METODOS MATEMATICOS PARA LA PREDICCION DE TRAFICO

TESIS DOCTORAL

MARIA NOGAL MACHO

Director: ENRIQUE CASTILLO

UNIVERSIDAD DE CANTABRIA

ETS INGENIEROS
DE CAMINOS,
CANALES Y PUERTOS

To my grandparents Eloina and Quintín,
for being an example of sacrifice, goodness and happiness.

Para mis abuelos Eloina y Quintín,
por ser un ejemplo de sacrificio, bondad y felicidad.

Acknowledgments

I am glad to have this section that provides me the opportunity to write some words that must be said. However I regret that there is no a good enough sentence to express my deep feeling of gratitude to some people.

The first one of these persons is Enrique Castillo, my thesis director. It is a “thanks” in capital letters, full of respect and admiration. I sincerely feel myself fortunate for having been able to approach to his way of understanding job, research and life. Enrique has given me his time and support. In addition, he has given me the chance of meeting and working with the Ciudad Real team: José María, Ana, Inmaculada, Santos and Pilar. All of them have been very kind to me, treating me as one more of the team since the very first day. They have also provided me with all the necessary resources in order not to miss anything. In all of them I have found the friendship, in particular, in Pilar and Santos.

The other pillar of this thesis is, of course, my family: parents and my three sisters, Esther, Lili and Anita. They knew that doing research and becoming a doctor were always my dreams so that, they have encouraged me to reach my objective. Specially my mother, who lives each of my troubles and achievements like her own ones.

I cannot forget my couple, Mario. He has been waiting me with a great smile on his face everyday. He has adapted to my (difficult)time schedule and made my life easier.

Furthermore, I am also grateful to the Universities of Cantabria and Castilla-La Mancha, and to the Spanish Ministry of Science and Technology for their economic support.

To finish, to Emilio Muela, a great boss who let me to combine job with studies, helping me to begin this fantastic experience.

With all my heart, thank you very much.

Me alegro de disponer de esta sección para escribir algunas palabras que deberían ser dichas. Sin embargo, lamento que no haya frase suficientemente buena para expresar mi profundo sentimiento de gratitud a ciertas personas.

La primera de esas personas es Enrique Castillo, mi director de tesis. Se trata de un “gracias” en letras mayúsculas, lleno de respeto y admiración. Me siento francamente afortunada por haberme podido acercar a su forma de entender el trabajo, la investigación y la vida. Enrique me ha dado su tiempo y apoyo. Además, me ha brindado la oportunidad *de conocer y trabajar con el equipo de Ciudad Real: José María, Ana, Inmaculada, Santos y Pilar. Todos ellos se han portado muy bien conmigo, tratándome como una más del equipo desde el primer día. También me han proporcionado todos los recursos necesarios con el fin de que no me faltara nada. En todos ellos he encontrado la amistad, en particular, en Pilar y Santos.*

El otro pilar de esta tesis es, por supuesto, mi familia: mis padres y mis tres hermanas, Esther, Lili y Anita. Ellos sabían que la investigación y llegar a ser doctor fue siempre mi ilusión y, por eso, me han animado a alcanzar mi objetivo. Especialmente mi madre, que vive cada uno de mis problemas y logros como los suyos propios.

No puedo olvidar a mi pareja, Mario. Él me ha esperado todos los días con una gran sonrisa en la cara. Se ha adaptado a mi (difícil) horario y me ha hecho la vida más fácil.

Además, también estoy agradecida a las Universidades de Cantabria y Castilla-La Mancha, y al Ministerio español de Ciencia y Tecnología por el apoyo económico.

Para terminar, a Emilio Muela, un gran jefe que me permitió combinar trabajo y estudios, ayudándome a empezar esta fantástica experiencia.

Con todo mi corazón, muchas gracias.

Abstract

Traffic engineering raises a great deal of aspects and problems related to mobility of people and goods. In this thesis we have centered on traffic models for studying the vehicular flow in traffic networks. We have tried to expose a general vision of the most common current existing approaches and give some solutions for three concrete problems selected among a great collection of them. The proposed models have been sent for publication to the best especialized journals and they have been accepted with congratulation words.

This thesis provides the reader the following contributions:

- **Literature review.** A literature review about existing traffic approaches, classified in static and dynamic models. Much of these approaches are illustrated with some examples for a better understanding of the main concepts and ideas.
- **A traffic flow reconstruction and estimation Bayesian approach based on plate scanning.** We present a new conjugate Bayesian model for reconstructing and estimating traffic flows, based on plate scanning information, which permits us to identify, totally or partially, the vehicle route, origin-destination and link flows by scanning their corresponding plate numbers at an adequately selected subset of links.
- **A dynamic traffic loading model.** A FIFO rule consistent model for solving the continuous dynamic network loading problem is provided. The model assumes a non-linear link travel time function of the link volumes and considers the congestion effect taking into account the interaction of flows of all paths and their coincidence at different times and locations. In addition, it considers the effect of a link congestion on the upstream route links.
- **A dynamic traffic models with stochastic demand.** We present a dynamic traffic model with stochastic demand for predicting some traffic variables such as link travel times, link flows or link densities and their time evolution in real networks. The model is a Beta-normal Bayesian network, whose probabilistic structure is learnt by analyzing the correlation coefficients among variables.

- **Practical applications.** All the proposed new models are tested in fictitious and real traffic networks in order to analyze their characteristics and performances together with the validity of the results and the associated computacional requirements.
- **Programm codes.** We present the computational implementation of the formula and algorithms proposed 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

Con frecuencia, la ingeniería y las políticas de transporte y planificación precisan conocer los futuros usuarios en el diseño de una nueva vía de transporte o el impacto en el tráfico a largo plazo de un nuevo aeropuerto, urbanización, etc., con el fin de conseguir un dimensionamiento adecuado o estimar la viabilidad financiera y social de los proyectos. Por otra parte, a corto plazo, debido al aumento de congestión del tráfico en las ciudades, se hacen necesarias medidas de gestión del tráfico, como pueden ser los sistemas urbanos de control de tráfico tipo SCOOT (Split Cycle Offset Optimization Technique) o el sistema SCAT (Sydney Coordinated Adaptive Traffic), que coordinan la operación de todas las señales de tráfico en un área para dar fluidez a los vehículos a través de la red. Estos sistemas obtienen información sobre los flujos de tráfico en tiempo real, que es enviada a una unidad central donde se procesa, y se toman decisiones sobre ese tráfico. Los beneficios no sólo se plasman en la reducción de los tiempos de recorrido, en la mejora del flujo, y la disminución de la congestión, sino también en la reducción de accidentes, ahorro de combustible, la disminución de la contaminación del aire, y una mejora de las comodidades residenciales.

Los modelos de tráfico son una de las herramientas más utilizadas para resolver los problemas mencionados. Hay muchos modelos diferentes, en función de los objetivos que se persigan, a saber, si predomina el corto o largo plazo, si lo que se quiere es predecir o estimar el tráfico, etc. Una respuesta técnicamente adecuada para estos problemas debe ser lo más simple posible, pero debe ofrecer una solución práctica al problema principal en cada caso.

1.2 Contenido de la tesis

Esta tesis está organizada básicamente en tres partes más un apéndice.

La primera parte incluye una revisión de la literatura existente acerca de los distintos enfoques en el estudio del tráfico. Mas concretamente, se presentan los modelos de tráfico estáticos y dinámicos. Gran parte de los métodos revisados se ilustran con ejemplos para una mejor comprensión de los conceptos e ideas principales.

En la segunda parte, se presentan los siguientes modelos matemáticos originales:

- *Un modelo para la reconstrucción y estimación de los flujos de tráfico a partir de la lectura de las matrículas.* Se aporta un modelo bayesiano conjugado para la reconstrucción y estimación de los flujos de tráfico, basado en modelos α-Gamma $\Gamma(\theta, \lambda)$ desplazada, $\mathcal{H}(\alpha, \theta, \lambda)$. Si se supone que el número de usuarios que viajan a través de diferentes rutas son variables independientes $\mathcal{H}(\alpha, \theta, \lambda)$ con λ común, los flujos en arcos, origen-destino (OD) y nodos son también variables aleatorias $\mathcal{H}(\alpha, \theta, \lambda)$. Suponemos que la principal fuente de información proviene de la lectura automática de las matrículas, lo que nos permite identificar, total o parcialmente, el flujo en ruta, arco u OD, mediante el escaneo de las matrículas que pasan por un subconjunto adecuadamente seleccionado de arcos. La reconstrucción de los flujos de la muestra se puede hacer exacta o aproximadamente, dependiendo de la intensidad del proceso de muestreo de matrículas. Con este fin, se usa una técnica de mínimos cuadrados generalizada junto con las leyes de conservación. Se propone un método bayesiano usando familias conjugadas, que permite calcular los diferentes flujos de tráfico, tales como los de rutas, OD, arcos escaneados o arcos aforados. Además se proporciona una descripción detallada de cómo se realiza la evaluación previa, la toma de muestras, la posterior actualización y la obtención de la distribución bayesiana.

- *Un modelo de recarga de tráfico dinámico innovador.* Se aporta un modelo consistente con la regla FIFO para resolver el problema dinámico, continuo en el tiempo, de recarga de red. El modelo calcula las funciones de los tiempos de recorrido de los arcos en un conjunto finito de tiempos básicos equidistantes que se utilizan para interpolar un spline monótono para obtener un tiempo continuo. El modelo supone que el tiempo de recorrido del arco es función no lineal de los volúmenes de tráfico del mismo, pero se realizan algunas correcciones con el fin de cumplir la regla FIFO en ese conjunto básico de tiempos estudiado. Además, el uso de splines monótonos cúbicos garantiza la monotonía que se precisa para cumplir la regla FIFO en todo momento. El modelo consta de cinco unidades: (1) una unidad de definición de la onda de flujo de la ruta, es decir, las funciones de intensidad de flujo de las rutas en sus orígenes mediante una función temporal; (2) una unidad de propagación de esas ondas individuales de cada ruta a través de sus correspondientes rutas; (3) una unidad de análisis del grado de congestión de los distintos arcos y cómo cambia con el tiempo; (4) una unidad para combinar todas las ondas de las rutas resultando las ondas de flujo total que se propaga a través de los arcos y nodos; y (5) una unidad de compatibilidad de todo el proceso. La onda de intensidad flujo de la ruta, que es la información básica de la que se parte, se modela como una combinación lineal de ondas básicas. A continuación, las ondas se propagan a través de la red por sus rutas, mediante el uso de la ecuación de conservación que aumenta o disminuye la longitud de onda, modificando la altura de las ondas en función del grado de congestión en los diferentes arcos. Entonces, esas ondas individuales de cada ruta se combinan entre sí para generar las ondas de los arcos y nodos. Para terminar, el sistema combina toda la información para hacerla compatible en horarios y lugares utilizando un método iterativo hasta la convergencia.

- *Un modelo dinámico de tráfico con demanda estocástica.* Se aporta un modelo de tráfico dinámico con demanda estocástica para la predicción de algunas variables de tráfico, tales como los tiempos de recorrido, flujos o densidad de los arcos y su evolución en el tiempo en redes reales. El modelo considera que las variables son variables Beta generalizadas, de manera que, cuando se transforman sus marginales a normal estándar, pasan a ser normales multivariadas. Esto le da suficientes grados de libertad para reproducir (aproximadamente) las variables consideradas en un conjunto discreto de pares tiempo-situación. Se dan dos opciones para conocer los parámetros del modelo: (a) una basada en observaciones anteriores de las mismas variables, y (b) otra basada en datos simulados utilizando modelos dinámicos. El modelo es capaz de proporcionar una estimación puntual, un intervalo de confianza o la función de densidad de la variable que se predice. Con este fin, se proporciona una fórmula cerrada para obtener los valores de las variables futuras condicionadas (tiempos de recorrido de los arcos o flujos), a partir de la información disponible sobre las variables en tiempos recientes. Ya que sólo la información local y reciente es relevante para realizar las predicciones de los flujos de los arcos a corto plazo, el modelo es aplicable a las redes de gran tamaño.

Todos estos modelos propuestos se han probado en redes de tráfico ficticias y reales con el fin de analizar sus características, validez de los resultados y los correspondientes requisitos computacionales.

En la tercera parte, se describen las conclusiones derivadas de cada uno de los modelos y se muestran posibles líneas de actuación futuras.

Por último, se añade un apéndice donde se presenta la implementación computacional de las fórmulas y algoritmos propuestos en esta tesis, que han sido utilizados para obtener las aplicaciones prácticas mencionadas.

1.3 Redes de tráfico utilizadas

Cada uno de estos modelos se desarrolla en un capítulo diferente de esta tesis, en la parte de contribuciones originales. Además, los tres modelos se ilustran con algunos ejemplos:

1. *La red de Nguyen Dupuis.* Se trata de una pequeña red ficticia de uso común en la literatura del transporte. Se utiliza en los tres modelos ya que, debido a su reducido tamaño, permite explicar y analizar fácilmente los procedimientos y discutir los resultados.

2. *La red de Cuenca y la de Ciudad Real (España).* Son redes reales de tamaño medio. Sus topologías se utilizan para simular los flujos de tráfico mediante el modelo consistente con la regla FIFO. Los resultados, después de simular 100 veces cada uno de los ejemplos, se aplican al modelo de demanda estocástica. Por otra parte, la red de Cuenca se utiliza en el modelo bayesiano de reconstrucción-estimación.

3. *El ejemplo del Estado de Vermont (EE.UU.).* Se trata de una red real de gran tamaño. Se cuenta con los datos (número de usuarios) registrados en 85 estaciones fijas en todo el estado de Vermont durante dos años, los 365 días, todas las horas del día. Esta red se utiliza en el modelo de demanda estocástica.

1.4 Contribuciones

Las aportaciones originales de esta tesis se basan en los tres modelos mencionados. A continuación se destaca el interés de cada uno de ellos:

Las principales aportaciones del modelo de estimación bayesiana basada en la lectura de matrículas (véase el capítulo 6) son los siguientes:

1. El modelo *conjugado Gamma bayesiano* propuesto tiene una propiedad importante: si los flujos de ruta siguen esta distribución, el resto de flujos (OD, arcos, nodos y flujos escaneados) siguen esta distribución. Esto significa que una sola familia nos permite reproducir todos los flujos. Una vez seleccionado un modelo conjugado, se proporcionan fórmulas cerradas para actualizar los valores a posteriori a partir de los datos disponibles de la muestra.

2. El método bayesiano, mediante las distribuciones a posteri, permite conocer los intervalos de confianza, es decir, no sólo proporciona una estimación puntual, sino

los intervalos con sus probabilidades asociadas. Esta es una diferencia importante con respecto a otros modelos existentes.

3. La técnica de la lectura de matrículas permite reproducir exactamente los flujos reales si el conjunto de arcos escaneados están elegidos adecuadamente. Esto no es posible con la técnica estándar de aforar los arcos.

4. Los costes computacionales requeridos por este método para los casos de redes reales son bajos. Esto significa que se puede aplicar a grandes redes.

Las principales aportaciones del modelo de dinámico recarga de la red consistente con la regla FIFO (véase el capítulo 7) son los siguientes:

1. Se trata de un modelo continuo para redes reales de tráfico, que parece más conveniente que una versión discretizada del problema continuo. Aunque las funciones de tiempo implican infinitos grados de libertad, se interpolan estas funciones mediante splines monótonos cúbicos. Esto implica que el número de parámetros utilizados para definir los modelos sigue siendo limitado y pequeño.

2. Se supone que el tiempo de recorrido del arco es una función no lineal de los volúmenes del mismo, incluyéndose una serie de correcciones con el fin de cumplir la regla FIFO. La idea consiste en añadir el retraso mínimo de tal forma que se cumpla (esto es equivalente a considerar una cola de vehículos debido a la congestión y su posterior disipación), que se complementa con el uso de la interpolación cúbica monótona que garantiza la satisfacción de la regla FIFO en el puntos interpolados.

3. Se considera el efecto de la congestión en la ubicación y momento adecuado, es decir, el modelo evalúa el efecto de la congestión teniendo en cuenta la interacción de los flujos de todas las rutas y su coincidencia en diferentes momentos y lugares. Por otra parte, se tiene en cuenta el efecto de la congestión del arco en los arcos aguas abajo de la ruta.

4. Se usa un esquema iterativo. En una iteración dada, primero se determinan los tiempos de recorrido del arco asociado a un conjunto seleccionado cuidadosamente de usuarios basándose en la interpolación mediante splines cúbicos de las funciones de tiempo de recorrido del arco de una iteración anterior y, después, se actualizan los splines monótonos cúbicos, ajustando los nuevos tiempos de recorrido. El proceso se repite hasta la convergencia.

5. El método propuesto tiene costes computacionales moderados.

Las principales contribuciones del modelo de red bayesiana beta generalizada-gaussiana (véase el capítulo 8) son los siguientes:

1. La red beta generalizada gaussiana, cuyas marginales son variables beta generalizadas, es decir, variables con un rango de $[b_i, c_i]$ en vez del rango de las variables beta estándar $[0, 1]$, permite definir la distribución marginal de cada variable independiente, de acuerdo con sus características.

2. La estructura de covarianza de las variables de flujo se modela por medio de una distribución normal multivariada. El alto número de parámetros del modelo propuesto es muy fácil de estimar utilizando el método de la máxima verosimilitud como método de estimación.

3. La selección de las variables relevantes para realizar una predicción dada se realiza con la ayuda de los coeficientes de correlación entre las variables.

4. El modelo es espacio-temporal, es decir, se incluyen los tiempos y lugares.

5. Se aporta una comparación exhaustiva con otros modelos bayesianos existentes.

6. El modelo propuesto para hacer predicciones requiere un tiempo de CPU pequeño, de forma que el método se puede aplicar a grandes redes.

Part II

Introduction and State-of-the-art

Chapter 2

Introduction and contributions

Contents

2.1 Introduction

Frequently engineering, transportation policy and planning require to know the future users on a planned road or the long-term traffic impact of a new airport, housing development, etc., in order to dimension adequately or estimate the financial and social viability of the projects. On the other hand, on a short-term, due to the more and more congested traffic, cities have been forced to adopt some traffic management measures. One of these measures - increasingly common- is the use of urban traffic control systems, such as the Split Cycle Offset Optimization Technique (SCOOT) system or the Sydney Coordinated Adaptive Traffic (SCAT) system, which coordinate the operation of all the traffic signals in an area to give good progression to vehicles through the network. They obtain real-time information on traffic flows that is sent to a central. Here, the information 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.

Traffic models are very useful tools to solve the above mentioned problems. There are a lot of different models according to the objectives to be reached, namely, sort 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 this thesis we present the following mathematical models:

- *A traffic flow reconstruction and estimation model based on plate scanning.* A conjugate Bayesian model for reconstructing and estimating traffic flows, based on α-shifted Gamma $\Gamma(\theta, \lambda)$ models $\mathcal{H}(\alpha, \theta, \lambda)$ is given. If the numbers of users traveling through different routes are assumed to be independent $\mathcal{H}(\alpha, \theta, \lambda)$ variables with common λ, the link, origin-destination (OD) and node flows are also $\mathcal{H}(\alpha, \theta, \lambda)$ random variables. We assume that the main source of information is plate scanning, which permits us to identify, totally or partially, the vehicle route, OD and link flows, by scanning their corresponding plate numbers at an adequately selected subset of links. The reconstruction of the sample flows can be done exactly or approximately, depending on the intensity of the plate scanning sampling procedure. To this end, a generalized least squares technique is used together with the conservation laws. A Bayesian approach using conjugate families is proposed that allows us to estimate different traffic flows, such as route, OD-pair, scanned link or counted link flows. A detailed description of how the prior assessment, the sampling, the posterior updating and the obtention of the Bayesian distribution is given.

- *A new dynamic traffic loading model.* A FIFO rule consistent model for the continuous dynamic network loading problem is given. 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. The model assumes a non-linear link travel time function of the link volumes, but some corrections are made in order to satisfy the FIFO rule at the basic set. Furthermore, the use of monotone cubic splines preserving monotonicity guarantees that the FIFO rule is satisfied at all points. The model consists of five units: a path origin flow wave definition unit, a path wave propagation unit, a congestion analysis unit, a network flow propagation unit and an inference engine unit. The path flow intensity wave, which is the basic information, is modeled as a linear combination of basic waves. Next, the individual path waves are propagated throughout the paths by using a conservation equation that stretches or enlarges the wave lengths and increase or reduce the wave heights depending on the degree of congestion at the different links. Then, the individual path waves are combined together to generate the link and node waves. Finally, the inference engine unit combines all information items to make them compatible in times and locations using an iterative method until convergence.

- *A dynamic traffic model with stochastic demand.* A stochastic demand dynamic traffic model for predicting some traffic variables such as link travel times, link flows or link densities and their time evolution in real networks is given. The model considers that the variables are generalized Beta variables such that when marginally transformed to standard normal become multivariate normal. This gives sufficient degrees of freedom to reproduce (approximate) the considered variables at a discrete set of time-location pairs. Two options to learn the parameters of the model are provided: (a) one based on previous observations of the same variables, and (b) based on simulated data using existing dynamic models. The model is able to provide a point estimate, a confidence interval or the density of the variable being predicted. To this end, a closed formula for the conditional future variable values (link travel times or flows) given the available past variable information, is provided. Since only local information is relevant to short-term link flow predictions, the model is applicable to very large networks.

Each of these models is developed in a different chapter of this thesis, in the original contributions part. In addition, the three models are illustrated by some examples:

1. The Nguyen Dupuis network. It is a small fictitious network commonly used in the transportation literature. It is utilized by the three models because due to its small size it allows us to explain and analyze easily the procedures and discuss the results.

2. The Cuenca (Spain) and Ciudad Real (Spain) networks. They are real networks of medium size. Their topologies are used to simulate traffic flows through the FIFO rule consistent model. The results, after simulating 100 times each one, are applied to the stochastic demand model. Moreover, the Cuenca network is used in the reconstruction and estimation Bayesian model.

3. The Vermont-State (US) example. It is a large real network. We have the collection of data (number of users) registered in 85 fixed stations located throughout the state of Vermont for the 365 days for two years, every hour of the day. This network is used in the stochastic demand model.

2.2 Contributions

The original contributions of this thesis are based on the three mentioned models. Now we highlight the interest of each of them:

The main contributions of the Bayesian estimation model based on plate scanning (Chapter 6) are:

1. The proposed conjugate Bayesian Gamma model has an important property: if the path flows follow this distribution, all other flows (OD, link, node and scanned flows) follow this distribution. This means that a single family permits us to reproduce all flows. Since a conjugate model has been selected, closed formulas are given to update posteriors when the sample data becomes available.

2. The Bayesian approach through the posterior distributions allows us to provide credible intervals, that is, not only a point estimate can be provided, but the associated uncertainty. This is an important difference with respect to other existing models.

3. The use of the plate scanning technique permits us to reproduce exactly the real flows if the set of scanned links is adequately chosen. This is not possible with the standard technique of counting link flows.

4. The computation time required to apply the proposed method to real cases is low. This means that it can be applied to large networks.

The main contributions of the FIFO Rule consistent dynamic network model (Chapter 7) are:

1. We deal with a continuous model for real traffic networks, which seems more convenient than a discretized version of the real continuous problem. Though time functions involve infinite degrees of freedom, we approximate these functions by monotone cubic splines. This implies that the number of parameters used to define the models remains finite and small.

2. We consider that the link travel time is a non-linear function of the link volumes, but we make some corrections in order to satisfy the FIFO rule. The idea consists of adding the minimum delay for such a condition to be satisfied (this is equivalent to considering a queue of vehicles due to congestion and its later dissipation), which is complemented with the use of monotone cubic interpolation that guarantees the satisfaction of the FIFO rule at the interpolated points.

3. 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. Moreover, we consider the effect of a link congestion on the upstream route links.

4. We use an iterative scheme. At a given iteration, we first determine the link travel times associated with a carefully selected 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.

5. We propose a method that has moderate cpu requirements.

The main contributions of the Generalized Beta-Gaussian Bayesian network model (Chapter 8) are:

1. The Generalized Beta-Gaussian Bayesian network, whose marginals are generalized Beta variables, that is, variables with range $[b_i, c_i]$ instead of the $[0, 1]$ range of standard Beta variables, allows us to define the marginal distribution of each individual variable independently, according to its characteristics.

2. The covariance structure of the flow variables is modeled by means of a multivariate normal distribution. The large number of parameters of the proposed model are very easy to estimate, using the maximum likelihood as estimation method.

3. The selection of relevant variables to make a given prediction is done with the help of correlation coefficients among variables.

4. The model is spatio-temporal, that is, they include times and locations.

5. An exhaustive comparison with other existing Bayesian models are given.

6. The proposed model requires small cpu times to make predictions so that, the method can be applied to large networks.

Chapter 3

Static traffic models

Contents

3.1 Notation

β_a	parameter of the travel cost function associated with link a.
δ_{pqr}	parameter of the Gumbel distribution associated with route r with origin-destination p-q.
η_{pq}	dual variable associated with the origin-destination p-q.
γ	parameter of the travel cost function.
γ_e	Euler constant.
γ_{ijpq}	parameters of calibration of the maximum entropy problem.
λ_a	dual variable. Travel cost associated with link a.
μ_{pqr}	dual variable. Over-cost associated with route r with origin-destination p-q.
π_{pq}	equilibrium travel cost to travel from origin p to destination q.

ρ_{iq}	minimum travel costs to go from node i to node q.
θ_{pq}	parameter of the Gumbel distribution associated with origin-destination p-q.
ξ_{pqr}	random error term associated with the route r with origin-destination p-q.
a	link.
a_{ij}	link between nodes i and j.
c_a	travel cost function associated with link a.
c_{a0}	free-flow conditions cost associated with link a.
C_a	integral of the travel cost function associated with link a.
C_{ij}	integral of travel cost function associated with link a_{ij}.
c_{ij}	travel cost function associated with link a_{ij}.
C_{pqr}	perceived travel cost associated with route r with origin-destination p-q.
c_{pqr}	travel cost function associated with route r with origin-destination p-q.
d_{pq}	demand for the flows from origin p to destination q.
$cost_{ij}$	minimum travel cost between any pair of nodes (i, j).
$G(\cdot)$	survival function of a cumulative distribution function.
h_{pqr}	flow on route r with origin-destination p-q.
P_{pqr}	share of drivers choosing a route r with origin-destination p-q.
p_{pqr}	probability of a user to select path r of OD-pair p-q.
q_a	constant measuring the flow producing a given congestion level in the link a.
r	route.
R_{pq}	set of routes with origin-destination p-q.
$\hat{v}$	observed flow.
v^*	link flows resulting from an assignment problem.
v_a	flow on link a.
v_{ij}	total flow through link a_{ij}.
v_{ijpq}	flow through link a_{ij} going from origin node p to destination node q.
v_{ijq}	flow on link a_{ij} with destination q.
w_{ijp}	flow coming from a given origin node p and using link a_{ij}.
z_{ipq}	flow going from origin node p to destination node q and passing through node i.
$\mathcal{A}$	set of links of a traffic network.
$\mathcal{D}$	set of origin-destination pairs of nodes.
$\mathcal{N}$	set of nodes of a traffic network.

3.2 Introduction

Some models try to reproduce the traffic during a certain period assuming that the existing conditions are stationary or that only mean values are of interest. They are called static traffic models. However, other models assume that the traffic conditions change with time. They are the so called dynamic models. We devote this chapter to the static traffic models, more exactly, to the static traffic assignment (STA).

The problem of traffic assignment consists in describing how users of different origin-destinations (OD) select their routes according to the traffic conditions. This is very important because the traffic flow governs the network performance in terms of travel time. In this problem the origin-destination traffic flows (demands) are assumed as data, and the users traveling the different routes are the unknowns. Thus, it is crucial to know how the users behave, i.e., how they select the adequate paths to go from origins to destinations. In this chapter we assume that the problem is static and that there is a constant-over-time travel demand.

The chapter is structured as follows. In Section 3.3, we introduce the User Equilibrium (UE), or the Wardrop's first principle and we analyze several formulations to reach this equilibrium according to the route knowledge. In Section 3.4, the System optimization (SO), or the Wardrop's second principle, is presented. In Section 3.5, we show the Stochastic User Equilibrium (SUE) and two kinds of stochastic route choice models for STA problems, the probit and the logit models. Finally, in the Appendix, we provide a method to obtain the minimum travel times between any pair of nodes and all actually used routes. Moreover, we explain how the previous method can be incorporated to obtain disaggregated flows by ODs and to perform route enumeration.

3.3 User Equilibrium (UE). Wardrop's first principle

Definition 1 (User equilibrium (UE)) *A user equilibrium is reached when no vehicle can improve their travel time by unilaterally changing routes, and it is assumed that all the drivers have a perfect knowledge of the network and, hence, of the travel times.*

Wardrop (1952) adopted the supply-demand equilibrium concept of economics, which suggests that travel demand should be balanced against the performance of the transport system in servicing that level of demand. This gives Wardrop's first principle, or the user equilibrium principle:

> *"Journey times in all routes actually used are equal, and less than*
> *those which would be experienced by a single vehicle on any unused route".*

Mathematically, this principle can be expressed by the following variational inequality:

$$h_{pqr}(c_{pqr} - \pi_{pq}) = 0, \qquad \forall r \in R_{pq}, \tag{3.1}$$

$$c_{pqr} - \pi_{pq} \geq 0, \qquad \forall r \in R_{pq}, \tag{3.2}$$

where h_{pqr} is the flow on route r with origin-destination p-q, c_{pqr}, the travel cost function associated with route r with origin-destination (p,q), π_{pq}, the equilibrium travel cost to travel from origin p to destination q and R_{pq}, the set of routes with origin-destination p-q.

3.3.1 Traffic model with route knowledge

Let us suppose a connected traffic network with set of nodes $\mathcal{N}$ and set of links $\mathcal{A}$. For certain origin-destination pairs of nodes, $p, q \in \mathcal{D}$, where $\mathcal{D}$ is a subset of $\mathcal{N}$ x $\mathcal{N}$, there are given positive demands d_{pq} for origin-destination flows which give rise to a link flow pattern $(v_a)_{a \in A}$ when distributed through the network. Further, assume that for each link a there is a positive and strictly increasing travel cost function $c_a(v_a)$.

Beckmann et al. (1956) transformed the user equilibrium principle into the following mathematical programming problem

$$\underset{\mathbf{h},\mathbf{v}}{\text{Minimize}}\, Z(\mathbf{v}) = \sum_{a \in A} \int_0^{v_a} c_a(s)ds = \sum_{a \in A} C_a(v_a), \tag{3.3}$$

subject to:

$$\sum_{r \in R_{pq}} h_{pqr} = d_{pq} : \ \eta_{pq}, \qquad \forall (p,q) \in D \tag{3.4}$$

$$\sum_{(p,q) \in D} \sum_{r \in R_{p,q}} \delta_{apqr} h_{pqr} = v_a : \ \lambda_a, \qquad \forall a \in A \tag{3.5}$$

$$h_{pqr} \geq 0 : \ \mu_{pqr}, \qquad \forall r \in R_{pq}, \forall (p,q) \in D \tag{3.6}$$

$$\tag{3.7}$$

with

$$\delta_{apqr} = \begin{cases} 1, & \text{if route } r \text{ from node } p \text{ to node } q \text{ contains arc } a; \\ 0, & \text{otherwise,} \end{cases} \tag{3.8}$$

where $C_a(\cdot)$ is the integral of the travel cost function associated with link a, and η_{pq}, λ_a and μ_{pqr} are the dual variables[1], and it has been assumed that the travel cost on a link depends only on the flow on that link.

Theorem 1 (Variational relation associated with Wardrop's first principle.) *Equations (3.3)-(3.8) imply Expressions (3.1)-(3.2)*

Proof.

This can be proved by using the Karush-Kuhn-Tucker (KKT) conditions (see Sheffi (1985))[2].

[1] Whatever value of the objective function $\theta(\lambda, \mu)$ of the dual problem is a lower bound of the primal objective function $f(h)$.

[2] These conditions must be satisfied by any stationary solution of the primal problem. Depending on the convexity of the problem, we can have local or global optima (Castillo et al. (2001)).

The Lagrangian function of this problem is

$$\mathcal{L}(\mathbf{h},\mathbf{v};\boldsymbol{\lambda},\boldsymbol{\mu}) = \sum_{a\in\mathcal{A}} C_a(v_a) + \sum_{(p,q)\in D} \eta_{pq}\left(d_{pq} - \sum_{r\in R_{pq}} h_{pqr}\right) + \tag{3.9}$$

$$+\sum_{a\in\mathcal{A}}\lambda_a\left(\sum_{(p,q)\in D}\sum_{r\in R_{p,q}}\delta_{apqr}h_{pqr} - v_a\right) - \sum_{(p,q)\in D, r\in R_{pq}}\mu_{pqr}h_{pqr}$$

and the Karush-Kuhn-Tucker conditions are:

$$\begin{aligned}
\frac{\partial\mathcal{L}}{\partial v_a} = c_a(v_a) - \lambda_a &= 0; \quad \forall a\in\mathcal{A} && (3.10)\\
\frac{\partial\mathcal{L}}{\partial h_{pqr}} = -\eta_{pq} + \sum_a \lambda_a\delta_{apqr} - \mu_{pqr} &= 0; \quad \forall(p,q)\in\mathcal{D}, \forall r\in\mathcal{R}_{pq} && (3.11)\\
\sum_r h_{pqr} &= d_{pq}; \quad \forall(p,q)\in\mathcal{D} && (3.12)\\
\sum_{(p,q)}\sum_r \delta_{apqr}h_{pqr} &= v_a; \quad \forall a\in A && (3.13)\\
h_{pqr} &\geq 0; \quad \forall(p,q)\in\mathcal{D}, \forall r\in\mathcal{R}_{pq} && (3.14)\\
h_{pqr}\mu_{pqr} &= 0; \quad \forall(p,q)\in\mathcal{D}, \forall r\in\mathcal{R}_{pq} && (3.15)\\
\mu_{pqr} &\geq 0; \quad \forall(p,q)\in\mathcal{D}, \forall r\in\mathcal{R}_{pq}. && (3.16)
\end{aligned}$$

From (3.10) we get

$$c_a(v_a) = \lambda_a, \tag{3.17}$$

that is, the dual variable λ_a is the link cost.

From Equation (3.11) and considering (3.16) we have:

$$\begin{aligned}
\mu_{pqr} &= \sum_a \lambda_a\delta_{apqr} - \eta_{pq}\\
&= c_{pqr} - \eta_{pq} \geq 0; \quad \forall(p,q)\in\mathcal{D}, \forall r\in\mathcal{R}_{pq},
\end{aligned} \tag{3.18}$$

that implies

$$c_{pqr} \geq \eta_{pq}, \quad \forall(p,q)\in\mathcal{D}, \forall r\in\mathcal{R}_{pq}. \tag{3.19}$$

In addition, due to (3.15) if $h_{pqr} > 0$ then $\mu_{pqr} = 0$ that implies $c_{pqr} = \eta_{pq}$. In other words, $\eta_{pq} \equiv \pi_{pq}$ is the equilibrium travel cost to travel from origin p to destination q. This concludes the proof. ■

From Equation (3.18) we can conclude that the dual variable μ_{pqr} can be interpreted as the over-cost (larger than the equilibrium one) associated with route r with origin-destination (p, q).

This proves that Equations (3.1) and (3.2) can be written in the following variational form:

$$\begin{aligned} h_{pqr}\left(\sum_a c_a(v_a)\delta_{apqr} - \eta_{pq}\right) &= 0; \quad \forall (p,q) \in \mathcal{D}, \forall r \in \mathcal{R}_{pq} && (3.20)\\ \sum_a c_a(v_a)\delta_{apqr} - \eta_{pq} &\geq 0; \quad \forall (p,q) \in \mathcal{D}, \forall r \in \mathcal{R}_{pq}. && (3.21) \end{aligned}$$

Another important issue is the uniqueness of the solution of Problem (3.3)–(3.8), established by the following theorem. First we introduce the notion of Hessian matrices.

Definition 2 *The Hessian matrix $H(f) =$ of a function $f(x_1, x_2, \ldots, x_n)$ is the square matrix of its second-order partial derivatives, that is,*

$$H(f) = \begin{bmatrix} \dfrac{\partial^2 f}{\partial x_1^2} & \dfrac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \dfrac{\partial^2 f}{\partial x_1 \partial x_n} \\ \dfrac{\partial^2 f}{\partial x_2 \partial x_1} & \dfrac{\partial^2 f}{\partial x_2^2} & \cdots & \dfrac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \dfrac{\partial^2 f}{\partial x_n \partial x_1} & \dfrac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \dfrac{\partial^2 f}{\partial x_n^2} \end{bmatrix}. \qquad (3.22)$$

If the Hessian is positive definite at a non-degenerate point x^*, then f attains its local minimum at x. If the Hessian is negative definite at x^*, then f attains a local maximum at x^*. If the Hessian has both positive and negative eigenvalues then x^* is a saddle point for f (this is true even if x^* is degenerate). Otherwise the test is inconclusive.

