(OD,AA):6:1;););put "\\"/;
put "\hline"/;
put "\end{tabular}"/;
put "\caption{Cars and motorcycles example. Mixed BPR model. Link flows disaggregated by
                OD and $\alpha$-classes for the Nguyen-Dupuis network (congested case).}"/;
put "}"/;
put "\end{table}"/;


put /;
put "\begin{table}"/;
put "\centering"/;
put "\begin{tabular}{|c|c|";loop(SS,put "c";);put "|";loop(AA, put "c";);put "|}"/;
put "\hline"/;
put " OD & Routes & \multicolumn{",card(SS):2:0,"}{|c|}{path links} &
 \multicolumn{",card(AA):2:0,"}{|c|}{Classes}\\"/;
put " & & \multicolumn{",card(SS):2:0,"}{|c|}{} ",loop(AA,put " & ",ord(AA):2:0;);
put " \\"/;
put "\hline"/;
aux=0;
loop(OD(K,S), loop(PATH$(pathorigin(PATH)=ord(K) and pathdestination(PATH)=ord(S)),
    aux=aux+1;put ord(K):2:0,"-",ord(S):2:0 "&";put aux:3:0;
loop(SS, if(pathlinks(PATH,SS)>0,put " & ",pathlinks(PATH,SS):3:0;else put " & ";););
loop(AA,if(abs(traveltime(OD,AA)-routecost(path,aa))<0.00001, PUT " & {\bf ",
    routecost(path,aa):8:3,"}"; else PUT " & ",routecost(path,aa):8:3;););PUT "\\"/;);
```

```
put "\hline"/;
);
put "\end{tabular}"/;
put "\caption{Cars and motorcycles example. Route travel times classified by OD and
        $\alpha$-classes for the Nguyen-Dupuis network (uncongested case).
         Used routes are boldfaced.}"/;
put "\end{table}"/;
putclose out1;

*******************************************************************************************
* Print the codes to create the figures in Mathematica
*******************************************************************************************
put out1;
put "Arrowpos = 0.4;"/;
put "ArrowSize = 0.15;"/;
put "FontSize1 = 12;"/;
put "Radious = 0.4;"/;
put "Points = {";
loop(I,if(ord(I)<card(I),put "{",U(I):7:3,",",V(I):7:3,"},"/;  else put "{",U(I):7:3,",",
        V(I):7:3,"}};"/;););
put /;
nn=0;
nn=0;
nlinks=SUM(LINK,1);
put "LinkNodes = {";
loop(LINK(I,J),nn=nn+1;if(nn<nlinks, put "{",ord(I):3:0,",",ord(J):3:0,"},"/;
else put "{",ord(I):3:0,",",ord(J):3:0,"}}"/; ););
put "bb = NetworkPlot[Points, LinkNodes, Arrowpos, ArrowSize, FontSize1, Radious];"/;
put "Show[bb, ImageSize -> 360, Axes -> False, AxesLabel -> None, AxesStyle -> False];"/;

nlinks=SUM(LINK,1);
put "nlinks=",nlinks:5:0/;
nOD=SUM(OD,1);
put "Frequencies1={";
ilink=0;
loop(LINK,
         ilink=ilink+1;
         put "{";
         iOD=0;
         loop(OD,
                 iOD=iOD+1;
                 put "{";
                 loop(AA,
                         if(ord(AA)=card(AA),
                                 if(iOD=nOD,
                                         if(ilink=nlinks,
                                            put Frequencies1(LINK,OD,AA):8:4,"}}};"/;
                                         else
                                            put Frequencies1(LINK,OD,AA):8:4,"}},"/;
```

```
                                        );
                                else
                                        put Frequencies1(LINK,OD,AA):8:4,"},"/;
                                );
                        else
                                put Frequencies1(LINK,OD,AA):8:4,","/;
                        );
                );
        );
);
mm=0;
iAA=0;
Loop(AA,
        iAA=iAA+1;
        iOD=0;
        loop(OD(K,S),
                iOD=iOD+1;
                put "OriginNode=",K.tl:5:0,";"/;
                put "DestinationNode=",S.tl:5:0,";"/;
                mm=mm+1;
                nn=0;
                indic=0;
                nlinks=SUM(LINK$(Frequencies1(LINK,OD,AA)>0),1);
                loop(LINK(I,J)$(Frequencies1(LINK,OD,AA)>0),
                        if(nn=0,
                                indic=1;
                                put "LinkNodes1= {";
                        );
                        nn=nn+1;
                        if(nn<nlinks,
                                put "{",ord(I):3:0,",",ord(J):3:0,"},"/;
                        else
                                put "{",ord(I):3:0,",",ord(J):3:0,"}};"/;
                        );
                );

                if(indic=1,
                        put "ff = Table[Frequencies1[[i,",iOD:5:0,",",iAA:5:0,"]],
                                   {i, 1,", SUM(LINK,1):5:0,"}]"/;
                        put "a[",mm:5:0,"]=NetworkPlot2[Points, LinkNodes, LinkNodes1,
                         Arrowpos, ArrowSize, FontSize1, Radious,ff];"/;
                );
        );
);
put "Show[GraphicsGrid[{{a[1], a[5], a[9]}, {a[13], a[17], a[21]}}], ImageSize -> 800,
    Axes -> False, AxesLabel -> None, AxesStyle -> False]"/;
put "Show[GraphicsGrid[{{a[2], a[6], a[10]}, {a[14], a[18], a[22]}}], ImageSize -> 800,
    Axes -> False, AxesLabel -> None, AxesStyle -> False]"/;
put "Show[GraphicsGrid[{{a[3], a[7], a[11]}, {a[15], a[19], a[23]}}], ImageSize -> 800,
```

```
    Axes -> False, AxesLabel -> None, AxesStyle -> False]"/;
put "Show[GraphicsGrid[{{a[4], a[8], a[12]}, {a[16], a[20], a[24]}}], ImageSize -> 800,
    Axes -> False, AxesLabel -> None, AxesStyle -> False]"/;
putclose out2;

*******************************************************************************************
* Print the Matlab codes to compute the paths used in the model without path enumeration
*******************************************************************************************
put out3;

SCALAR iii;
put "maxss=",card(SS):3:0,";"/;
put "QQQ=zeros(",nlinks:3:0,",",nOD:3:0,",",card(AA):3:0,");"/;
iii=0;
loop(OD,iii=iii+1;loop(LINK(I,J),
put "QQQ(",order(LINK):3:0,",",iii:3:0,",:)=[";
loop(AA,if(Q.l(LINK,OD,AA)>CRITICALFLOW,put sign(Q.l(LINK,OD,AA)):2:0; else put 0:2:0;););
    put "];"/;);
);
put "Links=["/;
loop(LINK(I,J),put I.tl:5:0,J.tl:5:0,";"/;);
put "];"/;
put "U=["/;
loop(I,put U(I):8:3," "/;);
put "];"/;
put "V=["/;
loop(I,put V(I):8:3," "/;);
put "];"/;
put "ORIGIN=["/;
loop(ORIGIN(I),put I.tl:5:0," "/;);
put "];"/;
put "DESTINATION=["/;
loop(DESTINATION(I),put I.tl:5:0," "/;);
put "];"/;
put "ODExtremes=["/;
loop(OD(K,S),put K.tl:3:0,",",S.tl:3:0/;);
put "];"/;
putclose out3;
```

A.2.2 Matlab code for the program of path enumeration

This section deals with the Matlab code for the program of path enumeration. This program permits obtaining the used paths by all classes when the assignment model without path enumeration ((8.45) subject to (8.40)-(8.42) and (8.44))is applied. The set of paths obtained by this program coincides with the set of used paths obtained with the approach with path enumeration.

Main program

```
clc
clear all
clear
close all

global maxss;
global NodeLinks;
global TestedLinks;
global TotalNodeLinks;
global ActualPath;
global ActualNode;
global PreviousNode;
global ActualLink;
global PreviousLink;
global Links;
global UsedNode;
global OriginNode;
global DestinationNode;
global Paths;
global ODLINKS;
global nPaths;
global nlinks;
global PathNodes;
global PathLinks;
global nodesinpath;
global linksinpath;
global aa;
global nclasses;
global OD;
global QQQ;

dataNGuyenDupuisMixed;

[nOD,nn]=size(ODExtremes);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Prints all ODs
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for OD=1:nOD
```

```
    fprintf('OD %3.0f : %3.0f - %3.0f\n',OD,ODExtremes(OD,1),ODExtremes(OD,2));
end

[nlinks,nn]=size(Links);
nNodes=max(max(Links));
Paths=zeros(1,nlinks);
ODPaths=zeros(1,nlinks);
TestedLinks=zeros(1,nNodes);
TotalNodeLinks=zeros(1,nNodes);
for i=1:nlinks
    TotalNodeLinks(Links(i,1))=TotalNodeLinks(Links(i,1))+1;
    NodeLinks(Links(i,1),TotalNodeLinks(Links(i,1)))=i;
end
alllinks=1:nlinks;
nclasses=length(QQQ(1,1,:));
numberofPaths=zeros(nOD,nclasses);
nPaths=0;
fprintf('Start process\n');
ODlist=[1,2,3,4];

for ii=1:length(ODlist)
    OD=ODlist(ii);
    for aa=1:nclasses
        ODLINKS=zeros(nlinks);
        PathNodes=zeros(1,nNodes);
        nodesinpath=0;
        linksinpath=0;
        ActualPath=zeros(1,nlinks);
        UsedNode=zeros(1,nNodes);
        OriginNode=ODExtremes(OD,1);
        DestinationNode=ODExtremes(OD,2);
        ActualNode=OriginNode;
        TestedLinks(ActualNode)=0;
        PreviousLink=0;
        ActualLink=0;
        done=0;
        started=0;
        while done==0 && ActualNode>0
            if ActualNode==DestinationNode
                nodesinpath=nodesinpath+1;
```

```
                PathNodes(nodesinpath)=ActualNode;
                linksinpath=linksinpath+1;
                ActualPath(linksinpath)=ActualLink;
                StoresPath;
                for i=1:linksinpath
                    if ActualPath(i)>0
                        ODLINKS(ActualPath(i))=ActualPath(i);
                    end
                end
                BackNodeEnd;
                NewNode;
            elseif ActualNode==OriginNode && started==1
                done=1;
            else
                UsedNode(ActualNode)=1;
                nodesinpath=nodesinpath+1;
                PathNodes(nodesinpath)=ActualNode;
                if ActualLink>0
                    linksinpath=linksinpath+1;
                    ActualPath(linksinpath)=ActualLink;
                end
                started=1;
                NewNode;
            end
        end
        numberofPaths(OD,aa)=nPaths;
    end
end

previous=1;
for OD=1:nOD
    for aa=1:nclasses
        fprintf('class %3.0f OD %5.0f\n',aa,OD);
        PrintPaths(previous,numberofPaths(OD,aa));
        previous=numberofPaths(OD,aa)+1;
    end
end
fprintf('%9.0f paths found\n',length(Paths(:,1)));

UniquePaths=union(Paths,Paths,'rows');
```

```
length(UniquePaths);
Generatepathlinks(UniquePaths);
fprintf('%9.0f Unique paths found\n',length(UniquePaths(:,1)));
```

Function Backnode

```
function Backnode

global TestedLinks;
global ActualPath;
global ActualNode;
global PreviousNode;
global ActualLink;
global PreviousLink;
global UsedNode;
global PathNodes;
global nodesinpath;
global linksinpath;

TestedLinks(ActualNode)=0;
UsedNode(ActualNode)=0;
if nodesinpath>1
    ActualNode=PathNodes(nodesinpath-1);
else
    ActualNode=0;
end
PathNodes(nodesinpath)=0;
nodesinpath=nodesinpath-1;
if nodesinpath>1
    PreviousNode=PathNodes(nodesinpath-1);
end
if linksinpath>0
    ActualPath(linksinpath)=0;
end
linksinpath=linksinpath-1;
if linksinpath>0
    ActualLink=ActualPath(linksinpath);
end
if linksinpath>1
    PreviousLink=ActualPath(linksinpath-1);
```

```
end
NewNode;
```

Function NewNode

```
function NewNode

global NodeLinks;
global ActualLink;
global PreviousLink;
global Links;
global ActualNode;
global TestedLinks;
global UsedNode;
global PreviousNode;
global TotalNodeLinks;
global ActualPath;
global PathNodes;
global nodesinpath;
global linksinpath;
global aa;
global nclasses;
global OD;
global QQQ;

if ActualNode>0
    ss=TestedLinks(ActualNode)+1;
    if ss>TotalNodeLinks(ActualNode)
        BackNode;
    else
        TestedLinks(ActualNode)=ss;
        TemptativeLink=NodeLinks(ActualNode,ss);
        if QQQ(TemptativeLink,OD,aa)==1
            TemptativeNode=Links(TemptativeLink,2);
            if UsedNode(TemptativeNode)==0
                PreviousNode=ActualNode;
                ActualNode=TemptativeNode;
                PreviousLink=ActualLink;
                ActualLink=TemptativeLink;
            else
```

```
                NewNode
            end
        else
            NewNode;
        end
    end
end
```

Function BacknodeEnd

```
function BacknodeEnd

global NodeLinks;
global TestedLinks;
global ActualPath;
global ActualNode;
global PreviousNode;
global ActualLink;
global PreviousLink;
global UsedNode;
global PathNodes;
global nodesinpath;
global linksinpath;

TestedLinks(ActualNode)=0;
UsedNode(ActualNode)=0;
if nodesinpath>1
    ActualNode=PathNodes(nodesinpath-1);
else
    ActualNode=0;
end
PathNodes(nodesinpath)=0;
nodesinpath=nodesinpath-1;
if nodesinpath>1
    PreviousNode=PathNodes(nodesinpath-1);
end
ActualPath(linksinpath)=0;
linksinpath=linksinpath-1;
if linksinpath>1
    ActualLink=ActualPath(linksinpath);
```

```
else
    ActualLink=0;
end
if linksinpath>1
    PreviousLink=ActualPath(linksinpath-1);
else
    PreviousLink=0;
end
```

Function Generatepathlinks

```
function Generatepathlinks(UniquePaths)

filename='pathlinks.gms';
fid=fopen(filename,'w');
fprintf(fid,'TABLE\n');

fprintf(fid,'pathlinks(PATH,SS)\n');
[m,n]=size(UniquePaths);

jmax=0;
for i=1:m
    for j=1:n
        if UniquePaths(i,j)>0
            jmax=max(jmax,j);
        end
    end
end
fprintf(fid,'    ');
for j=1:jmax
    fprintf(fid,'%4.0f',j);
end
fprintf(fid,'\n');
for i=1:m
    fprintf(fid,'%-4.0f',i);
    for j=1:jmax
        fprintf(fid,'%4.0f',UniquePaths(i,j));
    end
    fprintf(fid,'\n');
end
```

```
fprintf(fid,';\n');
fclose(fid);
```

Function PrintPaths

```
function PrintPaths(n1,n2)

global nPaths;
global Paths;
global nlinks;

fprintf('Paths\n');
for i=n1:n2
    for j=1:nlinks
        if Paths(i,j)>0
            fprintf('%4.0f',Paths(i,j));
        end
    end
    fprintf('\n');
end
```

Function StoresPath

```
function StoresPath

global ActualPath;
global nPaths;
global Paths;
global nlinks;
global ActualLink;
global PathNodes;
global ActualNode;
global nodesinpath;
global linksinpath;
global ODLINKS;

nPaths=nPaths+1;
for i=1:linksinpath
    Paths(nPaths,i)=ActualPath(i);
end
```

DupuisEquilibriumMixedMathlab

The following code is generated by the GAMS code in the previous section and it is the input for the enumeration paths program.