Theorem 2 (Uniqueness.) *If the cost functions $c_a(v_a); a \in A$ are monotonically increasing with respect to link flows v_a, then the Problem (3.3)–(3.8) has unique solution in terms of link flows v_a.*

Proof. To prove the uniqueness of the user equilibrium solution we can demonstrate that the feasible region and the objective function in the Beckmann formulation are both convex. Given the linearity of the constraints, the resultant feasible region of the optimization problem is obviously convex.

We demonstrate the convexity of the objective function (3.3) with respect to the link flows v_a by showing that the Hessian is positive definite.

Since $\dfrac{\partial Z(\mathbf{v})}{\partial v_a} = c_a(v_a)$, $\dfrac{\partial^2 Z(\mathbf{v})}{\partial v_a^2} = c_a'(v_a)$ and $\dfrac{\partial^2 Z(\mathbf{v})}{\partial v_a v_b} = 0$; $\forall b \neq a$, the Hessian becomes

$$\nabla^2 z(v_a) = \begin{bmatrix} c_a'(v_a) & 0 & 0 & \cdots \\ 0 & c_b'(v_b) & 0 & \cdots \\ 0 & 0 & \ddots & \vdots \\ \vdots & \vdots & \cdots & c_A'(v_A) \end{bmatrix}, \qquad (3.23)$$

and since c_a has been assumed monotonically increasing with respect to link flow v_a, this matrix is positive definite, and therefore, the equilibrium solution with respect to link flows is unique. ∎

On the contrary, we can have multiple solutions with respect to path flows h_{pqr}.

3.3.2 Traffic model without route knowledge

The Ferris, Meeraus and Rutherford model

The Ferris, Meeraus and Rutherford model Ferris et al. (1999) provides an optimization problem that not only gives the link flows but it identifies their corresponding destination nodes directly without requiring neither route enumeration nor route choice probabilities. In addition, they show how the optimization approaches to the traffic equilibrium problem can be implemented using standard modelling software. The problem they propose is the Beckmann model in terms of some disaggregated variables, which gives the link flows v_{ijq} disaggregated by destinations without path enumeration:

$$\underset{\mathbf{v}}{\text{Minimize}}\, Q = \sum_{a_{ij}\in\mathcal{A}} C_{ij}\left(\sum_q v_{ijq}\right) \tag{3.24}$$

subject to

$$\sum_{a_{ij}\in\mathcal{A}} v_{ijq} - \sum_{a_{ji}\in\mathcal{A}} v_{jiq} \;=\; d_{iq} \;:\; \lambda_{iq}; \quad i\in\mathcal{N}, q\in\mathcal{D}; \quad i\neq q, \tag{3.25}$$

$$v_{ijq} \;\geq\; 0 \;:\; \mu_{ijq}; \quad a_{ij}\in\mathcal{A}, q\in\mathcal{D}, \tag{3.26}$$

where v_{ijq} is the flow through link a_{ij} with destination node q, $C_{ij}(\cdot)$ is the integral of the travel cost function associated with link a_{ij}, d_{iq} are the OD flows associated with origin and destination nodes i and q, respectively, λ_{iq} and μ_{ijq} are the dual variables, and it has been assumed that the travel cost on a link depends only on the flow on that link.

We note that the balance equation for $i = q$ does not need to be written because it is a linear combination of the balance equations for the remaining nodes.

Finally, we indicate that the sensitivity of Q with respect to d_{iq} is λ_{iq}.

The role of dual variables. The Lagrangian function of this problem is

$$\mathcal{L}(\mathbf{v};\boldsymbol{\lambda},\boldsymbol{\mu}) = \sum_{a_{ij}\in\mathcal{A}} C_{ij}\left(\sum_q v_{ijq}\right) + \sum_{i\neq q}\lambda_{iq}\left(d_{iq} - \sum_{a_{ij}\in\mathcal{A}} v_{ijq} + \sum_{a_{ji}\in\mathcal{A}} v_{jiq}\right) - \sum_{a_{ij}\in\mathcal{A},\, q\in\mathcal{D}} \mu_{ijq}v_{ijq} \tag{3.27}$$

and the corresponding Karush-Kuhn-Tucker (KKT) conditions become

$$\frac{\partial\mathcal{L}}{\partial v_{ijq}} = c_{ij}\left(\sum_q v_{ijq}\right) - (1-\delta_{iq})\lambda_{iq} + (1-\delta_{jq})\lambda_{jq} - \mu_{ijq} \;=\; 0; \quad a_{ij}\in\mathcal{A}, q\in\mathcal{D}, \tag{3.28}$$

$$\begin{aligned}
\sum_{a_{ij}\in\mathcal{A}} v_{ijq} - \sum_{a_{ji}\in\mathcal{A}} v_{jiq} &= d_{iq}; \quad i \in \mathcal{N}, q \in \mathcal{D}; \quad i \neq q && (3.29)\\
v_{ijq} &\geq 0; \quad a_{ij} \in \mathcal{A}, q \in \mathcal{D} && (3.30)\\
v_{ijq}\mu_{ijq} &= 0; \quad a_{ij} \in \mathcal{A}, q \in \mathcal{D} && (3.31)\\
\mu_{ijq} &\geq 0; \quad a_{ij} \in \mathcal{A}, q \in \mathcal{D}, && (3.32)
\end{aligned}$$

where $c_{ij}(\cdot)$ is the derivative of $C_{ij}(\cdot)$, and δ_{iq} is the Dirac Delta whose value is 1 when $i = q$ and zero otherwise.

From the system (3.28)-(3.32) we obtain

$$\begin{aligned}
c_{ij}\left(\sum_q v_{ijq}\right) + \rho_{jq} &\geq \rho_{iq}; \quad a_{ij} \in \mathcal{A}, q \in \mathcal{D}; \quad \perp \quad v_{ijq} \geq 0, && (3.33)\\
\sum_{a_{ij}\in\mathcal{A}} v_{ijq} - \sum_{a_{ij}\in\mathcal{A}} v_{jiq} &= d_{iq}; \quad i \in \mathcal{N}, q \in \mathcal{D}, && (3.34)
\end{aligned}$$

where $\rho_{iq} = (1 - \delta_{iq})\lambda_{iq}$ and $\perp$ represents complementary slackness, i.e., both conditions must be satisfied but at least one with equality, λ_{iq} represents the minimum travel cost to travel from node i to node q.

Note that (3.33) shows that the variables ρ_{iq} are the minimum costs to go from node i to node q.

Equation (3.28) also shows that the dual variable μ_{ijq} gives the differences between the travel cost $c_{ij}\left(\sum\limits_q v_{ijq}\right)$ associated with link a_{ij} and the travel cost associated with that link if we travel from node i to node j through the least expensive path. If these two costs are identical, this means that the less expensive path from i to j passes through link a_{ij}, and then $\mu_{ijq} = 0$. Otherwise, $\mu_{ijq} < 0$ (note that constraint (3.30) is a $\geq$ constraint).

We end this section by saying that a traffic model analogous to this one can be given in which instead of the destination nodes, the origin nodes are identified. So, a similar problem to (3.24)-(3.26) can be formulated if we disaggregate link flows by origins.

The Castillo et al. model

In this section we propose a model for disaggregating link flows by OD-pair flows by origin and destination, i.e., to identify the origins and destinations of the link flows directly, i.e., without the need of path enumeration, route choice probabilities or postprocessing calculations.

Let v_{ijpq} be the flow through link a_{ij} going from origin node p to destination node q. Given the origin-destination (OD) traffic flows, one can estimate the link flows associated with the different OD-pairs, using the following optimization problem, which corresponds to the Wardrop equilibrium problem

$$\underset{\mathbf{v}}{\text{Minimize}}\, Z = \sum_{a_{ij}\in A} C_{ij}\left(\sum_{p,q} v_{ijpq}\right) \tag{3.35}$$

subject to

$$\begin{aligned} d_{pq}\delta_{ip} &= \sum_{a_{ij}\in\mathcal{A}} v_{ijpq} - \sum_{a_{ji}\in\mathcal{A}} v_{jipq} \;:\; \lambda_{ipq}; \quad (p,q)\in OD; \quad i\neq q, & (3.36)\\ v_{ijpq} &\geq 0 \;:\; \mu_{ijpq}; \quad a_{ij}\in\mathcal{A}; \quad (p,q)\in OD, & (3.37)\end{aligned}$$

where v_{ijpq} is the flow through link (i,j) with origin node p and destination node q, λ_{ipq} are the dual variables associated with the corresponding constraints and δ_{ip} is the Dirac Delta whose value is 1 when $i = p$ and zero otherwise. We have also assumed that the travel cost on a link depends only on the flow on that link.

The problem (3.35)-(3.37) is a statement of the Beckmann et al. formulation of the Wardrop equilibrium problem, but stated for each OD pair. As the travel cost function one can select the Bureau of Public Roads (BPR) type cost functions, because it is generally accepted and has nice regularity properties (see Cantarella (1997)), but other alternative travel cost functions with the same regularity properties (increasing with flow, monotonic and continuously differentiable) can be used instead.

Note that equation (3.36) represent the flow balance associated with the OD-pair (p,q), for all nodes.

Note that the sensitivities of the objective function to changes in the $d_{pq}\delta_{ip}$ values are the dual variable values λ_{ipq}, because they appear as the independent term of Equation (3.36) (see Castillo et al. (2006, 2005)).

Once the values of the OD link flows v_{ijpq} have been estimated using the optimization problem (3.35)-(3.37), one can easily calculate important flow information. Thus, the statement of the flow problem using the set of variables v_{ijpq} has the following important advantages:

1. It avoids path enumeration.

2. We can easily calculate the flows w_{ijp} coming from a given origin node p and using link a_{ij}:

$$w_{ijp} = \sum_{q} v_{ijpq}. \tag{3.38}$$

3. We can easily calculate the flows v_{ijq} going to a given destination node q and using link a_{ij}:

$$v_{ijq} = \sum_{p} v_{ijpq}. \tag{3.39}$$

4. We can easily calculate the total flow v_{ij} through link a_{ij}, as follows:

$$v_{ij} = \sum_{p,q} v_{ijpq}. \tag{3.40}$$

5. We can calculate the flow z_{ipq} going from origin node p to destination node q and passing through node i as

$$z_{ipq} = \sum_{i:a_{ij}\in\mathcal{A}} v_{ijpq}. \tag{3.41}$$

6. We can identify and/or enumerate the flow paths very easily. To enumerate paths of an OD-pair (p, q), we can simply build the tree with accessible (with non-null flow v_{ijpq} in some solution) branches (links) starting from the origin node p and ending with the destination node q (see Figures 3.2 and 3.3, 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).

7. Equations (3.38) to (3.41) and the selected variables v_{ijpq} for the optimization problem (3.35)-(3.37) allow us incorporating new estimation techniques, based on information about $w_{ijp}, v_{ijq}, z_{ipq}$ and/or v_{ijpq}, which are not possible for other methods. This has important practical implications.

   If the model (3.35)-(3.37) is to be used for estimation purposes of link or OD-pair flows, one must add one or two of the following constraints:

$$\sum_{(p,q)\in\mathcal{OD}} v_{ijpq} = \hat{v}_{ij}; \quad \forall a_{ij}\in\mathcal{B}\subset\mathcal{A}, \tag{3.42}$$

$$v_{ijpq} = \hat{v}_{ijpq}; \quad \forall a_{ij}\in\mathcal{S}\subset\mathcal{A}, \tag{3.43}$$

   where $\hat{v}_{ijpq}$ and $\hat{v}_{ij}$ are observations of the selected observation sets $\mathcal{S}$ and $\mathcal{B}$, respectively, that is, we assume that we observe the flows of the links in the set $\mathcal{B}$, or we observe the flows of the links in set $\mathcal{S}$ and ask the users their origin and destinations (a more informative observation).

   In particular, the OD-pair flow estimates can benefit from using information of the type in (3.43), when compared with estimates based only on link flows as in (3.42), the standard technique.

8. The problem can be solved as a standard non-linear programming problem using standard software packages as GAMS, for example. For the illustrative example at the end of the section, we have used GAMS and the results were very precise.

We note that though the problem (3.35)-(3.37) has unique solution in terms of total link flows v^*_{ij}, it can have infinitely many solutions in terms of v_{ijpq}[3], though they are equivalent in terms of link costs (they have the same link costs). Thus, this fact does not affect to the link costs but has practical implications in terms of the distributions of link flows among the OD-pairs. That is why the v_{ijpq} observations are needed.

We end by pointing out that in SUE (Stochastic User Equilibrium), introduced in Section 3.5, this problem is resolved and those path flows are unique, because of the behavioral mechanism used in stochastic loading.

To calibrate the resulting link flows we can play with the travel cost function $C_{ij}(v^*_{ij})$. In fact, use of the model in real cases must be done after a good calibration of the model parameters.

Looking for unique flows. As we have indicated, the v_{ijpq} solution of the problem (3.35)-(3.37) are not necessarily unique. In Castillo et al. (2008e), the uniqueness problem is solved by a minimum variance approach.

Another way of solving the uniqueness problem consists in using the maximum entropy principle to select one solution among all optimal solutions of the problem (3.35)-(3.37), that is, solving the problem:

$$\underset{\mathbf{v}}{\text{Minimize}}\, Z = \sum_{a_{ij}\in A} \sum_{p,q} \gamma_{ijpq} v_{ijpq} \log\left(v_{ijpq}\right) \tag{3.44}$$

where γ_{ijpq} are parameters, subject to

$$\begin{aligned} d_{pq}\delta_{ip} &= \sum_{a_{ij}\in\mathcal{A}} v_{ijpq} - \sum_{a_{ji}\in\mathcal{A}} v_{jipq} \;:\; \lambda_{ipq}; \quad (p,q)\in OD; \quad i\neq q, && (3.45)\\ \sum_{(p,q)\in OD} v_{ijpq} &= v^*_{ij};\; a_{ij}\in A, && (3.46)\\ v_{ijpq} &\geq 0 \;:\; \mu_{ijpq}; \quad \forall a_{ij}\in A; \quad (p,q)\in OD. && (3.47) \end{aligned}$$

where v^*_{ij} are the link flows resulting from the problem (3.35)-(3.37).

Note that the set of feasible solutions of the problem (3.35)-(3.37) is the solution of the linear system (3.45)-(3.47), which is a polytope (see Castillo et al. (2000, 2002) for methods to identify this polytope).

We note that the behavior of the users coming from different ODs at different links a_{ij} can be calibrated by means of an adequate selection of the γ_{ijpq} parameters.

[3] For example, in Figure 3.2 the costs of flows through nodes 6-7-11 and 6-10-11 are identical, because of the Wardrop equilibrium, and then any exchange of users between both sub-paths does not alter the link flows nor the corresponding costs. So, given an optimal solution to the problem, exchanging different OD users from one sub-path to the other leads to another optimal solution with different v_{ijpq} values, though the same link flows v_{ij}.

The dual variables. Since it is important for practical purposes, in this section we give the physical meaning of the dual variables of the optimization problem (3.35)-(3.37), which can be obtained as follows.

The Lagrangian function of this problem is

$$\begin{aligned}\mathcal{L}(\mathbf{v};\boldsymbol{\lambda}_1,\boldsymbol{\lambda}_2,\boldsymbol{\lambda}_3,\boldsymbol{\mu}) &= \sum_{a_{ij}\in\mathcal{A}} C\left(\sum_{p,q} v_{ijpq}\right) - \sum_{i,j,p,q} \mu_{ijpq}v_{ijpq} \\ &\quad + \sum_{i,p\neq q}\lambda_{ipq}\left(\sum_{a_{ij}\in\mathcal{A}} v_{ijpq} - \sum_{a_{ji}\in\mathcal{A}} v_{jipq} - d_{pq}\delta_{ip}\right) \\ &= \sum_{a_{ij}\in\mathcal{A}} C\left(\sum_{p,q} v_{ijpq}\right) - \sum_{i,j,p,q} \mu_{ijpq}v_{ijpq} - \sum_{j,p\neq q}\lambda_{jpq}\left(\sum_{a_{ij}\in\mathcal{A}} v_{ijpq}\right) \\ &\quad + \sum_{i,p\neq q}\lambda_{ipq}\left(\sum_{a_{ij}\in\mathcal{A}} v_{ijpq} - d_{pq}\delta_{ip}\right). \end{aligned} \tag{3.48}$$

and the KKT conditions:

$$\begin{aligned}\frac{\partial\mathcal{L}}{\partial v_{ijpq}} &= c\left(\sum_{(p,q)\in OD} v_{ijpq}\right) + (1-\delta_{iq})\lambda_{ipq} - (1-\delta_{jq})\lambda_{jpq} - \mu_{ijpq} = 0; \\ &\qquad a_{ij}\in\mathcal{A};\ (p,q)\in OD, \qquad (3.49) \\ d_{pq}\delta_{ip} &= \sum_{a_{ij}\in\mathcal{A}} v_{ijpq} - \sum_{a_{ji}\in\mathcal{A}} v_{jipq}\ :\ \lambda_{ipq};\quad (p,q)\in OD;\ i\neq q \qquad (3.50) \\ v_{ijpq} &\geq 0;\quad a_{ij}\in\mathcal{A};\ (p,q)\in OD, \qquad (3.51) \\ \mu_{ijpq}v_{ijpq} &= 0;\quad a_{ij}\in\mathcal{A};\ (p,q)\in OD, \qquad (3.52) \\ \mu_{ijpq} &\geq 0;\quad a_{ij}\in\mathcal{A};\ (p,q)\in OD, \qquad (3.53)\end{aligned}$$

where $c(\cdot)$ is the first derivative of $C(\cdot)$ with respect to its argument and δ_{iq} is the Dirac Delta whose value is 1 when $i=q$ and zero otherwise, that leads to

$$c\left(\sum_{(p,q)\in OD} v_{ijpq}\right) \geq \rho_{jpq} - \rho_{ipq};\quad a_{ij}\in\mathcal{A};\ (p,q)\in OD;\quad i,j\neq q \perp \quad v_{ijpq}\geq 0, \tag{3.54}$$

where $\rho_{ipq} = \lambda_{ipq}(1-\delta_{iq})$ and this permits obtaining the minimum travel cost $cost_{ij}$ from a node i to a node j based on the dual variables, when the corresponding path exists, that is,

$$cost_{ij} = \rho_{jpq} - \rho_{ipq};\quad a_{ij}\in\mathcal{A}, \tag{3.55}$$

and this value must be independent of $(p,q)\in OD$. This shows that path costs depend only on the origin and destination nodes involved.

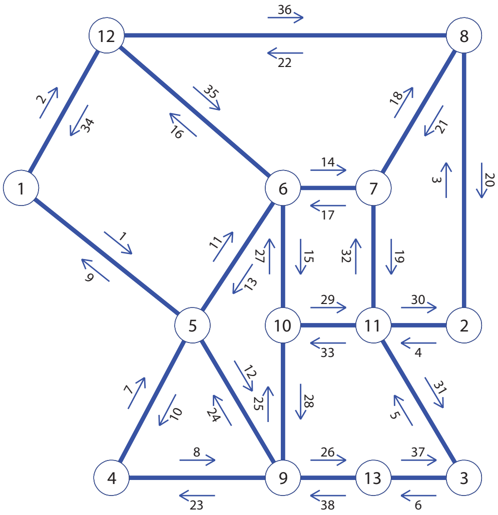

Figure 3.1: The Nguyen-Dupuis network, showing the nodes and links.

Example 1 (The Nguyen-Dupuis network) *With the purpose of illustrating the proposed concepts and methods, they are applied to the Nguyen-Dupuis network. We use an example of bidirectional flow. 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 is connected in both directions by links a_{ij} and a_{ji} (see Figure 3.1). In this example, we use the BPR (Bureau of Public Roads) cost function, i.e.,*

$$c_a(v_a) = c_{a0}\left[1 + \beta_a \left(\frac{v_a}{q_a}\right)^{\gamma}\right],$$

where for a given link a, v_a refers to its flow, c_{a0} is the travel cost associated with free flow conditions, q_a is a constant measuring the flow producing congestion, and β_a and γ are constants defining how the cost increases with traffic flow.

The integral of this cost function is the following:

$$C_a(v_a) = v_a c_{a0}\left[1 + \beta_a \left(\frac{v_a}{q_a}\right)^{\gamma} \frac{1}{\gamma + 1}\right],$$

The network data used in this example are $\beta_a = 1$, $\gamma = 4$ and the c_{a0} and q_a travel cost constants used for every link a are shown in Table 6.3.

In this section we illustrate the traffic assignment problem, i.e., the OD-pair flows are given and the flows v_{ijpqr} are determined. We consider the following OD-pair non-symmetric flows:

$$\begin{array}{llll} d_{12} = 320 & d_{13} = 640 & d_{42} = 480 & d_{43} = 160 \\ d_{21} = 500 & d_{31} = 640 & d_{24} = 480 & d_{34} = 300 \end{array}$$

Link	c_{a0}	q_a
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
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

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

To solve the traffic assignment problem with disaggregation of link flows, we go through the following steps:

Step 1: Obtaining link flows and travel costs. *Initially, the problem (3.35)-(3.37) is solved, and then the problem (3.44)-(3.47) to get uniqueness. Tables 3.2 and 3.3 show the resulting link costs and total link flows, classified by origins and destinations.*

Step 2: Obtaining the OD cost matrix. *Following the method presented in Appendix 3.5.2 and using these link costs, we obtain the minimum travel cost problem (3.93)-(3.98), resulting the minimum costs matrix shown in Table 3.4. Note that there are no empty values, showing that there are no unreachable nodes from a given one, and no null values apart from those in the main diagonal.*

Step 3: Enumerating paths. *All minimum cost paths for each OD-pairs are found, using the algorithm 2 given in Appendix 3.5.2. Table 3.5 shows all the paths utilized by the users for different OD-pairs and their associated travel costs, which interpretation will be given later. To clarify, we explain the meaning of the information contained in Table 3.5 corresponding to the OD-pair* $(1, 3)$*. It contains* 6 *rows with the six corresponding paths and travel costs. The paths are all the minimum cost paths for this OD-pair (the only paths considered). The travel costs are* 53.7 *and are all identical, as it corresponds to Wardrop's equilibrium.*

Figures 3.2 and 3.3 show the sets of links $\mathcal{A}_{OD}$ *used by the eight OD-pair users. They contain all minimum path costs obtained by algorithm 2. It is interesting to see that apart from the OD-pair* $(1, 2)$*, which has a single path, all OD-pairs have several possible options for minimum cost paths. This is just a consequence of Wardrop equilibrium.*

			Origin-Destination							
Link	Cost	Total flow	1- 2	1- 3	2- 1	2- 4	3- 1	3- 4	4- 2	4- 3
1 -5	8.9	506.9	0.0	506.9	0.0	0.0	0.0	0.0	0.0	0.0
1 -12	12.9	453.1	320.0	133.1	0.0	0.0	0.0	0.0	0.0	0.0
2 -8	19.4	725.9	0.0	0.0	495.0	230.9	0.0	0.0	0.0	0.0
2 -11	15.1	254.1	0.0	0.0	5.0	249.1	0.0	0.0	0.0	0.0
3 -11	17.6	586.3	0.0	0.0	0.0	0.0	583.5	2.8	0.0	0.0
3 -13	12.8	353.7	0.0	0.0	0.0	0.0	56.5	297.2	0.0	0.0
4 -5	10.8	376.7	0.0	0.0	0.0	0.0	0.0	0.0	298.8	77.9
4 -9	21.4	263.3	0.0	0.0	0.0	0.0	0.0	0.0	181.2	82.1
5 -1	9.2	526.0	0.0	0.0	0.2	0.0	525.8	0.0	0.0	0.0
5 -4	14.1	485.6	0.0	0.0	0.0	480.0	0.0	5.6	0.0	0.0
5 -6	16.6	613.4	0.0	237.1	0.0	0.0	0.0	0.0	298.5	77.9
5 -9	10.5	270.2	0.0	269.8	0.0	0.0	0.0	0.0	0.4	0.0
6 -5	23.4	678.4	0.0	0.0	0.0	479.7	196.0	2.8	0.0	0.0
6 -7	6.3	498.3	0.0	199.8	0.0	0.0	0.0	0.0	298.5	0.0
6 -10	8.1	248.2	0.0	170.3	0.0	0.0	0.0	0.0	0.0	77.9
6 -12	10.6	119.0	0.0	0.0	4.7	0.0	114.2	0.0	0.0	0.0
7 -6	6.8	545.2	0.0	0.0	0.7	231.5	310.2	2.8	0.0	0.0
7 -8	5.2	298.5	0.0	0.0	0.0	0.0	0.0	0.0	298.5	0.0
7 -11	9.1	199.8	0.0	199.8	0.0	0.0	0.0	0.0	0.0	0.0
8 -2	14.5	618.5	320.0	0.0	0.0	0.0	0.0	0.0	298.5	0.0
8 -7	5.1	230.9	0.0	0.0	0.0	230.9	0.0	0.0	0.0	0.0
8 -12	22.5	495.0	0.0	0.0	495.0	0.0	0.0	0.0	0.0	0.0
9 -4	26.7	294.4	0.0	0.0	0.0	0.0	0.0	294.4	0.0	0.0
9 -5	12.6	333.2	0.0	0.0	0.2	0.3	329.8	2.9	0.0	0.0
9 -10	14.2	225.3	0.0	43.8	0.0	0.0	0.0	0.0	181.5	0.0
9 -13	22.2	308.2	0.0	226.0	0.0	0.0	0.0	0.0	0.0	82.1
10-6	8.3	252.2	0.0	0.0	4.1	248.1	0.0	0.0	0.0	0.0
10-9	19.1	273.8	0.0	0.0	0.2	0.3	273.3	0.0	0.0	0.0
10-11	7.3	473.5	0.0	214.2	0.0	0.0	0.0	0.0	181.5	77.9
11-2	10.6	181.5	0.0	0.0	0.0	0.0	0.0	0.0	181.5	0.0
11-3	12.8	491.8	0.0	414.0	0.0	0.0	0.0	0.0	0.0	77.9
11-7	9.4	314.3	0.0	0.0	0.7	0.7	310.2	2.8	0.0	0.0
11-10	7.9	526.1	0.0	0.0	4.3	248.5	273.3	0.0	0.0	0.0
12-1	22.0	614.0	0.0	0.0	499.8	0.0	114.2	0.0	0.0	0.0
12-6	12.7	133.1	0.0	133.1	0.0	0.0	0.0	0.0	0.0	0.0
12-8	15.5	320.0	320.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
13-3	12.0	308.2	0.0	226.0	0.0	0.0	0.0	0.0	0.0	82.1
13-9	31.9	353.7	0.0	0.0	0.0	0.0	56.5	297.2	0.0	0.0

Table 3.2: Link costs, link total flows, and link flows associated with different origin-destination nodes (INITIAL).

			Origin-Destination							
Link	Cost	Total flow	1- 2	1- 3	2- 1	2- 4	3- 1	3- 4	4- 2	4- 3
1 -5	8.9	506.9	0.0	506.9	0.0	0.0	0.0	0.0	0.0	0.0
1 -12	12.9	453.1	320.0	133.1	0.0	0.0	0.0	0.0	0.0	0.0
2 -8	19.4	725.9	0.0	0.0	495.2	230.8	0.0	0.0	0.0	0.0
2 -11	15.1	254.1	0.0	0.0	4.8	249.2	0.0	0.0	0.0	0.0
3 -11	17.6	586.3	0.0	0.0	0.0	0.0	406.7	179.5	0.0	0.0
3 -13	12.8	353.7	0.0	0.0	0.0	0.0	233.3	120.5	0.0	0.0
4 -5	10.8	376.7	0.0	0.0	0.0	0.0	0.0	0.0	312.9	63.8
4 -9	21.4	263.3	0.0	0.0	0.0	0.0	0.0	0.0	167.1	96.2
5 -1	9.2	526.0	0.0	0.0	5.0	0.0	521.0	0.0	0.0	0.0
5 -4	14.1	485.6	0.0	0.0	0.0	324.1	0.0	161.6	0.0	0.0
5 -6	16.6	613.4	0.0	244.5	0.0	0.0	0.0	0.0	310.9	58.0
5 -9	10.5	270.2	0.0	262.4	0.0	0.0	0.0	0.0	2.0	5.7
6 -5	23.4	678.4	0.0	0.0	3.7	304.3	244.2	126.3	0.0	0.0
6 -7	6.3	498.3	0.0	166.3	0.0	0.0	0.0	0.0	301.6	30.5
6 -10	8.1	248.2	0.0	211.3	0.0	0.0	0.0	0.0	9.3	27.6
6 -12	10.6	119.0	0.0	0.0	0.0	0.0	119.0	0.0	0.0	0.0
7 -6	6.8	545.2	0.0	0.0	2.2	266.4	199.5	77.0	0.0	0.0
7 -8	5.2	298.5	0.0	0.0	0.0	0.0	0.0	0.0	298.5	0.0
7 -11	9.1	199.8	0.0	166.3	0.0	0.0	0.0	0.0	3.1	30.5
8 -2	14.5	618.5	320.0	0.0	0.0	0.0	0.0	0.0	298.5	0.0
8 -7	5.1	230.9	0.0	0.0	0.1	230.8	0.0	0.0	0.0	0.0
8 -12	22.5	495.0	0.0	0.0	495.0	0.0	0.0	0.0	0.0	0.0
9 -4	26.7	294.4	0.0	0.0	0.0	155.9	0.0	138.4	0.0	0.0
9 -5	12.6	333.2	0.0	0.0	1.3	19.8	276.9	35.3	0.0	0.0
9 -10	14.2	225.3	0.0	23.3	0.0	0.0	0.0	0.0	169.1	33.0
9 -13	22.2	308.2	0.0	239.2	0.0	0.0	0.0	0.0	0.0	69.0
10-6	8.3	252.2	0.0	0.0	1.5	37.9	163.6	49.3	0.0	0.0
10-9	19.1	273.8	0.0	0.0	1.3	175.7	43.6	53.3	0.0	0.0
10-11	7.3	473.5	0.0	234.6	0.0	0.0	0.0	0.0	178.4	60.5
11-2	10.6	181.5	0.0	0.0	0.0	0.0	0.0	0.0	181.5	0.0
11-3	12.8	491.8	0.0	400.8	0.0	0.0	0.0	0.0	0.0	91.0
11-7	9.4	314.3	0.0	0.0	2.1	35.6	199.5	77.0	0.0	0.0
11-10	7.9	526.1	0.0	0.0	2.7	213.6	207.2	102.5	0.0	0.0
12-1	22.0	614.0	0.0	0.0	495.0	0.0	119.0	0.0	0.0	0.0
12-6	12.7	133.1	0.0	133.1	0.0	0.0	0.0	0.0	0.0	0.0
12-8	15.5	320.0	320.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
13-3	12.0	308.2	0.0	239.2	0.0	0.0	0.0	0.0	0.0	69.0
13-9	31.9	353.7	0.0	0.0	0.0	0.0	233.3	120.5	0.0	0.0

Table 3.3: Link costs, link total flows, and link flows associated with different origin-destination nodes.

Node	1	2	3	4	5	6	7	8	9	10	11	12	13
1	0.0	42.8	53.7	23.0	8.9	25.6	31.9	28.3	19.5	33.7	40.9	12.9	41.7
2	64.0	0.0	27.9	68.8	54.7	31.3	24.5	19.4	42.2	23.0	15.1	42.0	40.6
3	66.5	28.2	0.0	71.3	57.2	33.8	27.0	32.1	44.7	25.5	17.6	44.5	12.8
4	20.1	53.4	55.6	0.0	10.8	27.5	33.8	38.9	21.4	35.6	42.8	32.9	43.6
5	9.2	42.6	44.8	14.1	0.0	16.6	22.9	28.1	10.5	24.7	32.0	22.1	32.7
6	32.7	25.9	28.1	37.5	23.4	0.0	6.3	11.4	27.2	8.1	15.3	10.6	40.9
7	39.5	19.6	21.8	44.4	30.3	6.8	0.0	5.2	34.1	14.9	9.1	17.5	34.6
8	44.6	14.5	26.9	49.4	35.3	11.9	5.1	0.0	39.1	20.0	14.1	22.5	39.6
9	21.8	32.0	34.2	26.7	12.6	22.5	28.8	33.9	0.0	14.2	21.5	33.1	22.2
10	40.9	17.8	20.0	45.8	31.7	8.3	14.6	19.7	19.1	0.0	7.3	18.9	32.8
11	48.9	10.6	12.8	53.7	39.6	16.2	9.4	14.5	27.1	7.9	0.0	26.9	25.5
12	22.0	30.0	40.8	45.0	30.9	12.7	19.0	15.5	40.0	20.8	28.1	0.0	53.6
13	53.7	40.2	12.0	58.6	44.5	45.8	39.0	44.2	31.9	37.5	29.6	56.5	0.0

Table 3.4: Minimum travel cost between two given nodes.

Paths	Cost
OD-pair 1-2	
1 12 8 2	42.8
OD-pair 1-3	
1 5 6 7 11 3	53.7
1 5 6 10 11 3	53.7
1 5 9 10 11 3	53.7
1 5 9 13 3	53.7
1 12 6 7 11 3	53.7
1 12 6 10 11 3	53.7
OD-pair 2-1	
2 8 7 6 5 1	64.0
2 8 7 6 12 1	64.0
2 8 12 1	64.0
2 11 7 6 5 1	64.0
2 11 7 6 12 1	64.0
2 11 10 6 5 1	64.0
2 11 10 6 12 1	64.0
2 11 10 9 5 1	64.0
OD-pair 2-4	
2 8 7 6 5 4	68.8
2 11 7 6 5 4	68.8
2 11 10 6 5 4	68.8
2 11 10 9 4	68.8
2 11 10 9 5 4	68.8
OD-pair 3-1	
3 11 7 6 5 1	66.5
3 11 7 6 12 1	66.5
3 11 10 6 5 1	66.5
3 11 10 6 12 1	66.5
3 11 10 9 5 1	66.5
3 13 9 5 1	66.5
OD-pair 3-4	
3 11 7 6 5 4	71.3
3 11 10 6 5 4	71.3
3 11 10 9 4	71.3
3 11 10 9 5 4	71.3
3 13 9 4	71.3
3 13 9 5 4	71.3
OD-pair 4-2	
4 5 6 7 8 2	53.4
4 5 6 7 11 2	53.4
4 5 6 10 11 2	53.4
4 5 9 10 11 2	53.4
4 9 10 11 2	53.4
OD-pair 4-3	
4 5 6 7 11 3	55.6
4 5 6 10 11 3	55.6
4 5 9 10 11 3	55.6
4 5 9 13 3	55.6
4 9 10 11 3	55.6
4 9 13 3	55.6

Table 3.5: All paths utilized by the users for different OD-pairs and associated travel costs.

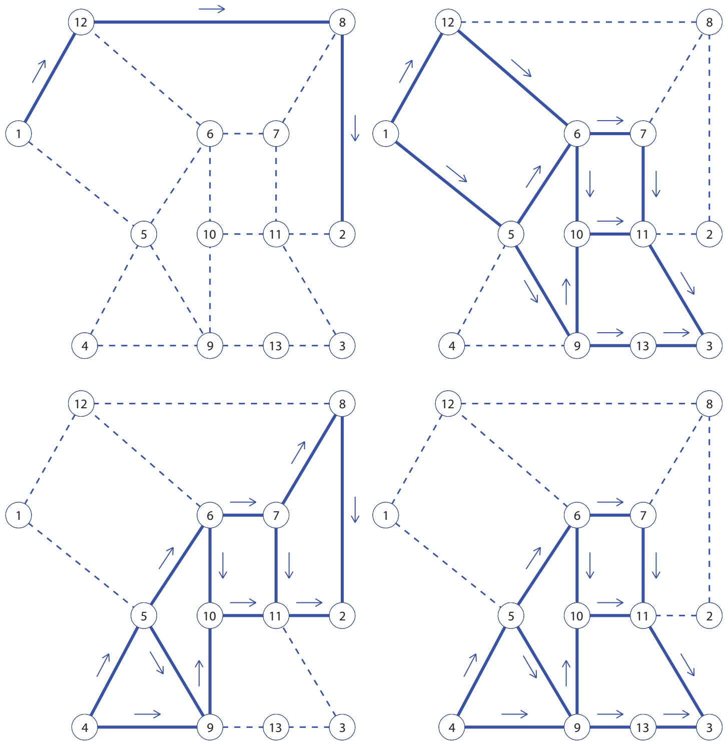

Figure 3.2: Different subsets $\mathcal{A}_{OD}$ of links used by the four OD-pairs (Flow from left to right).