```
maxss=  5;
QQQ=zeros(  5,  4,  6);
QQQ(  1,  1,:)=[ 0 0 0 1 1 0];
QQQ(  2,  1,:)=[ 1 1 1 0 1 1];
QQQ(  3,  1,:)=[ 0 0 0 0 0 0];
QQQ(  4,  1,:)=[ 0 0 0 0 0 0];
QQQ(  5,  1,:)=[ 0 0 0 1 1 0];
QQQ(  6,  1,:)=[ 0 0 0 0 0 0];
QQQ(  7,  1,:)=[ 0 0 0 1 1 0];
QQQ(  8,  1,:)=[ 0 0 0 0 0 0];
QQQ(  9,  1,:)=[ 0 0 0 1 1 0];
QQQ( 10,  1,:)=[ 0 0 0 0 0 0];
QQQ( 11,  1,:)=[ 1 1 1 1 1 1];
QQQ( 12,  1,:)=[ 0 0 0 0 0 0];
QQQ( 13,  1,:)=[ 0 0 0 0 0 0];
QQQ( 14,  1,:)=[ 0 0 0 0 0 0];
QQQ( 15,  1,:)=[ 0 0 0 0 0 0];
QQQ( 16,  1,:)=[ 0 0 0 0 0 0];
QQQ( 17,  1,:)=[ 0 0 0 0 0 0];
QQQ( 18,  1,:)=[ 1 1 1 0 1 1];
QQQ( 19,  1,:)=[ 0 0 0 0 0 0];
QQQ(  1,  2,:)=[ 1 1 0 1 1 1];
QQQ(  2,  2,:)=[ 0 1 1 0 0 0];
QQQ(  3,  2,:)=[ 0 0 0 0 0 0];
QQQ(  4,  2,:)=[ 0 0 0 0 0 0];
QQQ(  5,  2,:)=[ 0 0 0 1 1 1];
QQQ(  6,  2,:)=[ 1 1 0 0 0 0];
QQQ(  7,  2,:)=[ 0 1 1 0 0 1];
QQQ(  8,  2,:)=[ 0 0 0 1 1 1];
QQQ(  9,  2,:)=[ 0 0 0 0 0 0];
QQQ( 10,  2,:)=[ 0 1 1 0 0 1];
QQQ( 11,  2,:)=[ 0 0 0 0 0 0];
QQQ( 12,  2,:)=[ 1 0 0 0 0 0];
QQQ( 13,  2,:)=[ 1 1 0 0 0 0];
QQQ( 14,  2,:)=[ 1 0 0 1 1 1];
QQQ( 15,  2,:)=[ 0 0 0 0 0 0];
```

```
QQQ( 16,  2,:)=[ 1 1 1 1 1 1];
QQQ( 17,  2,:)=[ 0 1 1 0 0 0];
QQQ( 18,  2,:)=[ 0 0 0 0 0 0];
QQQ( 19,  2,:)=[ 1 1 0 0 0 0];
QQQ(  1,  3,:)=[ 0 0 0 0 0 0];
QQQ(  2,  3,:)=[ 0 0 0 0 0 0];
QQQ(  3,  3,:)=[ 0 1 1 1 1 1];
QQQ(  4,  3,:)=[ 1 1 0 0 0 0];
QQQ(  5,  3,:)=[ 0 1 1 1 1 1];
QQQ(  6,  3,:)=[ 0 0 1 0 0 0];
QQQ(  7,  3,:)=[ 0 1 1 1 1 1];
QQQ(  8,  3,:)=[ 0 0 0 0 0 0];
QQQ(  9,  3,:)=[ 0 1 0 1 1 1];
QQQ( 10,  3,:)=[ 0 0 1 0 0 0];
QQQ( 11,  3,:)=[ 0 1 0 1 1 1];
QQQ( 12,  3,:)=[ 1 1 1 0 0 0];
QQQ( 13,  3,:)=[ 0 0 0 0 0 0];
QQQ( 14,  3,:)=[ 1 1 1 0 0 0];
QQQ( 15,  3,:)=[ 1 1 1 0 0 0];
QQQ( 16,  3,:)=[ 0 0 0 0 0 0];
QQQ( 17,  3,:)=[ 0 0 0 0 0 0];
QQQ( 18,  3,:)=[ 0 0 0 0 0 0];
QQQ( 19,  3,:)=[ 0 0 0 0 0 0];
QQQ(  1,  4,:)=[ 0 0 0 0 0 0];
QQQ(  2,  4,:)=[ 0 0 0 0 0 0];
QQQ(  3,  4,:)=[ 0 1 1 1 1 1];
QQQ(  4,  4,:)=[ 1 1 0 0 0 0];
QQQ(  5,  4,:)=[ 0 0 0 1 1 1];
QQQ(  6,  4,:)=[ 0 1 1 0 0 0];
QQQ(  7,  4,:)=[ 0 0 0 0 0 0];
QQQ(  8,  4,:)=[ 0 0 0 1 1 1];
QQQ(  9,  4,:)=[ 0 0 0 0 0 0];
QQQ( 10,  4,:)=[ 0 0 0 0 0 0];
QQQ( 11,  4,:)=[ 0 0 0 0 0 0];
QQQ( 12,  4,:)=[ 0 0 0 0 0 0];
QQQ( 13,  4,:)=[ 1 1 1 0 0 0];
QQQ( 14,  4,:)=[ 0 0 0 1 1 1];
QQQ( 15,  4,:)=[ 0 0 0 0 0 0];
QQQ( 16,  4,:)=[ 0 0 0 1 1 1];
QQQ( 17,  4,:)=[ 0 0 0 0 0 0];
```

```
QQQ( 18,  4,:)=[ 0 0 0 0 0 0];
QQQ( 19,  4,:)=[ 1 1 1 0 0 0];
Links=[
1    5    ;
1    12   ;
4    5    ;
4    9    ;
5    6    ;
5    9    ;
6    7    ;
6    10   ;
7    8    ;
7    11   ;
8    2    ;
9    10   ;
9    13   ;
10   11   ;
11   2    ;
11   3    ;
12   6    ;
12   8    ;
13   3    ;
];
U=[
   0.000
  10.000
  10.000
   2.043
   3.871
   5.914
   7.957
  10.000
   5.914
   5.914
   7.957
   1.935
   7.957
];
V=[
   6.552
```

```
   3.448
   0.000
   0.000
   3.448
   6.552
   6.552
  10.000
   0.000
   3.448
   3.448
  10.000
   0.000
];
ORIGIN=[
1
4
];
DESTINATION=[
2
3
];
ODExtremes=[
1  ,2
1  ,3
4  ,2
4  ,3
];
```

A.3 Gams code for the Bayesian matrix estimation model

In this section the GAMS programming codes of the Bayesian Matrix Estimation Model presented in Chapter 9, Section 9.4 is shown. In particular, we show the program codes for the Nguyen-Dupuis example (including its data).

```
  $Title Gamma Bayesian model
*******************************************************************************
*The output file is defined
*******************************************************************************
file out/gamma.out/;
file out1/gammaLatex.out/;
put out;
OPTION nlp=CONOPT;
```

```
SETS
simul            number of simulations/1*100/
iter             maximum number of Global iterations /1*100/
iter_2           maximum number of iterations for DTRUE smulation /1*2/
I                set of nodesnodes /1*13/
LINK(I,I)        set of links
OBSERVED(I,I)    observed links  /1.5,12.8,9.10,9.13/
OD(I,I)          set of OD pairs
KK               terms of gamma function/1*6/
case             cases for m and n values/1*5/
param            link parameters table headers / a, k /;

ALIAS(I,J,K,S);

SCALARS
m                Sample size of fictitious sample
n                sample size of the observed sample
lambda           wighting factor for WMV model /0.000001/
rho              relaxation factor to control the OD matrix updating /0.5/
etaOD            weight factor for least squares OD estimation  /0.2/
epsilon1         auxiliary scalar/0.000000000001/
epsilon2         auxiliary scalar/0.000000000001/
nn               auxiliary scalar
mmm              auxiliary scalar
error            error in bayes OD flows updating
tol              tolerance error for bayes OD flows updating /0.01/
iteration        scalar for iteration control
phi              auxiliary scalar
e1               auxiliary value of the first component of the hypermarater
e2               auxiliary value of the second component of the hypermarater;

PARAMETERS
simulatedOD4(simul,I,I)  vector of estimated OD flows by simulation for case 4
simulatedODLS(simul,I,I) vector of estimated OD flows by simulation for LS method
simulatedTrue(simul,I,I) vector of true OD flows by simulation
cputime(case)            vector of CPU times by case
Ca(I,J)                  uncongested link costs
Qa(I,J)                  link capacities
alpha(I,J)               link proportionality constants for the BPR function
GAMMA(I,J)               link power constants for the BPR function
delta(I,I)               kronecker's delta matrix
DTRUE(I,J)               true OD flows
D(I,J)                   OD flows
D0(I,J)                  prior OD flows
D00(I,J)                 stored prior OD flows
D1(I,J)                  modified prior OD flows
DLS(I,J)                 OD flows estimated by LS method
eta1(I,J)                eta1 hyperparameters
eta2(I,J)                eta2 hyperparameters
```

```
etapost1(I,J)            posterior eta1 hyperparameters
etapost2(I,J)            posterior eta1 hyperparameters
BETA(I,J,K,S)            beta matrix
WREAL(I,I)               real link flows obtained from the WMV model
W(I,I)                   total link flow
W0(I,I)                  prior total link flow
WLS(I,I)                 link flows estimated by LS method
LINKA(case,I,I)          auxiliary parameter to store the estimated link flows by case
TA(case,I,I)             auxiliary parameter to store the estimated OD flows by case
OBSERVATION(I,J)         observed link flows
COEFF(KK)                Gamma funcion parameters
/1       76.18009172947146
2        -86.50532032941677
3        24.01409824083091
4        -1.231739572450155
5        0.001208650973866179
6        -0.000005395239384953/
;

*****************************************************************************************
*The parameters for the BPR function for each link are defined
*****************************************************************************************

table arc_cost(I,I,param)  link parameter data
           a       k
1.5        7      70
1.12       9      56
4.5        9      56
4.9       12      70
5.6        3      42
5.9        9      42
6.7        5      70
6.10       5      28
7.8        5      70
7.11       9      70
8.2        9      70
9.10      10      56
9.13       9      56
10.11      6      70
11.2       9      56
11.3       8      56
12.6       7      14
12.8      14      56
13.3      11      56
;

Ca(i,j) = arc_cost(i,j,"a");
*****************************************************************************************
*The links are defined using the parameter a in arc_cost table
```

```
*******************************************************************************************
LINK(i,j)=Ca(i,j);
Qa(i,j) = arc_cost(i,j,"k");
alpha(LINK(I,J)) = 1;
GAMMA(LINK)=4;
delta(I,I)=1;

TABLE
D0(I,J) prior trip matrix
         1       2       3       4
1                0.4     0.8
4                0.6     0.2;
D0(I,J)=D0(I,J)*100;
D00(I,J)=D0(I,J);
*******************************************************************************************
*The set of OD pairs are defined using the prior OD matrix
*******************************************************************************************
OD(I,J)=no;
OD(I,J)$(D0(I,J)>0)=yes;

*******************************************************************************************
*The link parameters and the prior OD matrix are printed
*******************************************************************************************
  mmm=0;
  put "LINK PARAMETERS"/;
  put "        ca    alpha   qa    gammaa"/;
  loop(Link(I,J),mmm=mmm+1;put I.tl:2:0,"-",J.tl:1:0,
  Ca(I,J):7:1, alpha(I,J):7:1, Qa(I,J):7:1, gamma(I,J):7:1/;);
  put /;
  put "Total number of links=",mmm:6:0/;
  put /;
  put "PRIOR OD FLOWS"/;
  loop(I,put I.tl:2:0;loop(J,put D0(I,J):6:2;);put/;);
  put /;

*******************************************************************************************
*The variables are defined
*******************************************************************************************
VARIABLES
Z               objective variable for models
med             mean of the disaggregated link flows;

POSITIVE VARIABLES
V(I,J,K,S)      disssagregate link flows
T(K,S)          OD flows associated with the observed sample
WW(I,I)         link flows
TT              auxiliary variable to compute the mode;

EQUATIONS
```

```
*****************************************************************************************
*Equations for solving the start problem
*****************************************************************************************
zstart          objective function for start model (obtains a feasible solution)
*****************************************************************************************
*Equations for solving the WMV problem
*****************************************************************************************
zWMV             objective function for WMV model
balance1         flow balance at each node
mean             definition of the mean of the disaggregated link flows
*****************************************************************************************
*Equations for solving the Least Squares model
*****************************************************************************************
zLS              objective funtion for Least Squares model
propor           flow conservation function
*****************************************************************************************
*Equations for solving the  Gamma Bayesian  model
*****************************************************************************************
zGBay            numeric equivalent objective function to compute the mode of the Gamma model;


*****************************************************************************************
*The equations are defined
*****************************************************************************************
zstart..z=e=SUM((LINK,OD),V(LINK,OD));

zWMV..z=e=SUM(LINK,(Ca(LINK)*((SUM(OD,V(LINK,OD))*(1+ALPHA(LINK)*(SUM(OD,V(LINK,OD))/
    Qa(LINK))**(GAMMA(LINK))/(GAMMA(LINK)+1))))))
    +SUM((LINK,OD),SQR(V(LINK,OD)-med))*lambda/sum((LINK,OD),1);
balance1(I,OD(K,S))$(ord(K) ne ord(S))..D(OD)*(delta(I,K)-delta(I,S))=e=
    SUM(LINK(I,J),V(LINK,OD))-SUM(LINK(J,I),V(LINK,OD));
mean..med=e=SUM((LINK,OD),V(LINK,OD))/sum((LINK,OD),1);

zLS..z=e=etaOD*SUM(OD,sqr((T(OD)-D0(OD))/D0(OD)))
    +SUM(LINK$(OBSERVATION(LINK)>0),sqr((WW(LINK)-OBSERVATION(LINK))/OBSERVATION(LINK)));
propor(LINK)..WW(LINK)=e=SUM(OD,BETA(LINK,OD)*T(OD));

zGBay..z=e=e1*TT-e2*(-(TT + 5.5) + (TT + 0.5)*log(TT + 5.5)
    +log(2.506628274631*(1.00000000019 + Sum(kk,Coeff(kk)/(TT + ord(kk))))/max(TT,epsilon1)));

*****************************************************************************************
*The models are defined
*****************************************************************************************
MODEL start/zstart,balance1/;
MODEL WMV/zWMV,balance1,mean/;
MODEL LS/zLS,propor/;
MODEL GammaBayes/zGBay/;

*****************************************************************************************
```

```
*The true OD flows are simulated
*******************************************************************************************
loop(simul,
D0(OD)=D00(OD);
loop(iter_2,D(OD)=uniform(1,1.3)*D0(OD););
         put "REAL OD FLOWS"/;
         loop(I,put I.tl:2:0;loop(J,put D(I,J):6:2;);put/;);
         put /;
         DTRUE(OD)=D(OD);
*******************************************************************************************
*Solves the start problem to obtain a feasible solution
*******************************************************************************************
         SOLVE start USING lp MINIMIZING z;
         put "zstart=",z.l:12:3," modelstat=",start.modelstat," solvestat=",start.solvestat,"
         resusd=",start.resusd:12:8/;
*******************************************************************************************
*Solves the WMV problem with the real OD flows to obtain the real link flows (observed)
*******************************************************************************************
         SOLVE WMV USING nlp MINIMIZING z;
         put "WMV=",z.l:12:3," modelstat=",WMV.modelstat," solvestat=",WMV.solvestat,
         " resusd=",WMV.resusd:12:8/;
*******************************************************************************************
*The observed link flows are stored
*******************************************************************************************
         OBSERVATION(LINK)=0;
         WREAL(LINK)=SUM(OD,V.l(LINK,OD));
         OBSERVATION(OBSERVED)=WREAL(OBSERVED);
         put "OBSERVATIONS"/;
         loop(OBSERVED(I,J),put I.tl:4:0,"-",J.tl:4:0,OBSERVATION(OBSERVED):9:2/;);
*******************************************************************************************
*The prior OD flows are stored and printed
*******************************************************************************************
         D(OD)=D0(OD);
         put "PRIOR OD FLOWS"/;
         loop(I,put I.tl:2:0;loop(J,put D(I,J):6:2;);put/;);
         put /;
*******************************************************************************************
*Solves the start problem to obtain a feasible solution
*******************************************************************************************
         SOLVE start USING lp MINIMIZING z;
         put "zstart=",z.l:12:3," modelstat=",start.modelstat," solvestat=",start.solvestat,
         " resusd=",start.resusd:12:8/;
*******************************************************************************************
*Solves the WMV problem with the prior OD flows to obtain the prior link flows
*******************************************************************************************
         SOLVE WMV USING nlp MINIMIZING z;
         put "WMV=",z.l:12:3," modelstat=",WMV.modelstat," solvestat=",
         WMV.solvestat," resusd=",WMV.resusd:12:8/;
         W(LINK)=SUM(OD,V.l(LINK,OD));
```

```
         W0(LINK)=W(LINK);

*******************************************************************************************
*Beginning the case iterations
*******************************************************************************************
         loop(case,
*******************************************************************************************
*The cases for m and n values are defined
*******************************************************************************************
                 if(ord(case)=1,m=100;n=1;phi=0;);
                 if(ord(case)=2,m=10;n=10;phi=0;);
                 if(ord(case)=3,m=1;n=10;phi=0;);
                 if(ord(case)=4,m=1;n=100;phi=0;);
                 if(ord(case)=5,m=1;n=100;phi=0.3;);
                 put "CASE=",case.tl:5:0, " m=" m:3:0, " n=" n:3:0/;
*******************************************************************************************
*The prior hyperparameters are calculated
*******************************************************************************************
                 D0(OD)=(1+phi)*D0(OD)
                 if(ord(case)=5, D1(OD)=D0(OD););
                 eta1(OD)=m*log(max(D0(OD),epsilon1));
                 eta2(OD)=m;
                 put "PRIOR HYPERPARAMETERS"/;
                 loop(OD(I,J), put I.tl:3:0, "-", J.tl:3:0, eta1(OD):6:2, eta2(OD):6:2;
                 put /;);put/;
*******************************************************************************************
*The prior OD flows are stored for each case iteration and the error is defined
*******************************************************************************************
                 error=1000;
                 D(OD)=D0(OD);
*******************************************************************************************
*Beginning the convergence iterations
*******************************************************************************************
                 cputime(case)=0;
                 loop(iter$(error>tol),
                         iteration=ord(iter);
                         put "CASE=",case.tl:5:0, " m=" m:3:0, " n=" n:3:0,
                            " iter= " iteration:4:0/;
*******************************************************************************************
*The WMV model is solved with the actual OD flows
*******************************************************************************************
                         SOLVE WMV USING nlp MINIMIZING z;
                         put "WMV=",z.l:12:3," modelstat=",WMV.modelstat," solvestat=",
                         WMV.solvestat," resusd=",WMV.resusd:12:8/;
                         cputime(case)=cputime(case)+WMV.resusd;
*******************************************************************************************
*The Beta matrix is calculated
*******************************************************************************************
                         loop(OD,if(D(OD)>0,BETA(LINK,OD)=V.l(LINK,OD)/D(OD);
```

```
                          else BETA(LINK,OD)=0;););
*******************************************************************************
*The total link flows are calculated to be used as initial values
*******************************************************************************
                        WW.l(LINK)=SUM(OD,V.l(LINK,OD));
*******************************************************************************
*The LS problem is solved to obtain the "observed" OD flows T.l(OD)
*******************************************************************************
                        SOLVE LS USING nlp MINIMIZING z;
                        put "LS=",z.l:12:3," modelstat=",LS.modelstat," solvestat=",
                        LS.solvestat," resusd=",LS.resusd:12:8/;
                        cputime(case)=cputime(case)+LS.resusd;
                        put "OD FLOWS"/;
                        loop(I,put I.tl:2:0;loop(J,put T.l(I,J):6:2;);put/;);
                        put /;
*******************************************************************************
*The posterior hyperparameters are calculated
*******************************************************************************
                        etapost1(OD)=eta1(OD)+n*log(max(T.l(OD),epsilon1));
                        etapost2(OD)=eta2(OD)+n;
                        put "POSTERIOR HYPERPARAMETERS"/;
                        loop(OD(I,J),put I.tl:3:0,"-",J.tl:3:0,etapost1(OD):6:2,
                           etapost2(OD):6:2; put /;);
                        put/;
*******************************************************************************
*The Gamma Bayesian models are solved to obtain the OD mode estimates
*******************************************************************************
                        loop(OD(I,J),
                                TT.l=D(OD);
                                e1=etapost1(OD);
                                e2=etapost2(OD);
                                SOLVE GammaBayes USING dnlp MAXIMIZING z;
                                put "GammaBayes z=",z.l:15:8," modelstat=",
                                GammaBayes.modelstat," solvestat=",GammaBayes.solvestat,
                                " resusd=", GammaBayes.resusd:12:8/;
                                cputime(case)=cputime(case)+GammaBayes.resusd;
                                T.l(OD)=TT.l;
                        );
*******************************************************************************
*The results are stored for the corresponding case
*******************************************************************************
                        TA(case,OD)=T.l(OD);
                        LINKA(case,LINK)=SUM(OD,T.l(OD)*BETA(LINK,OD));
*******************************************************************************
*The current OD flows are compared with the previous
*******************************************************************************
                        put "PREVIOUS OD FLOWS "/;
                        loop(I,put I.tl:2:0;loop(J, put D(I,J):6:2;);put/;);
                        put /;
```

```
                        put "CURRENT OD FLOWS "/;
                        loop(I,put I.tl:2:0;loop(J, put T.l(I,J):6:2;);put/;);
                        put /;
*****************************************************************************************
*The error is calculated to control the convergence process
*****************************************************************************************
                        error=SUM(OD,sqr(D(OD)-T.l(OD)));
                        put "ERROR IN THE CURRENT ITERATION:"/;
                        put "iter=",iter.tl:4:0," error=",error:12:9," tol=",tol:12:8/;
*****************************************************************************************
*Update the OD flows stimation
*****************************************************************************************
                        D(OD)=rho*D(OD)+(1-rho)*T.l(OD);
*****************************************************************************************
*End of the convergence procedure
*****************************************************************************************
                );
*****************************************************************************************
*The resulting flows are printed
*****************************************************************************************
                put "Resulting flows"/;
                put "m=" m:3:0/;
                put "n=" n:3:0/;
                put " True Prior and Estimated OD flows "/;
                loop(OD, put DTRUE(OD):8:2, D0(OD):8:2, D(OD):8:2/;);
                put/;
                put " True Prior and Estimated link flows "/;
                loop(LINK, put WREAL(LINK):8:2, W0(LINK):8:2, SUM(OD,T.l(OD)*
                   BETA(LINK,OD)):8:2/;);
                put/;
                put "*****************************************************************"/;
*****************************************************************************************
*End of the case iteration
*****************************************************************************************
        );
        put "cputime(case)"/;
        loop(case,put case.tl:3:0,cputime(case):12:5/;);
        T.l(OD)=D0(OD);
*****************************************************************************************
*The problem is solved using a standard LS-bilevel procedure
*****************************************************************************************
        error=1000;
        D(OD)=D0(OD);
        loop(iter$(error>tol),
                iteration=ord(iter);
                put "CASE= LS procedure"/;
*****************************************************************************************
*The WMV model is solved
*****************************************************************************************
```

```
                SOLVE WMV USING nlp MINIMIZING z;
                put "WMV=",z.l:12:3," modelstat=",WMV.modelstat," solvestat=",
                WMV.solvestat," resusd=",WMV.resusd:12:8/;
*******************************************************************************
*The Beta matrix is calculated
*******************************************************************************
                loop(OD,if(D(OD)>0,BETA(LINK,OD)=V.l(LINK,OD)/D(OD);else BETA(LINK,OD)=0;););
*******************************************************************************
*The total link flow is calculated and the LS problem solved
*******************************************************************************
                WW.l(LINK)=SUM(OD,V.l(LINK,OD));
                SOLVE LS USING nlp MINIMIZING z;
                put "LS=",z.l:12:3," modelstat=",LS.modelstat," solvestat=",
                LS.solvestat," resusd=",LS.resusd:12:8/;
                put "Standard Method OD FLOWS"/;
                loop(OD, put T.l(OD):8:2/;);
                put /;
*******************************************************************************
*The error is calculated to control the convergence process
*******************************************************************************
                error=SUM(OD,sqr(D(OD)-T.l(OD)));
                put "ERROR IN THE CURRENT ITERATION:"/;
                put "iter=",iter.tl:4:0," error=",error:12:9," tol=",tol:12:8/;
*******************************************************************************
*Update the OD flows
*******************************************************************************
                D(OD)=rho*D(OD)+(1-rho)*T.l(OD);
        );
*******************************************************************************
*End of the standard iteration
*The OD flow estimations using LS method are stored
*******************************************************************************
        DLS(OD)=D(OD);
        WLS(LINK)=WW.l(LINK);
        simulatedOD4(simul,OD)=TA('4',OD);
        simulatedODLS(simul,OD)=DLS(OD);
        simulatedTrue(simul,OD)=DTRUE(OD);
);
*******************************************************************************
*End of the simul iteration
*******************************************************************************

put "SIMULATED OD ESTIMATES"/;
loop(simul,loop(OD,put simulatedOD4(simul,OD):12:5,simulatedODLS(simul,OD):12:5;);put /;);

put "RMSE"/;
loop(OD(K,S), put K.tl:1:0,"-",S.tl:1:0;
put sqrt(sum(simul,power((simulatedOD4(simul,OD)-simulatedTrue(simul,OD))/simulatedTrue
    (simul,OD),2))/card(simul)):12:5,sqrt(sum(simul,power((simulatedODLS(simul,OD)-
```

```
    simulatedTrue(simul,OD))/simulatedTrue(simul,OD),2))/card(simul)):12:5/;
);

*****************************************************************************************
*Link flows obtained for different cases are printed
*****************************************************************************************

put "PRIOR AND POSTERIOR LINK FLOWS"/;
put " LINK   Observed  Prior  m=100  m=10   m=1   m=1"/;
put "                          n=1    n=10   n=1   n=100"/;
loop(LINK(I,J),put I.tl:2:0,"-",J.tl:2:0;if(OBSERVATION(LINK)=0,put "      -   ":6:0;
         else put "   ", OBSERVATION(LINK):6:2;); put "   ",W0(I,J):6:2,"   ",
         LINKA('1',I,J):6:2,"   ", LINKA('2',I,J):6:2,"   ", LINKA('3',I,J):6:2,"   ",
         LINKA('4',I,J):6:2/;
);
put /;

put "PRIOR AND POSTERIOR OD FLOWS"//;
put "                     ";
loop(J,put"   ", J.tl:3:2;);put /;
put "_____________________________________________________________________"/;
loop(I,put I.tl:2:0; put "   Prior     ";loop(J, put"   ", D0(I,J):6:2;);put/;
put "_____________________________________________________________________"/;
put "  ";put "   m=10, n=1 ";loop(J, put"   ", TA('1',I,J):6:2;);put /;
put "  ";put "   m=10, n=10";loop(J, put"   ", TA('2',I,J):6:2;);put /;
put "  ";put "   m=1, n=1  ";loop(J, put"   ", TA('3',I,J):6:2;);put /;
put "  ";put "   m=1, n=10 ";loop(J, put"   ", TA('4',I,J):6:2;);put /;
put "_____________________________________________________________________"/;);
put /;
put /;

put "MODIFIED PRIOR AND POSTERIOR OD FLOWS"//;
put "    Case ";loop(J,put"    ", J.tl:3:2;);put/;
put "_____________________________________________________________________"/;
loop(I,put I.tl:2:0; put "    Prior   ";loop(J, put"    ", D0(I,J):6:2;);put/;
put "_____________________________________________________________________"/;
put "   ";put " Mod. prior ";loop(J, put"   ", D1(I,J):6:2;);put /;
put "   ";put " Posterior  ";loop(J, put"   ", TA('5',I,J):6:2;);put /;
put "_____________________________________________________________________"/;);

*****************************************************************************************
*Results are printed in tables for latex
*****************************************************************************************
put out1;
put "PRIOR AND POSTERIOR LINK FLOWS"//;
put "\begin{table}"/;
put "\centering"/;
put "\begin{tabular}{|c|c|c|c|c|c|c|c|}"/;
put "\hline "/;
```

```
put " LINK & True & Prior & $m=100$ & $m=10$ & $m=1$  & $m=1$  & LS \\"/;
put "  &  &  &  $n=1$ &  $n=10$ & $n=10$ & $n=100$ & \\"/;
put "\hline"/;
loop(LINK(I,J),put I.tl:2:0,"-",J.tl:2:0;if(OBSERVATION(LINK)=0,put "&", WREAL(LINK):6:2;
    else put " & {\bf ", WREAL(LINK):6:2,"}";); put " & ",W0(I,J):6:2," & ",
    LINKA('1',I,J):6:2," & ", LINKA('2',I,J):6:2," & ", LINKA('3',I,J):6:2," & ",
    LINKA('4',I,J):6:2," & ", WLS(I,J):6:2,"\\"/;
);
put "\hline";
put "\end{tabular}"/;
put "\caption{\label{}True, prior and resulting link flows when the proposed algorithm is used"/;
put " for four different cases of relative weight of the prior and the sample with
    respect to information."/;
put " The bolfaced values corresponds to the observed link flows}"/;
put "\end{table}"/;
put /;

put "PRIOR AND POSTERIOR OD FLOWS"//;
put "\begin{table}"/;
put "\centering"/;
put "\renewcommand{\arraystretch}{0.9}"/;
put "\begin{tabular}{|c|c|c|c|c|c|c|}"/;
put "\hline "/;
put " OD &  True & Prior & $m=100$ & $m=10$ & $m=1$  & $m=1$ & LS   \\"/;
put "  &   &  &  $n=1$ &  $n=10$ & $n=10$ & $n=100$ & \\"/;
put "\hline"/;
loop(OD(K,S),put K.tl:2:0,"-",S.tl:2:0;put " & ",DTRUE(K,S):6:2, " & ",D00(K,S):6:2," & ",
    TA('1',K,S):6:2," & ", TA('2',K,S):6:2," & ", TA('3',K,S):6:2," & ", TA('4',K,S):6:2, " & ",
    DLS(K,S):6:2"\\"/;);
put "\hline"/;
put "\end{tabular}"/;
put "\caption{\label{} True, prior and resulting $\nu_{ks}$ values (OD mean flow estimates)
    when the proposed algorithm"/;
put " for four different cases of relative weight of the prior and the sample with respect
    to information. A comparison with an standard LS method is provided.}"/;
put "\end{table}"/;
put /;

put /;
put "PRIOR AND POSTERIOR OD FLOWS"//;
put "\begin{table}"/;
put "\centering"/;
put "\renewcommand{\arraystretch}{0.9}"/;
put "\begin{tabular}{|c|c|c|c|}"/;
put "\hline "/;
put " \multirow{2}{*}{OD} & \multirow{2}{*}{Prior}& \multirow{2}{*}{Mod. prior} & Posterior \\"/;
put "  & & & ($m=1;n=100$)\\"/;
put "\hline";
loop(OD(K,S),put K.tl:2:0,"-",S.tl:2:0;put" & ",D00(K,S):6:2, " & ",D1(K,S):6:2,
```

```
    " & ", TA('5',K,S):6:2,"\\"/;);
put "\hline"/;
put "\end{tabular}"/;
put "\caption{Modified prior and resulting $\nu_{ks}$ values (OD mean flow estimates)
    when solving the gamma model.}"/;
put "\end{table}"/;
```

A.4 Mathematica code for the Observability problem

This section is devoted to the Mathematica implementation of the algorithm explained in Chapter 10, Section 10.6, based on the Γ-algorithm of Castillo and Jubete (2004). This algorithm permits testing if a set of paths is linearly independent and hence, it can be used to find a set of linearly independent paths. In particular, the following codes permit obtaining the results in Table 10.2.

Procedures;