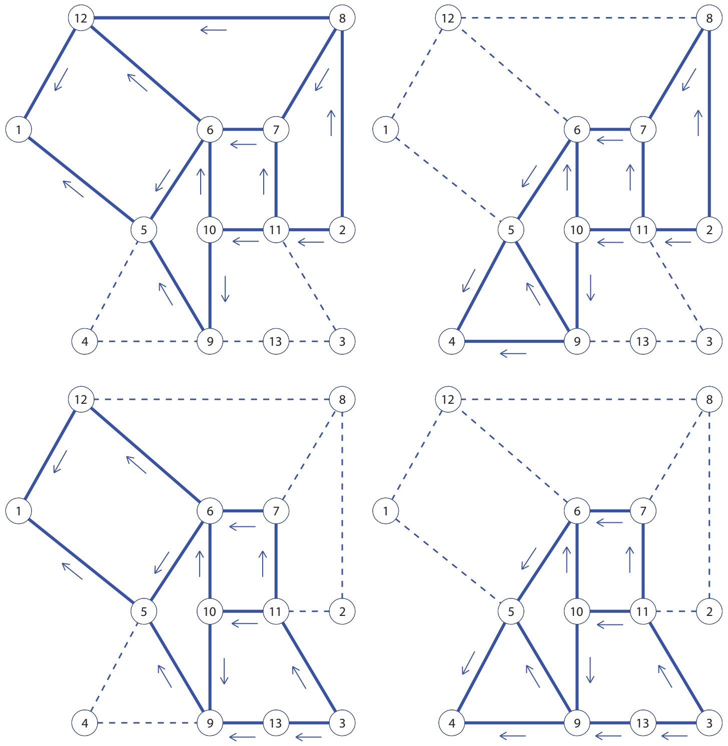

Figure 3.3: Different subsets $\mathcal{A}_{OD}$ of links used by the four OD-pairs (Flow from right to left).

3.4 System optimization (SO). Wardrop's second principle

The first Wardrop's principle does not lead to the best possible use of the network system. This is because user equilibrium considers that individual travelers act only in their own interests, but not necessarily in the interest of the system as a whole. To optimize general interests Wardrop proposed his second principle:

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

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

$$\underset{\mathbf{h},\mathbf{v}}{\text{Minimize}}\, Z = \sum_{a\in A} v_a c_a(v_a), \tag{3.56}$$

subject to

$$\sum_{r\in R_{pq}} h_{pqr} = d_{pq} : \ \xi_{pq}, \qquad \forall (p,q)\in D \tag{3.57}$$

$$\sum_{(p,q)\in D}\sum_{r\in R_{p,q}} \delta_{apqr} h_{pqr} = v_a : \ \phi_a, \qquad \forall a\in A \tag{3.58}$$

$$h_{pqr} \geq 0 : \ \mu_{pqr}, \qquad \forall r\in R_{pq}, \forall (p,q)\in D \tag{3.59}$$

Under the system optimum, some travelers may be assigned to routes that have travel costs higher than the minimal that they could achieve by deciding by themselves independently.

Theorem 3 (Variational relation associated with Wardrop's second principle.) *Under Wardrop's second principle, the marginal contribution to the total cost of the system of a new user is identical for all used routes in the same OD and larger for the unused routes of this OD. In other words, the following variational relation holds:*

$$h_{pqr}\left(\sum_a \phi_a \delta_{apqr} - \xi_{pq}\right) = 0, \qquad \forall (p,q)\in \mathcal{D}, \forall r\in R_{pq}, \tag{3.60}$$

$$\sum_a \phi_a \delta_{apqr} - \xi_{pq} \geq 0, \qquad \forall (p,q)\in \mathcal{D}, \forall r\in R_{pq}, \tag{3.61}$$

Proof. The Lagrangian function of problem 3.56–3.59 is

$$\mathcal{L}(\mathbf{h},\mathbf{v};\boldsymbol{\lambda},\boldsymbol{\mu}) = \sum_{a\in\mathcal{A}} v_a c_a(v_a) + \sum_{(p,q)\in D} \xi_{pq}\left(d_{pq} - \sum_{r\in R_{pq}} h_{pqr}\right) + \tag{3.62}$$
$$+\sum_{a\in\mathcal{A}} \phi_a \left(\sum_{(p,q)\in D}\sum_{r\in R_{p,q}} \delta_{apqr} h_{pqr} - v_a\right) - \sum_{(p,q)\in D, r\in R_{pq}} \mu_{pqr} h_{pqr}$$

The Karush-Kuhn-Tucker (KKT) conditions are:

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial v_a} = c_a(v_a) + v_a c_a'(v_a) - \phi_a &= 0; \quad \forall a \in \mathcal{A} && (3.63)\\
\frac{\partial \mathcal{L}}{\partial h_{pqr}} = -\xi_{pq} + \sum_a \phi_a \delta_{apqr} - \mu_{pqr} &= 0; \quad \forall (p,q) \in \mathcal{D}, \forall r \in \mathcal{R}_{pq} && (3.64)\\
\sum_r h_{pqr} &= d_{pq}; \quad \forall (p,q) \in \mathcal{D} && (3.65)\\
\sum_{(p,q)} \sum_r \delta_{apqr} h_{pqr} &= v_a; \quad \forall a \in A && (3.66)\\
h_{pqr} &\geq 0; \quad \forall (p,q) \in \mathcal{D}, \forall r \in \mathcal{R}_{pq} && (3.67)\\
h_{pqr}\mu_{pqr} &= 0; \quad \forall (p,q) \in \mathcal{D}, \forall r \in \mathcal{R}_{pq} && (3.68)\\
\mu_{pqr} &\geq 0; \quad \forall (p,q) \in \mathcal{D}, \forall r \in \mathcal{R}_{pq}. && (3.69)
\end{aligned}$$

Equation (3.64) can be written as:

$$\mu_{pqr} = \sum_a \phi_a \delta_{apqr} - \xi_{pq}; \quad \forall r \in \mathcal{R}_{pq}. \tag{3.70}$$

If $h_{pqr} > 0$ then from (3.68) we have $\mu_{pqr} = 0$, that according to (3.70) leads to $\xi_{pq} = \sum\limits_a \phi_a \delta_{apqr}$. In addition, due to (3.69) $\mu_{pqr} = \sum\limits_a \phi_a \delta_{apqr} - \xi_{pq} \geq 0$, that is $\sum\limits_a \phi_a \delta_{apqr} \geq \xi_{pq}$. This is equivalent to (3.60)-(3.61), that ends the proof. ■

In addition, using (3.63) we get:

$$\phi_a = c_a(v_a) + v_a c_a'(v_a); \quad \forall a \in \mathcal{A}. \tag{3.71}$$

In summary, we can draw the following interesting conclusions.

1. The dual variable ϕ_a can be interpreted as the link marginal contribution to total travel time in the network of an additional traveler that increments the link cost in $c_a(v_a)$ and the cost of other travelers in $v_a c_a'(v_a)$ ($c_a'(v_a)$ each).

2. The dual variables ξ_{pq} can be interpreted as the route (r of OD (p,q)) marginal contribution to total travel time in the network of an additional traveler. In fact if $h_{pqr} > 0$ because of (3.68) we have $\mu_{pqr} = 0$, that is, from (3.70) we get $\xi_{pq} = \sum\limits_a \phi_a \delta_{apqr}$. We note that this marginal contribution is the same for all used routes in the same OD.

3. From Equation (3.70) we can conclude that the dual variable μ_{pqr} can be interpreted as the over-contribution (larger than other routes in the same OD) associated with unused route r with origin-destination (p,q).

Theorem 4 *Both, the Wardrop first and second principles lead to the same solution for uncongested networks (Sheffi (1985)).*

Proof. When the link is uncongested, then $c_a(v_a) \simeq C_{a0}$ and then

$$C_a(v_a) = \int_0^{v_a} c_a(s)ds = c_{a0}(v_a),$$

i.e. both objective functions coincide. Since the constraints are common for both problems, they share the same optimal solution. ∎

The SO problem has no practical sense because it does not provide an equilibrium condition because the users will try not to follow the common rule, but their own optimal choices.

3.5 Stochastic User Equilibrium (SUE)

Definition 3 (Stochastic User Equilibrium (SUE)) *An stochastic user equilibrium is reached when no traveler* believes *that his travel time can be improved by unilaterally changing route and it is assumed that driver's perceptions and preferences are random.*

Due to variations in perception and factors such as weather, lighting, etc., the path times are perceived differently by each driver, therefore, the perceived travel time of each path can be modeled as a random variable with its associated probability density function. Given the perceptions of travel time, each driver is assumed to choose the shortest travel time path from origin to destination.

To treat this problem analytically, let the random variable C_{pqr} represent the perceived travel time on route r between origin p and destination q, where $r \in \mathcal{R}_{pq}$, the set of paths associated with OD pair p-q, and let c_{pqr} be the measured, or actual travel time on route r between p and q. Assume that:

$$C_{pqr} = c_{pqr} + \xi_{pqr} \quad \forall p, q, r, \tag{3.72}$$

where ξ_{pqr} is a random error term associated with the route consideration. Furthermore, assume that $E[\xi_{pqr}] = 0$, or $E[C_{pqr}] = c_{pqr}$, in other words, the average perceived travel time is the actual travel time. The average travel time can be interpreted as the travel time used in the deterministic analysis. The share of drivers choosing the $r - th$ route, P_{pqr}, is given by:

$$P_{pqr}(\mathbf{c}_{pq}) = Prob\left[C_{pqr} \leq C_{pql} \quad \forall l \in \mathcal{R}_{pq}\right] \quad \forall p, q, r. \tag{3.73}$$

In other words, the probability that a given route is chosen is the probability that its travel time is perceived to be the lowest of the all alternative routes.

One of the practical reasons for using stochastic network loading models is the sensitivity of the flows in deterministic models to small changes in the network. Consider, for example, a two-paths network whose travel times are t and $t' = t + \Delta t$ respectively, connecting a single OD pair. While $\Delta t \geq 0$, all the flow should be assigned to the lowest-travel-time path, i.e., the path 1. Although $\Delta t \to 0$, the path 2 has no flow. Treated by a stochastic network loading model, the probability that route 1 is chosen will be approximately 0.50 (when $\Delta t \geq 0$). For very low variance, travelers perceive the difference among the travel cost accurately and act accordingly, as the deterministic model (all-or-nothing) and when the variance is larger, the change in the choice probability is more moderate since the perception of travel times is less accurate. At the limit, when the variance is extremely large, the actual travel times do not affect the perception, which is completely random.

Theorem 5 (Equilibrium.) *Assuming a network $(\mathcal{N}, \mathcal{A})$, where $\mathcal{N}$ is the set of nodes and $\mathcal{A}$ is the set of links, the mathematical equations associated with this model are:*

$$d_{pq} P_{pqr}(\mathbf{c}_{pq}) = h_{pqr}; \qquad \forall (p,q) \in D, r \in R_{pq} \tag{3.74}$$

$$c_{pqr} = \sum_{a \in \mathcal{A}} c_a(v_a)\delta_{apqr}; \qquad \forall (p,q) \in D, r \in R_{pq}, \tag{3.75}$$

where D is the set of OD-pairs, R_{pq} is the set of paths associated with OD pair p-q, h_{pqr} is the flow associated with OD p-q and path r, $P_{pqr}(\mathbf{c}_{pq})$ is the probability that a random individual traveling through origin-destination p-q chooses path r.

Therefore, a stochastic user equilibrium method can be stated as:

$$\underset{\mathbf{v}}{Minimize}\, Z_{SUE} = -\sum_{p,q \in D} h_{pq} E\left[\min_{r \in R_{pq}} (C_{pqr})\right] + \sum_{a \in A} v_a c_a(v_a) - \sum_{a \in A} \int_0^{v_a} c_a(s)ds, \tag{3.76}$$

where Z_{SUE} is the objective function to be minimized, and $\mathbf{v} \equiv \{v_a | a \in A\}$ is the vector containing the link flows to be estimated.

Proof. Since we have an unconstrained problem, the optimal solution must satisfy the following equation

$$\begin{aligned}\frac{\partial Z_{SUE}}{\partial v_a} &= -\sum_{p,q \in D} h_{pq} \sum_{r \in R_{pq}} \frac{\partial E\left[\min_{r \in R_{pq}}(C_{pqr})\right]}{\partial c_{pqr}} c'_a(v_a)\delta_{apqr} + v_a c'_a(v_a) \\ &= c'_a(v_a)\left[v_a - \sum_{p,q \in D} h_{pq} \sum_{r \in R_{pq}} \frac{\partial E\left[\min_{r \in R_{pq}}(C_{pqr})\right]}{\partial c_{pqr}} \delta_{apqr}\right] = 0, \end{aligned} \tag{3.77}$$

which leads to

$$v_a = \sum_{p,q \in D} h_{pq} \sum_{r \in R_{pq}} \frac{\partial E\left[\min_{r \in R_{pq}}(C_{pqr})\right]}{\partial c_{pqr}} \delta_{apqr} \quad \text{or} \quad c'_a(v_a) = 0, \tag{3.78}$$

Based on the assumption of common regularity conditions (see, e.g., Sheffi (1985)) one obtains

$$p_{pqr} = \frac{\partial E\left[\min_{r \in R_{pq}}(C_{pqr})\right]}{\partial c_{pqr}}, \tag{3.79}$$

where p_{pqr} are the probabilities of a user to select path r of OD-pair p-q. We note that (3.79) is not a result of solving the SUE problem, indeed it has nothing to do with equilibrium problems at all; from reading textbooks on random utility theory, one will find that it is a fundamental property of such models, under certain regularity conditions.

Since

$$v_a = \sum_{p,q \in D} h_{pq} \sum_{r \in R_{pq}} p_{pqr} \delta_{apqr}, \quad a \in A, \tag{3.80}$$

one could could identify (3.78) and (3.80) to get (3.79). However, we must note that identification of (3.78) and (3.80) leads to a homogeneous linear system of equations, which can have an infinite number of solutions, some or all of which could not be probabilities. So, if the p_{pqr} resulting from (3.79) are probabilities, we have one solution to our traffic problem. Otherwise, we only have the solution to a mathematical optimization problem, which does not represent our physical traffic situation.

We noted in (3.78) that Equation (3.77) has also the solution $c'_a(v_a) = 0$, that in some cases leads to $v_a = 0$. For example, this occurs with the commonly BPR function (used in Example 7.11)

$$c_a(v_a) = c_{a0} \left[1 + \beta_a \left(\frac{v_a}{q_a} \right)^{\gamma} \right]. \tag{3.81}$$

Since

$$\frac{\partial^2 Z_{SUE}}{\partial v_a \partial v_b} = \begin{cases} c''(v_a) \left[v_a - \sum\limits_{p,q \in D} h_{pq} \sum\limits_{r \in R_{pq}} \dfrac{\partial E\left[\min_{r \in R_{pq}}(C_{pqr})\right]}{\partial c_{pqr}} \delta_{apqr} \right] + c'(v_a) & \text{if} \quad a = b \\ 0 & \text{otherwise} \end{cases}$$

and for the first solution in (3.78) we have $c'(v_a) > 0$, i.e., the travel cost function has to be strictly increasing, we can guarantee that we are in front of a local minimum in this case. On the contrary, the alternative solution coming from $c'(v_a) = 0$ is not a local minimum. ■

The perceived travel time is assumed to have some probability distribution functions on each path. Depending on the probability distribution of the random variable, we can find different stochastic route choice models. The most widely used are the following ones:

3.5.1 The multinomial probit based models (MNP)

The probit based model was formulated by Burrel (1968) and was developed by Daganzo and Sheffi (1977) later on.

As we explained (see Eq. (3.72)), the perceived travel time C_{pqr} has a deterministic part c_{pqr}, such that $E[C_{pqr}] = c_{pqr}$, and a random part ξ_{pqr}, such that $E[\xi_{pqr}] = 0$.

In these models, the joint probability distribution of the perceived travel times is assumed to be the **multivariate normal** distribution function (MVN). Then, the travel times can be modeled as

$$C_{pqr} \sim MNV[c_{pqr}, \Sigma_{pqr}], \tag{3.82}$$

where c_{pqr} is the vector of means and Σ_{pqr} is a nonsingular, symmetric and positive define covariance matrix. Due to the independence between the $r \in R$ alternatives, the covariance terms are zero, i.e., the covariance matrix is a diagonal matrix.

The MVN distribution has the property of the reproductivity, however, their cumulative density function (cdf) does not have this property. When we calculate the probability of choice (see Eq. (3.73)) we need to use the cdf, that implies to evaluate multiple integrals. In order to avoid this difficulty there exist several methods like the approximation method or Montecarlo simulation method.

We note that the normality assumption may not be entirely appropriate for modeling the distributions of the perceived link travel times, since these times cannot be negative.

3.5.2 The multinomial logit based models (MNL)

In these models it is assumed that the probability distribution of the perceived travel time on each path are the **reversed Gumbel** distributions.

If $C_{pqr} \sim \text{Gumbel}_{rev}(\theta_{pq}, \delta_{pqr})$, i.e., its survival function $G(h)$ is[4]

$$G(h) = \exp[-\exp(\theta_{pq}(h - \delta_{pqr}))] \tag{3.83}$$

with mean

$$E[C_{pqr}] = \delta_{pqr} - \gamma_e/\theta_{pq} \tag{3.84}$$

and variance

$$Var[C_{pqr}] = \frac{\pi^2}{6\theta_{pq}^2} \tag{3.85}$$

Assuming independence, we have

$$\begin{aligned} G_{\min_{r\in R_{pq}}(C_{pqr})}(h) &= \prod_{r\in R_{pq}} \exp[-\exp(\theta_{pq}(h - \delta_{pqr}))] \\ &= \exp\left[-\exp(\theta_{pq}h)\sum_{r\in R_{pq}} \exp(-\theta_{pq}\delta_{pqr})\right] \\ &= \exp[-\exp(\theta_{pq}(h - \delta_0)), \end{aligned} \tag{3.86}$$

where

$$\delta_0 = -\frac{1}{\theta_{pq}} \log \sum_{r\in R_{pq}} \exp(-\theta_{pq}\delta_{pqr}), \tag{3.87}$$

which proves the stability under minimum operations of the selected reversed Gumbel family.

[4] We work with survival functions because it is more convenient than the cumulative distribution functions (cdf) for dealing with minima. The survival function is $S(x) = 1 - F(x)$, where $F(x)$ is the cumulative distribution function.

Taking now into account that

$$E[C_{pqr}] = \delta_{pqr} - \gamma_e/\theta_{pq} = c_{pqr}, \tag{3.88}$$

where γ_e is the Euler constant, and that

$$E[\min_{r\in R_{pq}} C_{pqr}] = \delta_0 - \gamma_e/\theta_{pq}, \tag{3.89}$$

one obtains

$$E[\min_{r\in R_{pq}} C_{pqr}] = -\frac{1}{\theta_{pq}} \log \sum_{r\in R_{pq}} \exp(-\theta_{pq}c_{pqr} - \gamma_e) - \frac{\gamma_e}{\theta_{pq}} = -\frac{1}{\theta_{pq}} \log \sum_{r\in R_{pq}} \exp(-\theta_{pq}c_{pqr}), \tag{3.90}$$

and then Equation (3.76) becomes

$$\underset{\mathbf{v}}{\text{Minimize}}\, Z_{SUE} = \sum_{pq\in D} \frac{1}{\theta_{pq}} h_{pq} \log \left(\sum_{r\in R_{pq}} \exp\left(-\theta_{pq}c_{pqr}\right)\right) + \sum_{a\in A} v_a c_a(v_a) - \sum_{a\in A} \int_0^{v_a} c_a(s)ds, \tag{3.91}$$

which was obtained by Sheffi and Daganzo (1978), and from (3.79) and (3.90) one obtains the well known logit formula:

$$p_{pqr} = \frac{\partial E\left[\min_{r\in R_{pq}}(C_{pqr})\right]}{\partial c_{pqr}} = \frac{\exp(-\theta_{pq}c_{pqr})}{\sum\limits_{l} \exp(-\theta_{pq}c_{pql})}. \tag{3.92}$$

The multinomial logit SUE has long been known to suffer from the problem of assuming independent, identically distributed random path components. In addition, the reversed Gumbel family used implies the same variance for all routes, which is not realistic.

The main reason for forcing all distributions for the same OD-pair p-q to share the same value of the scale parameter θ_{pq} of the reversed Gumbel model is that only then, the reversed Gumbel family is stable with respect to minimum operations.

Appendix

A Travel times and routes

In this section we show how to obtain the minimum travel times (minimum cost) between any pair of nodes and how to obtain all actually used routes.

A1 Determining minimum trip cost between any pair of nodes

Apart from using the dual variables to calculate the minimum costs (time) associated with any pair of nodes, an alternative method, which also identifies the possible OD pairs and the unreachable nodes from a given node, is given below.

Once the problem (3.35)-(3.37) has been solved, one can immediately determine the costs (times) associated with the links, and the cost (time) of a given path can be calculated.

There are several available methods in the existing literature to calculate the table of minimum costs to travel between any pair of nodes, and the unreachable nodes from a given node. For the sake of completeness, we give below one of them, which can be stated as the following optimization problem:

$$\underset{\mathbf{cost}}{Maximize} \sum_{i,j} cost_{ij} \tag{3.93}$$

subject to

$$\begin{aligned} cost_{ip} &\leq cost_{ij} + c_{jp}; \quad \forall \text{ link } a_{jp} && (3.94)\\ cost_{ij} &\geq 0 && (3.95)\\ cost_{ii} &= 0 && (3.96)\\ cost_{ij} &\leq c_{ij}; \quad \forall \text{ link } a_{ij} && (3.97)\\ cost_{ij} &\leq c_0 && (3.98) \end{aligned}$$

where c_{ij} is the cost associated with link a_{ij}, with value 0 if $i = j$, and c_0 is a high enough upper bound of the costs c_{ij}.

The problem (3.93)-(3.98) gives the minimum trip cost (time) $cost_{ij}$ between any pair of nodes (i, j) if the resulting $cost_{ij} < c_0$, and the unreachable nodes from node i, if the resulting $cost_{ij} = c_0$. It is based on the Bellman dynamic programming constraint (3.94).

Note that the constraint (3.98) is required to deal with nodes not connected to the rest of nodes, and to avoid unboundedness.

A2 Route identification and minimum cost path finding algorithm

In this section we give some tools to identify and enumerate paths.

Given a candidate to path between nodes p and q, that is, a sequence of links joining two nodes p and q, it is very easy to decide whether or not it is a path with the help of matrices $cost_{ij}$ and the link costs c^*_{ij}. To this end, we sum the costs associated with all the links of the candidate to path and if the total cost equals $cost_{pq}$ it is a path; otherwise it is not.

Next, we supply an algorithm for finding all paths corresponding to a given OD-pair which all are of minimum cost. Note that an efficient use of the minimum cost table $cost_{ij}$ is used by cancelling the branching scheme as soon as the partial paths are not optimal.

Algorithm 1 (Minimum cost path finding algorithm)

```
Input: A graph, a table $cost_{ij}$ of minimum travel costs among, origin $X_i$ and destination
    node $X_j$ and a table with the link costs $c_{ij}$.

Output: All the minimum cost paths between origin node $X_p$ and a destination node
    $X_q$, or an empty set if there are no paths.

Initial Step: $X_i \leftarrow X_p$
    $Path \leftarrow X_p$
    $Visited \leftarrow X_p$
    $Pcost = 0.$
    $CandidatePaths = 1$

Iteration Step:
    while $CandidatePaths=1$
        if there exists $X_r \in Adj(X_i)$not in Visited, $Pcost + c_{ir} \leq cost_{pr}$ and
            $Pcost + c_{ir} \leq cost_{pq}$,
        then
            Add $X_r$ to Visited and Path.
            $Pcost=Pcost+c_{ir}$.
            if $X_r = X_q$, then
                A path has been found.
                Remove node $X_r$ from Path.
                $Pcost=Pcost - c_{ir}$
                $X_i \leftarrow$ last node in Path.
            else
                $X_i \leftarrow X_r$.
        else if $X_i = X_p$
                There is no path.
                $CandidatePaths=0.$
            else
                Remove node $X_i$ from Path.
                $X_i \leftarrow$ *last node* $X_p$ *in Path.*
                *Pcost=Pcost -* $c_{pr}$.
        **end if**.
    **end while**.
```

B Implementation of the proposed method

In this section we explain how the previous methods can be incorporated to obtain disaggregated flows by ODs and to perform route enumeration.

The traffic assignment proposed method together with a post-process of the resulting information to obtain the matrix of least costs and to enumerate optimal paths, can be practically implemented in the following steps:

Step 1: Obtaining link flows and costs. Initially, the problem (3.35)-(3.37) is solved to obtain all the link flows v_{ij} and the link cost matrix with elements c_{ij}. Once this problem has been solved and the link flows v^*_{ij} are known, the problem (3.44)-(3.47) is solved to get the unique v_{ijpq} disaggregated flow values.

Step 2: Obtaining the OD cost matrix. Using the link costs resulting from problem (3.35)-(3.37), the minimum cost problem (3.93)-(3.98) is solved supplying the minimum cost matrix $C \equiv cost_{ij}$.

Step 3: Enumerating paths. All minimum cost paths for each OD-pairs can be found, for example, using the algorithm given in Appendix 3.5.2. Note that only minimum cost (optimal) paths are considered, that is, a reduced number (the most important paths) with respect to all possible paths[5].

[5] We note that from a practical point of view it has no sense considering routes which are hardly used.

Chapter 4

Dynamic traffic models

Contents

4.1 Notation

α_a	free-flow travel time.
$\eta_a(t)$	link a travel time associated with a user that leaves link a at time t.
Λ	total number of queued vehicles at the link exit node.
ϕ_{pqr}	stochastic travel cost between the OD-pair p-q through the route r associated with stochastic demand $d_{pq}(\omega)$ or link capacity $q_a(\omega)$.
ϕ^*_{pqr}	stochastic minimum travel cost associated with route r with origin-destination p-q.
$\hat{\phi}_{pqr}(h,\omega)$	worst-case travel cost between the OD-pair p-q through the route r

	associated with the worst case demand $\hat{d}_{pq}$.
$\pi_{pq}(t)$	equilibrium travel cost to travel from origin p to destination q at time t.
ρ	traffic density.
ρ^*	dimensionless density.
ρ_c	critical density.
ρ_c^*	dimensionless critical density.
ρ_j	jam traffic density.
$\tau(t)$	delay function.
$\tau_a(t)$	link traversal time for a vehicle entering the link a at time t.
a	link.
$A(x,t)$	cumulative flow past any point x by time t.
c_a	travel cost associated with link a.
$c_{ij}(t)$	travel cost to go from the initial node i to the final node j associated with the departure time t.
C_{pqr}^*	least travel cost corresponding to route r with origin-destination p-q.
c_{pqr}	travel cost function associated with route r with origin-destination p-q.
D	abscissa coordinate of a given point.
D^*	dimensionless abscissa coordinate of a given point.
d_{pq}	demand for the flows from origin p to destination q.
$\hat{d}_{pq}$	worst case demand associated with the origin p to destination q at time t.
$E_a(t)$	cumulative inflow up to time t for link a.
$e_a(t)$	inflow rate for link a at time t.
$G_a(t)$	cumulative flow associated with the exit of link a by time t.
$g_a(t)$	outflow rate for link a at time t.
h_{pqr}	flow on route r with origin-destination p-q.
$h_{pqr}^*(t)$	flow on route r with origin-destination p-q at time t in a dynamic user equilibrium state.
L	link length.
$N_i(t)$	maximum number of vehicles that can be inside cell i at time t (cell capacity).
$n_i(t)$	number of vehicles in cell i at time t.
q	traffic flow.
q^*	dimensionless flow.
$q_a(\omega)$	stochastic link capacity.
$Q_i(t)$	maximum flow in cell i at time t.
q_m	maximum flow.
R_{pq}	set of routes with origin-destination p-q.

t	time.
t^*	dimensionless time.
$T_a(t)$	link a entry time associated with a user that leaves link a at time t.
u	traffic velocity.
u^*	dimensionless velocity.
u_0	free-flow speed.
u_c	critical velocity.
u_w	shock wave speed.
v_a	flow on link a.
x	abscissa coordinate.
x^*	dimensionless abscissa coordinate.
x_0	abscissa associated with the initial time.
x_0^*	dimensionless abscissa associated with the initial time.
$x_a(t)$	number of vehicles on link a at time t.
x_{max}	maximum number of vehicles in a link. Capacity.
$y_i(t)$	maximum flow in cell i and the number of vehicles that coming from cell $i-1$ can enter cell i during the time interval $(t, t+1)$.
$Y_{pqr}(t)$	cumulative number of vehicles entering the network at origin p and exiting at destination q over route r at any time t.
$y_{pqr}(t)$	instantaneous flows that arrives at destination q from origin p at time t.
$\mathcal{A}$	set of links of a traffic network.
$\mathcal{D}$	set of origin-destination pairs of nodes.
$\mathcal{N}$	set of nodes of a traffic network.

4.2 Introduction

Dynamic traffic models consist in describing traffic flow patterns over time and space on a transportation network for a given set of time-varying origin-destination (OD) demands based on some predefined conditions. The users select their routes and departures times according to the traffic conditions and desired arrival times.

Figure 4.1 shows a classification of dynamic traffic assignment problems described in this chapter.

These problems present two components: (i) the network loading, i.e., how traffic propagates on a transport network and hence governs the network performance in terms of travel time; and (ii) the assignment strategy, i.e., how the travelers select their routes, departures times, modes or destinations.

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:

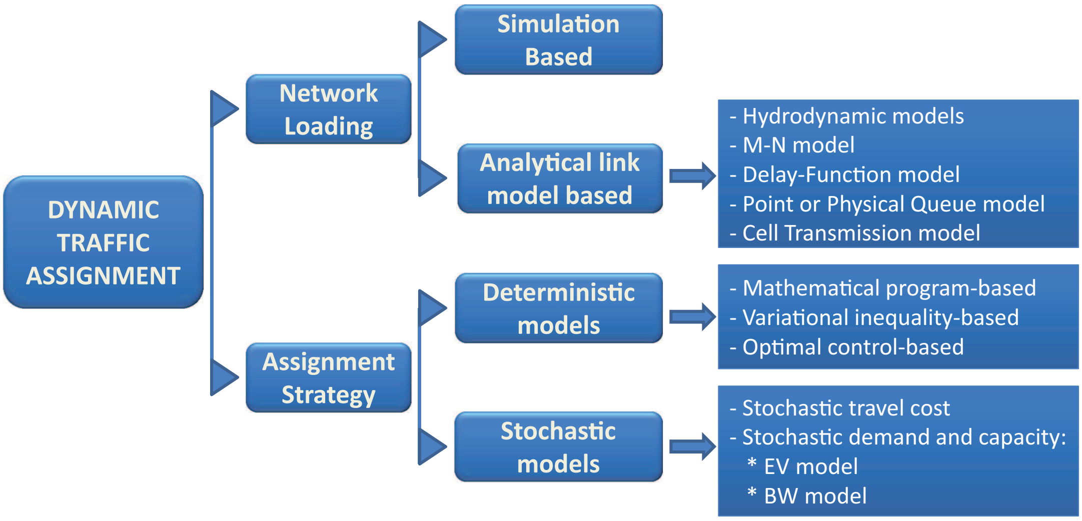

Figure 4.1: Classification of the dynamic traffic assignment problems.

- *Simulation-based approach model.* This approach emphasizes each individual driver's behavior, and has the advantage of closely approximating the behavior of individual drivers while readily considering several traffic control schemes. The number of attributes involved is normally quite large, and the properties of the solution remain uncertain. In this group it is worth mentioning the follow-the-leather models (see Chandler et al. (1958), Gazis et al. (1961) or Wagner et al. (1996)), which try to reproduce each driver's behavior due to some stimulus. These models assume that there is a high correlation between the response of a driver and the relative speed of his vehicle with respect to the vehicle ahead. However, such a deep detailed description of traffic flow implies a very high requirements of memory and cpu time, that make these models inapplicable for real cases of traffic flow.

- *Analytical-based approach model.* This approach concerns the average driver's behavior, and is essentially macroscopic. The significant advantages of the analytical models are: (i) the derived optimality conditions can be characterized by preset driver behavior principles, such as utility maximization or equilibrium condition, and (ii) the sensitivity analysis for different scenarios is easier to perform since the procedure is usually less time consuming than with a simulation-based approach.

In this chapter we refer to the analytical-based approach model.

The chapter is structured as follows. In Section 4.3, we present some basic concepts and relationships that are necessary to treat the dynamic traffic approach. In addition, we define causality and the FIFO rule, providing three different mathematical ways to observe this principle. In Section 4.4, we show the main network loading models, i.e., the hydrodynamic models, the Merchant-Nemhauser (M-N) model, the delay function model, the point and physical queue models and the cell transmission model. About the hydrodynamic models we develop a dimensional analysis in order to treat the problem in a dimensionless form. In Section 4.5, the assignment strategy is analyzed. We divide this part in deterministic and stochastic approach. Among the deterministic models, we describes the most used formulations: the mathematical program-based, the variational inequality-based and the optimal control-based. About the stochastic models, we include the cases when the random variable is the travel cost or, when the demand or the road capacity is stochastic. In Section 4.6, we draw some conclusions. Finally, in the Appendix of this chapter, we present the characteristic lines method.

4.3 Traffic flow components

This section presents some concepts and relationships that are used along the chapter.

Let $e_a(t)$ be the inflow rate for link a at time t, $g_a(t)$, the outflow rate for link a at time t, $E_a(t)$, the cumulative inflow up to time t for link a, $G_a(t)$, the cumulative outflow up to time t for link a, $\tau_a(t)$, the link a traversal time at time t and $x_a(t)$ the number of vehicles on link a at entering time t. By definition, we have:

$$E_a(t) = \int_0^t e_a(s)ds, \qquad or \qquad e_a(t) = \frac{\partial E_a(t)}{\partial t}, \tag{4.1}$$

$$G_a(t) = \int_0^t g_a(s)ds, \qquad or \qquad g_a(t) = \frac{\partial G_a(t)}{\partial t}. \tag{4.2}$$

The **flow conservation condition** requires the number of vehicles on a link 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 link exit:

$$x_a(t) = E_a(t) - G_a(t). \tag{4.3}$$

By taking derivatives, we have the following alternative expression for the flow conservation condition:

$$\dot{x}_a(t) = e_a(t) - g_a(t). \tag{4.4}$$

Assuming that the vehicles leave links in the same order as they enter, we can derive the link travel time of each vehicle. Then, the vehicle entering link a at time t_1 exits this link at time t_2 if and only if

$$E_a(t_1) = G_a(t_2). \tag{4.5}$$

As no overtaking is allowed[1], the link travel time of a vehicle is equal to the link exit time

[1] This condition is assumed in common traffic models and is called the FIFO condition, as explained in section 4.3.1.

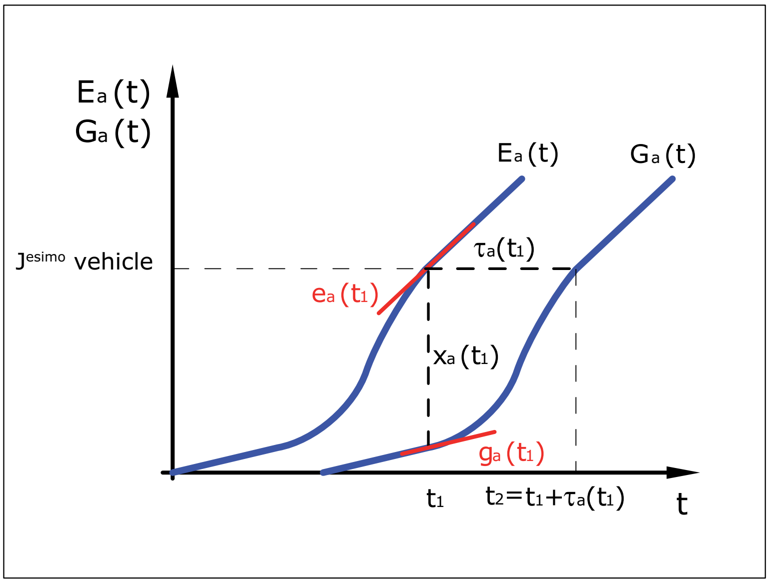

Figure 4.2: Relationships among the link occupancy (x_a), inflow rate (e_a), outflow rate (g_a), cumulative inflow (E_a) and cumulative outflow (G_a).

minus the corresponding entry time. So, we have the following important relationships:

$$E_a(t) = G_a\left(t + \tau_a(t)\right), \quad or \tag{4.6}$$

$$\tau_a(t) = G_a^{-1}[E_a(t)] - t, \tag{4.7}$$

which allow us to know how flow changes with time.