```
(*Procedure to find a non-null pivot*)
FindPivot[t_, type_] :=
 Module[{i = 1, p = 0, done = False, m = Length[type]},
  While[! done && i <= m,
   If[t[[i]] != 0  && type[[i]] == 0, p = i; done = True]; i++];
  Return[p]
  ]
(*Procedure to remove a row (or column)*)
RemoveRow[A_, j_] :=
 If[j > 0 && j <= Length[A],
  Join[Take[A, {1, j - 1}], Take[A, {j + 1, Length[A]}]], A]

(*Procedure to print the tables of the algorithm in Latex*)

PrintTable[T_, a_, label_, t_, p_, ort_, filename_, tablenumber_, m1_,
   h_, type_] :=
 Module[{i, j, s1 = "", s2 = "", s3 = "", aux, ivcont, iwcont, m},
  WriteString[ttt, "\\renewcommand{\\tabcolsep}{-0.02cm}\n"];
  WriteString[ttt, "\\renewcommand{\\arraystretch}{0.7}\n"];
  WriteString[ttt, "\\begin{tabular}{|c|"];
  WriteString[ttt, StringJoin[Table["c", {j, 1, Length[T]}]]];
  WriteString[ttt, "|}\n\hline\n\multicolumn{"];
  WriteString[ttt, ToString[Length[T] + 1]];
  WriteString[ttt, "}{|c|}{{\\bf ", label, "}} \\"];
  WriteString[ttt, "\\ \n\hline\n"];
  If[h != 0, WriteString[ttt, "${\\bf a}_{", ToString[h], "}$ "]];
  aux = If[h != 0, StringJoin["^{", ToString[h], "}"], aux = ""];
  ivcont = 0; iwcont = 0;
  Do[If[type[[j]] == 1, iwcont += 1;
    WriteString[ttt, "& ${\\bf w}", aux, "_{", ToString[iwcont],
     "}$ "], ivcont += 1;
```

```
    WriteString[ttt, "& ${\\bf v}", aux, "_{", ToString[ivcont],
     "}$ "]], {j, 1, Length[T]}];
  WriteString[ttt, "\\\\\n\\hline\n"];
  Do[If[h != 0, WriteString[ttt, a[[i]]], WriteString[ttt, ""]];
   Do[WriteString[ttt, " & "];
    If[j == p, WriteString[ttt, "{\\bf "]];
    WriteString[ttt, T[[j, i]]];
    If[j == p, WriteString[ttt, "}"]], {j, 1, Length[T]}];
   WriteString[ttt, " \\"];
   WriteString[ttt, "\\\n"], {i, 1, Length[a]}];
  WriteString[ttt, "\hline\n"];
  If[h != 0, WriteString[ttt, " ${\\bf t}^{", ToString[h], "}$"];
   Do[WriteString[ttt, " & "]; If[j == p, WriteString[ttt, "{\\bf "]];
     WriteString[ttt, t[[j]]];
    If[j == p, WriteString[ttt, "}"]], {j, 1, Length[T]}];
   WriteString[ttt, " \\"];
   WriteString[ttt, "\\\n"];
   WriteString[ttt, "\hline\n"]];
  m = Length[ort];
  Size = Max[Table[Length[ort[[j]]], {j, 1, m}]];
  T2 = Table["", {i, 1, Size}, {j, 1, m}];
  Do[T2[[i, j]] = ToString[ort[[j, i]]], {j, 1, m}, {i, 1,
    Length[ort[[j]]]}];
  Print[MatrixForm[T2]];
  If[Size > 0, WriteString[ttt, " $I", aux, "$ "]
    Do[
     Do[WriteString[ttt, " & ", T2[[i, j]]] , {j, 1, m} ];
     WriteString[ttt, "\\\\ \n"]
     , {i, 1, Size}];
   WriteString[ttt, "\hline\n"]
   ];
  WriteString[ttt, "\\end{tabular}\n"];
  ]

(*Procedure for the Gamma-algorithm *)
DualConeLatex2[B1_, C1_, filename_] :=
 Module[{h, n1, n2, m1, m2, n, m, type, ort, T, T1, p, pivot, i, j,
   Iplus, Iminus, I0, Candidates, Cand},
  (* Printing in Latex *)
  DeleteFile[filename];
  ttt = OpenWrite[filename];
  WriteString[ttt,
   "\\documentclass{article}\n\\newtheorem{example}{Example}\n\
\\oddsidemargin -2cm\n\evensidemargin -2cm\n\\begin{document}\n\
\\begin{center}\n"];
  (* End printing in Latex *)
  h = 1; n1 = 0; n2 = 0;
  m1 = Length[B1];
  If[m1 > 0, n1 = Length[B1[[1]]];
```

```
m2 = Length[C1];
If[m2 > 0, n2 = Length[C1[[1]]]];
n = Max[n1, n2];
m = n;
If[n == 0, Return[]];
T = IdentityMatrix[n];
type = Table[0, {n}];
ort = Table[{}, {n}];
Do[
 Print["m=", m];
 If[m > 0,
  Print["h=", h];
  (* Dot products *)
  If[h <= m1, a = B1[[h]], a = C1[[h - m1]]];
  t = Flatten[T.Transpose[{a}]];
  p = FindPivot[t, type];
  pivot = t[[p]];
  Print[MatrixForm[{type}]];
  Print[MatrixForm[a] MatrixForm[Transpose[T]]];
  Print[t];
  Print[ort];
  label = StringJoin["Iteration ", ToString[h]];
  tablenumber = 1;
  PrintTable[T, a, label, t, p, {}, filename, tablenumber, m1, h,
   type];
  If[p > 0,
   (* Gamma I *)
   T[[p]] /= -pivot;
   Do[If[i != p, T[[i]] = T[[i]] + t[[i]]*T[[p]];
     AppendTo[ort[[i]], h]]
    , {i, 1, m}];
   If[h <= m1, T = RemoveRow[T, p]; ort = RemoveRow[ort, p];
    type = RemoveRow[type, p]; m = m - 1, type[[p]] = 1];
   (* Gamma II *)
   , Do[If[t[[i]] == 0, AppendTo[ort[[i]], h]], {i, 1, m}];
   Iplus = {}; Iminus = {}; I0 = {}; T1 = {}; mm = 0; ort1 = {};
   type1 = {}; ort0 = {};
   Do[If[t[[i]] > 0 , AppendTo[Iplus, i],
     If[t[[i]] < 0, AppendTo[Iminus, i],
      If[type[[i]] != 0, AppendTo[I0, i];
       AppendTo[ort0, ort[[i]]]]]], {i, 1, m}];
   Print["Iplus=", Iplus, " Iminus=", Iminus, " I0=", I0];
   Do[If[(t[[i]] < 0 && h > m1) || t[[i]] == 0, mm += 1;
     AppendTo[T1, T[[i]]]; AppendTo[ort1, ort[[i]]];
     AppendTo[type1, type[[i]]]], {i, 1, m}];
   If[Iplus != {},
    Candidates = {};
    Do[ii = Iplus[[i]]; jj = Iminus[[j]];
     AppendTo[
```

```
        Candidates, {Join[
          Intersection[ort[[ii]], ort[[jj]]], {h}], {ii, jj}}]
       , {i, 1, Length[Iplus]}, {j, 1, Length[Iminus]}];
      Print["Candidates=", Candidates];
      Print["ort0=", ort0];
      Cand = {}; rem = {};
      Do[aux = Intersection[Candidates[[i, 1]], Candidates[[j, 1]]];
       If[aux == Candidates[[i, 1]], AppendTo[rem, i]];
       If[aux == Candidates[[j, 1]] && aux != Candidates[[i, 1]],
        AppendTo[rem, j]], {i, 1, Length[Candidates]}, {j, 1, i - 1}];
      Print["**Remove=", rem];
      Do[aux = Intersection[Candidates[[i, 1]], ort0[[j]]];
       If[aux == Candidates[[i, 1]], AppendTo[rem, i]], {i, 1,
        Length[Candidates]}, {j, 1, Length[ort0]}];
      rem = Sort[Union[rem]];
      Print["Remove=", rem];
      Do[
       Candidates = RemoveRow[Candidates, rem[[i]]], {i, Length[rem],
        1, -1}];
      Print["Candidates=", Candidates];
      Do[mm += 1;
       AppendTo[T1,
        Abs[t[[Candidates[[i, 2, 2]]]]]*T[[Candidates[[i, 2, 1]]]] +
         Abs[t[[Candidates[[i, 2, 1]]]]]*T[[Candidates[[i, 2, 2]]]]];
       AppendTo[type1, 1];
       AppendTo[ort1, Candidates[[i, 1]]], {i, 1, Length[Candidates]}];
      m = mm;
      ort = ort1;
      type = type1;
      T = T1]
     ], T = {0}; ort = {}; type = {}]
   , {h, 1, m1 + m2}];
  (* Printing in Latex *)
  If[m > 0, label = "Final";
   PrintTable[T, a, label, t, 0, {}, filename, tablenumber, m1, 0,
    type], WriteString[ttt,
    "The dual cone is the zero vector. \n\\end{center}\n\
\\end{document}\n"]];
  WriteString[ttt, "\n\\end{center}\n\\end{document}\n"];
  Close[ttt];
  (* End printing in Latex *)
  Return[{T, ort, type}]
  ]
```

Nine linearly independent paths example

```
H = {
  {1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0},
  {0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1},
  {0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0},
```

```
  {0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1},
  {1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1},
  {0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 1},
  {1, 0, 0, 0, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0},
  {1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0},
  {0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0}}
filename = "C:\gammaalgorithmtables.tex";
OUTPUT = DualConeLatex2[H, {}, filename];
```

A.5 Matlab code for the Network Loading Model including overtaking

This section is devoted to the Matlab implementation of the iterative algorithm used to solve the Network Loading Model including overtaking classes explained in Chapter 11. We will show the codes for the case of BPR and normal cumulative distribution functions. For the sake of brevity, only the main program and the more important functions will be shown (which contains the most important differences with respect to the model proposed by Castillo et al. (2012)), omitting the plot and reading data functions (for the complete list of functions, see Nogal (2011)).

Main program

```
global alpha;
global congestcoeff;
global correctcontrol;
global colors;
global DATA;
global delta;
global deviation;
global error;
global fbeta;
global gamma;
global GrapOption
global Ii;
global increment;
global labels;
global linklength;
global links;
global linksplotted;
global linkroutes;
global linksatmax;
global m;
global MaxIi;
global maxtime;
global mean;
global muroutes;
```

```
global nclasses;
global nlinks;
global nlinkroutes;
global nfigures;
global nmaxlinks;
global nnodes;
global nodesplotted;
global nPath;
global nroutelinks;
global Nroutes;
global OpenFig;
global pathsLinks;
global Points;
global pp;
global ProjectName;
global rho;
global S;
global samax;
global satur;
global saturationmax;
global selectedroute;
global sigmaroutes;
global tsize1;
global tsize;
global t;
global TA1;
global tmax;
global ttt;
global XAmax;


% INITIALIZATION OF VARIABLES
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
tiempo0=clock;
maxtimepermitted=5000;
nmaxnodelinks=6; %maximum number of links exiting from a link
nmaxlinkroutes=100;
itermax=100;
tol=0.1;
rho=1.0;    % relax coefficient
ReadingDATA;

% Calculates the output links of each node (outputnodelinks)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
nnodes=max(max(links));
Oi=zeros(nnodes,nmaxnodelinks);
MaxOi=zeros(1,nnodes);
actualnmaxnodelinks=0;
for i=1:nnodes
```

```
    kk=0;
    for j=1:nlinks
        if links(j,1)==i
            kk=kk+1;
            Oi(i,kk)=j;
            if actualnmaxnodelinks<kk
                actualnmaxnodelinks=kk;
            end
        end
    end
    MaxOi(i)=kk;
end

% Calculates the input links of each node (outputnodelinks)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Ii=zeros(nnodes,nmaxnodelinks);
MaxIi=zeros(1,nnodes);
for i=1:nnodes
    kk=0;
    for j=1:nlinks
        if links(j,2)==i
            kk=kk+1;
            Ii(i,kk)=j;
            if actualnmaxnodelinks<kk
                actualnmaxnodelinks=kk;
            end
        end
    end
    MaxIi(i)=kk;
end

% Calculate the number of routes per each link (linkroutes)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
linkroutes=zeros(nlinks,nmaxlinkroutes);
nlinkroutes=zeros(nlinks,1);
for r=1:nPath
    for j=1:nroutelinks(r)
        l=pathsLinks(r,j);
        nlinkroutes(l)=nlinkroutes(l)+1;
        linkroutes(l,nlinkroutes(l))=r;
    end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% SPLINE MONOTONE HERMITE CALCULATION
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if ~DATA.SMH.calculated

    % Estimated congestion control
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
res=zeros(nlinks,1);
for i=1:nlinks
    suma=0;
    if nlinkroutes(i)>0
        for j=1:nlinkroutes(i)
            for class=1:nclasses
                vol=sum(Nroutes(class,:,:),3)./tmax;
                suma=suma+vol(linkroutes(i,j))/XAmax(class,i);
            end
        end
    end
    res(i)=suma;
end
control=1;

if control
    % Spline initialization
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    ttt=zeros(nclasses,nlinks,tsize);
    TA1=zeros(nlinks,tsize);

    for i=1:tsize
        TA1(:,i)=t(i);  % link enter time
    end
    for class=1:nclasses
        for i=1:nlinks
            ttt(class,i,:)=t(:)+alpha(class,i);  % initial link exit time associated to TA1
        end
    end
    satur=zeros(nlinks,length(t));

    % Starts iterative program
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    ppp=pchip(t,t);
    S=zeros(nclasses,nlinks);
    for class=1:nclasses
        pp{class}=struct(ppp);
        for ll=1:nlinks
            % initial spline (exit time, enter time)
            pp{class,ll}=pchip(ttt(class,ll,:),TA1(ll,:));
        end
        for ll=1:nlinks
            S(class,ll)=alpha(class,ll)*congestcoeff*(1+fbeta(class,ll))/(XAmax(class,ll));
        end
    end
    error=100*tol;
    iter=0;
    maxtime=maxtimepermitted/2;
```

```
        crack=0;

        while error>tol && iter<itermax && maxtime<maxtimepermitted
            iter=iter+1;
            updatesplines;  %update the splines
            if maxtime<maxtimepermitted
                fprintf('iteration = %3.0f cc=%2.0f error=%12.6f saturationmax=%8.3f
                 linksatmax=%5.0f, cpu=%4.5g\n',iter,correctcontrol,error,
                 saturationmax,linksatmax,etime(clock,tiempo0));
            else
                fprintf('The proposed flow blocks the network with maxtime=%9.2f
                 exceeding maxtimepermitted=%9.2f\n',maxtime,maxtimepermitted);
                crack=1;
                break;
            end
        end

        % If the results are obtained correcty, we save them
        if crack==0
            DATA.SMH.calculated=1;
            DATA.SMH.pp=pp;
            DATA.SMH.TA1=TA1;  % link enter time
            DATA.SMH.ttt=ttt;  % initial link exit time associated to TA1
            DATA.SMH.maxtime=maxtime;
            DATA.SMH.control=control;
            tiempo1=clock;
            fprintf('\n Tiempo de cpu :  %0.5g segundos\n',etime(tiempo1,tiempo0));
        end
    end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Graphic Output
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
nfigures=0;
OpenFig=[];
control=DATA.SMH.control;

if and(DATA.SMH.calculated==1,control)
    pp=DATA.SMH.pp;
    TA1=DATA.SMH.TA1;  % link enter time
    ttt=DATA.SMH.ttt;  % initial link exit time associated to TA1
    maxtime=DATA.SMH.maxtime;
    if GrapOption(7)
        PlotNetwork(SpecialLinks); % network with the special links highlighted
    end
    if GrapOption(1)
        plotroutes(routesplotted); % flow at the origin of each path
    end
```

```
    if GrapOption(2)
        plotlinks(linksplotted); % traffic flow intensity at the exit each link
    end
    if GrapOption(3)
        plotnodes(nodesplotted); % flow intensity at each node with the time
    end
    if GrapOption(4)
        plotRouteFlow(selectedroute,GrapOption(9)); % Link flow evolution
    end
    if GrapOption(5)
        plotlinksN(linksplotted2); % Link travel time evolution
    end
    if GrapOption(6)
        plotintensity(routesplotted2); % route start time with the link exit time of each path
    end
    if GrapOption(8)
        plotTTF(linksplotted3); % route start time with the link exit time of each path
    end
end

fprintf('END OF PROGRAM\n');
```

Function updatesplines

This function permits updating the splines for each iteration.

```
function updatesplines

global nlinks;
global error;
global maxtime;
global pp;
global TA1;
global ttt;
global nclasses;
global tsize;

GenerateXA; % link exit time of each link (ttt)
error=0;
for class=1:nclasses
    for ll=1:nlinks
        coeff0=pp{class,ll}.coefs;
        pp{class,ll}=pchip(ttt(class,ll,:),TA1(ll,:));
        error=error+max(max(abs(coeff0-pp{class,ll}.coefs)));
    end
end
maxtime=max(max(max(ttt)));
```

Function GenerateXA

This function calculates the link exit time of each link.

```
function GenerateXA

global nPath;
global t;
global nlinks;
global nroutelinks;
global pathsLinks;
global TA1;
global ttt;
global tsize;
global alpha;
global gamma;
global rho;
global XAmax;
global saturationmax;
global linksatmax;
global fbeta;
global XA;
global correctcontrol;
global S;
global satur;
global delta;
global nclasses;
global mean;
global deviation;

saturationmax=0;
XA=zeros(nclasses,nlinks,tsize);
E=zeros(nclasses,nlinks,length(t));

correctcontrol=1;
Numbercorrections=0;
for kk=1:length(t)
    for ll=1:nlinks
        E0=zeros(1,nclasses);
        XA0=zeros(1,nclasses);
        sat=0;
        for p=1:nPath
            found=0;
            j=1;
            while j<=nroutelinks(p) && found==0
                a=pathsLinks(p,j);
                if a==ll
                    for class=1:nclasses
                        tkk(class)=ttt(class,ll,kk);     % link exit time
                        tkk1=TA1(ll,kk);    % link enter time associated with tkk
```

```
                % starting time of a user who reaches the end of link
                % of a route p at time tkk
                tt0=theta(class,ll,p,tkk(class));
                % starting time of a user who reaches the end of link
                % of a route p at time tkk1
                tt1=theta(class,ll,p,tkk1);
                %cumulative flow at origin node of the path p at tt0
                [NI,c0]=qroute(class,p,tt0,1);
                %cumulative flow at origin node of the path p at tt1
                [NI,c1]=qroute(class,p,tt1,1);
                % cumulative flow at the link exit by the time ttk
                E0(class)=E0(class)+c0;
                % number of vehicles inside the link
                XA0(class)=XA0(class)+c0-c1;
                found=1;
                if j<nroutelinks(p)
                    % saturation of upstream links
                    sat=max(sat,satur(pathsLinks(p,j+1),kk));
                end
            end
        end
        j=j+1;
    end
end
E(:,ll,kk)=E0(:);
XA(:,ll,kk)=XA0(:);
aux2=sum(XA0(:)./XAmax(1:nclasses,ll));     % link ll congestion ratio at time kk
satur(ll,kk)=aux2;
if saturationmax<aux2
    saturationmax=aux2;
    linksatmax=ll;
end
% link ll travel time for a user who enters in this link at time kk
tau=alpha(nclasses+1,ll)*(1+fbeta(nclasses+1,ll)*aux2^gamma)*normcdf(aux2,mean',
    deviation')+alpha(1:nclasses,ll).*(1+fbeta(1:nclasses,ll)*aux2^gamma).*
    (1-normcdf(aux2,mean',deviation'))+alpha(1:nclasses,ll).*delta(1:nclasses,ll)
    *sat^gamma/3;
if kk==1
    Q=E(:,ll,kk).*S(:,ll);
else
    % queue dissipation time at link ll at time kk
    Q=(E(:,ll,kk)-E(:,ll,kk-1)).*S(:,ll);
end
for class=1:nclasses
    tkk(class)=tkk1+tau(class); % updates the link exit time
    if tau(class)<0.00000001
       fprintf('tau=%12.9f\n',tau(class));
    end
```

```
            if kk>1 && (ttt(class,ll,kk-1)+Q(class)>tkk(class))
                Numbercorrections=Numbercorrections+1;
                if correctcontrol==1
                    fprintf('Corrects kk=%3.0f and link=%3.0f ttt(ll,kk-1)=%8.3f
                     tkk=%8.3f Q=%8.3f\n',kk,ll,ttt(class,ll,kk-1),tkk(class),Q(class));
                    correctcontrol=0;
                end
                % updates the link exit time with the queue dissipation term
                tkk(class)=ttt(class,ll,kk-1)+Q(class);
            end
            if kk>1 && (Q(class)<0)
                fprintf('Error kk=%3.0f and link=%3.0f ttt(ll,kk-1)=%8.3f
                 ttt(ll,kk)=%8.3f Q=%8.3f\n',kk,ll,ttt(class,ll,kk-1),tkk(class),Q(class));
            end
            ttt(class,ll,kk)=rho*tkk(class)+(1-rho)*ttt(class,ll,kk);
            if kk>1
                if ttt(class,ll,kk)-ttt(class,ll,kk-1)==0
                    fprintf('Q(class)=%15.8f tau(class)=%15.8f\n',Q(class),tau(class));
                end
            end
        end
    end
end
if Numbercorrections>0
    fprintf('Numbercorrections=%3.0f \n',Numbercorrections);
end
```

Function theta

This function returns the starting time of a user who reaches the end of link a of route p at time tt.

```
function tt1=theta(class,a,p,tt)

global nroutelinks;
global pathsLinks;
global pp;

found=0;
tt1=tt;
for j=nroutelinks(p):-1:1
    ll=pathsLinks(p,j);
    if a==ll
        found=1;
    end
    if found==1
        tt1=ppval(pp{class,ll},tt1);
    end
end
```

Function qroute

This function calculates the cumulative flow and flow intensity at origin node.

```
function [z,cumflow]=qroute(class,route,tt,option)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% option = 1 : calculates cumulative flow at origin node
% option = 2 : calculates flow intensity at origin node
% option > 2 : calculates both cumulative flow and flow intensity at origin node
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

global Nroutes;
global muroutes;
global sigmaroutes;
global m;

z=0;
cumflow=0;
for i=1:m
    if option>1
    % normal distribution of the m waves
    z=z+Nroutes(class,route,i)*normpdf(tt,muroutes(class,route,i),sigmaroutes(class,route,i));
    end
    if option==1 || option>2
        cumflow=cumflow+Nroutes(class,route,i)*(normcdf(tt,muroutes(class,route,i),
        sigmaroutes(class,route,i)));
    end
end
```

A.6 Matlab code for the trajectory plots

This section is devoted to the Matlab codes used to generate the trajectory plots shown in Chapter 12. As already noted, the data used for the plots comes from the Dynamic Network Loading Model with Overtaking classes explained in Chapter 11 and which Matlab codes are shown in Section A.5 in this Appendix. Thus, the plots are obtained from a function which is executed from the Main program. Other functions that are needed to obtain the plots are also included.

Function plottrajectories

This function generates the plots shown in the corresponding chapter depending on the options chosen.

```
function plottrajectories(p)

global nroutelinks;
global pathsLinks;
```

```
global pp;
global t;
global linklength;
global ProjectName;
global nfigures;
global OpenFig;
global nclasses;
global class;
global lllink;
global tttt1;
global increment;
global labels;
global alpha;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% OPTIONS
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% classindices=1:nclasses;
classindices=[1 2 3];
% --- GENERAL
optiondeparture=0;  % 0. equal delayed  1. equal flow
bbmax=1;            % 1/2: without/with wide lines of some trajectories
Optionreverse=0;    % second figure;
% --- FREE FLOW TRAJECTORIES
together=0;         % 0/1: trajectories combined without/with free flow trajectories
referenceclass=3;   % class whose free flow trajectories are plotted
optionsplinefree=0; % 0/1: free flow trajectories plotted without/with splines
onlyfree=0;         % 0/1: Not/yes plot only free flow trajectories
% --- LATERAL PLOTS
plotmaxmin=0;       % lateral plots
optionoriginplot=0; % left plot: 0. travel time evolution 1. flow evolution
factor=10;          % factor for travel time lateral plot
factorflow=0.06;    % factor for flow lateral plot
% --- TRANSPARENCY
optiontrans=0;      % 0/1: opaque/transparent lines
transp=0.5;         % transparent level
transflinewidth=60; % transparent trajectories width
% --- BANDS
optionplotcruce=1;  % 0/1: Not/yes plot the overtaking bands
ClassOvertak=3;     % class to compare with other ones when there are three classes
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

if length(classindices)>1
    plotmaxmin=0;
end
txt=cell(1,length(classindices));
for iclass=1:length(classindices)
    class=classindices(iclass);
    txt{1,iclass}=labels{class};
```

```
end

colors=GenColor(nclasses, 1711);
ts=t(1:1:end);
ntotal=length(ts);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Direct trajectories
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
nfigures=nfigures+1;
h=figure('Name',[ProjectName '_trajectories']);
OpenFig=[OpenFig h];
es=0.001;
imax=100;
leg=zeros(1,length(classindices));
maxy=ts(end);
disc=200;
for bb=1:bbmax
    if bb==2
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        % Select particular trajectories to be plotted and their colors
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        ts=t([1,33,54,77,96]);
        ntotal=length(ts);
        linewidth=4;
        arrivaltimes=zeros(ntotal);
        colors(1,:)=[11/255 104/255 11/255];
        colors(3,:)=[36/255 17/255 211/255];
    else
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        % Select all trajectories to be plotted
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        ts=t(1:1:end);
        ntotal=length(ts);
        linewidth=1.5;
    end
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    % Calculates travel times to end of all links
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    x=zeros(1,nroutelinks(p)+1);
    for j=1:nroutelinks(p)
        ll=pathsLinks(p,j);
        if j==1
            x(j+1)=linklength(j);
        else
            x(j+1)=x(j)+linklength(j);
        end
    end
    xmin=min(x);
```

```
xmax=max(x);
%---Intersections
BVx=x(end):-(x(end)-x(1))/disc:x(1);  % bands: vector x
BMy1=zeros(ntotal,disc+1);    % bands: matrix y1
BMy2=zeros(ntotal,disc+1);    % bands: matrix y2
BMy3=zeros(ntotal,disc+1);    % bands: matrix y3
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Initializes xxa, xxaflow, yya0 and yya1 variables
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
xxa=zeros(length(classindices),ntotal);
xxaflow=zeros(length(classindices),ntotal);
yya0=zeros(length(classindices),ntotal);
yyaflow0=zeros(length(classindices),ntotal);
yya1=zeros(length(classindices),ntotal);

for iclass=1:length(classindices)
    class=classindices(iclass);
    if bb==1
        if optiondeparture==1
            %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
            % Calculates departure times associated with a given subset
            % of
            %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
            [~,qqqmax]=qroute(class,p,ts(end),1);
            [~,qqqmin]=qroute(class,p,ts(1),1);
            qqq=qqqmin:(qqqmax-qqqmin)/ntotal:qqqmax;
            ntotal=length(qqq);
            tss=zeros(1,ntotal);
            for ii=1:ntotal
                [tss(ii),~,~]=qrouteinverse(class,p,qqq(ii),ts(1),ts(end),es,imax);
            end
        else
            tss=ts;
        end
    else
        tss=ts;
    end

    for k=1:ntotal
        if together==1 && bb==1
            %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
            % Plots free traveltime trajectories
            %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
            yyy(1)=tss(k);
            for j=1:nroutelinks(p)
                ll=pathsLinks(p,j);
                yyy(j+1)=yyy(j)+alpha(referenceclass,ll);
            end
            if yyy(end)<=t(end)-0.001
```

```
        if optionsplinefree==1
            xxxi=x(end):-(x(end)-x(1))/disc:x(1);
            yyyi=pchip(x,yyy,xxxi);
            %---Intersections
            BMy2(k,:)=yyyi;    % bands: matrix y2
            %---
            fillLine(xxxi,yyyi,linewidth,transflinewidth,colors(class,:),
            transp,optiontrans);
        else
            %---Intersections
            [~, VectY0]=Interpolate(x,yyy,disc+1);
            BMy2(k,:)=fliplr(VectY0);    % bands: matrix y2
            %---
            fillLine(x,yyy,linewidth,transflinewidth,colors(class,:),
            transp,optiontrans);
        end
    end
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
end
y=zeros(1,nroutelinks(p)+1);
y(1)=tss(k);
tttt1=y(1);
lll=1;
for j=1:nroutelinks(p)
    ll=pathsLinks(p,j);
    lllink=ll;
    [tttt1 ea iter]=Bisectiontau(t(1),t(end),es,imax);
    lll=lll+1;
    y(lll)=tttt1;
end
xx=x;
minaax=min(x);
maxaax=max(x);
yy=y;
xxi=x(end):-(x(end)-x(1))/disc:x(1);
if onlyfree
    yyi(1)=tss(k);
    for j=1:nroutelinks(p)
        ll=pathsLinks(p,j);
        yyi(j+1)=yyi(j)+alpha(iclass,ll);
    end
    %---Intersections
    [~, VectY0]=Interpolate(x,yyi,disc+1);
    if iclass==1
        BMy1(k,:)=fliplr(VectY0);    % bands: matrix y1
    elseif iclass==2
        BMy2(k,:)=fliplr(VectY0);    % bands: matrix y2
    else
        BMy3(k,:)=fliplr(VectY0);    % bands: matrix y3
```

```
            end
            %---
            leg(iclass)= fillLine(x,yyi,linewidth,transflinewidth,
                colors(class,:),transp,optiontrans);
            plotmaxmin=0;
        else
            yyi=pchip(x,yy,xxi);
            if yyi(1)<=t(end)-0.001
                xxa(iclass,k)=yyi(1)-yyi(end);
                [xxaflow(iclass,k),~]=qroute(class,p,yyi(end),2);
                yya1(iclass,k)=yyi(1);
                yya0(iclass,k)=yyi(end);
                %---Intersections
                if iclass==1
                    BMy1(k,:)=yyi;    % bands: matrix y1
                elseif iclass==2
                    BMy2(k,:)=yyi;    % bands: matrix y2
                else
                    BMy3(k,:)=yyi;    % bands: matrix y3
                end
                if class==ClassOvertak
                    iClassOvertak=iclass;
                end
                %---
                leg(iclass)= fillLine(xxi,yyi,linewidth,transflinewidth,
                    colors(class,:),transp,optiontrans);
                maxy=max(maxy,max(yyi));
            end
        end
        hold on;
        if bb==2 && iclass==1
            arrivaltimes(k)=y(end);
        end
    end
    yyaflow0=yya0;
    if plotmaxmin==1
        xxa=xxa(xxa>0);
        yya0=yya0(xxa>0);
        yya1=yya1(xxa>0);
        minxxa=min(xxa(iclass,:));
        xxa=xxa-minxxa;
        minaax=min(min(factor*(-xxa)-0.02*(xmax-xmin)));
        maxaax=max(max(factor*(xxa)+1.02*(xmax-xmin)));
        xxa0=factor*(-xxa)-0.02*(xmax-xmin);
        xxaflow0=factorflow*(-xxaflow)-0.02*(xmax-xmin);
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        % Determines trajectories associated with min and max travel times
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        [minxxa0 k0]=max(xxa0);
```