Taking the derivative of (4.6) with respect to t we obtain the well known **flow propagation condition**:

$$g_a(t + \tau_a(t)) = \frac{e_a(t)}{1 + \dfrac{d\tau_a(t)}{dt}}. \tag{4.8}$$

Figure 4.2 shows the relationships among the link occupancy (x_a), inflow rate (e_a), outflow rate (g_a), cumulative inflow (E_a) and cumulative outflow (G_a) that have been discussed mathematically in (4.1)–(4.8).

Following Nie and Zhang (2005a), analytical-based approach models present a common framework, i.e., these models consider (i) the flow conservation, (ii) the flow propagation and (iii) the flow behavior. With "flow behavior" they assume that the link traversal time for a vehicle entering the link at the beginning of the interval $\tau(t)$, or the link outflow during the studied time interval $g(t)$, is a function of the physical characteristics of the link (e.g. free flow travel time, bottleneck capacity...) and the current link state (e.g. volume), considering the causality and the FIFO (First Input First Output) rule:

$$\tau(t) \text{ or } g(t) = f(\text{free flow travel time } \alpha_a, \text{ bottleneck capacity } q_m, \\ \text{causality, FIFO, queue spillback...}).$$

4.3.1 Causality and the FIFO rule

Causality refers to the property that the link travel times for traffic entering at time t_2 only depend on the traffic entering at time $t_1 \leq t_2$. 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 (Heydecker and Addison (1996)).

FIFO (First Input First Output) rule (on the link level) means that users who enter the link earlier will leave it sooner. We can consider three level FIFO rules: link, route and OD. Link (or route) FIFO is satisfied if every user who enters the link (or route) earlier will leave it sooner. Similarly, OD FIFO is satisfied if users on the same OD pair who depart the origin earlier will arrive the destination sooner. Link FIFO can prevent unrealistic situation such as the fast traffic “jump over” the preceding slow traffic. In particular, on a single-lane road and in a queue, no overtaking can occur and capturing FIFO for this situation in modeling is a mandatory.

In the following paragraphs we provide three different mathematical ways to observe this principle.

A. In terms of $\boldsymbol{\tau}_{\mathbf{a}}(\mathbf{t})$. Let $\tau_a(t)$ be the travel time associated with a vehicle that enters link a at time t, then the FIFO condition can be expressed as indicated by the following theorem.

Theorem 6 *The FIFO condition is equivalent to:*

$$\tau_a'(t) > -1. \tag{4.9}$$

Proof. First we proof that condition (4.9) is necessary. Let $\omega > 0$, then the FIFO condition, expressed in terms of the link exit times of two vehicles entering the link at times t and $t + \omega$, respectively, can be written as

$$t + \tau_a(t) < t + \omega + \tau_a(t + \omega); \quad \omega > 0, \tag{4.10}$$

that is,

$$\tau_a(t+\omega) - \tau_a(t) > -\omega \;\Leftrightarrow\; \frac{\tau_a(t + \omega) - \tau_a(t)}{\omega} > -1 \;\Rightarrow \tau_a'(t) = \lim_{\omega \to 0} \frac{\tau_a(t + \omega) - \tau_a(t)}{\omega} > -1. \tag{4.11}$$

Condition (4.9) is sufficient too. If $\tau_a'(t) > -1$, then we have

$$\tau_a(t + \omega) = \tau_a(t) + \int\limits_t^{t+\omega} \tau_a'(t)dt > \tau_a(t) + \int\limits_t^{t+\omega} -1dt = \tau_a(t) - \omega,$$

which implies (4.10). ■

B. In terms of $\boldsymbol{\eta}_\mathbf{a}(\mathbf{t})$. Let $\eta_a(t)$ be the link a travel time associated with a vehicle that leaves link a at time t, then we have the following theorem.

Theorem 7 *The FIFO condition is equivalent to:*

$$\eta'_a(t) < 1. \tag{4.12}$$

Proof. *Necessary condition:* Let $\omega > 0$, then the FIFO condition, expressed in terms of the link entry times of two vehicles leaving the link at times t and $t+\omega$, respectively, can be written as

$$t - \eta_a(t) < t + \omega - \eta_a(t+\omega); \quad \omega > 0, \tag{4.13}$$

that is,

$$\eta_a(t+\omega) - \eta_a(t) < \omega \;\Leftrightarrow\; \frac{\eta_a(t+\omega) - \eta_a(t)}{\omega} < 1 \;\Rightarrow \eta'_a(t) = \lim_{\omega\to 0} \frac{\eta_a(t+\omega) - \eta_a(t)}{\omega} < 1. \tag{4.14}$$

Sufficient condition: If $\eta'_a(t) < 1$, then we have

$$\eta_a(t+\omega) = \eta_a(t) + \int\limits_t^{t+\omega} \eta'_a(t)dt < \eta_a(t) + \int\limits_t^{t+\omega} 1dt = \eta_a(t) + \omega,$$

which implies (4.13). ■

C. In terms of $\mathbf{T_a(t)}$. Let $T_a(t)$ be the link a entry time associated with a vehicle that leaves link a at time t, we have:

Theorem 8 *The FIFO condition is equivalent to:*

$$T'_a(t) > 0. \tag{4.15}$$

Proof. *Necessary condition:* Let $\omega > 0$, then the FIFO condition can be written as

$$T_a(t) < T_a(t+\omega); \quad \omega > 0, \tag{4.16}$$

that is,

$$T_a(t+\omega) - T_a(t) > 0 \;\Leftrightarrow\; \frac{T_a(t+\omega) - T_a(t)}{\omega} > 0 \;\Rightarrow T'_a(t) = \lim_{\omega\to 0} \frac{T_a(t+\omega) - T_a(t)}{\omega} > 0. \tag{4.17}$$

Sufficient condition: If $h'_a(t) > 0$, then we have

$$T_a(t+\omega) = T_a(t) + \int\limits_t^{t+\omega} T'_a(t)dt > T_a(t) + \int\limits_t^{t+\omega} 0dt = T_a(t),$$

which is (4.16). ∎

Theorems 6, 7 and 8 are three equivalent forms of expressing the FIFO condition. Depending on the way of dealing with traffic problems one of them can be more convenient than the other two.

Accordingly, linear travel time functions like $\tau(t) = k_1 + k_2 x(t)$, where k_1 and k_2 are positive constants, obey FIFO. On the other hand, Daganzo (1995b) proved that the travel time functions depending on the inflow $e(t)$ can violate FIFO.

4.4 Network loading Models

4.4.1 Hydrodynamic Models

The analogies of traffic flow with fluid and gas dynamics has attracted the attention of physicists and engineers in the past. Consequently, a great collection of papers dealing with the problem of modeling traffic flow using well known equations from hydraulics have been published. They led to the so-called macroscopic models for traffic simulation.

Thus, one possibility for analyzing the dynamic case consists in using the hydraulic analogy, that is, assuming that the arcs are channels and that the traffic flow is replaced by a fluid (see Hopf (1950), Lax (1954), Whitham (1974)). Two important works of this approach were Lighthill and Whitham (1955) and Richards (1956), who independently proposed similar methods for traffic flow. An important contribution of these models to traffic problems is in the Newell's kinematic wave trilogy (Newell (1993a,b,c)). Newell, instead of using the Lighthill-Whitham-Richards (LWR) theory to evaluate flows or densities, proposes to evaluate the cumulative flow $A(x, t)$ past any point x by time t, because a solution for $A(x, 1)$ can be evaluated directly from boundary or initial conditions without its evaluation at intermediate times and positions. Using this idea, Newell considers several examples including single and multiple O-D pairs. Other interesting results using this model, as the analysis of a red light turning green or red can be seen in Childress (2005), but unfortunately most of its applications are for single arcs and/or particular cases.

In this section, we discuss the mathematical bases of the hydrodynamic models. Due to the simplicity of the formulation that let us a better understanding, we have chosen a dimensionless point of view. In the appendix of this chapter we present the *characteristic lines method*, in terms of density and in terms of flow, and then we solve the red to green traffic light case. Finally, we give some examples with different density distributions.

Variables involved

To begin, we define and expose the relationships between the traffic stream variables that are involved in a traffic model, namely, density, flow and speed.

Density (ρ) is defined as the number of vehicles per unit area of the roadway. In traffic flow, the two most important densities are the critical density (ρ_c) and the jam density (ρ_j). The maximum density achievable under free flow is ρ_c, while ρ_j is the minimum density achieved under congestion. In general, the jam density is in the order of seven times the critical density.

Flow (q), or traffic intensity, is the number of vehicles passing a reference point per unit time, and is measured in vehicles per hour. The inverse of flow is headway, which is the time that elapses between the i^{th} vehicle and the $i+1$ vehicle passing a reference point in space. In congestion, the headway remains constant and reaches a minimum value.

Speed (u) in traffic flow is defined as the distance traveled per unit time. The speed of every single vehicle is almost impossible to track on a roadway; therefore, in practice, average speed is calculated by sampling vehicles over a period of time or area and is calculated and used in formulas.

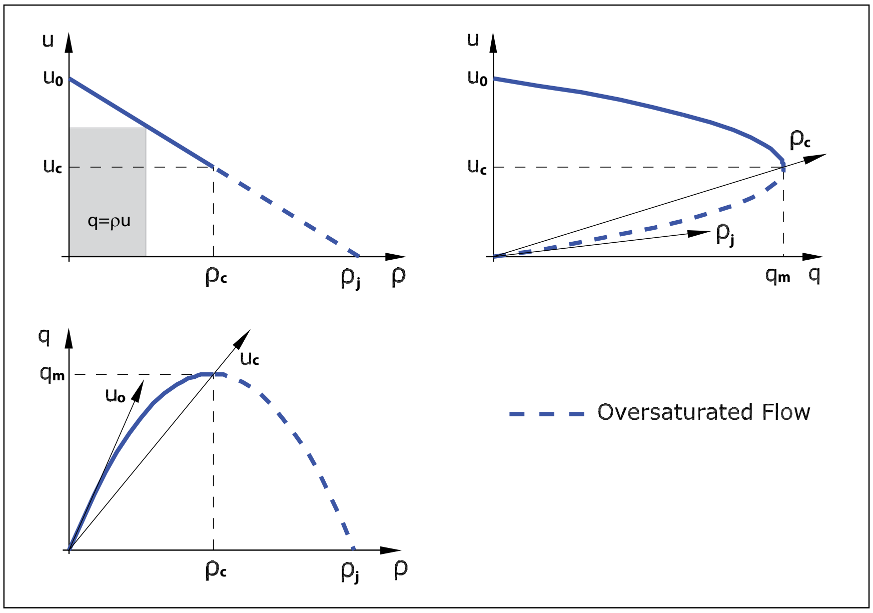

Figure 4.3: Relationships among flow, density and speed for vehicle movement.

Figure 4.3 shows the usually assumed relationships between flow (q), density (ρ) and speed (u). The maximum flow rate, q_m, represents the highest rate of traffic flow that the highway is capable of handling. The traffic density that corresponds to this maximum flow rate is ρ_c, and the corresponding speed is u_c (critical speed).

The basic premise of traffic flow models is that speed is a decreasing function of density. As the density increases, the space between vehicles decreases and drivers react by lowering

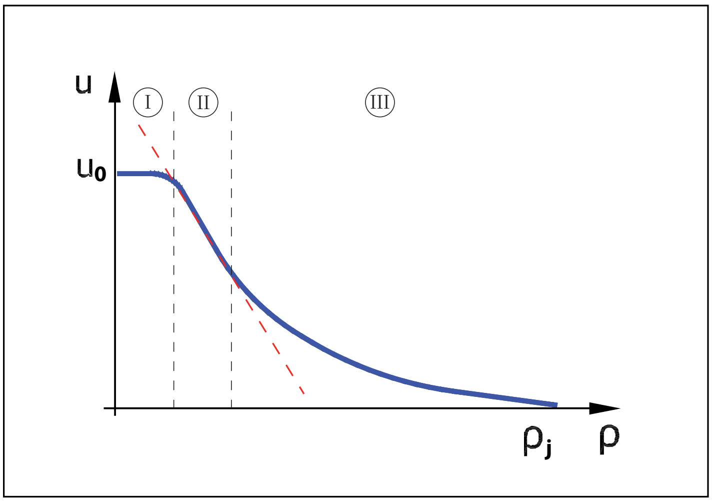

Figure 4.4: Empirical relationship density-velocity.

their speed. Mathematically, it can be expressed as:

$$u = u_0 \left(1 - \frac{\rho}{\rho_j}\right), \tag{4.18}$$

where u_0 is the free-flow speed (which is observed at zero density). Note that when the density is the traffic jam density (ρ_j), the highway section becomes so congested that the traffic stops ($u = 0$).

According to Mannering and Kilareski (1998), it is important to note that field studies have shown that the speed-density relationship tends to be nonlinear at low densities and high densities. In fact, the overall speed-density relationship is better represented by three relationships as indicated in Figure 4.4: (i) a nonlinear relationship at low densities whose speed slowly declines from the free-flow value, (ii) a linear relationship over the large medium-density region (as shown in Equation (4.18)), and (iii) a nonlinear relationship near the traffic jam density as the speed asymptotically approaches zero with increasing density. In any case, along this section we will work with traffic stream models that are based on the assumption of a linear speed-density relationship.

Since $q = \rho u$, from (4.18) the following parabolic flow-density model can be obtained

$$q = u_0 \left(\rho - \frac{\rho^2}{\rho_j}\right). \tag{4.19}$$

The general form of Equation (4.19) is shown in Figure 4.3 (lower-left diagram).

Finally, the following equation shows the relationship between flow and speed:

$$q = \rho_j \left(u - \frac{u^2}{u_0}\right). \tag{4.20}$$

This relationship is also a parabolic function as shown in Figure 4.3 (upper-right diagram). This figure shows that two speeds are possible for each q value, up to the highway's capacity q_m. It is desirable, for any given flow, to keep the average space-mean speed on the upper portion of the speed-flow curve (i.e., above u_c). When speeds drop below u_c, traffic is in a highly congested and unstable condition.

Finally, we define the shockwave how the propagation of congestion upstream from a traffic bottleneck. Congestion shock waves will vary in propagation length, depending upon the upstream traffic flow and density. Figure 4.5 shows the relationship among flow, density and speed in a traffic bottleneck.

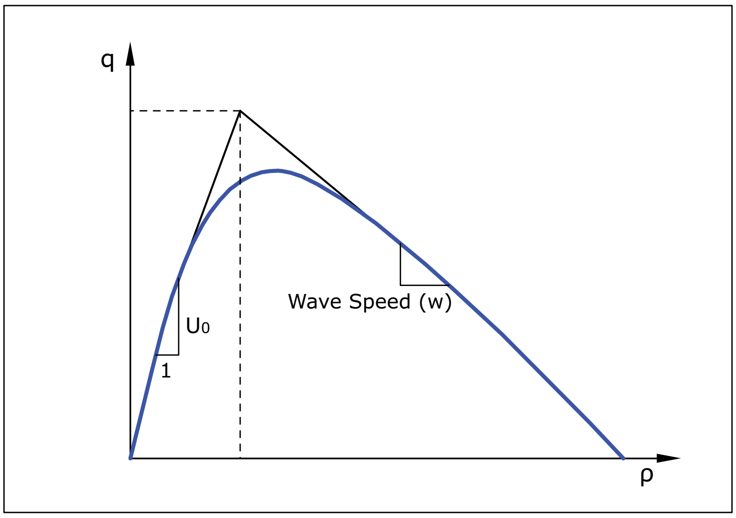

Figure 4.5: Relationship among flow, density and speed in a traffic bottleneck.

Dimensional analysis

We consider the following list of 11 variables involved in our traffic problem:

$$\{q, \rho, \rho_j, \rho_c, u, u_0, x, x_0, D, L, t\}, \tag{4.21}$$

i.e., traffic intensity, density, jam traffic density, density associated with maximum intensity of flow, velocity, maximum velocity, the abscissa coordinate, the abscissa associated with the initial time, the abscissa coordinate of a given point, the link length and time.

The Buckingham theorem states that any relation among these 11 variables is equivalent to a relation among the reduced set of 8 dimensionless variables (see the variable dimensional Table 4.1):

$$\left\{q^* = \frac{q}{\rho_j u_0}; \rho^* = \frac{\rho}{\rho_j}; \rho_c^* = \frac{\rho_c}{\rho_j}; u^* = \frac{u}{u_0}; x^* = \frac{x}{L}; x_0^* = \frac{x_0}{L}; D^* = \frac{D}{L}; t^* = \frac{tu_0}{L}\right\}. \tag{4.22}$$

Note that we have used u_0, ρ_j and L as basic or normalizing variables.

Table 4.1: Dimensional table of the variables involved in the traffic problem.

	q	ρ	ρ_j	ρ_c	u	u_0	x	x_0	D	L	t
Veh	1	1	1	1	0	0	0	0	0	0	0
L	0	-1	-1	-1	1	1	1	1	1	1	0
T	-1	0	0	0	-1	-1	0	0	0	0	1

Traffic variables in dimensionless form

If we apply the Buckingham theorem to obtain the relations among the following set of variables: $\{\rho, \rho_j, u, u_0\}$, $\{q, \rho, \rho_j, u_0\}$ and $\{q, \rho_j, u, u_0\}$, we get (see Table 4.2)

$$u^* = f(\rho^*), \tag{4.23}$$

$$q^* = g(\rho^*) \tag{4.24}$$

and

$$q^* = h(u^*), \tag{4.25}$$

where $f(\cdot)$, $g(\cdot)$ and $h(\cdot)$ are arbitrary functions. As one example, we can easily check that the well known relationships (4.18), (4.19) and (4.20) using the dimensionless variables in (4.22), can be written in dimensionless form as (see Table 4.2)

$$u^* = f(\rho^*) = (1 - \rho^*), \quad 0 \leq \rho^* \leq 1, \tag{4.26}$$

$$q^* = g(\rho^*) = \rho^* \left(1 - \rho^*\right), \quad 0 \leq \rho^* \leq 1 \tag{4.27}$$

and

$$q^* = h(u^*) = u^* \left(1 - u^*\right), \quad 0 \leq u^* \leq 1, \tag{4.28}$$

respectively.

From Equation (4.27) we obtain that

$$0 \leq q^* \leq 1/4.$$

This proves that Expressions (4.18), (4.19) and (4.20) satisfy the Buckingham theorem.

Note that

$$w(\rho) = \left.\frac{dq(\rho)}{d\rho}\right|_{\rho(x,t)} = u_0(1 - 2\rho/\rho_j), \tag{4.29}$$

that in dimensionless form becomes

$$w^*(\rho^*) = \left.\frac{dq^*(\rho)}{d\rho^*}\right|_{\rho^*(x,t)} = (1 - 2\rho^*). \tag{4.30}$$

Table 4.2: Dimensional tables of the variables involved in three traffic problems.

	ρ	ρ_j	u	u_0		q	ρ	ρ_j	u_0		q	ρ_j	u	u_0
Veh	1	1	0	0	Veh	1	1	1	0	Veh	1	1	0	0
L	-1	-1	1	1	L	0	-1	-1	1	L	0	-1	1	1
T	0	0	-1	-1	T	-1	0	0	-1	T	-1	0	-1	-1

Formulation

Let $q(x,t)$, $u(x,t)$ and $\rho(x,t)$ be the traffic intensity, velocity and density, respectively, associated with location x at time t. These three magnitudes are related by

$$q(x,t) = F(\rho(x,t)) = \rho(x,t)u(x,t). \tag{4.31}$$

The conservation law stated at the space and time intervals $(x, x+dx)$ and $(t, t+dt)$, respectively, leads to (flow entering the space interval minus flow leaving the interval equal to increment of users, all referred to the time interval dt):

$$\rho(x,t+dt)dx - \rho(x,t)dx = q(x,t)dt - q(x+dx,t)dt, \tag{4.32}$$

that produces the partial differential equation

$$\frac{\partial \rho}{\partial t} + \frac{\partial q}{\partial x} \;=\; 0, \tag{4.33}$$

which is the differential equation sought after.

Equation (4.33) in dimensionless form becomes

$$\frac{\partial \rho^*}{\partial t^*} + \frac{\partial q^*}{\partial x^*} \;=\; 0, \tag{4.34}$$

Differential equation in terms of $\rho(x,t)$

Assuming the functional relation (4.31), between $q(x,t)$ and $\rho(x,t)$ we have

$$\frac{\partial \rho}{\partial t} + \frac{\partial \rho u(\rho)}{\partial x} = 0 \qquad \Leftrightarrow \qquad \frac{\partial \rho}{\partial t} + F'(\rho)\frac{\partial \rho}{\partial x} = 0. \tag{4.35}$$

From (4.18) we obtain

$$F(\rho) = u_0\rho\left(1 - \frac{\rho}{\rho_j}\right). \tag{4.36}$$

Equation (4.35) in dimensionless form becomes

$$\frac{\partial \rho^*}{\partial t^*} + (F^*)'(\rho^*)\frac{\partial \rho^*}{\partial x^*} \;=\; \frac{\partial \rho^*}{\partial t^*} + (1-2\rho^*)\frac{\partial \rho^*}{\partial x^*} = 0, \tag{4.37}$$

where

$$q^* = F^*(\rho^*) = \rho^*(1-\rho^*).$$

An important advantage of model (4.35) that facilitates the solution of many particular problems is that its general solution can be obtained analytically as:

$$x = tF(\rho(x,t)) + \Phi(\rho(x,t)) \tag{4.38}$$

where $\Phi(\cdot)$ is an arbitrary function, which can be verified by taking partial derivatives and replacing into (4.35).

Note that for $t = 0$, Expression (4.35) becomes $x = \Phi(\rho(x,0))$. So, if $\rho(x,0)$ is invertible (strictly monotone) with respect to its first argument, we have

$$\Phi(x) = \rho^{-1}(x,0). \tag{4.39}$$

The family of characteristic lines passing throughout the points $(x_0, 0)$ are:

$$x = F'(\rho)t + x_0 = u_0\left(1 - \frac{2\rho}{\rho_j}\right)t + x_0 \tag{4.40}$$

and in dimensionless form the family of characteristic lines passing throughout the points $(x_0^*, 0)$ are:

$$x^* = (1 - 2\rho^*)\,t^* + x_0^*. \tag{4.41}$$

Shock wave (ρ equation)

In this section we derive the propagation speed of a shock wave.

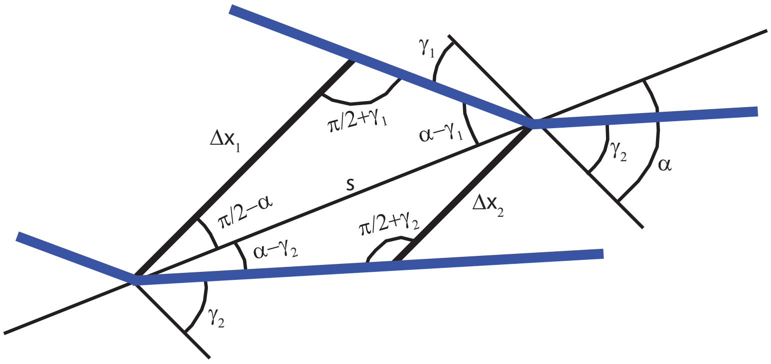

Figure 4.6: Two close trajectories in the neighborhood of a shock wave. The ρ flow density function case.

Consider Figure 4.6, where we have plotted the trajectories of two vehicles running in close times in the neighborhood of the shock wave discontinuity. Using the seen theorem, from the figure we get

$$\frac{s}{\cos\gamma_1} = \frac{\Delta x_1}{\sin(\alpha - \gamma_1)} \tag{4.42}$$

$$\frac{s}{\cos\gamma_2} = \frac{\Delta x_2}{\sin(\alpha - \gamma_2)}, \tag{4.43}$$

that leads to

$$\begin{aligned}
\frac{\Delta x_1}{\Delta x_2} &= \frac{\cos\gamma_2 \sin(\alpha-\gamma_1)}{\cos\gamma_1 \sin(\alpha-\gamma_2)} && (4.44)\\
&= \frac{\cos\gamma_2(\sin\alpha\cos\gamma_1 - \cos\alpha\sin\gamma_1)}{\cos\gamma_1(\sin\alpha\cos\gamma_2 - \cos\alpha\sin\gamma_2)} && (4.45)\\
&= \frac{\tan\alpha - \tan\gamma_1}{\tan\alpha - \tan\gamma_2} = \frac{u_w - u_1}{u_w - u_2}, && (4.46)
\end{aligned}$$

where $u_w = \tan\alpha$ is the shock wave speed and $u_1 = \tan\gamma_1$ and $u_2 = \tan\gamma_2$ are the vehicle speeds before and after the shock wave discontinuity.

Since the flow circulating between both vehicle trajectories must be identical before and after crossing the shock wave, we must have:

$$\rho_1\Delta x_1 = \rho_2\Delta x_2 \quad \Leftrightarrow \quad \frac{\Delta x_1}{\Delta x_2} = \frac{\rho_2}{\rho_1}, \tag{4.47}$$

and then from (4.46) we get

$$u_w = \frac{\rho_2 u_2 - \rho_1 u_1}{\rho_2 - \rho_1} = \frac{q_2 - q_1}{\rho_2 - \rho_1}. \tag{4.48}$$

If now we consider the relationship (4.19), we get

$$\begin{aligned}
u_w &= \frac{q_2 - q_1}{\rho_2 - \rho_1} && (4.49)\\
&= \frac{u_0\left(\rho_2 - \frac{\rho_2^2}{\rho_j}\right) - u_0\left(\rho_1 - \frac{\rho_1^2}{\rho_j}\right)}{\rho_2 - \rho_1} && (4.50)\\
&= \frac{u_0\left(\rho_2 - \rho_1 - \frac{(\rho_2 - \rho_1)(\rho_2 + \rho_1)}{\rho_j}\right)}{\rho_2 - \rho_1} && (4.51)\\
&= \frac{u_0}{\rho_j}(\rho_j - \rho_1 - \rho_2), && (4.52)
\end{aligned}$$

that in dimensionless form becomes

$$u_w^* = 1 - \rho_1^* - \rho_2^* \tag{4.53}$$

and if $\rho_1 = \rho_2 = \rho$ gives

$$u_w^* = 1 - 2\rho^*. \tag{4.54}$$

Differential equation in terms of $q^*(x,t)$

In the previous paragraph we have eliminated $q(x,t)$ to obtain a differential equation for $\rho(x,t)$. In this section we eliminate $\rho(x,t)$ to obtain a differential equation for $q(x,t)$, but in order to illustrate its advantages, we use a dimensionless form.

Since

$$q^* = \rho^*(1-\rho^*), \tag{4.55}$$

solving for ρ^* we obtain

$$\rho^* = G^*(q^*) = \frac{1 \pm \sqrt{1-4q^*}}{2}, \tag{4.56}$$

and taking partial derivatives in (4.56) with respect to t^* we get

$$\frac{\partial \rho^*}{\partial t^*} = G^{*\prime}(q^*)\frac{\partial q^*}{\partial t^*} = (1-4q^*)^{-1/2}\frac{\partial q^*}{\partial t^*}, \tag{4.57}$$

and substituting (4.57) into (4.34) we get the differential equation in terms of $q^*(x^*, t^*)$

$$\frac{\partial q^*}{\partial x^*} + G^{*\prime}(q^*(x^*, t^*))\frac{\partial q^*}{\partial t^*} = \frac{\partial q^*}{\partial x^*} \pm (1-4q^*)^{-1/2}\frac{\partial q^*}{\partial t^*} = 0.$$

The family of characteristic lines passing throughout the points $(0, t_0)$ are:

$$t^* = G^{*\prime}(q^*(0, t_0^*))x^* + t_0^* = \pm(1-4q^*(0, t_0^*))^{-1/2}x^* + t_0^*. \tag{4.58}$$

The intersection of two infinitely close characteristic lines neglecting infinitesimal terms is given by

$$\begin{aligned} t^* &= G^{*\prime}(q^*(t_0^*))x^* + t_0^* \\ &= G^{*\prime}(q^*(t_0^* + \Delta t_0^*))x^* + t_0^* + \Delta t_0^* \\ &= G^{*\prime}(q^*(t_0^*) + \Delta t_0^*(q^*)'(t_0^*))x^* + t_0^* + \Delta t_0^* \\ &= (G^{*\prime}(q^*(t_0^*)) + \Delta t_0^*(q^*)'(t_0^*)G^{*\prime\prime}(q^*(t_0^*))x^* + t_0^* + \Delta t_0^*, \end{aligned} \tag{4.59}$$

that leads to

$$x^* = -\frac{1}{(q^*)'(t_0^*)G^{*\prime\prime}(q^*(t_0^*))}, \tag{4.60}$$

$$t^* = t_0 - \frac{G^{*\prime}(q^*(t_0^*))}{(q^*)'(t_0^*)G^{*\prime\prime}(q^*(t_0^*))}, \tag{4.61}$$

that are the parametric equations of the intersection curve.

Shock wave (q^* equation)

In this section we derive the propagation speed of a shock wave of the q^* differential equation.

Figure 4.7 shows the trajectories of two vehicles running in close times in the neighborhood of the shock wave discontinuity. Note that in this case, the x and t axes have been exchanged with respect to those in subsection 4.4.1. This means that the angles γ_i and γ_i' for $i = 1, 2$, on one side, and α and α', on the other, are complementary (add up to $\pi/2$).

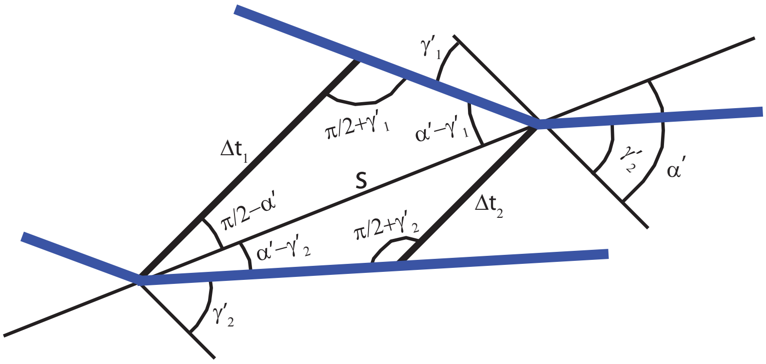

Figure 4.7: Two close trajectories in the neighborhood of a shock wave. The q^* flow intensity function case.

Similarly, we denote ${u^*}_w^{-1} = 1/u_w^*$, ${u^*}_1^{-1} = 1/u_1^*$ and ${u^*}_2^{-1} = 1/u_2^*$, to the dimensionless velocity inverses.

Using the sine theorem, from the figure we get

$$\frac{s}{\cos\gamma_1'} = \frac{\Delta t_1^*}{\sin(\alpha' - \gamma_1')} \tag{4.62}$$

$$\frac{s}{\cos\gamma_2'} = \frac{\Delta t_2^*}{\sin(\alpha' - \gamma_2')}, \tag{4.63}$$

that leads to

$$\frac{\Delta t_1^*}{\Delta t_2^*} = \frac{\cos\gamma_2' \sin(\alpha' - \gamma_1')}{\cos\gamma_1' \sin(\alpha' - \gamma_2')} \tag{4.64}$$

$$= \frac{\cos\gamma_2'(\sin\alpha' \cos\gamma_1' - \cos\alpha' \sin\gamma_1')}{\cos\gamma_1'(\sin\alpha' \cos\gamma_2' - \cos\alpha' \sin\gamma_2')} \tag{4.65}$$

$$= \frac{\tan\alpha' - \tan\gamma_1'}{\tan\alpha' - \tan\gamma_2'} = \frac{{u^*}'_w - {u^*}'_1}{{u^*}'_w - {u^*}'_2}, \tag{4.66}$$

where, as indicated, ${u^*}_w^{-1} = \tan\alpha'$ is the inverse of the shock wave speed and ${u^*}_1^{-1} = \tan\gamma_1'$ and ${u^*}_2^{-1} = \tan\gamma_2'$ are the inverses of the vehicle speeds before and after the shock wave discontinuity.

Denoting by q_1^* and q_2^* to the flow intensities before and after the shock wave, in this case, we have:

$$q_1^* \Delta t_1^* = q_2^* \Delta t_2^* \quad \Leftrightarrow \quad \frac{\Delta t_1^*}{\Delta t_2^*} = \frac{q_2^*}{q_1^*}, \tag{4.67}$$

and then similar to (4.66) we get

$$\frac{\Delta t_1^*}{\Delta t_2^*} = \frac{{u^*}_w^{-1} - {u^*}_1^{-1}}{{u^*}_w^{-1} - {u^*}_2^{-1}}$$

and then

$$u^*{}_w^{-1} = \frac{q_2^* u^*{}_2^{-1} - q_1^* u^*{}_1^{-1}}{q_2^* - q_1^*} = \frac{\rho_2^* - \rho_1^*}{q_2^* - q_1^*}. \tag{4.68}$$

If now we consider the relationship (4.56), we get

$$u^*{}_w^{-1}(q_1^*, q_2^*) = \frac{\rho_2^* - \rho_1^*}{q_2^* - q_1^*} = \frac{\pm\sqrt{1-4q_1^*} \mp \sqrt{1-4q_2^*}}{2(q_2^* - q_1^*)}, \tag{4.69}$$

where we have indicated explicitly that $u^*{}_w^{-1}$ depends on q_1^* and q_2^*.

If $q_2^*, q_1^* \to q^*$ we get $u^*{}_w^{-1}(q^*, q^*) = \dfrac{1}{\sqrt{1-4q^*}}$.

Conclusion

In addition, this model has some limitations, such as: (a) the shock waves lead to discontinuous wave profiles, which do not seem very realistic in practice, (b) the assumed relations between velocity and density are not adequate to describe nonequilibrium situations, (c) the interactions among vehicles is not adequately taken into consideration, and many others. These limitations motivated important modifications proposed by several authors, leading to complex models that involved well known equations, such as the Burgers equation (see Whitham (1974)), the Boltzmann equation (see Prigogine and Herman (1971) or Paveri Fontana (1975)) and the Navier-Stokes like equations (see Stokes (1845)). These modifications include additional terms to consider important traffic effects, such as: (a) the acceleration and (b) the interaction of vehicles, (c) a dynamic equation for the average velocity, (d) a convection term to describe velocity changes caused by average vehicle motion, (e) an anticipation term to consider drivers awareness of the traffic conditions ahead, (f) a viscosity term to smooth out sudden density and velocity changes, (g) a relaxation term, (h) an interaction term to describe the deceleration of vehicles due to next vehicle ahead, etc.

In conclusion, very complex models are required to reproduce the real and complicated traffic flow behavior but unfortunately, there are no analytical solutions for the general Navier-Stokes equation and only in very special cases this solution is known. Consequently, either simplified versions of the Navier-Stokes equation are used to deal with traffic problems, or complicated numerical methods that are not practical for traffic networks need to be used.

4.4.2 M-N Model (Merchant-Nemhauser)

Merchant and Nemhauser (1978a,b) proposes the following model, based on an exit-flow function that determines the share of the number of users leaving the link during a time interval. It can be expressed as:

$$x_{i+1} = x_i + e_i - g_i \tag{4.70}$$

$$g_i \;=\; f(x_i). \tag{4.71}$$

Equation (4.70) forces the flow conservation. Equation (4.71) models the flow behavior (exit flow approach), being $f(x_i)$ a nondecreasing and concave function to model congestion. In addition, $x_i \geq f(x_i)$, that is, the number of users leaving the link cannot exceed the number of users inside the link. We can use the function $f(x) = min\{\gamma_m x, q_m\}$, being q_m the maximum flow and the function $\gamma_m < 1$. Note that g_i is bounded from above by the maximum flow that can be discharged during a given interval of time Δt.

We have included this model because the first assignment models used this loading approach. The disadvantage of this model is the difficulty in defining the function γ_m. For example, if we assume uniform flow distribution, γ_m will be 1.

4.4.3 Delay-Function Model

The delay function model is based on choosing a form of the delay function $\tau(t)$ which is FIFO-consistent ($\dot{\tau}(t) > -1$). Whether the delay function is a linear one like $\tau(t) = k_1 + k_2 x(t)$, where k_1 and k_2 are positive constants, which always holds the FIFO condition, the model can be defined as follows:

$$\dot{x}_a(t) \;=\; e_a(t) - g_a(t) \tag{4.72}$$

$$\tau_a(t) \;=\; \alpha_a + q_m x_a(t) \tag{4.73}$$

$$g_a(t + \tau_a(t)) \;=\; \frac{e_a(t)}{1 + \dot{\tau}_a(t)}. \tag{4.74}$$

Equation (4.72) is the flow conservation, Equation (4.73) models the link flow behavior (travel time approach) and Equation (4.74), the flow propagation.