```
[maxxxa0 k1]=min(xxa0);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Plots trajectories associated with k0 and k1
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for kk=[k0 k1]
    y(1)=tss(kk);
    tttt1=tss(kk);
    lll=1;
    for j=1:nroutelinks(p)
        ll=pathsLinks(p,j);
        lllink=ll;
        [tttt1 ea iter]=Bisectiontau(t(1),t(end),es,imax);
        lll=lll+1;
        y(lll)=tttt1;
    end
    xx=x;
    yy=y;
    xxi=x(end):-(x(end)-x(1))/100:x(1);
    yyi=pchip(x,yy,xxi);
    plot(xxi,yyi,'color','r','LineWidth',3*linewidth);

end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Plots traffic flows at the path origin
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if optionoriginplot==0
    [yya0,xxa0]=preparefill(yya0,xxa0);
    fill(xxa0,yya0,colors(class,:));
    plot(xxa0,yya0,'LineWidth',linewidth);
    minyya0=yya0(k0);
    maxyya0=yya0(k1);
    minxxa00=min(xxa0(1),xxa0(end));
    plot([minxxa0 minxxa00],[minyya0 minyya0],'r','LineWidth',2*linewidth);
    plot([maxxxa0 minxxa00],[maxyya0 maxyya0],'r','LineWidth',2*linewidth);
    plot(minxxa0,minyya0,'ko','MarkerSize',10,'MarkerFaceColor','r',
        'LineWidth',1.75)
    plot(maxxxa0,maxyya0,'ko','MarkerSize',10,'MarkerFaceColor','r',
        'LineWidth',1.75)
else
    yyaflow0=[0 yyaflow0(yyaflow0>0)];
    xxaflow0=[xxaflow0(1) xxaflow0(yyaflow0>0)];
    [yyaflow0,xxaflow0]=preparefill(yyaflow0,xxaflow0);
    fill(xxaflow0,yyaflow0,colors(class,:));
    plot(xxaflow0,yyaflow0,'LineWidth',linewidth);
end
xxa0=factor*(xxa)+1.02*(xmax-xmin);
[yya1,xxa0]=preparefill(yya1,xxa0);
fill(xxa0,yya1,colors(class,:));
plot(xxa0,yya1,'LineWidth',linewidth);
```

```
            [minxxa0 k0]=min(xxa0);
            [maxxxa0 k1]=max(xxa0);
            minyya0=yya1(k0);
            maxyya0=yya1(k1);
            minxxa00=min(xxa0(1),xxa0(end));
            plot([minxxa0 minxxa00],[minyya0 minyya0],'r','LineWidth',2*linewidth);
            plot([maxxxa0 minxxa00],[maxyya0 maxyya0],'r','LineWidth',2*linewidth);
            plot(minxxa0,minyya0,'ko','MarkerSize',10,'MarkerFaceColor','r','LineWidth',1.75)
            plot(maxxxa0,maxyya0,'ko','MarkerSize',10,'MarkerFaceColor','r','LineWidth',1.75)
        end
    end
    if bb==2
        for ii=1:length(ts)
            text(0.98*min(x)+0.02*max(x),ts(ii)+increment,['T' num2str(ii)],
            'HorizontalAlignment','center','FontWeight','bold','BackgroundColor',
            [.8 .8 .8],'FontSize',16);
        end
    end
    set(legend(leg,txt),'Location','SouthEast','FontSize',16);
end
for j=2:nroutelinks(p)
    plot([x(j) x(j)], [0 maxy],'-.r','LineWidth',1.5);
    text((x(j)+x(j-1))*0.5,0.97*maxy,['Link ' num2str(pathsLinks(p,j-1))],'HorizontalAlignment',
    'center','FontWeight','bold','BackgroundColor',[.8 .8 .8],'FontSize',16);
    text(0.5*max(x),0.04*maxy,['Path ' num2str(p)],'HorizontalAlignment','center',
    'FontWeight','bold','BackgroundColor',[.8 .8 .8],'FontSize',16);
end
j=nroutelinks(p)+1;
text((x(j)+x(j-1))*0.5,0.97*maxy,['Link ' num2str(pathsLinks(p,j-1))],'HorizontalAlignment',
'center','FontWeight','bold','BackgroundColor',[.8 .8 .8],'FontSize',16);
axis([1.01*min(min(xx),minaax)-0.01*max(max(xx),maxaax) -0.01*min(0,minaax)+
1.01*max(max(xx),maxaax) 0 max(maxy)]);
xlabel('x','fontsize',16);
ylabel('t','Rotation',0,'fontsize',16);

%---Intersections
if optionplotcruce
    colors1=GenColor(nclasses, 1605);
    if length(classindices)>2
        switch iClassOvertak
            case 1
                plotcruce(BVx,BMy1,BMy2,colors1(2,:))
                plotcruce(BVx,BMy1,BMy3,colors1(3,:))
            case 2
                plotcruce(BVx,BMy2,BMy1,colors1(1,:))
                plotcruce(BVx,BMy2,BMy3,colors1(3,:))
            case 3
                plotcruce(BVx,BMy3,BMy1,colors1(1,:))
```

```
                plotcruce(BVx,BMy3,BMy2,colors1(2,:))
        end
    else
        plotcruce(BVx,BMy1,BMy2,colors1(1,:))
    end
end
%---
print(h,'-dpdf','-r600',sprintf('%s.pdf',[ProjectName '_trajectories']))

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Reverse trajectories
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if Optionreverse
    nfigures=nfigures+1;
    h=figure('Name',[ProjectName '_trajectories']);
    OpenFig=[OpenFig h];
    colors=GenColor(nclasses, 1711);
    for bb=1:bbmax
        if bb==2
            ts=arrivaltimes;
            ntotal=length(ts);
            linewidth=4;
            colors(1,:)=[11/255 104/255 11/255];
            colors(3,:)=[36/255 17/255 211/255];
        else
            ts=t(1:1:end);
            ntotal=length(ts);
            linewidth=1.5;
        end
        for iclass=1:length(classindices)
            class=classindices(iclass);
            if bb==1
                if optiondeparture==1
                    [~,qqqmax]=qroute(class,p,ts(end),1);
                    [~,qqqmin]=qroute(class,p,ts(1),1);
                    qqq=qqqmin:qqqmax/ntotal:qqqmax;
                    ntotal=length(qqq);
                    tss=zeros(1,ntotal);
                    for ii=1:ntotal
                        [tss(ii),~,~]=qrouteinverse(class,p,qqq(ii),ts(1),ts(end),es,imax);
                    end
                else
                    tss=ts;
                end
            else
                tss=ts;
            end
            for k=1:ntotal
                x=zeros(1,nroutelinks(p)+1);
```

```
            y=zeros(1,nroutelinks(p)+1);
            y(1)=tss(k);
            for j=1:nroutelinks(p)
                ll=pathsLinks(p,j);
                if j==1
                    x(j+1)=linklength(j);
                else
                    x(j+1)=x(j)+linklength(j);
                end
            end

            tt1=y(1);
            lll=1;
            for j=nroutelinks(p):-1:1
                ll=pathsLinks(p,j);
                tt1=ppval(pp{class,ll},tt1);
                lll=lll+1;
                y(lll)=tt1;
            end
            xx=x(end:-1:1);
            yy=y(end:-1:1);
            xxi=x(end):-(x(end)-x(1))/disc:x(1);
            yyi=pchip(x,yy,xxi);
            leg(iclass)= fillLine(xxi,yyi,linewidth,transflinewidth,
                colors(class,:),transp,optiontrans);
            maxy=max(maxy,max(yyi));
            hold on;
        end
    end
    if bb==2
        for ii=1:length(ts)
            text(0.02*min(x)+0.98*max(x),ts(ii)-5*increment,['T' num2str(ii)],
            'HorizontalAlignment','center','FontWeight','bold','BackgroundColor',
            [.8 .8 .8],'FontSize',16);
        end
    end
    set(legend(leg,txt),'Location','NorthWest','FontSize',16);
end
for j=2:nroutelinks(p)
    plot([x(j) x(j)], [0 maxy],'-.r','LineWidth',1.5);
    text((x(j)+x(j-1))*0.5,0.03*maxy,['Link ' num2str(pathsLinks(p,j-1))],
    'HorizontalAlignment','center','FontWeight','bold','BackgroundColor',[.8 .8 .8],'FontSize',16);
    text(0.5*max(x),0.96*maxy,['Path ' num2str(p)],'HorizontalAlignment','center',
    'FontWeight','bold','BackgroundColor',[.8 .8 .8],'FontSize',16);
end
j=nroutelinks(p)+1;
text((x(j)+x(j-1))*0.5,0.03*maxy,['Link ' num2str(pathsLinks(p,j-1))],
'HorizontalAlignment','center','FontWeight','bold','BackgroundColor',
[.8 .8 .8],'FontSize',16);
```

```
    axis([0 max(xx) 0 maxy]);
    xlabel('x','fontsize',16);
    ylabel('t','Rotation',0,'fontsize',16);

    print(h,'-dpdf','-r600',sprintf('%s.pdf',[ProjectName '_trajectories']))

    yyy(1)=tss(k);
    for j=1:nroutelinks(p)
        ll=pathsLinks(p,j);
        yyy(j+1)=yyy(j)+alpha(class,ll);
    end
    plot(x,yyy,'color',colors(class,:),'LineWidth',linewidth);
    hold on;

end;
```

Function fillLine

This function permits plotting filled lines.

```
function leg=fillLine(x,y,linewidth,transflinewidth,color,transp,option)

if option==1
n=length(x);
deltax=max(x)-min(x);
deltay=max(y)-min(y);
a=deltay^2;
b=deltax^2;
u=(x(1:n-1)+x(2:end))/2;
v=(y(1:n-1)+y(2:end))/2;
u2=x(2:end)-x(1:n-1);
v2=(y(2:end)-y(1:n-1));
u1=-b/a*v2./u2;
v1=ones(1,n-1);
m=sqrt(a*u1.^2+b*v1.^2);
aux=transflinewidth./(2*m);
x1=u+u1.*aux;
y1=v+v1.*aux;
x2=u-u1.*aux;
y2=v-v1.*aux;

X=[x1 fliplr(x2)];
Y=[y1 fliplr(y2)];

leg=fill(X,Y,color,'EdgeColor','none');
hold on;
alpha(transp)
axis square;
else
    leg=plot(x,y,'Color',color,'LineWidth',linewidth);
```

```
end
```

Function interpolate

This function transforms vector $VectX0$ in 'dim' equally spaced points and computes the associated value in the curve $(VectX0, VectY0)$.

```
function [VectX, VectY]= Interpolate (VectX0, VectY0, dim)

VectX=linspace(VectX0(1),VectX0(end),dim);
VectY=zeros(1,dim);
for i=1:dim-1
   xc=VectX(i);
   [~,pos2]=find(VectX0>xc,1,'first');
   pos1=pos2-1;
   yc=VectY0(pos1)+(xc-VectX0(pos1))*(VectY0(pos1)-VectY0(pos2))/(VectX0(pos1)-VectX0(pos2));
   VectY(i)=yc;
end
VectY(end)=VectY0(end);
```

Function Bisectiontau

This function obtains the bisection between points xl and xu.

```
function [Bisect ea iter]=Bisectiontau(xl,xu,es,imax)

global class;
global lllink;
global tttt1;
global pp;

iter=0;

fl=tttt1-ppval(pp{class,lllink},xl);
ea=2*es;
xr=xl;
while ea>es && iter<=imax
    xrold=xr;
    xr=(xl+xu)/2;
    fr=tttt1-ppval(pp{class,lllink},xr);
    iter=iter+1;
    if xr ~= 0
        ea=abs((xr-xrold)/xr)*100;
    end
    test=fl*fr;
    if test<0
        xu=xr;
    else if test>0
            xl=xr;
            fl=fr;
```

```
        else
            ea=0;
        end
    end
end
Bisect=xr;
```

Function qrouteinverse

This function computes the inverse of the qroute function.

```
function [Bisect ea iter]=qrouteinverse(class,route,tt,xl,xu,es,imax)

iter=0;
[~,fl]=qroute(class,route,xl,1);
fl=fl-tt;
if abs(fl)>es
    [~,fr]=qroute(class,route,xu,1);
    if abs(fr)>es
        ea=2*es;
        xr=xl;
        while ea>es && iter<=imax
            xrold=xr;
            xr=(xl+xu)/2;
            [~,fr]=qroute(class,route,xr,1);
            fr=fr-tt;
            iter=iter+1;
            if xr ~= 0
                ea=abs((xr-xrold)/xr)*100;
            end
            test=fl*fr;
            if test<0
                xu=xr;
            else if test>0
                    xl=xr;
                    fl=fr;
                else
                    ea=0;
                end
            end
        end
        Bisect=xr;
    else
      Bisect=xu;
    ea=es;
    iter=1;
    end
else
    Bisect=xl;
    ea=es;
```

```
    iter=1;
end
```

Function plotcruce

This function plots the intersections, i.e., the moments when overtaking takes place.

```
function plotcruce(vecX,matY1,matY2,color)

%--- Sort the vectors and matrices
vecX=fliplr(vecX);
matY1=fliplr(matY1);
matY2=fliplr(matY2);
%--- Check the speeds and correct them
if matY1(1,2)>matY2(1,2)
    aux=matY1;
    matY1=matY2;
    matY2=aux;
end
%--- Compute the points where they cross and save them
[dim,~]=size(matY1);
Rxcruce=zeros(dim,dim);
Rycruce=zeros(dim,dim);
xant=0;
yant=0;
for i=1:dim
    for j=i+1:dim
        if max(matY1(j,:))>0 && max(matY2(i,:))>0
            %--- Eliminate the final zeros
            len=min(length(matY1(matY1(j,:)>0)),length(matY2(matY2(i,:)>0)));
            X=vecX(1:len);
            Y1=matY1(j,1:len);
            Y2=matY2(i,1:len);
            %---
            if i>1 && abs(Rxcruce(i-1,j-1))<999999999
                xant=Rxcruce(i-1,j-1);
                yant=Rycruce(i-1,j-1);
            end
            [Rxcruce(i,j),Rycruce(i,j)]=cruce(X,Y1,Y2,xant,yant);
        end
    end
end

for i=1:ceil(sqrt(dim))
    vx=diag(Rxcruce(1:dim-i,1+i:dim));
    vy=diag(Rycruce(1:dim-i,1+i:dim));
    aux=vx;
    auy=vy;
    q=1;
    while q
```

```
        p=find(aux==0,1,'first');
        plot(aux(1:p-1),auy(1:p-1),'Color',color,'linewidth',4)
        aux=aux(p:end);
        auy=auy(p:end);
        q=find(aux>0,1,'first');
        aux=aux(q:end);
        auy=auy(q:end);
    end
end
```

Bibliography

Aashtiani, M. (1979). *The multi-modal traffic assignment problem.*. Ph.D. thesis, Operation Research Center, MIT, Cambridge, MA.

Abdel-Aty, M., Kitamura, R., and Jovanis, P. (1995). Investigating effect of travel time variability on route choice using repeated-measurement stated preference data. *Transportation Research Record*, 1493:39–45.

Abur, A. and Gómez Expósito, A. (2004). *Power System State Estimation-Theory and Implementations*. Marcel Dekker, Inc., New York.

Aczél, J. (1966). *Lectures on functional equations and their applications*, vol. 19. Mathematics in Science and Engineering. Academic Press.

Ahuja, R., Magnanti, T., and Orlin, J. (1993). *Network Flows: Theory, Algorithms and Applications*. Prentice-Hall.

Akamatsu, P. (1997). Decomposition of path choice entropy in general transport networks. *Transportation Science*, 31:349–362.

Anagnostopoulos, C., Anagnostopoulos, I., Loumos, V., and Kayafas, E. (2006). A licence plate-recognition algorithm for intelligent transportation system applications. *IEEE Transactions on Intelligent Transportation Systems*, 7:3:377–392.

Arnold, B., Castillo, E., and Sarabia, J. (1993). Conjugate exponential family priors for exponential family likelihoods. *Statistics*, 25:71–77.

Arnold, B., Castillo, E., and Sarabia, J. (1996). Priors with convenient posteriors. *Statistics*, 28:347–354.

Asakura, Y. and Kashiwadani, M. (1991). Road network reliability caused by daily fluctuation of traffic flow. In *Proceedings of the* 19*th PTRC Summer Annual Meeting*, pp. 73–84. Brighton.

Ashok, K. and Ben Akiva, M. (2000). Alternative approaches for real-time estimation and prediction of time-dependent origin-destination flows. *Transportation Science*, 34:1:21–36.

Ban, X., Liu., H. X., Ferris, M. C., and Ran, B. (2008). A link-node complementary model and solution algorithm for dynamic user equilibria with exact flow propagations. *Transportation Research Part B*, 42:823–842.

Bar-Gera, H. (2002). Origin-based algorithm for the traffic assignment problem. *Transportation Science*, 36:398–417.

Bard, J. (1988). Convex two-level optimization. *Mathematical Programming*, 40:15–27.

Beckmann, M. J., McGuire, C. B., and Wisten, C. M. (1956). *Studies in the Economics of Transportation*. Yale University Press, New Haven.

Bell, M. G. H. (1983). The estimation of an origin-destination matrices from traffic counts. *Transportation Science*, 10:198–217.

Bell, M. G. H. (1991). The estimation of origin-destination matrices by constrained generalized least squares. *Transportation Research, Part B*, 25:1:13–23.

Bernstein, D., Friesz, T., Tobin, R. L., and Wie, B. (1993). A variational control formulation of the simultaneous route and departure-time choice equilibrium problem. In C. Daganzo, ed., *Proceedings of the 12th International Symposium on Transportation and Traffic Theory*, pp. 107–126. Elsevier, New York.

Bernstein, D. and Wynter, L. (2000). Issues of uniqueness and convexity in non-additive bi-criteria equilibrium models. In *8th Meeting of the EURO Working Group on Transportation*. Rome Jubilee 2000 Conference, La Sapienza, Rome, Italy.

Bianco, L., Confessore, G., and Gentili, M. (2006). Combinatorial aspects of the sensor location problem. *Annals of Operations Research*, 144:201–234.

Bianco, L., Confessore, L., and Reverberi, P. (2001). A network based model for traffic sensor location with implications on O-D matrix estimates. *Transportation Science*, 35:50–60.

Boyce, D. and Janson, B. (1981). The effect on equilibrium trim assignment of different link congestion functions. *Transportation Research, Part B*, 15A:223–232.

Bracken, J., Falk, J. E., and McGill, J. M. (1973). Mathematical programs with optimization problems in the constraints. *Operations Research*, 21:37–44.

Bracken, J., Falk, J. E., and McGill, J. M. (1974). Equivalence of two mathematical programs with optimization problem in the constraints. *Operations Research*, 22:1102–1104.

Branston, D. (1976). Link capacity functins: A review. *Transportation Research, Part B*, 10:223–236.

Brenninger-Gothe, M. and Jornsten, K. (1989). Estimation of origin-destination matrices from traffic counts using multiobjective programming formulations. *Transportation Research, Part B*, 23(4):257–269.

Brillouin, L. (1956). *Science and Information Theory*. Academic Press, New York.

Bureau of Public Roads (1964). *Traffic assignment manual*. U.S. Department of Commerce, Urban Planning Division, Washington, D.C.

Buric, L. and Janovsky, V. (2007). The overtaking in a class of traffic models: A filippov system formulation. In *AIP Conference Proceedings, Volume 936*, pp. 97–100. DOI: 10.1063/1.2790276, Capry, Italy.

Caceres, N., Romero, L. M., Benítez, F. G., and del Castillo, J. M. (2012). Traffic flow estimation models using cellular phone data. *IEEE Transactions on Intelligent Transportation Systems*, 13(3):1430–1441.

Caceres, N., Wideberg, J., and Benitez, F. (2007). Deriving origin-destination data from a mobile phone network. *Journal of Intelligent Transportation Systems*, 1(1):15–26.

Carey, M. (1992). Nonconvexity of the dynamic traffic assignment problem. *Transportation Research Part B*, 26:127–133.

Carey, M. (2004). Link travel times i: Desirable properties. *Network and Spatial Economics*, 4:257–268.

Carey, M. and Revelli, R. (1986). Constrained estimation of direct demand functions and trip matrices. *Transportation Science*, 20:3:143–152.

Carey, M. and Srinivasan, A. (1993). Externalities, average and marginal costs, and tolls on congested networks with time-varying flows. *Operations Research*, 41:217–231.

Carey, M. and Subrahmanian, E. (2000). An approach to modeling time-varying flows on congested networks. *Transportation Research Part B*, 34:157–183.

Cascetta, E. (1984). Estimation of trip matrices from traffic counts and survey data: a generalized least squares estimator. *Transportation Research, Part B*, 18:289–299.

Cascetta, E. and Nguyen, S. (1988). A unified framework for estimating or updating origin/destination matrices from traffic counts. *Transportation Research, Part B*, 22:437–455.

Castillo, E., Calviño, A., Sánchez-Cambronero, S., and Lo, H. (submitted). A user equilibrium approach considering overtaking. *Transportation Research Part B*.

Castillo, E., Cobo, A., Jubete, F., and Pruneda, R. (1999). *Orthogonal Sets and Polar Methods in Linear Algebra: Applications to Matrix Calculations, Systems of Equations and Inequalities, and Linear Programming*. John Wiley and Sons, New York.

Castillo, E., Cobo, A., Jubete, F., Pruneda, R. E., and Castillo, C. (2000). An orthogonally based pivoting transformation of matrices and some applications. *SIAM Journal on Matrix Analysis and Applications*, 22:666–681.

Castillo, E., Conejo, A., Menéndez, J. M., and Jiménez, P. (2008a). The observability problem in traffic network models. *Computer Aided Civil and Infrastructure Engineering*, 23:208–222.

Castillo, E., Conejo, A., Pruneda, E., and Solares, C. (2007). Observability in linear systems of equations and inequalities. Applications. *Computers and Operation Research*, 34:1708–1720.

Castillo, E., Conejo, A. J., Menéndez, J. M., and P, J. (2008b). The observability problem in traffic network models. *Computer Aided Civil and Infrastructure Engineering*, 23:208–222.

Castillo, E., Gallego, I., Menéndez, J. M., and Jiménez, M. P. (2011). Link flow estimation in traffic networks on the basis of link flow observations. *Journal of Intelligent Transportation Systems*, 15(4):205–222.

Castillo, E., Gallego, I., Sánchez-Cambronero, S., and Rivas, A. (2010). Matrix tools for general observability analysis in traffic networks. *IEEE Transactions on Intelligent Transportation Systems*, 11(4):799–813.

Castillo, E., Iglesias, A., and Ruiz-Cobo, R. (2005). *Functional Equations in Applied Sciences*. Elsevier.

Castillo, E., Jiménez, P., Menéndez, J. M., and Conejo, A. (2008c). The observability problem in traffic models: algebraic and topological methods. *IEEE Transactions on Intelligent Transportation System*, 9(2):275–287.

Castillo, E. and Jubete, F. (2004). The γ-algorithm and some applications. *International Journal of Mathematical Education in Science and Technology*, 35:369–389.

Castillo, E., Jubete, F., Pruneda, R. E., and Solares, C. (2002). Obtaining simultaneous solutions of linear subsystems of equations and inequalities. *Linear Algebra and its Applications*, 346:131–154.

Castillo, E., Menéndez, J. M., and Jiménez, P. (2008d). Trip matrix and path flow reconstruction and estimation based on plate scanning and link observations. *Transportation Research, Part B*, 42:5:455–481.

Castillo, E., Menéndez, J. M., Nogal, M., Jiménez, M. P., and Sánchez-Cambronero, S. (2012). A fifo rule consistent model for the continuous dynamic network loading problem. *IEEE Transactions on Intelligent Transportation Systems*, 13(1):264–283.

Castillo, E., Menéndez, J. M., and P, J. (2008e). Trip matrix and path flow reconstruction and estimation based on plate scanning and link observations. *Transportation Research Part B*, 42(5):455–481.

Castillo, E., Menéndez, J. M., and Sánchez-Cambronero, S. (2008f). Traffic estimation and optimal counting location without path enumeration using bayesian networks. *Computer-Aided Civil and Infraestructure Engineering*, 23:189–207.

Castillo, E., Menéndez, J. M., and Sánchez-Cambronero, S. (2008g). Traffic estimation and optimal counting location without path enumeration using Bayesian networks. *Computer Aided Civil and Infraestructure Engineering*, 23(3):189–207.

Castillo, E. and Ruiz, R. (1992). *Functional Equations in Science and Engineering*. Marcel Dekker.

Chandler, R. E., Herman, R., and Montroll, E. W. (1958). Traffic dynamics: studies in car following. *Operations Research*, 16(2):165–184.

Clements, K. A. and Wollenberg, B. F. (1975). An algorithm for observability determination in power system state estimation. *IEEE PES Summer Meeting*, A75:447–449.

Conejo, A., Castillo, E., Mínguez, and García-Bertrand, R. (2005). *Decomposition Techniques in Mathematical Programming. Engineering and Science Applications*. Springer, New York.

Dafermos, S. (1980). Traffic equilibrium and variational inequalities. *Transportation Science*, 14:42–54.

Daganzo, C. F. (1994). The cell transmission model: A dynamic representation of highway traffic consistent with the hydrodynamic theory. *Transportation Research Part B*, 28(4):269–287.

Daganzo, C. F. (1995). The cell transmission model, part II: Network traffic. *Transportation Research Part B*, 29(2):79–93.

Davidson, K. (1966). A flow travel time relationship for use in transportation planning. *Austrial Road Research Board*, 3(1):186–194.

Doblas, J. and Benítez, F. G. (2005). An approach to estimating and updating origin-destination matrices based upon traffic counts preserving the prior structure of a survey matrix. *Transportation Research, Part B*, 39:565–591.

Facchinei, F. and Soares, J. (1995). *Variational Inequalities and Network Equilibrium Problems*, chap. Testing a new class of algorithms for nonlinear complementary problem. Plenum Press, New York.

Ferris, M., Meeraus, A., and Rutherford, T. F. (1999). Computing wardropian equilibria in a complementary framework. *Optimization Methoeds and Software*, 10:669–685.

Fisk, C. (1988). On combining maximum entropy trip matrix estimation with user optimal assignment. *Transportation Research, Part B*, 22:69–79.

Fisk, C. (1989). Trip matrix estimation from link traffic counts: the congested network case. *Transportation Research, Part B*, 23:331–336.

Friesz, T., Bernstein, D., Smith, T., Tobin, R. L., and Wie, B. (1993). A variational inequality formulation of the dynamic network user equilibrium problem. *Operations Research*, 41:179–191.

Friesz, T. L., Bernstein, D., Suo, Z., and Tobin, R. L. (2001). Dynamic network user equilibrium with state-dependent time lags. *Networks and Spatial Economics*, 1:319–347.

Friesz, T. L., Kim, T., Kwon, C., and Rigdon, M. A. (2011). Approximate network loading and dual-time-scale dynamic user equilibrium. *Transportation Research Part B*, 45:176–207.

Fritsch, F. N. and Carlson, R. E. (1980). Monotone piecewise cubic interpolation. *SIAM Journal on Numerical Analysis*, 17(2):238–246.

Gabriel, S. and Bernstein, D. (1997). The traffic equilibrium problem with nonadditive path costs. *Transportation Science*, 31(4):337–348.

Gazis, D. C., Herman, R., and Rothery, R. W. (1961). Nonlinear follow-the-leader models of traffic flow. *Operations Research*, 19(4):545–567.

Gomez Exposito, A. and Abur, A. (1998). Generalized observability analysis and measurements classification. *IEEE Transactions on Power Systems*, 13:3:1090–1095.

Hall, P. R. (1983). Travel outcome and performance: the effect of uncertainty on accessibility. *Transportation Research. Part B*, 17(4):275–290.

Hazelton, M. L. (2000). Estimation of origin-destination matrices from link flows on uncongested networks. *Transportation Research, Part B*, 34:549–566.

Hazelton, M. L. (2001). Inference for origin-destination matrices: estimation, prediction and reconstruction. *Transportation Research, Part B*, 35:7:667–676.

Hearn, D. (1982). The gap function of a convex program. *Operations Research Letter*, 1(2):67–71.

Helbing, D. (1996). Gas-kinetic derivation of Navier-Stokes-like traffic equations. *Physical Review E*, 53:2366–2381.

Herrera, J. C., Work, D. B., Herring, R., Ban, X., Jacobson, Q., and Bayen, A. M. (2010). Evaluation of traffic data obtained via gps-enabled mobile phones: The mobile century field experiment. *Transportation Research Part C*, 18:568–583.

Hopf, E. (1950). The partial differential equation $u_t + uu_x = \mu_{xx}$. *Communications in Pure and Applied Mathematics*, 3:201–230.

Hu, S. H., Peeta, S., and Chu, C. (2009). Identification of vehicle sensor locations for link-based network traffic applications. *Transportation Research, Part B*, 43:873–894.

Huang, H. and Lam, W. (2002). Modeling and solving dynamic user equilibrium route and departure time choice problem in network with queues. *Transportation Research*, 36(B):253–273.

Jackson, W. and Jucker, J. (1981). An empirical study of travel time variability and travel choice behaviour. *Transportation Science*, 16(4):460–475.