This model is difficult to calibrate when realistic results are required.

4.4.4 Point and Physical Queue Models

- *The point queue model* is based on the following assumptions: (i) the vehicles have zero length; (ii) the link traversal speed is the free flow speed; (iii) the vehicles wait for exiting if the link exit-capacity is exceeded. There exist tree general formulations, namely, (a) the *exit flow function approach*, where the inflow rates and occupancies are given and the outflow rates are determined by the flow conservation condition and then, the travel time is calculated using the flow propagation condition; (b) the *travel time function approach*, where we determine the travel times based on the flow conservation and the travel times functions, given the link inflows and occupancies and then, the outflow rate can be obtained by flow propagation condition; and (c) the *mixed approach*, where the inflow rate and occupancy are known, also we require both predefined travel time functions and exit flow functions, but the travel times and the outflow rates are determined separately by their corresponding equation and the flow conservation condition, without satisfying the flow propagation condition.

<table>
<tr><td colspan="3">APPROACHES</td></tr>
<tr><td colspan="3"><b>Point Queue Model</b></td></tr>
<tr><td><i>Exit flow function approach</i></td><td><i>Travel time function approach</i></td><td><i>Mixed approach</i></td></tr>
<tr><td>Flow conservation
$g(t) = f(\cdot)$
Flow propagation</td><td>Flow conservation
$\tau(t) = f(\cdot)$
Flow propagation</td><td>Flow conservation
$g(t) = f(\cdot)$
$\tau(t) = h(\cdot)$</td></tr>
<tr><td colspan="3"><b>Physical Queue Model</b></td></tr>
<tr><td><i>Exit flow function approach</i></td><td colspan="2"><i>Combined approach</i></td></tr>
<tr><td rowspan="3">Flow conservation
$g(t) = f(x_{max})$
Flow propagation
Hydrodynamic relationship</td><td colspan="2">Flow conservation</td></tr>
<tr><td>running segment
$\tau(t) = f(\cdot)$</td><td>queuing segment
$g(t) = f(x_{max})$</td></tr>
<tr><td colspan="2">Flow propagation</td></tr>
</table>

Table 4.3: Different approaches in the point and physical queue models

- *The physical queue model* considers the vehicle lengths, whereupon it let us capture junction blockage and queue spillback[2] in a network model. There exist two general formulations, namely, (a) the *exit flow function approach*, which is similar to the exposed exit flow function approach, but this one uses the hydrodynamic wave model of traffic flow, considering the storage capacity (x_{max}) in the exit flow function to capture the effects of physical queues; and (b) the *combined approach*, 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 but does not consider shockwaves), combining flow conservation and propagation conditions.

Table 4.3 shows a summary of the different types of approaches in the point and physical queue models.

For a better understanding, we show an example of the point queue model evaluated with the exit flow function approach. The formulation used, whose conditions have been explained before, can be expressed as follows (see Nie and Zhang (2005a)):

$$\frac{d\Lambda(t)}{dt} = \begin{cases} 0 & \text{if } \Lambda(t) = 0 \quad \text{and} \quad e(t-\alpha_a) < q_m \\ e(t-\alpha_a) - q_m & \text{otherwise} \end{cases} \tag{4.75}$$

$$g(t) = \begin{cases} e(t-\alpha_a) & \text{if } \Lambda(t) = 0 \quad \text{and} \quad e(t-\alpha_a) < q_m \\ q_m & \text{otherwise} \end{cases} \tag{4.76}$$

$$\tau(t) = \alpha_a + \frac{\Lambda(t+\alpha_a)}{q_m}, \tag{4.77}$$

[2] spillback refers to the end of queue spilling backwards in the network.

where Λ is the total number of queued vehicles at the exit node, α_a and q_m represent the free-flow travel time and bottleneck capacity respectively. Equations (4.75) and (4.77) are the flow conservation and propagation conditions, and the Equation (4.76) means the flow behavior expressed by an exit flow function.

Example 2 (Four different cases of the point queue model) *We describe an example of a piece-wise constant density which reaches other different piece-wise constant densities for the cases:*

(*a*) *A group of vehicles with a density of* 25% *of the jam density that reach other group with a density of* 60% *of the jam density.*

(*b*) *A group of vehicles with a density of* 25% *of the jam density that reach other group with a density of the jam density.*

(*c*) *A group of vehicles with a density of* 60% *of the jam density that reach other group with a density of* 25% *of the jam density.*

(*d*) *A group of vehicles with a density of* 100% *of the jam density that reach other group with a density of* 25% *of the jam density.*

We assume a one-way road whose free speed is 100 *km/h, with* $q_m = 1600$ *vh/h and* $\rho_j = 65$ *vh/km. As this model offers two degrees of freedom: the maximum flow and the free flow speed (we must divide the link in S time ticks that are necessary to cover the link with a free flow speed), we choose the next values:* 50 *segments S of* 1 *km and the clock tick value of* 36 *sec. and the maximum flow is* 16 *vehicles/tick. Note that a user needs* 50 *ticks to cover the total link length (*50 *km) at* 100 *km/h. The piece-wise densities, which are applied on ten ticks each one, are the following:*

ρ^*	0.25	0.60	1.00
users/tick	16.25	39.00	65.00

Figure 4.8 shows the evolution of the number of vehicles in the link over time for each case. We consider three different bottleneck capacities, i.e., the blue line depicts a bottleneck capacity of $q_m = 800$ *vh/h, the red line,* $q_m = 1600$ *vh/h and the green line, an infinite capacity. At time* $= 0$ *there are no users in the link. With time, the total amount of vehicles is inside the link, for example, in case* (*b*) *there are*

$$10\,ticks\,\times\,16.25\,users/tick\,+\,10\,ticks\,\times\,65\,users/tick\,=\,812.5\,users.$$

In all cases, the first user leaves the link in 50 *ticks, the free flow travel time. However, last users requiere different times. Hence, the queue effect appears. We can also see that*

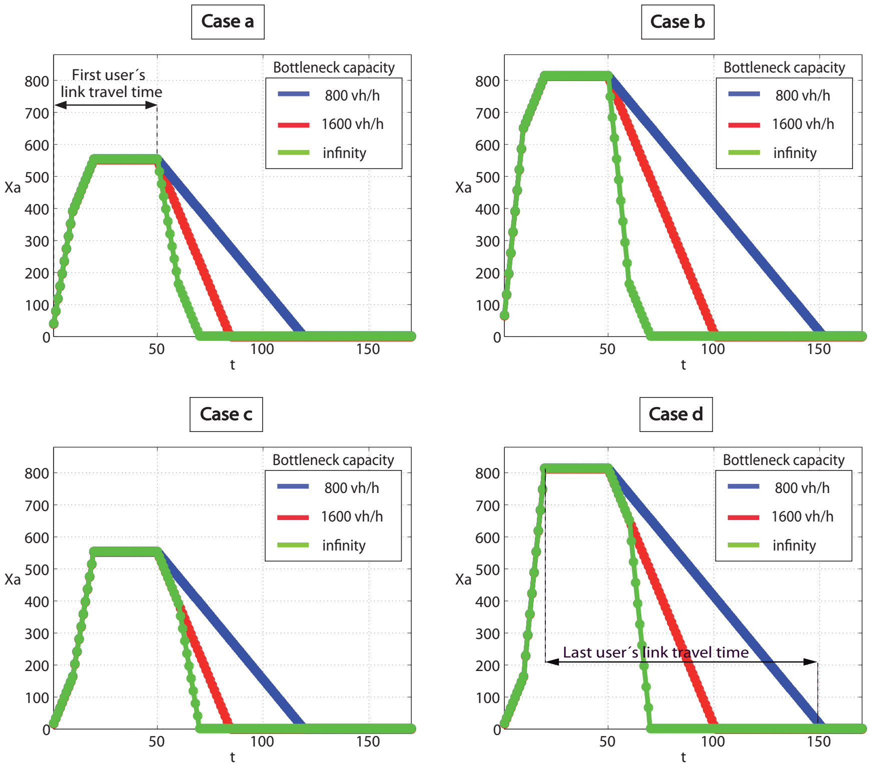

Figure 4.8: Evolution of the number of vehicles in the link over time (PQ model). Cases *a*, *b*, *c* and *d*

when the bottleneck capacity is unlimited, the last users link travel time is the free flow travel time. This implies that the link traversal speed is the free flow speed. Finally, we point out that the negative slice of the curve is always the same for the cases (*a*), (*b*), (*c*) *and* (*d*) *and it relies on the* q_m *value.*

Finally, Table 4.4 shows a comparative study of the solution properties of the point queue and the physical queue model, according to Szeto and Lo (2005).

4.4.5 Cell Transmission Model

The cell transmission model (CTM) is a discrete model that reproduces hydrodynamic flow models and permits predicting traffic evolution over time and space including transient phenomena, such as the building, propagation and dissipation of queues. It was first defined by Daganzo.

The CTM predicts macroscopic traffic behavior on a given corridor by evaluating the flow and density at a finite number of intermediate points (cells) and at a finite number of different time steps, such that relationship tick/cell is the free flow speed.

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 if 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 SDUO 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 flow functions	Possibly discontinuous. Possibly non-differentiable
Continuity of OD 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 links flows.	

Table 4.4: Comparative study of the solution properties of the point queue and the physical queue model.

Let $n_i(t)$, $N_i(t)$, $Q_i(t)$ and $y_i(t)$ be the number of vehicles in cell i at time t, the maximum number of vehicles that can be inside cell i (cell capacity), the maximum flow in cell i and the number of vehicles that coming from cell $i-1$ can enter cell i during the time interval $(t, t+1)$, respectively. Then, we have

$$n_i(t+1) = n_i(t) + y_i(t) - y_{i+1}(t), \tag{4.78}$$

that expresses the number of vehicles $n_i(t+1)$ in cell i at time $t+1$ as the number of vehicles in cell i at a previous time t plus the number of vehicles $y_i(t)$ entering cell i minus the number of vehicles $y_{i+1}(t)$ leaving this cell, all at time t.

According to Daganzo (1992, 1995b), the number of vehicles entering cell i is the minimum of three quantities, as follows:

$$y_i(t) = \min(n_{i-1}(t), Q_i(t), N_i(t) - n_i(t)), \tag{4.79}$$

that expresses three possible limits to $y_i(t)$: (a) the number of vehicles in cell $i-1$, (b) the maximum possible flow of cell i and (c) the cell i capacity, respectively.

Considering the hydrodynamic model equivalence, the last relationship can be expressed as:

$$q(x,t) = \min\left(u\rho(x,t), q_m, u(\rho_j - \rho(x,t))\right); \quad 0 \leq \rho(x,t) \leq \rho_j. \tag{4.80}$$

The occupancy restriction $y_i \leq N_i(t) - n_i(t)$ is conservative because it assumes that no vehicle leaves cell i at time t. This is equivalent to assuming that density waves propagate backwards at the free flow speed[3] when waves move much slower. In order to improve the model, Daganzo includes the relationship w/v (wave shock velocity/vehicular speed), changing the manner in which vehicles approach the bottleneck and the location of queues. Thus, equation (4.79) can be rewritten as follows:

$$y_i(t) = \min\{n_{i-1}(t), Q_i(t), (w/u)(N_i(t) - n_i(t))\}. \tag{4.81}$$

Example 3 (Four different cases of the cell transmission model) *Figures 4.9, 4.10, 4.11 and 4.12 represent the same four cases described in Example 2 but solved by the cell transmission model, showing the density evolution with time.*

We assume the previous example values of free speed, density and flow. As this model offers four degrees of freedom, the free flow speed (we must choose an adequate relationship between cell length and tick time), the maximum flow, the jam density and the wave speed, we choose the following values: the cell length is 1 *km, the clock tick value (cell travel time)* 36 *sec., the maximum flow is* 16 *vehicles/tick, the cell jam density is* 65 *vehicles/cell and the ratio* $w/v = 1$*.*

At time $= 0$ *there are no users in the link. With time, vehicles move forward conditioned by the maximum flow (*$Q^* = 0,246$*). In all cases, the first user leaves the link in* 50 *ticks, the free flow travel time. However, last users requiere different times. In cases* (a) *and* (c) *they need* 85 *ticks while in cases* (b) *and* (c)*, they need* 101 *ticks.*

In order to illustrate the factor w/v *effect, we add the case* (a) *considering the relationship* $w/v = 0.20$ *(see figure 4.13). Note how the forehead of the piece-wise constant density goes back, meanwhile for the* $w/v = 1$ *case this effect does not happen. Note that the traversal time increases with respect to the case of* $w/v = 1$*.*

Table 4.5 shows the required time to enter and leave the link, and the link travel time for the case (a) *with different* w/v *values. This table shows how the required time to leave the link decreases when the ratio* w/v *increases. We point out that the link traversal speed is the free flow speed.*

[3] If we assume $N_i(t)$ instead of $N_i(t) - n_i(t)$, it implies an instant download capability. However, it is assumed a download capability of n_i vehicles with a delay of a tick (note that the tick is fixed by mean of the free flow speed).

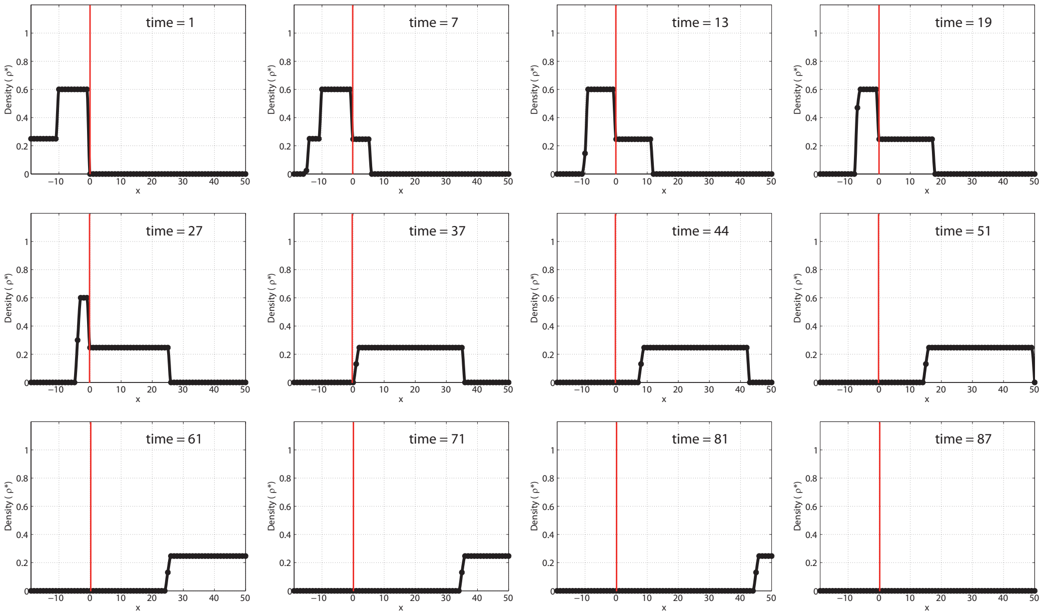

Figure 4.9: Time evolution of flow density (CTM). Case (a): $\rho_1^* = 0.60$ and $\rho_2^* = 0.25$.

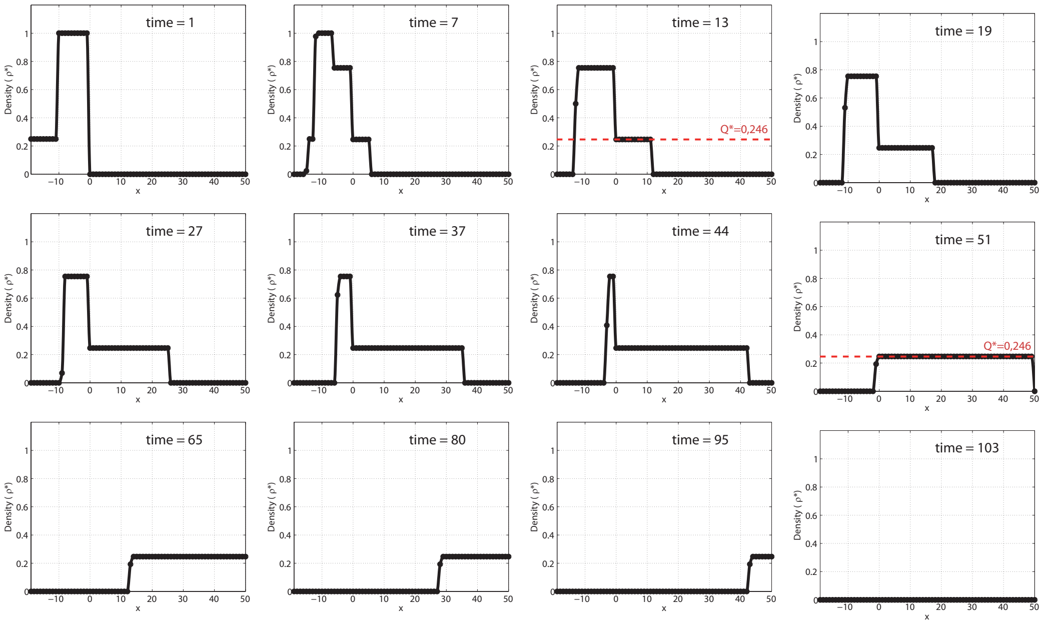

Figure 4.10: Time evolution of flow density (CTM). Case (b): $\rho_1^* = 1.00$ and $\rho_2^* = 0.25$.

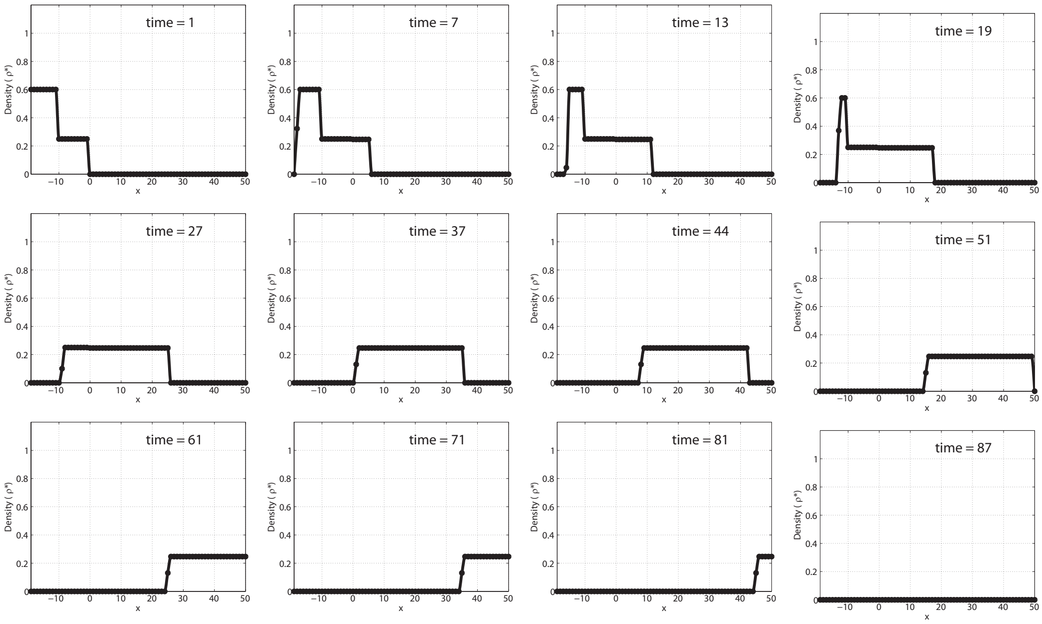

Figure 4.11: Time evolution of flow density (CTM). Case (c): $\rho_1^* = 0.25$ and $\rho_2^* = 0.60$.

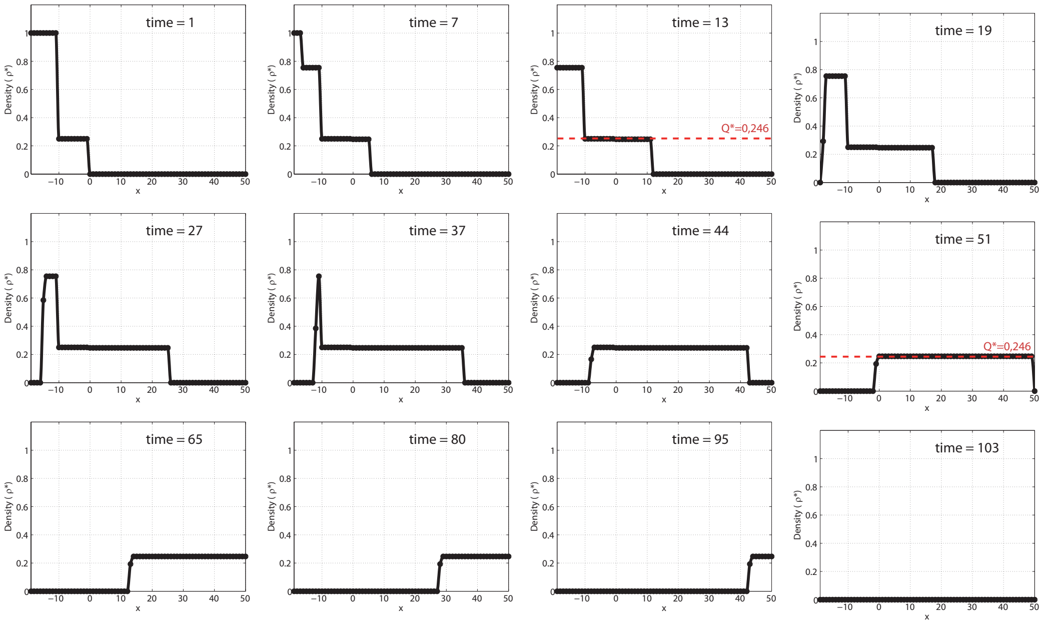

Figure 4.12: Time evolution of flow density (CTM). Case (d): $\rho_1^* = 0.25$ and $\rho_2^* = 1.00$.

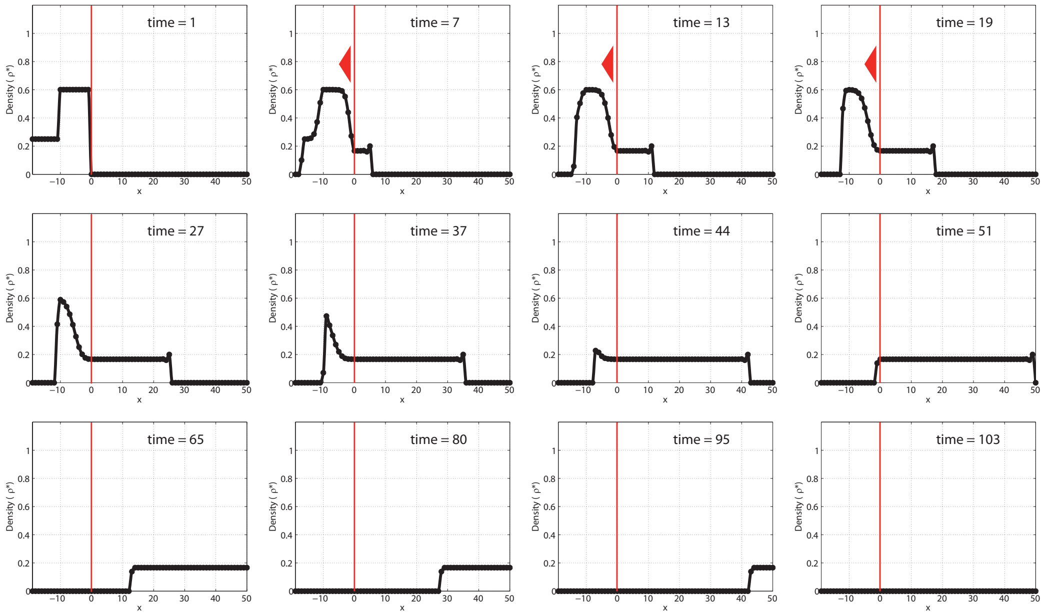

Figure 4.13: Time evolution of flow density (CTM). Case (a): $\rho_1^* = 0.60$ and $\rho_2^* = 0.25$, with $w/v = 0.20$.

	Link length = 50 Km				
w/v	0.10	0.20	0.50	0.85	1.00
Time to enter to the link	95	52	36	35	35
Time to leave the link	145	102	86	85	85
Link travel time	50	50	50	50	50

Table 4.5: Travel time (ticks) by Daganzo for case a with different w/v values.

4.5 Assignment Strategy

This part of the DTA problems deals with the travel choice, i.e., how the travelers select their departures times and routes. The difference with the STA problems is that, whereas in STA models the travel times are static (no time-dependent), in the DTA problems the travel costs depend on the departure time.

First of all, we discuss two concepts that commonly appears in the traffic literature, namely, dynamic user equilibrium (DUE) and dynamic user-optimal (DUO):

The dynamic user equilibrium is a temporal generalization of the static equilibrium assignment problem (UE) (see Section 3.3), i.e., at equilibrium, only the routes with minimal (actual o perceived) travel cost have flow. However, in the dynamic analysis we have: (i) the route choice and (ii) the departure time choice. The first authors dealing with DUE reach equilibrium only considering the route choice in a time-dependent network. For example, Merchant and Nemhauser (1978a,b) and Carey and Revelli (1986) minimize the cost over a specified planning horizon, and Janson (1991) studies the equilibrium in successive time intervals of 10-15 minutes and considers additional constrains to insure continuous flows. Later, papers such as Friesz et al. (1993b) or Ran et al. (1996) consider both the departure time choice and route choice in a simultaneous route-departure equilibrium.

The dynamic user-optimal approach assumes that users choose the least time paths, however, due to the dynamic nature of the problem, no equilibrium is reached. It is based on the Optimally Principle that states that the shortest path for a to b through c must be comprised of the shortest paths from a to c and c to b[4]. There exists three dominance criteria considered in the determination of a shortest path:

- *Deterministically Pareto optimal.* For a given departure time, there is one path (or several paths with the same travel cost associated) such that the other existing paths have zero-probability for being shortest.

  In this case, the Optimally Principle is guaranteed iff, for the departure times s and t such that $s \leq t$,
  $$s + c_{ij}(s) \leq t + c_{ij}(t), \tag{4.82}$$
  where $c_{ij}(t)$ is the travel cost to go from the initial node i to the final node j associated with the departure time t.

- *Stochastically Pareto optimal.* For a given departure time, there is one path (or several paths with the same travel cost associated) such that the other existing paths have smaller probability for being shortest.

[4] The Optimally Principle always holds when the travel costs are independent. However, to guarantee the principle the rest of the cases we have to add more conditions.

In this case, the Optimally Principle is guaranteed iff, for the departure times s and t such that $s \leq t$,

$$Pr(s + c_{ij}(s) \leq z) \leq Pr(t + c_{ij}(t) \leq z). \tag{4.83}$$

In other words, the probability of arriving by any given time z cannot be increased by leaving later. This condition is defined as *stochastic consistency* by Wellman et al. (1995).

- *Expecting value (EV) dominance.* For all departure times, there is one path (or several paths with the same travel cost associated) which has the lowest expected time.

  Miller-Hooks and Mahmassani (2000) define the following EV-dominance criteria and extend the Optimally Principle as follows: Let $\phi_{ijr}(t)$ be the random travel time for the rth path from node i to the destination node j at departure time interval t. For each $t \in T$, $E[\phi_{ir}(t)]$ is the path with the minimum expected time. Then, a path r is nondominated iff no path l exists such that

$$E[\phi_{ijl}(t)] \leq E[\phi_{ijr}(t)] \qquad \forall t \in T \tag{4.84}$$

$$\text{and} \qquad \exists t \in T | E[\phi_{ijl}(t)] < E[\phi_{ijr}(t)]; \tag{4.85}$$

  otherwise, the path r is dominated. Therefore, all subpaths of a nondominated path with the same destination node as this path must themselves be nondominated.

Once these concepts have been exposed, we present the DTA models divided in deterministic or stochastic ones.

4.5.1 Deterministic Models

In this part we introduce three different ways to treat the deterministic DTA, namely, the mathematical program-based, the variational inequality-based and the optimal control-based.

Previously, we define the dynamic user equilibrium (DUE) state according to the Wardrop principle as:

> *"The travel costs incurred by traffic on all routes entered by traffic during the same time interval are equal or less than those that would be on any unused route at that time interval."*

Mathematically, it can be expressed as:

$$h_{pqr}(t)(c_{pqr}(t) - \pi_{pq}(t)) = 0, \qquad \forall p, q \in D, \forall r \in R_{pq}, \forall t, \tag{4.86}$$

$$c_{pqr}(t) - \pi_{pq}(t) \geq 0, \qquad \forall p, q \in D, \forall r \in R_{pq}, \forall t, \tag{4.87}$$

where R_{pq} is the set of routes of OD pair (p, q), $h_{pqr}(t)$ the flow on route $r \in R_{pq}$ at time t, $c_{pqr}(t)$ is the cost associated with route r with origin-destination (p, q) at time t and $\pi_{pq}(t)$ is the equilibrium travel cost of any route of OD pair (p, q) at time t.

The mathematical program-based formulation

Janson (1991) proposed the following formulation by developing the Beckmann et al. (1956) formulation:

$$\underset{\mathbf{h},\mathbf{v}}{\text{Minimize}}\, Z(\mathbf{v}) = \int_t \sum_{a \in A} \int_0^{v_a(t)} c_a(w) dw dt, \tag{4.88}$$

subject to:

$$\sum_{r \in R_{pq}} h_{pqr}(t) = d_{pq}(t), \qquad \forall (p,q) \in D, \forall t \tag{4.89}$$

$$v_a(t) = \int_s \sum_{(p,q) \in D} \sum_{r \in R_{p,q}} h_{pqr}(s) \delta^t_{apqr}(s) ds, \qquad \forall a \in A, \forall t \tag{4.90}$$

$$h_{pqr}(t) \geq 0, \qquad \forall (p,q) \in D, \forall t \tag{4.91}$$

with

$$\delta^t_{apqr}(s) = \begin{cases} 1, & \text{if traffic on route } r \text{ of OD } (p,q) \text{ departing at time } s \text{ is} \\ & \text{present on link } a \text{ at time } t; \\ 0, & \text{otherwise,} \end{cases}$$

where $\mathcal{A}$ is the set of links of a connected transportation network with set of nodes $\mathcal{N}$, $\mathcal{D}$ is a subset of origin-destination pairs (p,q) and $c_a(t)$ is the cost associated with link a at time t. There are given positive demands $d_{pq}(t)$ for origin-destination flows -variable over time- which give rise to a time dependent link flow pattern $v_a(t)$ when distributed through the network.

This formulation has been adopted in much of the literature such as Ran et al. (1996) or Janson and Robles (1995). Han (2006) show how the Beckman *dynamic* model is not equivalent to the Wardrop *dynamic* principle.

The variational inequality (VI)-based approach

According Smith (1979), the *dynamic* Wardrop principle can be expressed by a variational inequality. We consider the path inflow vector $\mathbf{h}$ such that:

$$\sum_{r \in R_{p,q}} h_{pqr}(t) = d_{pq}(t), \qquad \forall (p,q) \in D, \forall t \tag{4.92}$$

$$h_{pqr}(t) \geq 0, \qquad \forall (p,q) \in D, \forall r \in R_{pq}, \forall t \tag{4.93}$$

then, the assignment at time t, expressed in form of a column vector of route inflows $\mathbf{h}^*(t) \in H(t)$, is an equilibrium if and only if:

$$-\left[\mathbf{h} - \mathbf{h}^*(t)\right]^T \cdot \mathbf{c}(t) \leq 0 \quad \forall \mathbf{h} \in H(t), \tag{4.94}$$

where $\mathbf{c}(t)$ is the column vector of route costs incurred by travelers departing at time t and $H(t)$ denotes the corresponding set of all route flows at time t.

From the DUE state we know that the flow entering a route r at time t would be greater than zero only if the travel cost on the route is equal to the minimum travel cost at that time. On the other hand, due to the fact that the cost vector is assumed non-negative, the equilibrium solution $\mathbf{h}^*(t)$ at time t can be obtained by maximizing Equation 4.94, i.e.,

$$Z_{VI} = Arg \underset{\mathbf{h}^* \in H(t)}{\text{Min}} \left[\underset{\mathbf{h} \in H(t)}{\text{Max}} \left(-[\mathbf{h} - \mathbf{h}^*(t)]^T \cdot \mathbf{c}(t) \right) \right]. \tag{4.95}$$

Therefore, we can write the dynamic user equilibrium assignment problem over time in the form of a dynamic programme as:

$$\underset{\mathbf{h}^*}{\text{Min}} \int_t \underset{\mathbf{h} \in H(t)}{\text{Max}} \left(-[\mathbf{h} - \mathbf{h}^*(t)]^T \cdot \mathbf{c}(t) \right) dt \tag{4.96}$$

subject to

$$\mathbf{h}^*(t) \in H(t), \quad \forall t. \tag{4.97}$$

We note that at each time t there is no contribution from future flows in the integration because the value of the integrand will be zero at equilibrium. This means that we can find optimal solutions at time t without knowledge of the inflows at future times.

The existence of solution requires (i) $\mathbf{c}(t)$ to be a continuous function of $\mathbf{h}^*$ and (ii) $\mathbf{h}^*$ to be a non-empty compact convex set[5].

The uniqueness of solution requires $\mathbf{c}(t)$ to be strictly monotone[6].

This approach has been developed by Smith (1993) who formulates a route-based dynamic route choice problem or Friesz et al. (1993b) who raises a route-based departure time and a route choice problem.

The optimal control-based approach

The optimal control-based approach consists on determining control strategies that cause a process to satisfy the physical constraints while at the same time minimize or maximize performance criterion. Using the optimal control theory, Ran et al. (1993) propose an instantaneous dynamic user optimal (DUO) traffic assignment problem, i.e, determining the vehicle flows at each instant of time of on each link that result from drivers using minimal-time routes under the currently prevailing travel times. They assume that the time-dependent origin-destination trip pattern is known a priori, and the departure times of travelers are given (they do not study the optimal choice of departure times).

They define the dynamic user optimal equilibrium (DUO) as follows:

[5] Given a variational inequality problem $VI(F, K)$, Theorem 1.4 in Nagurney (1993) states that if K is a compact convex set and $F(x)$ is continuous on K, then the V.I. problem admits at least one solution x^*.

[6] Given a variational inequality problem $VI(F, K)$, Theorem 1.6 in Nagurney (1993) states that if $F(x)$ is strictly monotone on K, then the solution of the V.I. is unique, if one exists.

Definition 4 (DUO) *If for each OD pair at each decision node at each instant of time, the travel times for all routes that are being used equal the minimal instantaneous route travel time, the dynamic traffic flow over the network is in a dynamic user-optimal state.*

The "instantaneous travel time" means the travel time that is incurred if traffic conditions on the link remain unchanged while traversing the link.

The instantaneous travel time can be expressed by a sum of an instantaneous flow-dependent running time $f_{1a}(\cdot)$ over link a and an instantaneous queuing delay $f_{2a}(\cdot)$:

$$c_a\left[x_a(t), e_a(t), g_a(t)\right] = f_{1a}\left[x_a(t), e_a(t)\right] + f_{2a}\left[x_a(t), g_a(t)\right], \tag{4.98}$$

being $x_a(t)$ the number of users traveling on link a at time t, and $e_a(t)$ and $g_a(t)$ the inflow and outflow rate to/from the link a at time t respectively. The instantaneous travel time, $c_a(\cdot)$, is assumed to be nonnegative, increasing and differentiable with respect to $x_a(t)$, $e_a(t)$ and $x_a(t)$ and $g_a(t)$, respectively.