Janson, B. (1991). Dynamic traffic assignment for urban road networks. *Transportation Research Part B*, 25(2,3):143–162.

Janson, B. and Robles, J. (1995). A quasy-continuous dynamic traffic assignment model. *Transportation Research Record*, 1493:199–206.

Kalaee, M. (2010). *Investigating freeway speed-flow relationships for traffic assignment applications.*. Ph.D. thesis, Portland State University, Oregon.

Klieter, G. (1992). Bayesian diagnosis in expert systems. *Proceedings of the AIJ92*.

Lam, W. and Huang, H. (1995). Dynamic user optimal traffic assignment model for many to one travel demands. *Transportation Research Part B*, 29(4):243–259.

Larsson, T., Lindberg, P. O., Lundgren, J., Patriksson, M., and Rydergren, C. (2002). *Transportation Planning - State of the Art*, chap. On traffic equilibrium models with a nonlinear time/money relation, pp. 19–31. Kluwer Academic Publishers, Boston, MA.

Lax, P. D. (1954). Week solutions of nonlinear hyperbolic equations and their numerical computation. *Communications in Pure and Applied Mathematics*, 7:159–193.

Lighthill, M. J. and Whitham, G. B. (1955). On kinematic waves. I: Flow movement in long rivers; II: A theory of traffic flow on long crowded roads. *Proceedings of Royal Society, Series A*, 229:281–345.

Lo, H. (1999). *Transportation and Traffic Theory*, chap. A Dynamic Traffic Assignment Formulation that encapsulates the Cell Transmission Model, pp. 327–350. Elsevier Science.

Lo, H. and Szeto, W. Y. (2002). A cell-based variational inequality formulation of the dynamic user optimal assignment problem. *Transportation Research Part B*, 36(5):421–443.

Lo, H. K. and Chen, A. (2000). Reformulating the traffic equilibrium problem via a smooth gap function. *Mathematical and Computer Modelling*, 31:179–195.

Lo, H. K., Luo, X. W., and Siu, B. (2006). Degradable transport network: Travel time budget of travelers with heterogeneous risk aversion. *Transportation Research, Part B*, 40:792–806.

Lo, H. K. and Tung, Y. K. (2003). Network with degradable links: Capacity analysis and design. *Transportation Research Part B*, 37:345–363.

Lo, H. P., Zhang, N., and Lam, W. (1996). Estimation of an origin-destination matrix with random link choice proportions: a statistical approach. *Transportation Research, Part B*, 30:4:309–324.

Maher, M. (1983). Inferences on trip matrices from observations on link volumes: a Bayesian statistical approach. *Transportation Research Part B*, 17B(6):435–447.

Mahmassani, H. S. and Sinha, K. (1981). A Bayesian updating of trip generation parameters. *Journal of Transportation Engineering (ASCE)*, 107.

Merchant, D. and Nemhauser, G. (1978a). A model and an algorithm for the dynamic traffic assignment model problem. *Transportation Science*, 12:183–199.

Merchant, D. and Nemhauser, G. (1978b). Optimality conditions for a dynamic traffic assignment model. *Transportation Science*, 12:200–207.

Mínguez, R., Sánchez-Cambronero, S., Castillo, E., and Jiménez, P. (2010). Optimal traffic plate scanning location in road networks. *Transportation Research B*, 44:282–298.

Monticelli, A. and Wu, F. (1985a). Network observability. Identification of observable islands and measurement placement. *IEEE Transactions on Power Apparatus Systems*, 104:5:1035–1041.

Monticelli, A. and Wu, F. (1985b). Network observability theory. *IEEE Transactions on Power Apparatus Systems*, 104:5:1042–1048.

Mosher, W. (1963). A capacity restraint algoritm for assigning flow to a transport network. *Highway Research Rec.*, 6:41–70.

Murchland, J. (1969). Road network traffic distribution in equilibrium. In *conference on Mathematical Methods in the Economic Sciences, Mathematisches Forschungsinstitut, Oberwolfach*. Oberwolfach, Germany.

Nagurney, A. (1999). *Network Economics: A Variational Inequality Approach*. Kluwer Academic Publishers, Boston, Massachusetts., second and revised edition ed.

Newell, G. F. (1993a). A simplified theory on kinematic wave in highway traffic, part I: general theory. *Transportation Research Part B*, 27(4):281–287.

Newell, G. F. (1993b). A simplified theory on kinematic wave in highway traffic, part II: queuing at freeway bottlenecks. *Transportation Research Part B*, 27(4):289–303.

Newell, G. F. (1993c). A simplified theory on kinematic wave in highway traffic, part III: multi-destination flows. *Transportation Research Part B*, 27(4):305–313.

Ng, M. W. (2012). Synergistic sensor location for link flow inference without path enumeration: A node-based approach. *Transportation Reseach Part B*, 46(6):781–788.

Nguyen, S. (1977). *Estimating an OD matrix from network data: A network equilibrium approach*. Publication 87, CRT, University of Montreal, Montreal, Canada.

Nguyen, S. and Dupuis, C. (1984). An efficient method for computing traffic equilibria innetworks with asymmetric transportation costs. *Transportation Science*, 18:185–202.

Nie, Y. (2011). Multi-class percentile user equilibrium with flow-dependent stochasticity. *Transportation Research Part B*, 45(10):1641–1659.

Nie, Y. and Wu, X. (2009). Shortest path problem considering on-time arrival probability. *Transportation Research Part B*, 43:597–613.

Nie, Y. and Zhang, H. M. (2005). Delay-function-based link models: their properties and computational issues. *Transportation Research Part B*, 39:729–751.

Nie, Y. and Zhang, H. M. (2008). A variational inequality formulation for inferring dynamic origin-destination travel demands. *Transportation Research Part B*, 42:635–662.

Nogal, M. (2011). *Mathematic Methods for Traffic Prediction*. Ph.D. thesis, University Of Cantabria, Santander, Spain.

Nucera, R. and Gilles, M. (1991). Observability analysis: A new topological algorithm. *IEEE Transactions on Power Systems*, 6:2:466–475.

Paveri Fontana, S. L. (1975). On Boltzmann like treatments for traffic flow. *Transportation Research Part B*, 9:225–235.

Praskher, J. N. and Bekhor, S. (2004). Route choice models used in the stochastic user equilibrium problem: a review. *Transport Reviews*, 24:437–463.

Prigogine, I. and Herman, R. (1971). *Kinetic theory of vehicular traffic*. American Elsevier, New York.

Ran, B. and Boyce, D. (1996). *Modeling dynamic transportation networks. An intelligent transportation system oriented approach*. Springer, Heidelberg, second revised edition ed.

Ran, B., Boyce, D., and Leblanc, L. (1993). A new class of instantaneous dynamic user-optimal traffic assignment models. *Operations Research*, 41(1):192–202.

Ran, B., Hall, R., and Boyce, D. (1996). A link-based variational inequality model for dynamic departure time/route choice. *Transportation Research Part B*, 30(1):31–46.

Richards, P. I. (1956). Shock waves on the highway. *Operations Research*, 4(1):42–51.

Rose, G., Daskin, M., and Koppelman, F. (1988). An examination of convergence errors in equilibrium traffic assignment models. *Transportation Rearch Part B*, 22(4):261–274.

Rossi, T. F., McNeil, S., and Hendrickson, C. (1989). An entropy model for consistent impact fee assessment. *Journal of Urban Planning and Development, ASCE*, 115:51–63.

Rubio-Ardanaz, J., Wu, J., and Florian, M. (2001). A numerical analytical model for the continuous dynamic network equilibrium problem with limited capacity and spill back. *IEEE Intelligent Transportation Systems Conference Proceedings*, pp. 263–267.

Rubio-Ardanaz, J. M., Wu, J. H., and Florian, M. (2003). Two improved numerical algorithms for the continuous dynamic network loading problem. *Transportation Research Part B*, 37:171–190.

Sheffi, Y. (1985). *Urban transportation networks: equilibrium analysis with mathematical programming methods*. N. Prentice Hall, Englewood Cliffs.

Smith, M. (1979). The existence, uniqueness and stability of traffic equilibria. *Transportation Research Part B*, 13(4):295–304.

Smith, M. (1983). The existence and calculation of traffic equilibria. *Transportation Research Part B*, 17:291–303.

Soriguera, F. and Robusté, F. (2011). Highway travel time accurate measurement and short-term prediction using multiple data sources. *Transportmetrica*, 7(1):85–109.

Soriguera, F., Rosas, D., and Robusté, F. (2010). Travel time measurement in closed toll highways. *Transportation Research Part B*, 44(10):1242–1267.

Soriguera, F., Thorson, L., and Robusté, F. (2007). Travel time measurement using toll infrastructure. *Transportation Research Record*, 2027:99–107.

Spiess, H. (1987). A maximum likelihood model for estimating origin-destinations matrices. *Transportation Research, Part B*, 21:5:395–412.

Spiess, H. (1990). Conical volume-delay functions. *Transportation Science*, 24(2):153–158.

Stokes, G. G. (1845). On the theories of internal friction of fluids in motion. *Transactions of the Cambridge Philosofical Society*, 8.

Suh, S. and Kim, T. (1990). A highway capacity function in korea: Measurement and calibration. *Transportation Research Part A*, 24(3):177–186.

Sun, S. L., Zhang, C. S., and Yu, G. Q. (2006). A Bayesian network approach to traffic flow forecasting. *IEEE Transactions on Intelligent Transportation Systems*, 7(1):124–132.

Svensson, A. (1978). An equilibrium equation for road traffic. *Transportation Research*, 12(5):309–313.

Szeto, W., O'Brien, L., and O'Mahony, M. (2006). Risk-averse traffic assignment with elastic demand: Ncp formulation and solution method for assesing performance reliability. *Network and Spatial Economics*, 6(3-4):313–332.

Szeto, W. Y. (2003). *Dynamic Traffic Assignment: Formulations, Properties and Extensions*. Ph.D. thesis, The Hong Kong University of Science and Technology, Hong Kong.

Szeto, W. Y. and Lo, H. (2004). A cell-based simultaneous route and departure time choice model with elastic demand. *Transportation Research Part B*, 38:593–612.

Szeto, W. Y. and Lo, H. K. (2005). Properties of dynamic traffic assignment with physical queues. *Journal of the Eastern Asia Society for Transportation Studies*, 6:2108–2123.

Szeto, W. Y. and Lo, H. K. (2006). Dynamic traffic assignment: Properties and extensions. *Transportmetrica*, 2(1):31–52.

Tebaldi, C. and West, M. (1998). Inference on network traffic using link count data. *Journal of the American Statistical Association*, 93:557–576.

Traffic-Research-Corporation (1966). *Winnipeg Area Transportation Study*. Streets an Transit Division of the Metropolitan Corporation of Greater Winnipeg, Winnipeg.

Uchida, T. and Iida, Y. (1993). Risk assignment: A new traffic assignment model considering risk of travel time variation. In C. Daganzo, ed., *Proceedings of the 12th International Symposium on Transportation and Traffic Theory*, pp. 89–105. Elsevier, New York.

Van Zuylen, H. J. (1978). The information minimization method: validity and applicability to transport planning. In *New Developments in Modeling Travel Demand and Urban Systems.*. Jansen, G. R. H. et al. (Eds), Saxon, Farnborough.

Van Zuylen, H. J. and Willumsen, L. (1980). Network tomography: The most likely trip matrix estimated from traffic-counts. *Transportation Research, Part B*, 14:291–293.

Vardi, Y. (1996). Network tomography: Estimating source-destination traffic intensities from link data. *Journal of the American Statistical Association*, 91:365–377.

Wagner, C., Hoffmann, C., Sollacher, R., Wagenhuber, J., and Schürmann, B. (1996). Second-order continuum traffic flow model. *Physical Review E*, 54(5):5073–5085.

Wardrop, J. (1952). Some theoretical aspects of road traffic research. In *Proceedings of the Institute of Civil Engineers II*, pp. 325–378.

Watling, D. (2006). User equilibrium traffic network assignment with stochastic travel times and late arrival penalty. *European Journal of Operational Research*, 175:1539–1556.

Whitham, G. B. (1974). *Linear and Nonlinear waves*. Wiley, New York.

Willumsen, L. G. (1978). Estimation of od matrix from traffic counts. a review. *Institute of Transportation Studies, University of Leeds*.

Wu, X. and Nie, Y. (2011). Modeling heterogeneous risk-taking behavior in route choice: a stochastic dominance approach. *Transportation Research Part A*, 45(9):896–915.

Xu, Y., Wu, J., Florian, M., Marcotte, P., and Zhu, D. (1999). Advances in the continuous dynamic network loading problem. *Transportation Science*, 33(4):341–353.

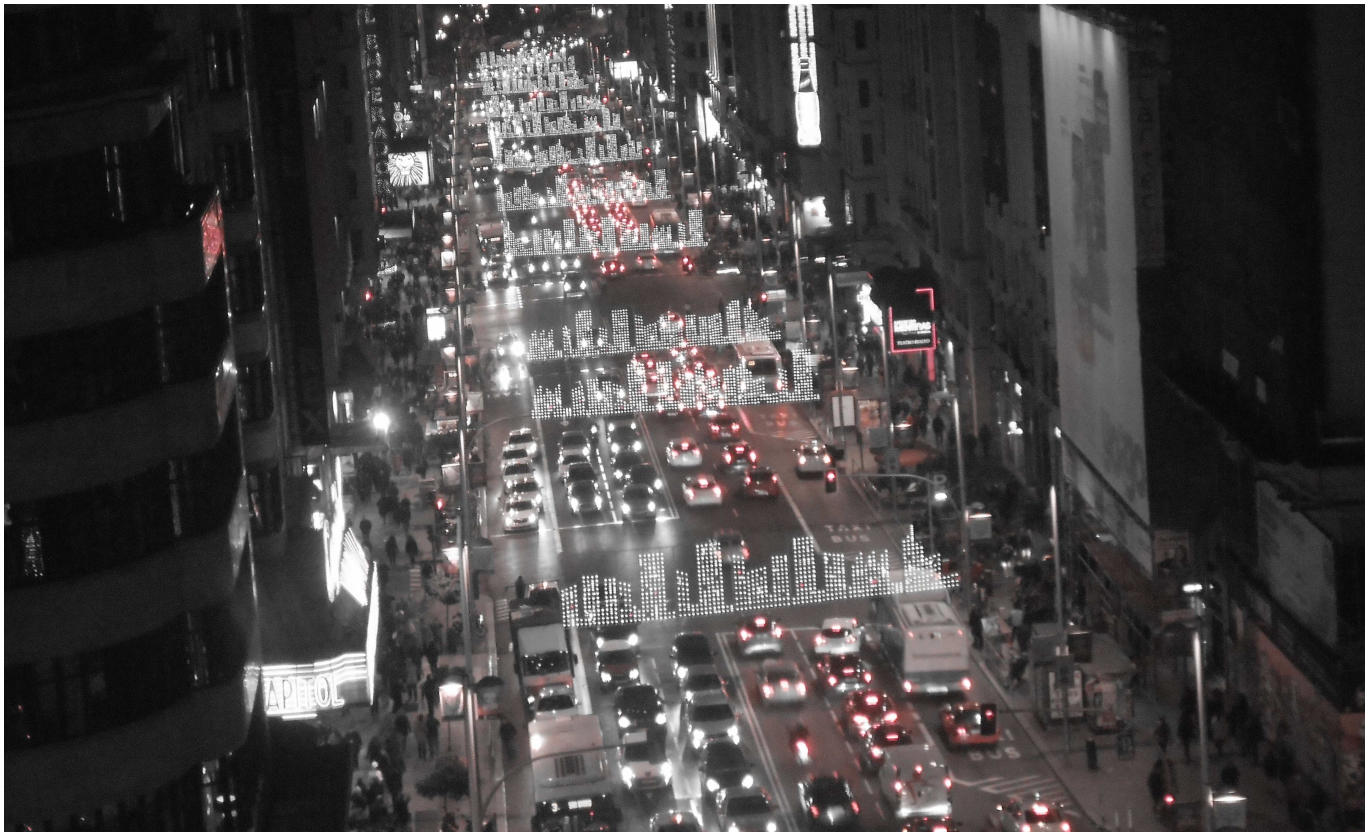
Yang, H. (1995). Heuristic algorithms for the bi-level origin-destination matrix estimation problem. *Transportation Research, Part B*, 29:231–242.

Yang, H. and Huang, H. (1997). Analysis of the time-varying pricing of a bottleneck with elastic demand using optimal control theory. *Transportation Research Part B*, 31:425–440.

Yang, H., Sasaki, T., Iida, Y., and Asakura, Y. (1992). Estimation of origin-destination matrices from link traffic counts on congested networks. *Transportation Research, Part B*, 26:417–434.

Zhou, X. and Mahmassani, H. (2006). Dynamic origin-destination demand estimation using automatic vehicle identification data. *IEEE Transactions on Intelligent Transportation Systems*, 7(1):105–114.

Index



Esta tesis presenta los siguientes modelos matemático-estadísticos originales:

- Dos modelos estáticos de asignación de tráfico con usuarios heterogéneos que permiten obtener los flujos de las rutas y los arcos, conocidos los flujos entre pares origen-destino. Dichos modelos consideran distintas clases de usuarios según su deseo de puntualidad y adelantamiento, respectivamente.
- Un modelo bayesiano de estimación de matrices origen-destino basado en técnicas de optimización jerárquica. Las estimaciones se obtienen a partir de la información ofrecida por arcos escaneados.
- Se calcula el mínimo conjunto de arcos que debe ser equipado con sensores para obtener observabilidad total a partir de los flujos en arcos.
- Un modelo continuo para el problema dinámico de recarga de red incluyendo adelantamientos que proporciona los flujos y tiempos de viaje en los arcos de la red en cualquier instante del intervalo de tiempo en estudio.
- Algunos métodos gráficos para analizar trayectorias de tráfico con y sin adelantamiento, que permiten analizar el estado de una red (velocidad, aceleración, etc.) a partir de las características físicas de los gráficos de trayectorias (pendiente, curvatura, etc.).

Todos los modelos han sido evaluados en redes de tráfico ficticias y reales (las ciudades españolas de Cuenca y Ciudad Real), con el fin de analizar sus características y evaluar la validez de los resultados.

Asimismo, se incluye una revisión de la literatura que permite contextualizar los modelos originales propuestos en esta tesis.