Then, the dynamic user-optimal traffic assignment problem is formulated as follows:

$$\text{Minimize}\, I = \int_0^T \sum_{a\in A} \left\{ \int_0^{e_a(t)} f_{1a}\left[x_a(t), w\right] dw + \int_0^{g_a(t)} f_{2a}\left[x_a(t), w\right] dw \right\} dt, \tag{4.99}$$

subject to

$$\begin{aligned}
\frac{dx_{apq}}{dt} &= e_{apq}(t) - g_{apq}(t) : \lambda_{apq}(t), \qquad \forall a, p, q; && (4.100)\\
\frac{dE_{pqr}(t)}{dt} &= y_{pqr}(t) : \nu_{pqr}(t), \qquad \forall r, p, q \neq p; && (4.101)\\
\sum_{a\in A} e_{apq}(t) &= d_{pq}(t) : \sigma_{pqj}(t) \qquad \forall q, p \neq q; && (4.102)\\
\sum_{a\in\mathcal{B}(j)} g_{apq}(t) &= \sum_{a\in\mathcal{A}(j)} e_{apq}(t) : \sigma_{pqj}(t) \qquad \forall p, q, j \neq p, q; && (4.103)\\
\sum_{a\in\mathcal{B}(j)} g_{apq}(t) &= y_{pq}(t) : \sigma_{pqj}(t) \qquad \forall p, q \neq p; && (4.104)\\
x_{apqr}(t) &= \sum_{h\in\overline{r}} \left[x_{hpqr}\left(t + \overline{\tau}_a(t)\right) - x_{hpqr}(t)\right] + \left[Y_{pqr}\left(t + \overline{\tau}_a(t)\right) - Y_{pqr}(t)\right] : \mu_{apqr}(t),\\
&\qquad \forall a \in \mathcal{B}(j), p, q, r, j \neq p; && (4.105)\\
e_{apq}(t) &\geq 0 \qquad \forall a, p, q; && (4.106)\\
g_{apq}(t) &\geq 0 \qquad \forall a, p, q; && (4.107)\\
x_{apqr}(t) &\geq 0 \qquad \forall a, p, q, r; && (4.108)\\
y_{pqr}(t) &\geq 0 \qquad \forall p, q, r; && (4.109)\\
Y_{pqr}(t) &\geq 0 \qquad \forall p, q, r; && (4.110)\\
Y_{pqr}(t_0) &= 0 \qquad \forall p, q, r; && (4.111)\\
x_{apqr}(t_0) &= 0 \qquad \forall a, p, q, r; && (4.112)
\end{aligned}$$

where $Y_{pqr}(t)$ is the cumulative number of vehicles entering the network at origin p and exiting at destination q over route r at any time t; $y_{pqr}(t)$ denotes the instantaneous flows that arrives at destination q from origin p at time t. $\mathcal{A}(j)$ is the set of links whose tail nodes is j (after j), and $\mathcal{B}(j)$ is the set of links whose head nodes is j (before j); $\overline{r}$ is a section of route r from node j to destination q; and $\overline{\tau}_a$ is the link-a travel time that is estimated and updated in an iterative procedure.

The control variables are $e_{apq}(t)$, $g_{apq}(t)$ and $y_{pqr}(t)$ and Equations (4.100) and (4.101) imply the flow conservation on each link and for cumulative arrivals at each destination. Equations (4.102)–(4.104) are flow conservation constraints at each node that includes origins and destinations. The other constrains include flow propagation, nonnegativity and boundary conditions. Additional definitional constraints are:

$$\begin{aligned} \sum_{pq} e_{apq}(t) &= e_a(t), & \sum_{pq} g_{apq}(t) &= g_a(t), \\ \sum_{r} x_{apqr}(t) &= x_{apq}(t), & \sum_{pqr} x_{apqr}(t) &= x_a(t), \\ \sum_{pq} x_{apq}(t) &= x_a(t), & \sum_{r} y_{pqr}(t) &= y_{pq}(t). \end{aligned}$$

The interpretation of Equation (4.105) is that for any link $a \in \mathcal{B}(j)$, vehicles on link a using route r at any time t must result in either:

1. Added vehicles on a downstream link or links on subroute $\overline{r}$ at time $t + \overline{\tau}_a(t)$, or
2. Increased exiting vehicles at destination q at time $t + \overline{\tau}_a(t)$.

Ran et al. (1993) demonstrate that the instantaneous DUO traffic assignment problem (4.99–4.112) satisfies the instantaneous dynamic user-optimal conditions (4.86 and 4.87). Moreover, they prove that the optimal solution is applicable to any subroute, even if all links on an entire route are not in use at the same time.

On the other hand, since the objective function of the optimal control program (4.99)–(4.112) is convex to respect to the control variables[7], there is a unique optimal solution.

4.5.2 Stochastic Models

Dynamic stochastic network equilibrium-optimal approach consists in predicting of the equilibrium-optimal flows based on the stochastic inputs in a time-dependent network. These equilibrium-optimal flows will follow some statistical distributions. So, we can find three relevant groups on traffic assignment under stochastic environment depending on the kind of the stochastic inputs: (i) when the cost of the trip (generally we speak of the travel time) is the random variable[8], (ii) when the demand is stochastic, or (ii) when the road capacity is stochastic.

[7] The state variables are x_a and E_a, the control variables are e_a, g_a and y_a, and t is the independent variable.

[8] The random travel cost is interpreted as the perceived travel time, as in stochastic static choice models happen.

Stochastic travel cost

This approach makes the assumption that there are systematic errors in the travelers perception of the cost of routes. In this case, the formulation is similar to that exposed in Section 3.5, i.e., probabilistic route choice models, as the probit and logit models, are used but assuming that the travel cost, demand and flows are time-dependent.

In line with the static case, Han (2002) defines the stochastic dynamic user equilibrium (SDUE) as follows:

Definition 5 (SDUE) *At each instant no traveler believes that he or she can improve his or her perceived travel cost by unilaterally changing route.*

This principle can be expressed as:

$$P_{pqr}(t) = \frac{h_{pqr}(t)}{d_{pq}(t)} \qquad \forall (p,q) \in D, \forall r \in R_{pq}; \tag{4.113}$$

$$\sum_{r \in R_{pq}} h_{pqr}(t) = d_{pq}(t) \qquad \forall (p,q) \in D; \tag{4.114}$$

$$h_{pqr}(t) \geq 0 \qquad \forall (p,q) \in D, \tag{4.115}$$

where the choice probability is

$$P_{pqr}(t) = Prob\left[C^*_{pqr}(t) \leq C^*_{pql}(t); \quad \forall l \in \mathcal{R}_{pq} | \mathbf{C}(t)\right] \quad \forall p,q,r, \tag{4.116}$$

and $C^*_{pqr}(t)$ is the least travel cost corresponding to the route r with origin-destination p-q at instant t and it is dependent on the route cost pattern $\mathbf{C}(t)$ at this instant.

Ran et al. (1996) treat this problem using the VI method (see Section 4.5.1), i.e., if $h^*_{pqr}(t)$ are the route inflows in a equilibrium state, Equations 4.117–4.120 are necessary and sufficient conditions, assuming that the travel cost is strictly monotone w.r.t. the path flow.

$$\sum_{(pq) \in D} \sum_{r \in R_{p,q}} K_{pqr}(t)[h_{pqr}(t) - h^*_{pqr}(t)] \geq 0, \quad \forall h_{pqr} \tag{4.117}$$

subject to

$$h_{pqr}(t) \geq 0 \qquad \forall (p,q) \in D, \forall t; \tag{4.118}$$

$$\sum_{r \in R_{pq}} h_{pqr}(t) = d_{pq}(t) \qquad \forall \forall t, \tag{4.119}$$

where

$$K_{pqr}(t) = [h_{pqr}(t) - d_{pq}(t) P_{pqr}(t)] \frac{\partial C_{pqr}(t)}{\partial h_{pqr}(t)} \geq 0. \quad \forall h_{pqr} \tag{4.120}$$

Stochastic demand and capacity

This approach makes the assumption that one or both demand and capacity are stochastic. For example, Watling (2002) or Zhou and Chen (2008) assume that demand is distributed as a random variable, stationary Poisson and log-normal, respectively. Lo and Tung (2003) consider the stochastic link capacity and Zhang et al. (2011), between others, analyze the stochastic demand and capacity together.

The two relevant models on traffic assignment under this stochastic environment are the Expected value (EV) model and the Best worst-case (BW) model.

Let $\phi_{pqr}(h,\omega)$ define the travel cost between the OD-pair p-q through the route r associated with some uncertain values, as the demand $d_{pq}(\omega)$ or link capacity $q_a(\omega)$[9].

Expected value (EV) model

This model considers that the users select paths to minimize their expected travel cost, $E[\phi_{pqr}(h,\omega)]$, assuming that the expected demand is $E[d_{pq}(\omega)]$. Then, the EV model solves the equilibrium problem as follows:

$$\underset{\mathbf{h},\boldsymbol{\phi}^*}{\text{Minimize}}\, Z_{EV} = \text{Min}\begin{bmatrix} E[\phi_{pqr}(h,\omega)] - \phi^*_{pqr} \\ \delta_{pqr}h_{pqr} - E[d_{pq}(\omega)] \end{bmatrix} = 0. \tag{4.121}$$

where ϕ^*_{pqr} is the stochastic minimum travel cost and δ_{pqr} is the Dirac Delta whose value is 1 when the route r has the OD-pair p-q and zero otherwise.

The solution is $[h_{pqr}, \phi^*_{pq}]$, where h_{pqr} has not uniqueness while the link flows are unique if ϕ^*_{pqr} is assumed strictly monotone w.r.t. the path flows (h_{pqr}).

Best worst-case (BW) model

This model considers that each user selects the path to minimize for each i the worst-case cost $\hat{\phi}^i_{pqr}(h,\omega) = \max_\omega \phi^i_{pqr}(h,\omega)$, assuming the worst case demand $\hat{d}^i_{pq} = \max_\omega d^i_{pq}(\omega)$.

$$\underset{\mathbf{h},\boldsymbol{\phi}^*}{\text{Minimize}}\, Z_{BW} = \text{Min}\begin{bmatrix} \hat{\phi}_{pqr}(h,\omega) - \phi^*_{pqr} \\ \delta_{pqr}h_{pqr} - \hat{d}_{pq}(\omega) \end{bmatrix} = 0. \tag{4.122}$$

4.6 Conclusions

In this chapter we have presented dynamic traffic models. More precisely, we have exposed the general points of the analytical-based approach of the main loading models, namely:

[9] Note that this concept is different to C_{pqr}, which is interpreted as the perceived travel cost function.

Condition	**Complementary conditions**	**Necessariness and Sufficiency**
CAUSALITY CONDITION		
Ensuring flow propagation condition.		It is neither necessary nor sufficient.
LINK FIFO CONDITIONS		
Ensuring flow propagation condition.	1. Single traffic type 2. $e_a(t) \geq 0$ and $g_a(t) \geq 0$	It is necessary and sufficient.
The derivative of the travel time associated with a user that enters link a at time t, $\tau(t)$: $\tau'_a(t) > -1$.	Travel time function $\tau(t)$ must be differentiable	It is necessary and sufficient.
The derivative of the travel time associated with a user that leaves link a at time t, $\eta(t)$: $\eta'_a(t) < 1$.	Travel time function $\eta(t)$ must be differentiable	It is necessary and sufficient.
The derivative of the entry time associated with a user that leaves link a at time t, $T(t)$: $T'_a(t) > 0$.	Entry time function $T(t)$ must be differentiable	It is necessary and sufficient.
Non-differentiable travel time functions $\tau(\cdot)$, $\eta(\cdot)$ or $T(\cdot)$ in a deterministic queuing model (Huang and Lam (2002)).		It is sufficient.
Travel time functions depending on the inflow $\tau(e_a(t))$ (Daganzo (1995b)).		Violate FIFO.
The inflow rate $e(\cdot) \leq B_a$ and the derivative of the travel time $\dfrac{\partial \tau_a(x,t)}{\partial(x)} \leq 1/B_a$, B_a is a constant for link a (Xu et al.(1999)).		It is sufficient.
ROUTE FIFO CONDITIONS		
Link FIFO satisfies.		It is necessary and sufficient.
OD FIFO CONDITIONS (Ran et al. (1996))		
Link FIFO satisfies for the route choice DUO problems. (Szeto and Lo (2004))	1. The travel cost is the sum of travel time and schedule delay cost. 2. The schedule delay cost is piecewise linear. 3. The unit travel time cost is higher than that of early arrival.	It is necessary and sufficient.

Table 4.6: Summary of different conditions that guaranty the causality and FIFO rule.

1. Kinematic models, based on the hydrodynamic analogy, which allowed the first dynamic approaches. With these models we can know the important traffic variables (flow, density, time, trajectory, etc.) in the studied link. In fact, in simple traffic cases, the solution is very realistic. However, when the conditions are more complex,

the problem is unapproachable.

2. Merchant-Nemhauser and delay-function models are conceptually very simple but, when we need realistic results, they are difficult to calibrate. On the other hand, they are black boxes, because they do not define what happens inside the link.

3. Queue models (point or physical) allow to simulate the delay due to the link saturation. Moreover, the physical queue models include the shock-wave effect in this delay, i.e., the variables inside the link can be approached.

4. Finally, the CTM gives the traffic evolution inside the link, taking account the generation, propagation and dissipation of queues. However, it is a very costly computational approach and, by that, impracticable in real networks.

Table 4.6 we provide a summary of different conditions that guaranty or not the necessariness and the sufficiency of the causality and FIFO rule.

On the other hand, we have analyzed the assignment strategy. The main conclusions drawn are:

1. Due the dynamic nature of the problem, the concept "equilibrium" is often replaced by "optimal". The criterion to evaluate this optimal state changes according to the kind of approach used.

2. The deterministic approach is a temporal generalization of the static equilibrium assignment problem.

3. The stochastic approach includes a great variety of problems depending on the kind of the stochastic inputs. These problems can be classify in (i) random travel cost (temporal generalization of the static stochastic assignment problem), and (ii) when the demand or the road capacity is stochastic.

It is evident that the stochastic analysis tends to be more realistic, however, the mathematical difficulties associated include the non-uniqueness of the solution, mathematical instability, etc.

Appendix

A Characteristic lines method

Childress (2005) suggests the characteristic lines method to solve the differential equation exposed in section 4.4.1.

Definition 6 (Characteristic lines method) *The characteristic lines method (CLM) consists of lines that emanate from the boundaries of the time-space domain and carry the value of density defined on the boundaries by the initial and boundary conditions. When two characteristic lines intersect, density at the point of the intersection should have two values which is physically unfeasible. The discontinuity of density at this point causes the generation of a shock wave which moves with the speed u_w.*

$$\frac{\partial\rho(x,t)}{\partial t}+\left.\frac{dq(\rho)}{d\rho}\right|_{\rho(x,t)}\frac{\partial\rho(x,t)}{\partial x}=0. \tag{4.123}$$

To work with the characteristic lines method we can use the following procedure:

Step 1: (Characteristic lines) Express the equation for the characteristic lines in the form:

$$x=\left.\frac{dq(\rho)}{d\rho}\right|_{\rho(x_0,0)}t+x_0. \tag{4.124}$$

Step 2: (Solve for x_0) Solve Equation (4.124) for x_0 as a function of x and t.

Step 3: (Write the solution) Obtain the solution by replacing the obtained function $x_0(x,t)$ into the initial condition $\rho_0(x)=\rho(x,0)$, that is,

$$\rho(x,t)=\rho_0(x_0(x,t)). \tag{4.125}$$

Example 4 (Queue traffic flow when a red light turns green) *We are interested in the case of a red light turning green. If this takes place at $t^*=0$, then at $t^*=0$ we have a density given by*

$$\rho^*(x^*,0)=\left\{\begin{array}{lll}1 & \textit{if} & x^*<0\\ 0 & \textit{if} & x^*>0.\end{array}\right. \tag{4.126}$$

To solve Equation (4.123) with the initial condition (4.129), we use the characteristic lines method described in Section A.

For the exterior regions we have:

Step 1: (Characteristic lines) *For the characteristic curves emerging from the region $x^* < 0$, the speed is $w^* = (1 - 2\rho^*) = 1 - 2 = -1$, so that the characteristic lines are $x^* = -t^* + x_0^*$. Similarly, for the characteristic curves emerging from the region $x^* > 0$, the speed is $u^* = (1 - 2\rho^*) = 1 - 0 = 1$, so that the characteristic lines are $x^* = t^* + x_0^*$.*

Step 2: (Solve for x_0) *Solving for x_0^* in the two cases above, we get*

$$x_0^*(x^*, t^*) = \begin{cases} x^* + t^* & \textit{if} \quad x^* < 0 \\ x^* - t^* & \textit{if} \quad x^* > 0. \end{cases} \tag{4.127}$$

Step 3: (Write the solution) *Equation (4.127) leads to the solution*

$$\rho^*(x^*, t^*) = \begin{cases} 1 & \textit{if} \quad x^* < -t^* \\ 0 & \textit{if} \quad x^* > t^*. \end{cases} \tag{4.128}$$

For the intermediate region, we can derive the characteristic curves using two different procedures:

Method 1. *Instead of a sudden change in the $\rho^*(x^*, t^*)$ function, such as that in (4.126), we can consider a continuous change as*

$$\rho_0(x) = \rho^*(x^*, 0) = \begin{cases} 1 & \textit{if} \quad x^* < 0 \\ (1 - x^*/\epsilon) & \textit{if} \quad 0 < x^* < \epsilon \\ 0 & \textit{if} \quad x^* > \epsilon, \end{cases} \tag{4.129}$$

where ϵ is a small positive number.

To solve Equation (4.123) with the initial condition (4.129), we use the three-step method described in Section A.

Step 1: (Characteristic lines) *The characteristic curves in the intermediate region are given by*

$$x^* = (1 - 2\rho^*(x_0^*, 0))t^* + x_0^* = (1 - 2(1 - x_0^*/\epsilon))t^* + x_0^* = -t^* + 2x_0^* t^*/\epsilon + x_0^*. \tag{4.130}$$

Step 2: (Solve for x_0) *Solving for x_0^*, we get*

$$x_0^* = \frac{x^* + t^*}{1 + 2t^*/\epsilon}. \tag{4.131}$$

Step 3: (Write the solution) *Equation (4.131) leads to the solution*

$$\rho^*(x^*, t^*) = \rho_0^*(x_0(x, t)) = 1 - \frac{x^* + t^*}{\epsilon + 2t^*} = \frac{\epsilon + t^* - x^*}{\epsilon + 2t^*}, \tag{4.132}$$

which letting $\epsilon \to 0$ gives

$$\rho^*(x^*, t^*) = \frac{t^* - x^*}{2t^*}. \tag{4.133}$$

Method 2. *It is clear that the characteristic curves in the intermediate region emanate from the point* $(0,0)$, *and then they must be of the form* $\rho^*(x^*,t^*) = r(x^*/t^*)$, *where* $r()$ *is a function to be determined. Replacing this expression for* $\rho^*(x^*,t^*)$ *into the differential equation* $\dfrac{\partial\rho^*(x^*,t^*)}{\partial t^*} + (1-2\rho^*(x^*,t^*))\dfrac{\partial\rho^*(x^*,t^*)}{\partial x^*} = 0$, *and letting* $\xi = x^*/t^*$, *we obtain*

$$0 = \frac{\partial r}{\partial\xi}\frac{\partial\xi}{\partial t^*} + (1-2r)\frac{\partial r}{\partial\xi}\frac{\partial\xi}{\partial x^*} = \frac{\partial r}{\partial\xi}\left[-\frac{x^*}{(t^*)^2} + (1-2r)\frac{1}{t^*}\right] = \frac{\partial r}{\partial\xi}\frac{1}{t^*}\left[-\xi + (1-2r)\right], \tag{4.134}$$

that is,

$$\rho^*(x^*,t^*) = r = \frac{1-\xi}{2} = \frac{t^*-x^*}{2t^*}, \tag{4.135}$$

which is the same result as (4.133).

Thus, we finally have:

$$\rho^*(x^*,t^*) = \begin{cases} 1 & \textit{if} \quad x^* < -t^* \\ \dfrac{t^*-x^*}{2t^*} & \textit{if} \quad -t^* \le x^* \le t^* \\ 0 & \textit{if} \quad x^* > t^* \end{cases} \tag{4.136}$$

and taking into account that $u^*(x^*,t^*) = 1 - \rho^*(x^*,t^*)$, *we obtain*

$$u^*(x^*,t^*) = \begin{cases} 0 & \textit{if} \quad x^* < -t^* \\ 1 - \dfrac{t^*-x^*}{2t^*} = \dfrac{t^*+x^*}{2t^*} & \textit{if} \quad -t^* \le x^* \le t^* \\ 1 & \textit{if} \quad x^* > t^*. \end{cases} \tag{4.137}$$

Finally, since $q^*(x^*,t^*) = \rho^*(x^*,t^*)u^*(x^*,t^*)$, *we have*

$$q^*(x^*,t^*) = \begin{cases} 0 & \textit{if} \quad x^* < -t^* \\ \dfrac{(t^*)^2-(x^*)^2}{4(t^*)^2} & \textit{if} \quad -t^* \le x^* \le t^* \\ 0 & \textit{if} \quad x^* > t^*. \end{cases} \tag{4.138}$$

Figure 4.14, 4.15 and 4.16 show the $\rho^*(x^*,t^*)$, $u^*(x^*,t^*)$ *and* $q^*(x^*,t^*)$ *respectively contours.*

Once we have the solution of our red light turning green problem, we can ask several questions, such as those treated below.

a. Time to start moving. *The first question is: how long does it take for a vehicle at a distance* $D > 0$ *in front of the traffic light to start moving? This question can be easily answered from (4.137) because the vehicle at its starting time has zero speed. So, we have*

$$u^* = 0 \quad\Leftrightarrow\quad \frac{t_D^* + x_D^*}{2t_D^*} = 0 \quad\Leftrightarrow\quad t_D^* = -x_D^* = D^*, \tag{4.139}$$

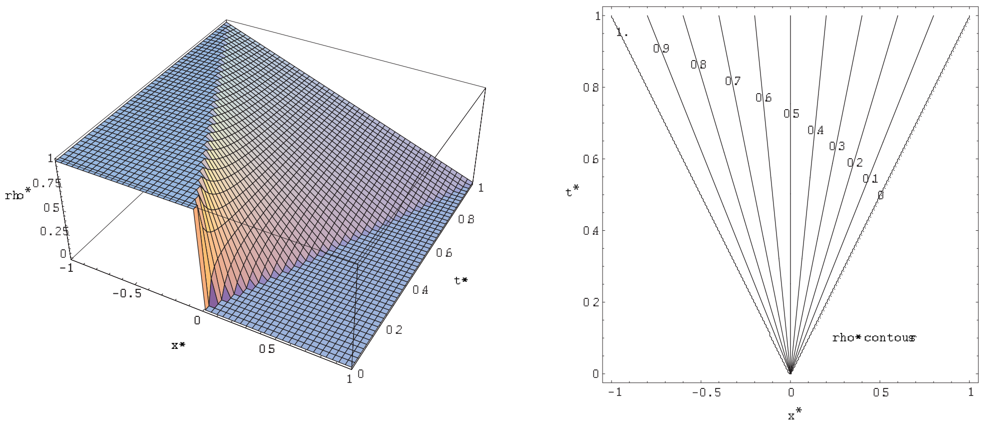

Figure 4.14: $\rho^*(x^*, t^*)$ contours.

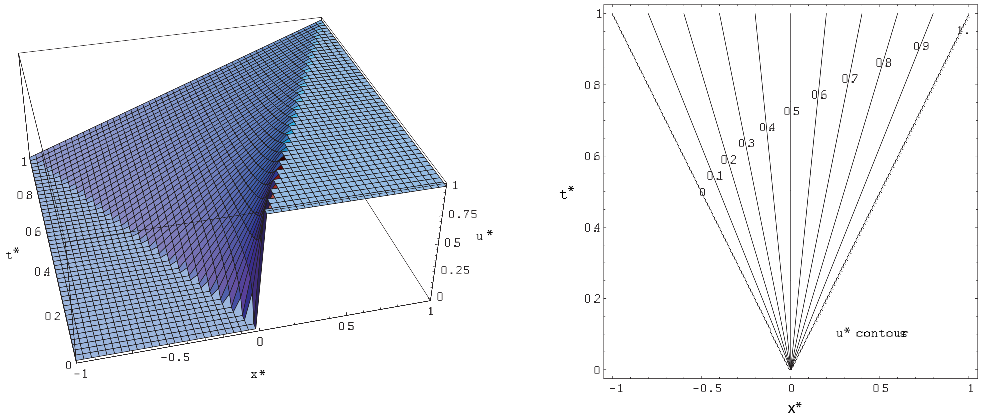

Figure 4.15: $u^*(x^*, t^*)$ contours.

where t_D^* is the dimensionless time associated with distance D^*.

To recover the initial variables we consider that

$$t_D^* = D^* \quad \Leftrightarrow \quad \frac{t_D u_0}{L} = \frac{D}{L} \quad \Leftrightarrow \quad t_D = \frac{D}{u_0}.$$

b. Car trajectories. *The second question is: what are the trajectories of the vehicles after they start moving? To answer this question, we need to solve the differential equation*

$$\frac{dx^*}{dt^*} = \frac{t^* + x^*}{2t^*}, \tag{4.140}$$

whose solution is

$$x^*(t^*) = t^* + C\sqrt{t^*}. \tag{4.141}$$

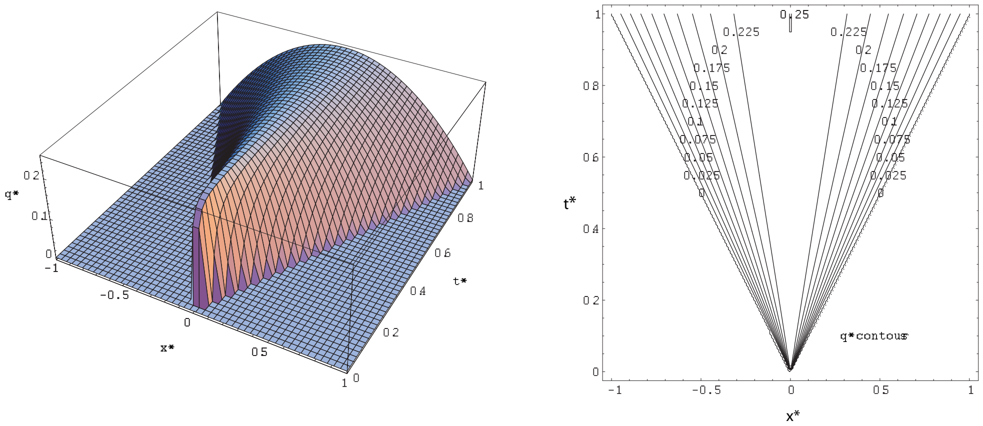

Figure 4.16: $q^*(x^*, t^*)$ contours.

In order to determine the constant C, we recall that at time $t^*_D = D^*$, the vehicle starts moving, that is,

$$x^*(t^*) = -D^* = t^*_D + C\sqrt{t^*_D} \quad \Rightarrow C = -2\sqrt{D^*}. \tag{4.142}$$

so that (4.141) becomes

$$x^*(t^*) = t^* - 2\sqrt{t^* D^*} \quad t^* > D^*, \tag{4.143}$$

that gives the looked after trajectories.

One example is given in Figure 4.17.

c. Time to reach the origin. *A third interesting question is: at what time t^*_0 the vehicle that was at $t^* = 0$ in x^*_D is reaching the origin? From (4.143), we can easily obtain the time required for a vehicle to reach the origin after start moving, as follows:*

$$0 = x^*(t^*_0) = t^*_0 - 2\sqrt{t^*_0 D^*} \quad \Leftrightarrow \quad t^*_0 = 4D^*, \tag{4.144}$$

that is,

$$\frac{t_0 u_0}{L} = 4\frac{D}{L} \quad \Leftrightarrow t_0 = \frac{4D}{u_0}.$$

d. Time to reach a given point. *Also, we ask: what is the time t^*_1 required for a vehicle to reach the point $x^* = D^*_1$ after start moving? This time is given by*

$$D^*_1 = x^*(t^*_1) = t^*_1 - 2\sqrt{t^*_1 D^*}, \tag{4.145}$$

that is,

$$\sqrt{t^*_1} = \sqrt{D^*} + \sqrt{D^* + D^*_1}, \tag{4.146}$$

that leads to

$$t^*_1 = (\sqrt{D^*} + \sqrt{D^* + D^*_1})^2 = 2D^* + D^*_1 + 2\sqrt{D^*(D^* + D^*_1)}, \tag{4.147}$$

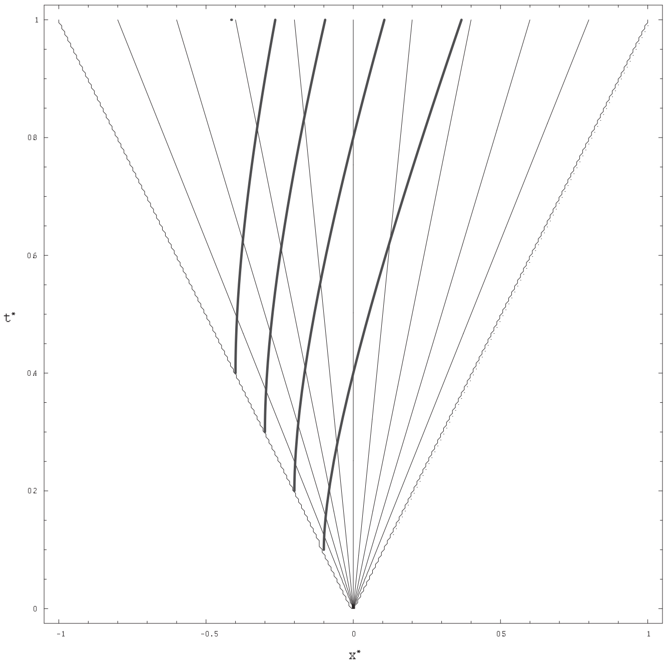

Figure 4.17: Trajectories followed by the vehicles after start moving.

That recovering the initial variables leads to

$$t_1 = \frac{2D + D_1 + 2\sqrt{D(D + D_1)}}{u_0}, \tag{4.148}$$

Consequently, the link travel time can be calculated as follows:
If we assume that $D = L_1$ and $D1 = L - L_1$, we get

$$t_a = \frac{L + L_1 + 2\sqrt{LL_1}}{u_0}. \tag{4.149}$$

If, additionally, we assume $L_1 = \alpha L$[10]*, we finally get*

$$t_a = \frac{L(1+\alpha) + 2L\sqrt{\alpha}}{u_0} = \frac{L}{u_0}(1+\sqrt{\alpha})^2 = t_0(1+\sqrt{\alpha})^2, \tag{4.150}$$

which in dimensionless terms becomes

$$t_a^* = t_0^*(1+\sqrt{\alpha})^2 = t_0(1+\sqrt{x_a^*})^2, \tag{4.151}$$

which implies

$$h\left(\alpha/2, \alpha\right) = (1+\sqrt{\alpha})^2. \tag{4.152}$$

[10] Note that in this case $\alpha = x_a^*$

If the jam is at the end of the link and has length D, we have

$$t^* = \max(1, 4D^*), \tag{4.153}$$

which implies

$$h\left(\alpha, 2(1-\alpha)\right) = \max(1, 4\alpha). \tag{4.154}$$

e. Flow behind the jam produced by a red light turning green. *Finally, the time required for a vehicle to travel a link in this case is given by*

$$t^* = \max(1, 4D^*), \tag{4.155}$$

because it is the maximum of the times required for a vehicle starting at the link origin and traveling at a maximum speed, and the time of the last vehicle in the jam region to reach the end of the link, that, according to (4.144) is $t^* = 4D^*$.

Example 5 (Four different cases by characteristic lines method) *To illustrate we consider four different cases. The graphics 4.18 and 4.19 shows the characteristic lines associated with a group of vehicles of constant density which reach other group with a different constant density for the cases (a), (b), (c) and (d). Due to the dimensionless nature of the method, we just need to define the grade of occupancy of each stretch to define completely the problem.*

The characteristics emanating from a stretch with constant density, ρ, are parallel with slope $1-2\rho$. When the density ρ_1 falls to another low value ρ_2, the "fan characteristic" appears. The instantaneous transition of density implies that density takes all the values between ρ_1 and ρ_2 and that there is no discontinuity on the slopes of the characteristics. However, when the density ρ_1 reaches to another higher value ρ_2, their characteristic lines intersect, causing the generation of a shock wave. The red lines, whose slope is $\dfrac{1}{u_w} = \dfrac{1}{1-\rho_1-\rho_2}$, are these discontinuity lines. In the same graphic the vehicular trajectories have been represented. We can see how the trajectory changes when it intersects the shock wave.

The graphics 4.20, 4.21, 4.22 and 4.23 show the density evolution over time. They have been obtained by sectioning the characteristic lines graphic for each time t.

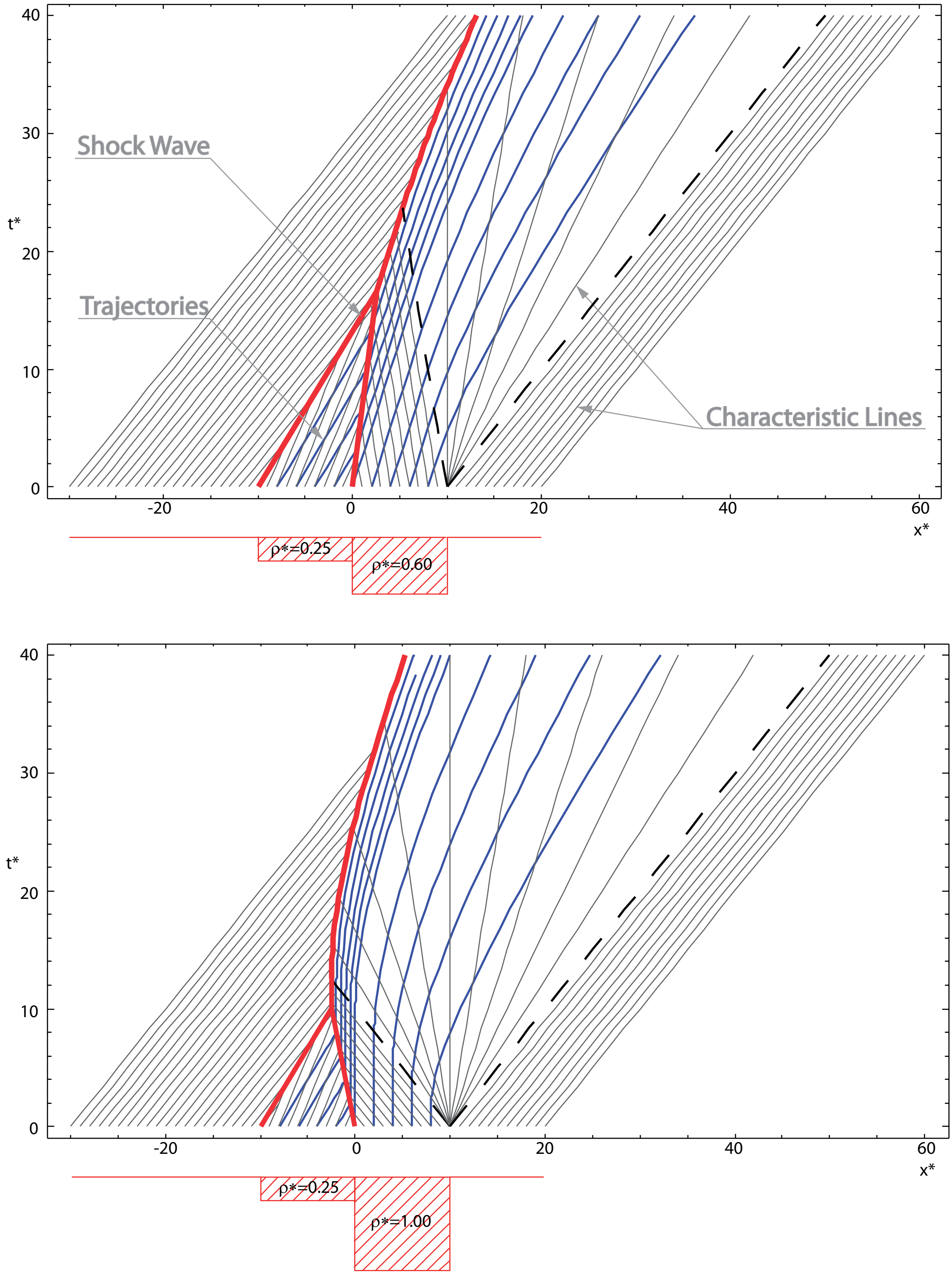

Figure 4.18: Four different problems solve by the characteristic lines method. Case a: $\rho_1^* = 0.60$ and $\rho_2^* = 0.25$ and case b: $\rho_1^* = 1.00$ and $\rho_2^* = 0.25$.

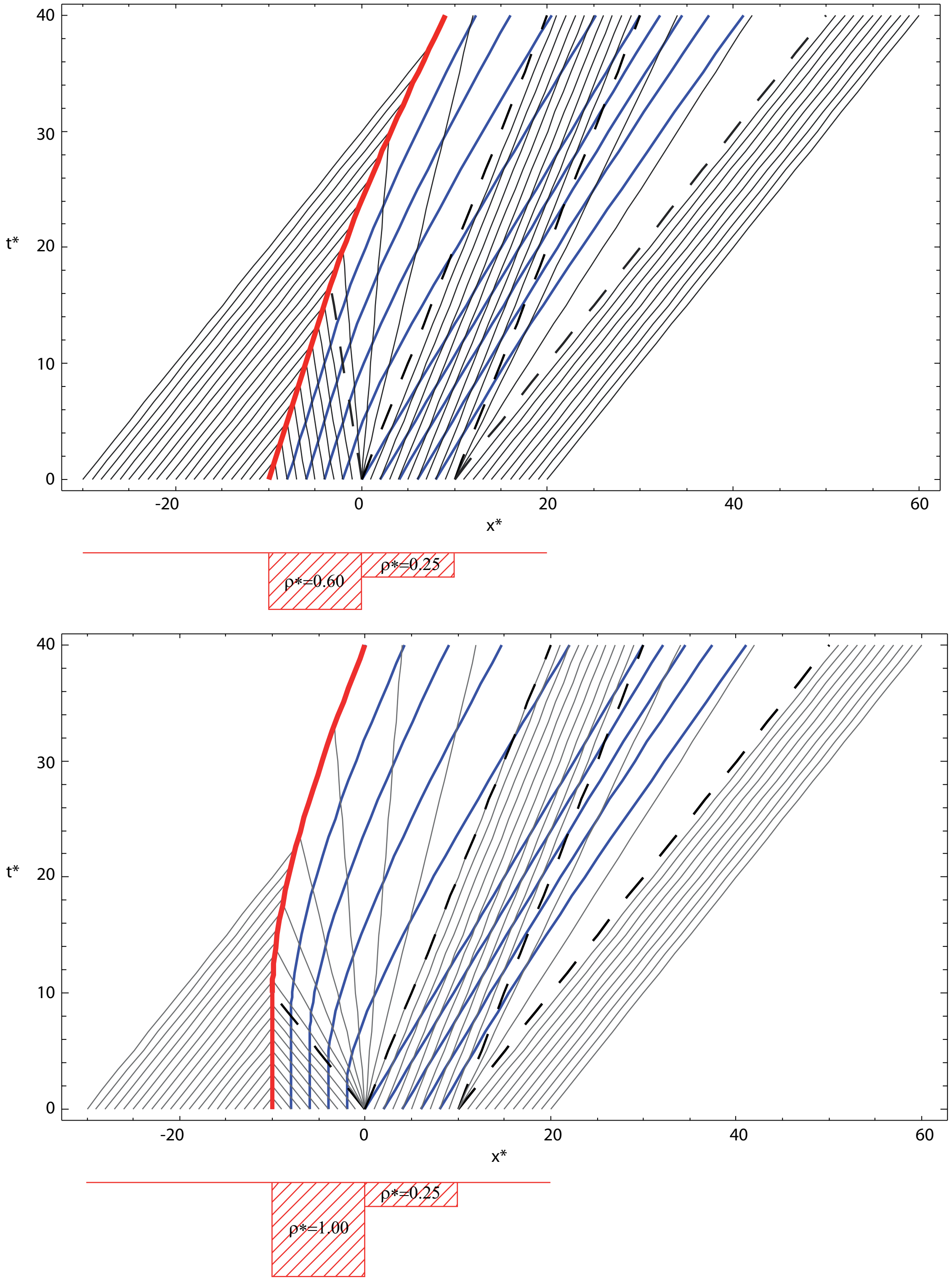

Figure 4.19: Four different problems solve by the characteristic lines method. Case c: $\rho_1^* = 0.25$ and $\rho_2^* = 0.60$ and case d: $\rho_1^* = 0.25$ and $\rho_2^* = 1.00$.

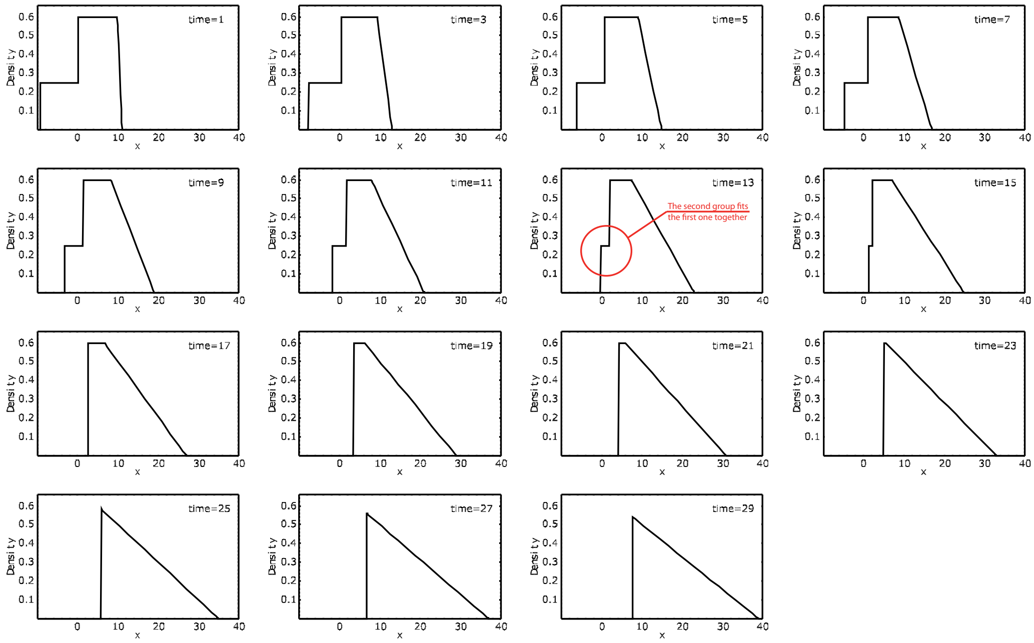

Figure 4.20: Evolution of the density with the time (characteristic lines method). Case a: $\rho_1^* = 0.60$ and $\rho_2^* = 0.25$.

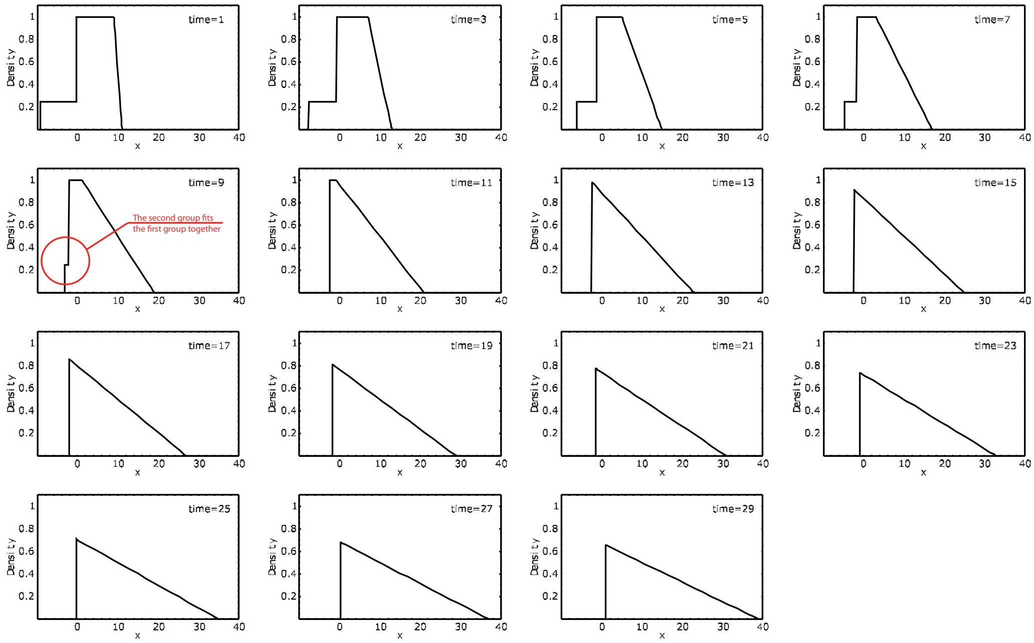

Figure 4.21: Evolution of the density with the time (characteristic lines method). Case b: $\rho_1^* = 1.00$ and $\rho_2^* = 0.25$

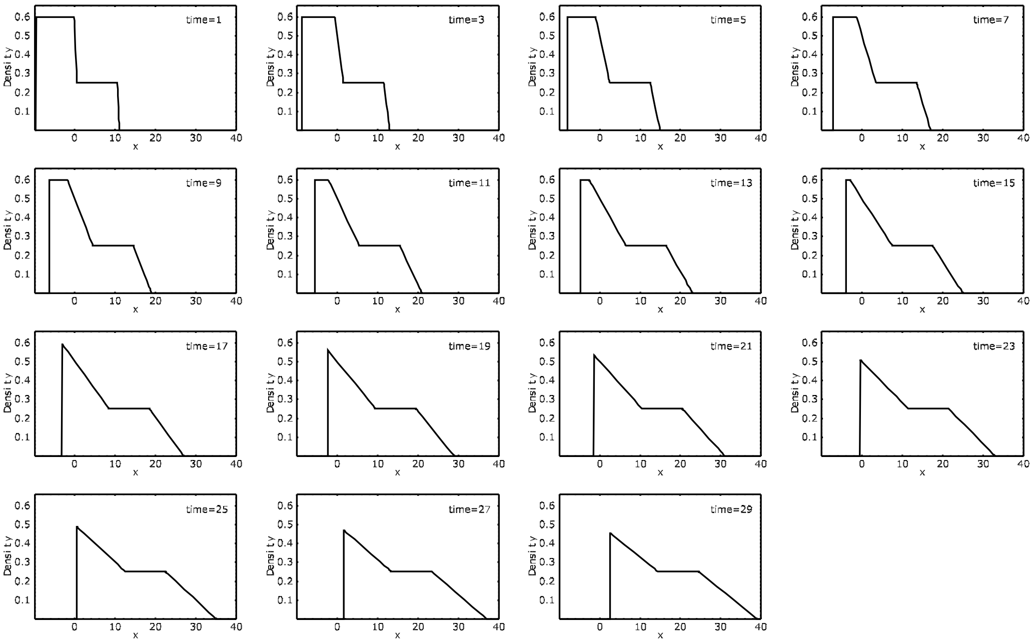

Figure 4.22: Evolution of the density with the time (characteristic lines method. Case *c*: $\rho_1^* = 0.25$ and $\rho_2^* = 0.60$

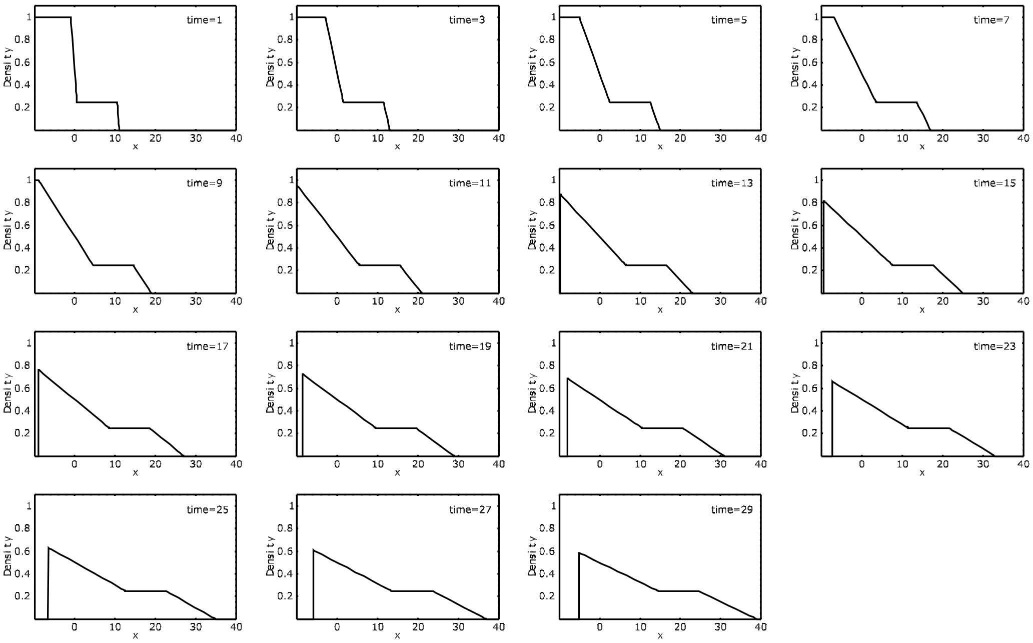

Figure 4.23: Evolution of the density with the time (characteristic lines method). Case *d*: $\rho_1^* = 0.25$ and $\rho_2^* = 1.00$

B $A(x,t)$ function

In this section we expose the $A(x,t)$ function and its interesting properties.

Theorem 9 *If there exists a function $A(x,t)$ such that*

$$\rho(x,t) = -\frac{\partial A(x,t)}{\partial x} \tag{4.156}$$

$$q(x,t) = \frac{\partial A(x,t)}{\partial t}, \tag{4.157}$$

then, the corresponding $q(x,t)$ and $\rho(x,t)$ functions satisfy the conservation law, that is,

$$\frac{\partial \rho}{\partial t} + \frac{\partial q}{\partial x} \;=\; 0. \tag{4.158}$$

Proof. Taking derivatives with respect to x and t in (4.156) and (4.157), respectively, and adding up the two values, we get

$$\frac{\partial \rho}{\partial t} + \frac{\partial q}{\partial x} \;=\; -\frac{\partial A}{\partial t \partial x} + \frac{\partial A}{\partial x \partial t} = 0. \tag{4.159}$$

■

In particular, we can define $A(x,t)$ in a convenient form.

Definition 7 ($A(x,t)$ function) *We define the $A(x,t)$ function as the number of vehicles that has passed x up to time t, which can be defined in two different forms, that is,*

$$A(x,t) = \int_x^{\infty} \rho(x,t)dx \tag{4.160}$$

or

$$A(x,t) = \int_0^{t} q(x,t)dt. \tag{4.161}$$

Note that $A(x,t)$ is an increasing function with x $\left(\frac{\partial A(x,t)}{\partial x} < 0\right)$ and a decreasing function with t $\left(\frac{\partial A(x,t)}{\partial t} > 0\right)$.

If the number of vehicles that is inside the interval $(x, x+u)$ at time $t+v$ is evaluated as the number of vehicles in $(x, x+u)$ at time t plus the vehicles that crosses x during the time interval $(t, t+v)$ minus the vehicles that leave $x+u$ at interval of time $(t, t+v)$, we obtain:

$$\int_x^{x+u} \rho(x,t+v)dx = \int_x^{x+u} \rho(x,t)dx + \int_t^{t+v} q(x,t)dt - \int_t^{t+v} q(x+u,t)dt, \tag{4.162}$$

which, expressed in terms of $A(x,t)$, becomes

$$\begin{aligned} A(x,t+v) - A(x+u,t+v) \;&=\; [A(x,t) - A(x+u,t)] + [A(x,t+v) - A(x,t)] - \\ &\quad -[A(x+u,t+v) - A(x+u,t)], \end{aligned} \tag{4.163}$$

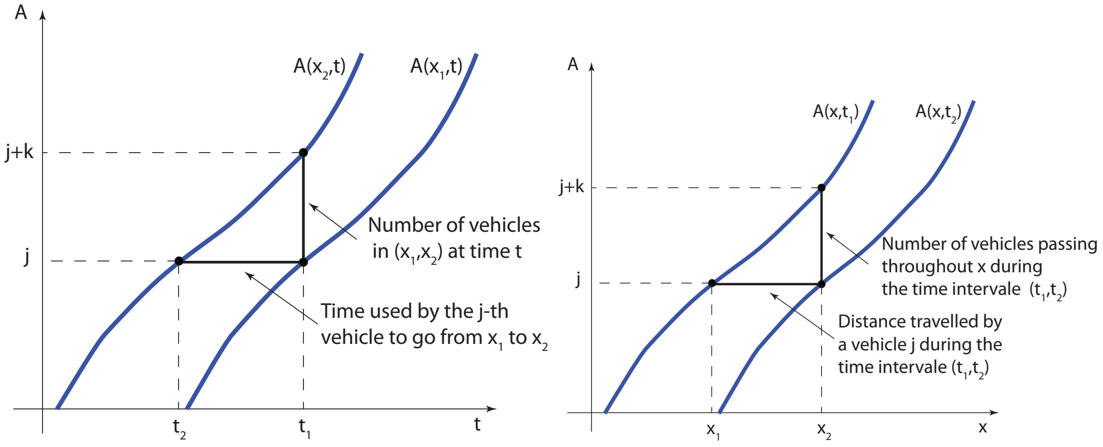

Figure 4.24: Illustration of the $A(x,t)$ function.

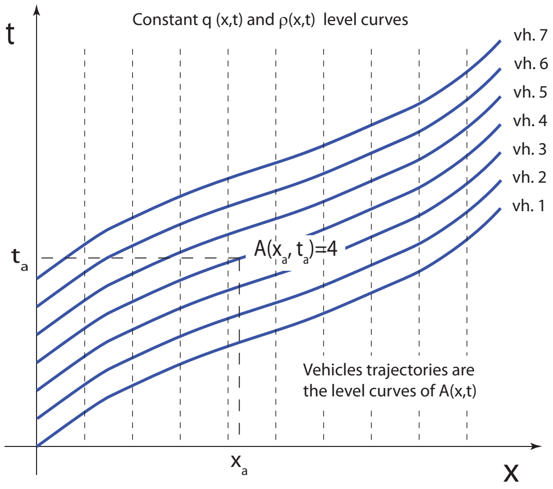

Figure 4.25: Illustration of the vehicle trajectories (level curves of the $A(x,t)$ function).

which is an identity. So, we conclude that the $A(x,t)$ function, defined as the number of vehicles that has passed x at time t satisfies the conservation law (4.158).

In order to illustrate all these concepts, the following graphics show the relationship between the function $A(x,t)$ and the variables x and t.

We emphasize the fact that the trajectories of the vehicles represented in Figure 4.24 must not cut themselves because of the FIFO condition (see Section 4.3.1).

Part III

Original Contributions

Chapter 5

Some statistical background

Contents

5.1 Notation

α	parameter of the statistical function.
$\beta(\alpha, \theta)$	Beta function with parameters α and θ.
β_{ij}	regression coefficient of X_j in the regression of X_i on their parents.
$\Gamma(\alpha)$	Gamma function.
λ	parameter of the shifted-Gamma model.
μ	mean.
Π_i	set of parents of node X_i in a directed acyclic graph.
π_i	parents of node i.
ψ_i^2	conditional variance of X_i given $\Pi_i = \pi_i$.
Σ	variance-covariance matrix.

σ^2	variance.
Σ_{YY}	variance-covariance matrix of Y.
Σ_{YZ}	covariance matrix of Y and Z.
σ_{zz}	variance of Z.
Θ	domain of definition of the θ parameter.
θ	parameter of the statistical function.
E	set of evidential nodes.
$I_\beta(x;\alpha,\theta)$	incomplete Beta function.
n	number of nodes in a Bayesian network.
$U(0,1)$	standard uniform random variable.
X	random variable.
$\mathbf{X}$	set of nodes (random variables) in a directed acyclic graph.
x_i	value of the random variable associated with node i in a Bayesian network.
$\mathcal{H}(\alpha,\theta,\lambda)$	shifted-Gamma family of densities.

5.2 Motivation

In order to get a better understanding of the following chapters, we have considered convenient to devote this chapter to explain some statistical tools to be used.

This chapter is organized as follows. In Section 5.3, we introduce some statistical families that are required in Chapters 6 and 8, namely, the shifted-Gamma function and the Beta distributions. We present the Bayesian approach in Section 5.4 and in Section 5.5, we provide some background on the Bayesian network models. In Section 5.6, we explain the particular case of Gaussian Bayesian networks and, in Section 5.7, the Gaussian mixture Bayesian network models are introduced.

5.3 Some statistical families

5.3.1 The shifted-Gamma function

In this section we introduce the reader to shifted-Gamma models that are required to understand Chapter 6.

Let X be a random variable having the probability density function (pdf)(see Castillo et al. (1996)):

$$g(x;\alpha,\theta,\lambda)=\frac{\lambda(\lambda(x-\alpha))^{\theta-1}e^{-\lambda(x-\alpha)}}{\Gamma(\theta)},x\geq\alpha, \tag{5.1}$$

where $\Gamma(\theta)$ is the Gamma function:

$$\Gamma(\alpha) = \int_{0}^{\infty} e^{-x} x^{\alpha-1} dx, \tag{5.2}$$

which for integer values of θ coincides with $(\theta-1)!$. We denote this family of densities as $\mathcal{H}(\alpha, \theta, \lambda)$.

The mean and variance of X are $E[X] = \alpha + \theta/\lambda$ and $Var[X] = \theta/\lambda^2$, respectively.

We indicate that in most practical cases we must have $\theta > 1$, and then, the mode is $\alpha + (\theta-1)/\lambda$ and the shape of the density is similar to the normal density (but non-symmetric).

An important practical property of the family of densities $\mathcal{H}(\alpha, \theta, \lambda)$ is its reproductivity, as indicated by the following theorem.

Theorem 10 *The family of densities $\{\mathcal{H}(\alpha, \theta, \lambda); \alpha \geq 0, \theta > 0, \lambda > 0\}$ described in (5.1) is reproductive, that is, the sum of two of more copies of independent random variables in this family belongs to this family.*

Proof. We use characteristic functions in this proof. The characteristic function of a random variable with density (5.1) is

$$\phi_{\mathcal{H}(\alpha,\theta,\lambda)}(t) = e^{it\alpha}\left(1 - \frac{it}{\lambda}\right)^{-\theta} \tag{5.3}$$

and the characteristic function of the sum of two random variables $X \sim \mathcal{H}(\alpha_1, \theta_1, \lambda_1)$ and $Y \sim \mathcal{H}(\alpha_2, \theta_2, \lambda_2)$ is

$$\begin{aligned}
\phi_{\mathcal{H}(\alpha_1,\theta_1,\lambda)+\mathcal{H}(\alpha_2,\theta_2,\lambda)}(t) &= \phi_{\mathcal{H}(\alpha_1,\theta_1,\lambda)}(t)\ \phi_{\mathcal{H}(\alpha_2,\theta_2,\lambda)}(t) \\
&= e^{it\alpha_1}\left(1 - \frac{it}{\lambda}\right)^{-\theta_1} e^{it\alpha_2}\left(1 - \frac{it}{\lambda}\right)^{-\theta_2} \\
&= e^{it(\alpha_1+\alpha_2)}\left(1 - \frac{it}{\lambda}\right)^{-(\theta_1+\theta_2)} \\
&= \phi_{\mathcal{H}(\alpha_1+\alpha_2,\theta_1+\theta_2,\lambda)}(t), \qquad (5.4)
\end{aligned}$$

that proves its reproductivity and shows how the parameters of the random variables X and Y must be combined to obtain the parameters of the random variable $X + Y$. ■

5.3.2 The standard Beta distributions

Due to we use the *generalized Beta distribution* in Chapter 8, we consider convenient to introduce the standard Beta distributions.

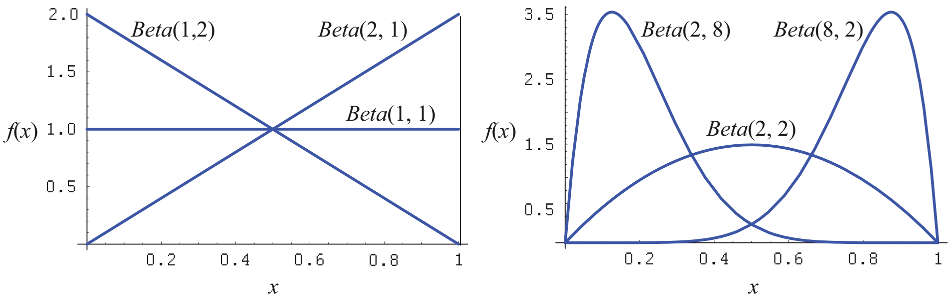

Figure 5.1: Examples showing that the pdf of Beta random variables take on a wide range of different shapes.

The Beta random variable is useful for modeling experimental data with range limited to the interval $[0, 1]$ (see Figure 5.1). Its name is due to the presence of the *Beta function* in its pdf, which is defined as

$$\beta(\alpha, \theta) = \int_0^1 x^{\alpha-1}(1-x)^{\theta-1}\,dx, \quad \alpha > 0,\ \theta > 0 \tag{5.5}$$

and is related to the Gamma function by the relation

$$\beta(\alpha, \theta) = \frac{\Gamma(\alpha)\Gamma(\theta)}{\Gamma(\alpha+\theta)}. \tag{5.6}$$

The pdf of a Beta random variable is given by

$$f(x) = \frac{x^{\alpha-1}(1-x)^{\theta-1}}{\beta(\alpha, \theta)} = \frac{\Gamma(\alpha+\theta)}{\Gamma(\alpha)\Gamma(\theta)} x^{\alpha-1}(1-x)^{\theta-1}, \quad 0 \le x \le 1, \tag{5.7}$$

where $\alpha > 0$ and $\theta > 0$. The Beta random variable is denoted by $Beta(\alpha, \theta)$. The cdf (cumulative distribution function) of the $Beta(\alpha, \theta)$ is

$$F(x) = \int_0^x f(t)dt = \int_0^x \frac{t^{\alpha-1}(1-t)^{\theta-1}}{\beta(\alpha, \theta)} dt = I_\beta(x; \alpha, \theta), \tag{5.8}$$

where $I_\beta(x; \alpha, \theta)$ is called the *incomplete Beta function*, which cannot be given in closed form, but can be obtained by numerical integration.

The mean and variance of the Beta random variable are

$$\mu = \frac{\alpha}{\alpha+\theta} \quad \text{and} \quad \sigma^2 = \frac{\alpha\theta}{(\alpha+\theta+1)(\alpha+\theta)^2},$$

respectively.

The interest in this variable is also based on its flexibility, because it can take on many different shapes, which can fit different sets of experimental data very well. For example, Figure 5.1 shows different examples of the pdf of the Beta distribution. Two particular cases of the Beta distribution are interesting. Setting ($\alpha = 1, \theta = 1$) gives the *standard uniform* random variable, $U(0,1)$, while setting ($\alpha = 2, \theta = 1$ or $\alpha = 1, \theta = 2$) gives the *triangular* random variable whose cdf is given by $f(x) = 2x$ or $f(x) = 2(1-x)$, $0 \leq x \leq 1$, respectively.

5.4 The Bayesian approach

Let X be a random variable belonging to the parametric family $f(x|\theta)$. Bayesian statisticians, instead of assuming $\boldsymbol{\theta}$ a fixed value, as classical statisticians do, consider $\boldsymbol{\theta}$ as a random variable itself. Bayesian statisticians combine the knowledge of experts with neutral knowledge obtained from samples $\mathbf{x}$. The first one is given as an "a priori" distribution, denoted $f(\theta;\eta)$, where η parameters are known. The neutral knowledge is later combined with the prior, using the Bayes theorem, to obtain the "posterior" distribution, $f(\theta;\eta|\mathbf{x})$. The Bayes' rule can be expressed as follows:

$$f(\theta;\eta|\mathbf{x}) = \frac{f(\mathbf{x}|\theta)f(\theta;\eta)}{\int\limits_{\theta} f(\mathbf{x}|\theta)f(\theta;\eta)d\theta} \doteq f(\mathbf{x}|\theta)f(\theta;\eta), \tag{5.9}$$

where $f(\mathbf{x}|\theta)$ is the *likelihood* of the data.

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.

The Bayesian approach is equivalent to assuming that the true probability function is a mixture (known as the Bayesian probability function) of $f(\theta)$ with weights given by the prior before obtaining the sample, or the posterior, after the sample is obtained. The Bayesian predictive probability function after the sample becomes known, is

$$f(\mathbf{x};\eta) = \int\limits_{\Theta} f(\theta;\eta|\mathbf{x})g(\mathbf{x};\theta)d\theta, \tag{5.10}$$

where Θ is the domain of definition of the θ parameter.

The Bayesian approach consists of the following steps:

1. Select the likelihood family.

2. Select the family of priors (normally but not always a conjugate family)[1].

3. Assess the prior distribution on the parameters.

[1] An alternative to conjugate models consists in using Markov chain Monte Carlo methods.

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 and use the initial distribution with these estimates or, alternatively, use the Bayesian distribution in (5.9).

5.5 Bayesian network models

We describe below some characteristics of Bayesian networks and at the same time we try to emphasize the importance of these models with respect to traffic problems. Bayesian network models have frequently been used to solve a wide variety of practical problems (see, for example, Sheffi (1988), Castillo et al. (1997b,a, 1999) or Jensen (2001)) and in particular in the field of traffic engineering (see Castillo et al. (2008d,e)). Next, the particular case of Gaussian Bayesian networks is presented in more detail.

Definition 8 (Bayesian network) *A Bayesian network is a pair* $(\mathcal{G},\mathcal{P})$*, where* $\mathcal{G}$ *is a directed acyclic graph (DAG) defined on a set of nodes* $\mathbf{X}$ *(the random variables),* $\mathcal{P}=\{p(x_1|\pi_1),\ldots,p(x_n|\pi_n)\}$ *is a set of* n *conditional probability densities, one for each variable, and* Π_i *is the set of parents of node* X_i *in* $\mathcal{G}$*. The set* P *defines the associated joint probability density (JPD) of all nodes as*

$$p(\mathbf{x})=\prod_{i=1}^{n}p(x_i|\pi_i). \tag{5.11}$$

Bayesian networks have two main elements:

- **Graph** $\mathcal{G}$**.** It contains all the qualitative information about the relationships among the variables, no matter what probability values are assigned to them. For example, the conditional independence relations among variables in $\mathbf{X}$ can be inferred directly from graph $\mathcal{G}$. This allows us to determine which information is relevant to given variables when the knowledge of other variables becomes available, and in particular, which variables contain redundant information and which ones add new information with respect to the already existing one. This is very useful in the case of traffic models, for which link flow information at different times and links contains such redundant information, which needs to be identified.

- **The set of conditional probabilities of the variables given their parents.** The probabilities in $\mathcal{P}$ contain the quantitative information, i.e., they quantify the qualitative properties revealed by the graphical structure. The factorization of the joint probability density implied by (5.11) is very simple and facilitates its calculation. In addition, the updating of probabilities when having evidence of some variables can be very easily done; in other words, the conditional probabilities of a set of variables given another set of evidential variables can be immediately calculated because Bayesian networks are especially designed to this end. This property is also relevant to traffic flow problems, in which we aim at predicting some traffic flows based on other traffic flow data.

We note that Bayesian networks are very general, i.e., any probability distribution can be represented by a Bayesian network.

Example 6 (Example of Bayesian traffic network) *Let us consider the simple traffic network in Figure (5.2). Assume that we have only the $\mathcal{OD}$ pair* $(1,3)$ *and two routes* $\{(1,2),(3)\}$*. Then, it is clear that the link flows v_1, v_2 and v_3 depend on the $\mathcal{OD}$ flow T, leading to the Bayesian network in the right part of Figure (5.2), where the arrows go from parents to sons. Note that link flow v_a has the T $\mathcal{OD}$ flow as a parent, and the T $\mathcal{OD}$ flow has v_a as son, if link a is contained in at least one path of such a $\mathcal{OD}$ pair.*

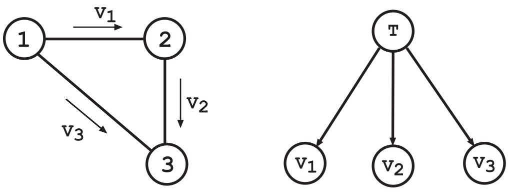

Figure 5.2: A traffic network and its associated Bayesian network.

Finally, we point out that working with Bayesian networks normally has two main phases:

1. *Learn the parameters of the model.* Learning the models means estimating the parameters that defines the network, i.e., the dependance structure and the associated conditional probabilities.

2. *Propagation of evidence.* The propagation of evidence[2] consists on updating the probabilities of the variables since the evidence. Consequently, we use the network to make predictions.

5.6 Gaussian Bayesian networks models

In this section we define the particular case of Gaussian Bayesian networks.

[2] There are three kinds of propagation algorithms: exact, approximate and symbolic ones.

Definition 9 (Gaussian Bayesian network) *Any Bayesian network $(\mathcal{G}, \mathcal{P})$ is said to be a Gaussian Bayesian network if and only if the joint probability distribution associated with its variables X is a multivariate normal distribution, $N(\mu, \Sigma)$, i.e., with joint probability density function*

$$f(x) = (2\pi)^{-n/2}|\Sigma|^{-1/2} \exp\left\{-1/2(x-\mu)^T\Sigma^{-1}(x-\mu)\right\}, \tag{5.12}$$

where μ is the n-dimensional mean vector, Σ is the $n \times n$ covariance matrix, $|\Sigma|$ is the determinant of Σ, and μ^T denotes the transpose of μ.

Gaussian Bayesian networks have been treated, among others, by Sheffi (1988), Kenley (1986), Shachter and Kenley (1989), and Castillo et al. (1997b,c).

5.6.1 Representation of the Gaussian Bayesian Network

We have two possible representations of the JPD of a normal Bayesian network, i.e., the JPD of the variables in a Gaussian Bayesian network can be specified in two different ways:

1. Using a factorization of the JPD as a product of Gaussian cumulative probability densities.

$$f(x_i|\pi_i) \sim N\left(\mu_i + \sum_{j=1}^{i-1} \beta_{ij}(x_j - \mu_j),\ \psi_i^2\right), \tag{5.13}$$

   where β_{ij} is the regression coefficient of X_j in the regression of X_i on the parents of X_i, Π_i, and

$$\psi_i^2 = \Sigma_i - \Sigma_{i\Pi_i}\Sigma_{\Pi_i}^{-1}\Sigma_{i\Pi_i}^T$$

   is the conditional variance of X_i, given $\Pi_i = \pi_i$, where Σ_i is the unconditional variance of X_i, $\Sigma_{i\Pi_i}$ is the matrix of covariances between X_i and the variables in Π_i, and Σ_{Π_i} is the covariance matrix of Π_i.

2. Alternatively, we can define the normal JPD function using the covariance matrix of the Gaussian JPD. Shachter and Kenley (1989) use the following recursive formula, in which $W(i)$ denotes the $i \times i$ upper left submatrix of W and $\boldsymbol{\beta}_i$ denotes the column vector $\{\beta_{ij} : j < i\}$:

$$W(i+1) = \left(\begin{array}{ccc} W(i) + \dfrac{\boldsymbol{\beta}_{i+1}\boldsymbol{\beta}_{i+1}^T}{\psi_{i+1}^2} & | & \dfrac{-\boldsymbol{\beta}_{i+1}}{\psi_{i+1}^2} \\ -------- & + & ---- \\ \dfrac{-\boldsymbol{\beta}_{i+1}^T}{\psi_{i+1}^2} & | & \dfrac{1}{\psi_{i+1}^2} \end{array}\right), \tag{5.14}$$

   with $W(1) = 1/\psi_1^2$.

Several algorithms advantageously use the structure provided by the first representation for evidence propagation (see Xu and Pearl (1989), and Chang and Fung (1991)). The following section introduces a kind of propagation used in the Gaussian networks models.

5.6.2 Exact propagation of evidence in Gaussian Network Models

In this section we present a conceptually simple and efficient algorithm that uses the covariance matrix representation. This algorithm illustrates the basic concepts underlying the exact propagation in Gaussian network models. An incremental implementation of the algorithm allows updating probabilities without the need to invert matrices[3], when a single piece of evidence is observed. It also allows us to perform a sequential selection of the most informative variables (flows) to be observed. This selection is based on the correlation coefficients of the updated conditional variances in order to take into account redundant information. The main result is given in the following theorem, which characterizes the CPDs obtained from a Gaussian JPD (see, for example, Anderson (1984)).

Theorem 11 Conditionals of a Gaussian distribution. *Let Y and Z be two sets of random variables having a multivariate Gaussian distribution with mean vector and covariance matrix given by*

$$\mu = \begin{pmatrix} \mu_Y \\ \mu_Z \end{pmatrix} \qquad \text{and} \qquad \Sigma = \begin{pmatrix} \Sigma_{YY} & \Sigma_{YZ} \\ \Sigma_{ZY} & \Sigma_{ZZ} \end{pmatrix},$$

respectively, where μ_Y and Σ_{YY} are the mean vector and covariance matrix of Y, μ_Z and Σ_{ZZ} are the mean vector and covariance matrix of Z, and Σ_{YZ} is the covariance of Y and Z. Then the CPD of Y given $Z = z$ (the evidence) is multivariate Gaussian with mean vector $\mu_{Y|Z=z}$ and covariance matrix $\Sigma_{Y|Z=z}$ that are given by[4]

$$\begin{aligned} \mu_{Y|Z=z} &= \mu_Y + \Sigma_{YZ}\Sigma_{ZZ}^{-1}(z - \mu_Z), & (5.15) \\ \Sigma_{Y|Z=z} &= \Sigma_{YY} - \Sigma_{YZ}\Sigma_{ZZ}^{-1}\Sigma_{ZY}. & (5.16) \end{aligned}$$

Note that the conditional mean $\mu_{Y|Z=z}$ depends on z but the conditional variance $\Sigma_{Y|Z=z}$ does not.

Theorem 11 suggests an obvious procedure to obtain the means and variances of any subset of variables $Y \subset X$, given a set of evidential nodes $E \subset X$ whose values are known to be $E = e$. Replacing Z in (5.15) and (5.16) by E, we obtain the mean vector and covariance matrix of the conditional distribution of the nodes in Y. Note that considering $Y = X \setminus E$ we get the joint distribution of the remaining nodes, and then we can answer

[3] It is based on Expressions (5.15) and (5.16).

[4] These evidence propagation formulas are valid only for normal Bayesian networks. Other Bayesian networks require other formulas (see Castillo et al. (1997b) or Jensen (2001)).

questions involving the joint distribution of nodes instead of the usual information that refers only to individual nodes.

Note that once we have calculated $\Sigma_{Y|Z}$ the variance of Y due to the knowledge of Z has been removed and we can other proceed to evaluate the extra variance of Y due to other Y variables. In other words, we can select the data that contain the best information that complements the already existing one.

In order to simplify the computations, it is more convenient to use an incremental method, updating one evidential node at a time (taking elements one by one from E)[5]. In this case, we do not need to calculate the inverse of a matrix because it degenerates to a scalar. Moreover, μ_y and Σ_{yz} are column vectors, and Σ_z is also a scalar, i.e., $\Sigma_{ZZ}^{-1} \equiv 1/\sigma_{zz}$. Thus, this algorithm provides a simple and efficient method for evidence propagation in Gaussian Bayesian network models. When the number of nodes in the transportation network is large, which is usually the case, this approach of updating the joint pdf can be replaced by other techniques, such as the Markov Chain Monte Carlo (MCMC) methods (see Gilks et al. (1996)) to update the joint pdf sequentially as evidences become available.

Bayesian networks are prepared to predict target variable distributions for any combination of evidential variables. This means that they are able to deal with missing information without any extra complication. This is useful in traffic problems where we can have missed information. In Section 8.5 we provide some examples.

5.7 Gaussian mixture Bayesian network models

In this section we describe the Gaussian mixture Bayesian network model. We start by defining Gaussian mixture Bayesian networks.

Definition 10 (Gaussian mixture Bayesian network) *A Bayesian network is said to be a Gaussian mixture Bayesian network if its joint pdf is a linear convex combination of multi-normal densities, that is, if its pdf is of the form*

$$f(\mathbf{x};\boldsymbol{\Theta}) = (2\pi)^{-n/2}\sum_{\ell=1}^{M}\alpha_\ell|\boldsymbol{\Sigma}_\ell|^{-1/2}\exp\left\{-1/2(\mathbf{x}-\boldsymbol{\mu}_\ell)^T\boldsymbol{\Sigma}_\ell{}^{-1}(\mathbf{x}-\boldsymbol{\mu}_\ell)\right\}, \tag{5.17}$$

where

$$\sum_{\ell=1}^{M}\alpha_\ell = 1; \quad \alpha_\ell \geq 0, \quad \ell = 1,2,\ldots,M, \tag{5.18}$$

[5] The incremental method produces very similar results as the method of updating the evidence in a single pass, as has been checked. Note that in addition, the numerical procedures of both methods are different, and the rounding errors act in a different way.

$\boldsymbol{\mu}_\ell$ are the n-dimensional mean vectors, $\boldsymbol{\Sigma}_\ell$ are the $n \times n$ covariance matrices, $|\boldsymbol{\Sigma}_\ell|$ is the determinant of $\boldsymbol{\Sigma}_\ell$, ${\boldsymbol{\mu}_\ell}^T$ denotes the transpose of $\boldsymbol{\mu}_\ell$, and $\boldsymbol{\Theta} = \{\alpha_1, \alpha_2 \dots, \alpha_M; \boldsymbol{\mu}_1, \boldsymbol{\mu}_2 \dots, \boldsymbol{\mu}_M; \boldsymbol{\Sigma}_1, \boldsymbol{\Sigma}_2, \dots, \boldsymbol{\Sigma}_M\}$ is the set of parameters of the model.

In this case, the Bayesian network is said to have M normal components.

5.7.1 Learning the Gaussian mixture Bayesian network

We estimate the parameters in the set

$$\boldsymbol{\Theta} = \{\alpha_1, \alpha_2 \dots, \alpha_M; \boldsymbol{\mu}_1, \boldsymbol{\mu}_2 \dots, \boldsymbol{\mu}_M; \boldsymbol{\Sigma}_1, \boldsymbol{\Sigma}_2, \dots, \boldsymbol{\Sigma}_M\}.$$

Since the maximum likelihood is considered as the best method from the point of view of statistical convenience, we adopt this method and use the well known Expectation-Maximization (EM) method, whose name was given in a classic 1977 paper (see Dempster et al. (1977)).

An expectation-maximization (EM) algorithm is a method for finding maximum likelihood or maximum "a posteriori" estimates of parameters in statistical models, where the model depends on unobserved variables. EM is an iterative method which, as its name indicates, consists of two steps: an expectation (E) step, which computes the expectation of the log-likelihood evaluated using the current estimate for the parameters, and a maximization (M) step, which computes parameters maximizing the expected log-likelihood found on the E step. The process is repeated until convergence[6].

Application of the EM-algorithm to our case here leads to the following estimates, which must be iterated until convergence.

$$\alpha_\ell^{(i)} = \frac{\sum_{k=1}^{N} p\left(\ell|\mathbf{x}_k; \boldsymbol{\Theta}^{(i-1)}\right)}{N}; \quad \ell = 1, 2, \dots, M, \tag{5.19}$$

where

$$p\left(\ell|\mathbf{x}_k; \boldsymbol{\Theta}^{(i-1)}\right) = \frac{\alpha_\ell^{(i-1)}\, p\left(\mathbf{x}_k|\boldsymbol{\Theta}^{(i-1)}\right)}{\sum_{\ell=1}^{M} \alpha_\ell^{(i-1)}\, p\left(\mathbf{x}_k|\boldsymbol{\Theta}_\ell^{(i-1)}\right)}; \quad \ell = 1, 2, \dots, M, \quad k = 1, 2, \dots, N, \tag{5.20}$$

and $\boldsymbol{\Theta}_\ell = \{\mu_\ell, \Sigma_\ell\}$

$${\boldsymbol{\mu}_\ell}^{(i)} = \frac{\sum_{k=1}^{N} \boldsymbol{x}\, p\left(\ell|\mathbf{x}_k; \boldsymbol{\Theta}_\ell^{(i-1)}\right)}{\sum_{k=1}^{N} p\left(\ell|\mathbf{x}_k; \boldsymbol{\Theta}_\ell^{(i-1)}\right)}; \quad \ell = 1, 2, \dots, M \tag{5.21}$$

[6] It has been shown that under very general conditions this method is convergent and has good properties. This justifies its common use (see Wu (1983)).

and

$$\boldsymbol{\Sigma}_\ell^{(i)} = \frac{\sum\limits_{k=1}^{N} (\boldsymbol{x} - \boldsymbol{\mu}_\ell^{(i)})(\boldsymbol{x} - \boldsymbol{\mu}_\ell^{(i)})^T p\left(\ell|\mathbf{x}_k; \boldsymbol{\Theta}_\ell^{(i-1)}\right)}{\sum\limits_{k=1}^{N} p\left(\ell|\mathbf{x}_k; \boldsymbol{\Theta}_\ell^{(i-1)}\right)}; \quad \ell = 1, 2, \ldots, M. \tag{5.22}$$

where $\mathbf{x}_k$ is the kth element of the sample, N is the sample size, M is the number of normal components of the model, and i is the iteration number.

In order to accelerate the convergence process it is convenient to start with a good guess of the right estimates. One possible option consists in using the K-means method, which provides us with initial estimates, that can be denoted as $\{\alpha_\ell^{(0)}, \boldsymbol{\mu}_\ell{}^{(0)}, \boldsymbol{\Sigma}_\ell^{(0)}\}$.

The K-means method

The K-means method aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean. It is similar to the EM algorithm for mixtures of Gaussians in that they both attempt to find the centers of natural clusters in the data as well as in the iterative refinement approach employed by both algorithms. The K-means algorithm proceeds by alternating between two steps until no change in the centroids takes place:

1. *Assignment step:* Assign each observation to the cluster with the closest mean (i.e. partition the observations according to the Voronoi diagram generated by the means).

2. *Update step:* Calculate the new centroids as the means of the observations in each cluster.

5.7.2 Propagation of evidence in Gaussian mixture Bayesian network

Let Y be the random variable (traffic variable) to be predicted and $\hat{Y}$ the predicted value. Our predictions $\hat{Y}$ will be based on the conditional probabilities of the random variable Y being predicted given (conditioned) the evidence $\mathbf{X}$. To this end, we will use the expected conditional value $\hat{Y} = E[Y|\mathbf{X}]$.

Since

$$P(Y|\mathbf{X}) = \frac{P(Y, \mathbf{X})}{P(\mathbf{X})} \tag{5.23}$$

$$= \frac{\sum\limits_{\ell=1}^{M} \alpha_\ell \; f(\mathbf{X}; \boldsymbol{\mu}_{\ell X}, \boldsymbol{\Sigma}_{\ell \mathbf{X}\mathbf{X}}) f(Y; \boldsymbol{\mu}_{\ell Y|\mathbf{X}}, \boldsymbol{\Sigma}_{\ell Y|\mathbf{X}})}{\sum\limits_{\ell=1}^{M} \alpha_\ell \; f(\mathbf{X}; \boldsymbol{\mu}_{\ell \mathbf{X}}, \boldsymbol{\Sigma}_{\ell \mathbf{X}\mathbf{X}})} \tag{5.24}$$

$$= \sum_{\ell=1}^{M} \beta_\ell f(Y; \boldsymbol{\mu}_{\ell Y|X}, \boldsymbol{\Sigma}_{\ell Y|\mathbf{X}}), \tag{5.25}$$

where

$$\begin{aligned}
\beta_\ell &= \frac{\sum_{\ell=1}^{M} \alpha_\ell \, f(\mathbf{X}; \boldsymbol{\mu}_{\ell\mathbf{X}}, \boldsymbol{\Sigma}_{\ell\mathbf{X}\mathbf{X}})}{\sum_{\ell=1}^{M} \alpha_\ell \, f(\mathbf{X}; \boldsymbol{\mu}_{\ell\mathbf{X}}, \boldsymbol{\Sigma}_{\ell\mathbf{X}\mathbf{X}})} && (5.26)\\
\boldsymbol{\mu}_{\ell Y|\mathbf{X}} &= \boldsymbol{\mu}_{\ell Y} - \boldsymbol{\Sigma}_{\ell Y\mathbf{X}} \boldsymbol{\Sigma}_{\ell\mathbf{X}\mathbf{X}}^{-1} (\boldsymbol{\mu}_{\ell\mathbf{X}} - \mathbf{X}) && (5.27)\\
\boldsymbol{\Sigma}_{\ell Y|\mathbf{X}} &= \boldsymbol{\Sigma}_{\ell YY} - \boldsymbol{\Sigma}_{\ell Y\mathbf{X}} \boldsymbol{\Sigma}_{\ell\mathbf{X}\mathbf{X}}^{-1} \boldsymbol{\Sigma}_{\ell\mathbf{X}Y} && (5.28)
\end{aligned}$$

Then, our prediction is given by

$$\hat{Y} = E[Y|\mathbf{X}] = \sum_{\ell=1}^{M} \beta_\ell \, \mu_{\ell Y|\mathbf{X}}. \tag{5.29}$$

Finally, we can use the matrix $\boldsymbol{\Sigma}_{\ell Y|\mathbf{X}}$ to obtain confidence bounds.

Chapter 6

Bayesian estimation based on plate scanning

Contents

6.1 Notation

α	location parameter of the shifted-Gamma function.
α_a	free-flow travel time of link a.

β_a parameter of the travel cost function associated with link a.

δ_a^r terms of the path-link incidence matrix.

ϵ_{rs} active hyperparameter of the shifted-Gamma function corresponding to route r and the s element in the sample.

$\hat{\epsilon}_{rs}$ posterior active hyperparameter of the shifted-Gamma function corresponding to route r.

η_r active hyperparameter of the shifted-Gamma function corresponding to route r.

$\hat{\eta}_r$ posterior active hyperparameter of the shifted-Gamma function corresponding to route r.

γ parameter of the travel cost function.

λ parameter of the shifted-Gamma function.

ν_i parameter of the shifted-Gamma function associated to the number of users traveling through the OD-pair i.

ρ_r active hyperparameter of the shifted-Gamma function corresponding to route r.

$\hat{\rho}_r$ posterior active hyperparameter of the shifted-Gamma function corresponding to route r.

ρ_i location parameter of the shifted-Gamma function associated to the number of users traveling through the OD-pair i.

ρ_i^a location parameter of the shifted-Gamma function associated to the number of users using link a and traveling the OD-pair i.

Θ domain of definition of the θ parameter.

θ parameter of the shifted-Gamma function.

θ_i^a parameter of the shifted-Gamma function associated to the number of users using link a and traveling the OD-pair i.

a link.

c_a travel cost function associated with link a.

c_{a0} free-flow conditions cost associated with link a.

C_{ir} set of links in route r of OD-pair i.

F_r random amount of users choosing route r.

f_r path flows.

$\hat{f}_r$ reconstructed path flows.

f_r^0 prior path flows.

$n(r;\mathcal{SC})$ number of routes that can be distinguished from route r using the set $\mathcal{SC}$ of scanned links.

n_0 size of the virtual prior sample.

n_r sample size corresponding to route r.

nc number of scanned links.

q_a constant measuring the flow producing a given congestion level in

	the link a.
r	route
t_i	OD-pair i flows.
T_i	number of users traveling through the OD-pair i.
v_a	unobserved link flows.
$\hat{v}_a$	link count information.
V_{ai}	number of users using link a and traveling the OD-pair i.
V_a	number of users using link a.
$\hat{w}_r$	flow associated to route r observed by plate scanning technique.
W_s	number of users passing through and only through a given subset $\mathcal{A}_s$ of links.
X	random variable.
x^0_{rs}	s element in a virtual sample.
$\mathcal{A}_s$	subset of observed links.
$\mathcal{F}$	feasible set of scanned links observations.
$\mathcal{H}(\alpha,\theta,\lambda)$	shifted-Gamma density function.
$\mathcal{R}$	set of routes.
$\mathcal{R}_i$	set of paths of each OD pair i.
$\mathcal{SC}$	set of links to be scanned.

6.2 Introduction

Predicting traffic flow is one of the main aims of traffic models. Two important problems in the traffic field are: the traffic assignment and the trip matrix estimation problems. In the first, the origin-destinations (OD) flows are assumed to be known and the link flows are sought, while in the second, some observations (link flows, path flows, plate scanned flows, etc.) are taken and the OD-flows are estimated.

Though there are many variants, the traffic assignment problem has been traditionally dealt with primarily using two different methods: the User Equilibrium (UE) model and the Stochastic User Equilibrium (SUE) model, as we have explained in Chapters 3 and 4.

In the matrix estimation (ME) problem, we observe traffic in some links and try to estimate the OD-pair flows (see Ashok and Ben Akiva (2000), Ben-Akiva and Lerman (1985), Tobin and Friesz (1988), Yang et al. (1992), Castillo et al. (2008a), Castillo et al. (2008d), etc.). Given independent error-free link flows, if the number of OD pairs is larger than the number of links, the problem is under-specified and there is an infinite set of solutions for the OD-pair flows satisfying the conservation laws. In order to have a unique solution, one has to give more information. Normally, people use a prior OD trip matrix, which can come from many different sources, including an old out-of-date, another OD matrix obtained by an alternative method, etc.

To obtain the OD flow estimates a wide variety of models can be utilized. They can be classified in two main groups:

Optimization based methods. Among these methods we can mention:

- *Least squares and generalized least squares methods* minimize the sum of squares of the differences between the predicted and the prior OD trip matrices (see Cascetta (1984), Carey and Revelli (1986), Cascetta and Nguyen (1988), Bell (1991), and Doblas and Benitez (2005)).

- *Entropy or information based methods* determine the most likely OD flow matrix that is consistent with the information contained in the observed link flow data (see Willumsen (1978), Van Zuylen (1978), Van Zuylen and Willumsen (1980), and Rossi et al. (1989)).

Statistical based methods. With respect to these methods we distinguish between:

- *Classical statistical methods* assume the traffic flows to be multivariate random variables from a given family, such as multivariate normal, Poisson (Lo et al. (1996), Vardi (1996) and Hazelton (2000)), gamma, etc. For estimation purposes normally they maximize the likelihood function (see also Spiess (1987) and Cascetta and Nguyen (1988), or Bayesian networks as Sun et al. (2006) and Castillo et al. (2008d,e)).

- In the *Bayesian methods*, the parameters themselves are considered as random variables. For example, Maher (1983) combines a multivariate normal distribution with a multivariate normal prior and considers random normal errors in observations, Tebaldi and West (1998) consider Poissonian flows, and citeMahmassaniS:81 deal with the problem of updating trip generation parameters.

While in the ME problem the route choice proportions or the link flows in the road network are assumed to be the input data and the trip flow matrix, the output, in the traffic assignment problem, the trip flow matrix is the input and the route choice proportions, the output. Thus, *bi-level approaches* that couple together both problems, have been proposed to solve some inconsistencies in the solutions of both problems when they are solved independently. Some examples are Yang (1995) and Maher and Zhang (1999) and Maher et al. (2001), Fisk (1988) or Lo and Chan (2003).

In this chapter we deal with the problem of estimating traffic flows in general: that is, to estimate origin-destination trip matrices (ME), path and link flows based on plate scanning.

In order to estimate traffic flows, some authors propose using probe vehicles, as Hellinga and Van Aerde (1994), who provide a statistical analysis of the OD estimators, Gentili and Mirchandiani (2005) who discuss the use of path-ID sensors, Zhou and Mahmassani (2006) or Asakura et al. (2000), who use AVI counts, or Bianco et al. (2006) who use node sensors. However, in this chapter we use regular vehicles without any special detectors installed.

The plate scanning approach to traffic flow estimation has become frequently used, due to the considerably larger amount of information it contains when compared with other standard methods, such as Watling (1994) or Castillo et al. (2008c)), and the advantage of being able to use the information gathered by surveillance and traffic control systems based on *Automatic Number Plate Recognition* (ANPR). The idea is to register the plate numbers of the circulating vehicles together with the corresponding times on some subsets of links and use this information to reconstruct vehicle routes from which OD and link flows are evaluated.

Following Hazelton (2001) we consider the two different problems:

- *The reconstruction problem.* This kind of problem tries to assess “the actual number of trips between each OD pair that occurred during the observational period”, that is, “the most probable set of OD movements that are consistent with a set of link flows” according to Bell (1983)

- *The estimation problem.* This problem aims to estimate the *expected* (or mean) number of OD trips based on several observations.

For the estimation problem, we combine the advantages of Bayesian methods with those of the plate scanning technique and treat the traffic flow problem as an integrated one in which we include OD-pairs, paths and link flows as related flows. We present a general conjugate Gamma Bayesian model that permits updating the model parameters very easily when information is available.

Maher (1983) assumes a multivariate normal distribution for the OD flows and observation link errors, and calculates the posterior density of the OD flows given the observations. Compared with Maher’s method, which is a well-based model because it uses a multivariate random variable, reproductive, infinitely divisible and justified from the point of view of approximations to other models, such as the binomial and Poisson, the proposed method uses gamma random variables which in addition to sharing these good properties with the normal model, are positive. In addition to point estimates of the parameters, as with Maher’s method, credible intervals can be obtained for path, OD and link flows, which is a clear advantage with respect to traditional traffic assignment models that do not supply information about the variability of the parameter estimates.

The chapter is organized as follows. In Section 6.3, we illustrate the plate scanning approach. Once this has been explained, in Section 6.4, we introduce the trip matrix estimation problem in a general way. In Section 6.5, we explain how a new model for estimating traffic flows based on the shifted-Gamma function can be applied and, in Section 6.6, how this model can be implemented in practice. With the purpose of illustrating

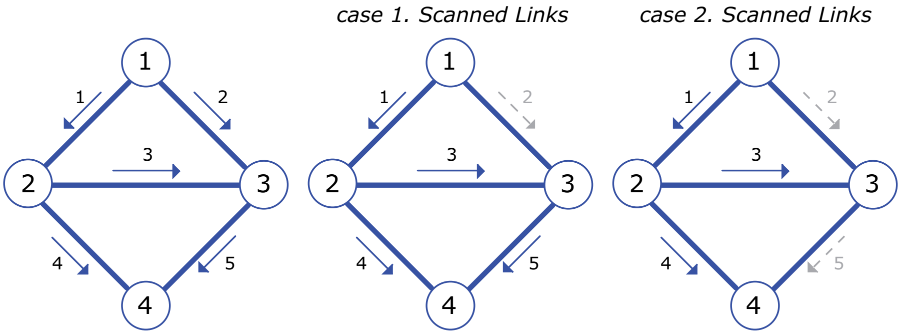

Figure 6.1: The elementary example network used for illustrative purposes, showing the nodes, links and the set of scanned links (continuous arrows).

the proposed methods, they are applied to the Nguyen-Dupuis network and the Cuenca network, in Sections 6.7 and 6.8, respectively. Finally, some conclusions are given in Section 6.9.

6.3 Traffic reconstruction based on plate scanning

In this section we illustrate the plate scanning approach. The idea consists in registering plate numbers and the corresponding times of the circulating vehicles at some subsets of links to reconstruct vehicle routes[1]. In this chapter we assume error-free scanned observations.

With the purpose of illustrating the concepts and the proposed methods, we use a simple network. It consists of 4 nodes and 5 unidirectional links (see Figure 6.1).

Table 6.1 shows the 4 OD-pairs considered and the corresponding 7 paths used in this example, which are the only possible paths for each OD pair due to the unidirectional links.

We assume that we observe the set $\mathcal{SC} \equiv \{1, 3, 4, 5\}$ (see the continuous arrows in Figure 6.1).

Once the set $\mathcal{SC}$ of scanned links is known, the next step consists in obtaining the feasible set of scanned link observations, set $\mathcal{F}$. Following Castillo et al. (2009), the set $\mathcal{F}$ of all subsets of scanned link feasible observations is determined by checking which links in $\mathcal{SC}$ are contained in each path, i.e., $\mathcal{F} \equiv \cup_{r \in \mathcal{R}_i} (\mathcal{SC} \cap C_{ir})$, where C_{ir} is the set of links in route r of OD-pair i, and $\mathcal{R}_i$ is the set of paths of each OD pair i. Consequently, if we had cameras on links $1, 3, 4$ and 5, then we would have counts on the sequences: $\{1\}$, $\{3\}$, $\{4\}$, $\{5\}$, $\{1, 4\}$, $\{1, 3, 5\}$, and $\{3, 5\}$, see the central part of Table 6.1. Therefore, the flow

[1] The times are used only to derive the order in which the scanned links were traveled.

			$\mathcal{SC}_1$						$\mathcal{SC}_2$				
			Scanned links						Scanned links				
OD	route code (r)	Links	1	3	4	5	$\mathcal{F}$	$n(r;\mathcal{SC}_1)$	1	3	4	$\mathcal{F}'$	$n(r;\mathcal{SC}_2)$
1-4	1	1 3 5	X	X		X	$\{1,3,5\}$	6	X	X		$\{1,3\}$	6
1-4	2	1 4	X		X		$\{1,4\}$	6	X		X	$\{1,4\}$	6
1-4	3	2 5				X	$\{5\}$	6				$\{-\}$	0
2-4	4	3 5		X		X	$\{3,5\}$	6		X		$\{3\}$	5
2-4	5	4			X		$\{4\}$	6			X	$\{4\}$	6
1-2	6	1	X				$\{1\}$	6	X			$\{1\}$	6
2-3	7	3		X			$\{3\}$	6		X		$\{3\}$	5

Table 6.1: Set of 4 OD-pairs, 7 routes considered in the elementary example, and scanned link feasible observations and associated values of r for the sets $\mathcal{SC}_1 = \{1,3,4,5\}$ and $\mathcal{SC}_2 = \{1,3,4\}$.

associated with each subset, $\hat{w}_r$, can be obtained.

Note that not all $2^{nc}-1$ possible combinations of scanned links are feasible for a given network and set of routes, being nc the number of scanned links (elements of set $\mathcal{SC}$). Clearly sequences $\{1,3,4\},\{1,4,5\},\{3,4\},\{3,4,5\},\{4,5\}$ and $\{1,3,4,5\}$ are not feasible in that no vehicle in this network with the routes in Table 6.1 can pass the cameras in that order.

Hence, with scanned data from certain links, we get more information (7 flow values, equivalent to 7 linear constraints) than simple counts on those links (4 flow values, equivalent to 4 linear constraints). This is extremely important because the OD estimation problem is usually under-specified, that is, the number of unknowns (OD or path flows) is much larger than the number of data items (link flows), although this does not happen in this simple example. Thus, the plate scanning technique, assuming measures to be error-free, permits the reduction of the degree of underspecification and eventually (with adequately selected scanned links) it becomes possible to completely estimate the OD and/or the path flows[2].

We point out that if some link is removed from the subset of scanned links $\mathcal{SC}$, complete route identification is impossible. The number of routes that can be distinguished from route r_1 using the set $\mathcal{SC}$ of scanned links was expressed by Castillo et al. (2009) as:

$$n(r_1;\mathcal{SC}) = \sum_{r_2 \neq r_1} \min\left(\sum_{a\in\mathcal{SC}} (\delta_a^{r_1}+\delta_a^{r_2})(1-\delta_a^{r_1}\delta_a^{r_2}), 1\right); \quad \forall r_1 \in \mathcal{R} \tag{6.1}$$

where δ_a^r is the path-link incidence matrix ($\delta_a^r = 1$ if link a belongs to route r, and $\delta_a^r = 0$

[2] Note that OD estimation requires fewer scanned links than route estimation.

otherwise). Note that if $n(r_1; \mathcal{SC}) = |R| - 1$, where $|R|$ is the cardinal of the route set $\mathcal{R}$, the route r_1 can be distinguished from all other routes by $\mathcal{SC}$.

For example, if we had cameras on links $1, 3$ and 4, we would obtain a new set $\mathcal{F}'$ (see Table 6.1). As Table 6.1 shows, in this case we cannot know the path flow 3 (empty set), and the flow of routes 4 and 7 are mixed due to them sharing the same sub-set of $\mathcal{F}'$. However, we have more information (5 flow values) than the case with link counts alone (3 flow values).

Therefore, given a set of scanned links in any network, all the possible feasible sequences (sub-paths) in which a vehicle can pass these cameras in that order can be constructed, and identifying such a sequences, the path flows contributing to the total flow attributed to that sequences can be identified too. Each of these sequences then provides a linear equality constraint on the path flows.

On the other hand, if together with the plate scanning approach we select the set of scanned links by means of optimization methods (see Castillo et al. (2008e,c)), the set $\mathcal{SC}$ of scanned links can become sufficient to identify all routes because every route has associated with it a different set of scanned links, as Table 6.1 shows. Note that given a car, we can determine its set of scanned links and then identify its route. Thus, that subset $\mathcal{SC}$ would be minimal for a complete identification of routes, and will allow estimating exact flows.

6.4 Matrix estimation problem based on plate scanning

Once the concept of plate scanning has been explained, we introduce the trip matrix estimation problem in a general way. The main objective of the trip matrix estimation problem is to find an OD matrix $\mathbf{T}$, close to a prior OD matrix $\bar{\mathbf{T}}$ subject to the assignment method, i.e a matrix coherent with the users behavior (how they selected the adequate paths to go from origins to destinations).

Based on the observations and the prior matrix, the OD matrix estimate can be obtained by many different methods; in particular, we work with the scanned link information ($\hat{w}_r$). These data can be used for reconstructing path flows and trip matrices, using the following model, which uses a quadratic objective function with the weighted sum of the differences between the predicted and the prior path flows:

$$\underset{f_r;\, r\in\mathcal{R}_i}{\text{Minimize}}\, Z = \sum_{r_1\in\mathcal{R}_{i_1}, r_2\in\mathcal{R}_{i_2}} \left(f_{r_1} - f^0_{r_1}\right) \gamma_{r_1 r_2} \left(f_{r_2} - f^0_{r_2}\right) \tag{6.2}$$

subject to

$$\hat{w}_r = \sum_{r\in\mathcal{R}} f_r\, \eta_r; \quad r \in \mathcal{R} \tag{6.3}$$

$$\hat{v}_a = \sum_{r\in\mathcal{R}} f_r\, \delta^a_r; \quad a \in \mathcal{A}_s, \tag{6.4}$$

$$f_r \geq 0, \tag{6.5}$$

where r refers to route, $\mathcal{R}$ is the set of routes, $\mathcal{R}_i$ is the set of feasible routes of OD-pair i, f_r are the path flows, f_r^0 are the prior path flows, $\gamma_{r_1 r_2}$ are the weights (normally the elements of the inverse of a covariance matrix), $\mathcal{A}_s$ is the subset of observed links, δ_r^a is 1 if route r contains link a and 0, otherwise, and η_r is one if the route r contains all and only the links in C_r (scanned sub-route of set $\mathcal{F}$), 0, otherwise.

The proposed model is able to work with two types of observations, which correspond to (i) the plate scanning link case ($\hat{w}_r$), and (ii) the usual link count information ($\hat{v}_a$), since the nature of information is the same for both set of data. However, note that constraints (6.4) for scanned links are redundant given constraints (6.3).

Alternatively, equation (6.2) can be replaced by

$$\underset{f_r;\, r\in\mathcal{R}_i}{\text{Minimize}}\, Z = \sum_i \left(\frac{\sum\limits_{r\in\mathcal{R}_i} (f_r - f_r^0)}{\sum\limits_{r\in\mathcal{R}_i} f_r^0} \right)^2 . \tag{6.6}$$

Without loss of generality, the matrix $\gamma_{r_1 r_2}$ can be assumed to be symmetric[3]. So, we make this assumption in this chapter. Note that the gamma γ weight matrix is used to take into account not only the variance but the covariance structure of the path flows. However, to keep things simple, we have considered the objective function in (6.2) as a least squares expression with weights

$$\gamma_{r_1 r_2} = \begin{cases} 1/\sigma_{r_1}^2 & \text{if} \quad r_1 = r_2 \\ 0 & \text{otherwise} . \end{cases}$$

Once the path flows have been reconstructed as $\hat{f}_r$, the OD flows t_i and the unobserved link flows v_a can be obtained as follows

$$v_a = \sum_{r\in\mathcal{R}} \hat{f}_r\, \delta_r^a; \quad a \notin \mathcal{A}_s, \quad t_i = \sum_{r\in\mathcal{R}_i} \hat{f}_r; \quad i \in \mathcal{OD}. \tag{6.7}$$

6.5 Proposed model for estimating traffic flows

In this chapter we propose a new model for estimating traffic flows based on the shifted-Gamma function described in Section 5.3.1. In this section we explain how the model can be applied. First, we start by discussing the following simplifying assumptions

[3] If it is not, we can make $\gamma^*_{r_1,r_2} = \gamma^*_{r_2,r_1} = (\gamma_{r_1 r_2} + \gamma_{r_2 r_1})/2$, where $\gamma^*_{r_1,r_2}$ are the elements of a symmetric matrix leading to an identical quadratic form to that associated with the initial non-symmetric matrix with elements $\gamma_{r_1 r_2}$.

and later, we analyzed some of the properties of the model.

Assumptions

1. The random amount of users F_r choosing route r follows a distribution in the family:

   $$\mathcal{F} \equiv \mathcal{H}(\alpha, \theta, \lambda)|\alpha \geq 0, \theta > 0, \lambda > 0\},$$

   i.e., $F_r \sim \mathcal{H}(\alpha_r, \theta_r, \lambda)$, where λ is the same for all routes. We note that this family $\mathcal{H}(\alpha, \theta, \lambda)$ of distributions has good properties, because the associated random variables are positive (as the number of users), it is reproductive (the sum of $\mathcal{H}(\alpha, \theta, \lambda)$ distributions in $\mathcal{F}$ belongs to $\mathcal{F}$), and it is infinitely divisible. In addition, it can have positive skewness as the observed traffic flows.

   The location α parameter is needed to reproduce flows whose lower bound is a positive number (a common case in practice). We note that using a $\mathcal{H}(\alpha, \theta, \lambda)$ probability model does not imply a prescribed link between the mean and variance of traffic flows, which could be not true in some cases (see Hazelton (2003)), as it occurs with Poisson models used by several authors.

   We denote $\mathcal{H}(\boldsymbol{\alpha}, \boldsymbol{\theta}, \lambda)$ to the model including all routes, where $\boldsymbol{\alpha}$ and $\boldsymbol{\theta}$ are the vectors of all $\mathcal{H}(\alpha, \theta, \lambda)$ parameters for all routes and λ is identical for all routes.

   For a given route there exists a minimum number of users (normally much larger than zero) that can be estimated, and since it is very far from zero, forcing it to be zero is a great limitation. This is the justification of the α parameter. Note that when a normal distribution $N(\mu, \sigma)$ is used, the probability of having flows smaller than $\mu - 3\sigma$ is very small, and this is practically equivalent to assuming a lower bound in $\mu - 3\sigma$ even though this value is well above zero. Since the α parameter can be estimated for each route, if in some routes it is convenient to use the value zero, we can use it.

2. All components of the multivariate shifted-Gamma random variable $U_i = (F_1, \ldots, F_{R_i})$, where R_i is the number of routes of OD i, are independent. This means that the number of users choosing the different routes of a given OD pair is independent. Note that here we are not assuming a given fixed number of OD users, in which case the independence assumption could be questioned.

3. The multivariate random variables $U_1, U_2, \ldots, U_I$, where I is the set of OD pairs, are independent. This implies that the users of different OD-pairs act independently.

These assumptions lead to the following derived properties.

Assumptions 2 and 3 above will be invalid if there are some effects inducing serial correlation in the path flow data or if significant events affect the network. In such a cases, some data preprocessing would be necessary to convert data before using the proposed methods below.

Derived Properties

1. The random number of users $T_i = \sum_{r \in \mathcal{R}_i} F_r$, where $\mathcal{R}_i$ is the set of routes of OD-pair i, traveling through the OD-pair i follows a $\mathcal{H}\left(\rho_i, \nu_i, \lambda\right)$ distribution, where $\rho_i = \sum_{r \in \mathcal{R}_i} \alpha_r$ and $\nu_i = \sum_{r \in \mathcal{R}_i} \theta_r$, i.e. it belongs to family $\mathcal{F}$ too.

2. The number of users V_{ai} using link a and traveling the OD-pair i is a $\mathcal{H}\left(\rho_i^a, \theta_i^a, \lambda\right)$ distribution, where $\rho_i^a = \sum_{r \in \mathcal{R}_i} \alpha_r \delta_r^a$ and $\theta_i^a = \sum_{r \in \mathcal{R}_i} \theta_r \delta_r^a$, which belongs to family $\mathcal{F}$ and δ_r^a is 1 if link a belongs to a route r, and 0 otherwise.

3. The number of users $V_a = \sum_i V_{ai}$ using link a is a $\mathcal{H}\left(\sum_i \rho_i^a, \sum_i \theta_i^a, \lambda\right)$ distribution, which belongs to family $\mathcal{F}$.

4. The number of users W_s passing through and only through a given subset $\mathcal{A}_s$ of links has a $\mathcal{H}(\sum_{r \in R(\mathcal{A}_s)} \alpha_r, \sum_{r \in R(\mathcal{A}_s)} \theta_r, \lambda)$ distribution, where $R(\mathcal{A}_s)$ is the subset of routes passing through and only through all links in $\mathcal{A}_s$.

Since the proposed model is very general, it can be applied not only to traffic but to other fields.

6.5.1 Conjugate of $\mathcal{F}$ families

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 of the conjugate family, are referred to as *hyperparameters*.

Since the likelihood of the $\mathcal{H}(\boldsymbol{\alpha}, \boldsymbol{\theta}, \lambda)$ family is

$$L(\mathbf{x}; \boldsymbol{\theta}, \boldsymbol{\alpha}, \lambda) \doteq \exp\left[\sum_{r \in \mathcal{R}} \left\{(\theta_r - 1)\sum_{i=1}^{n_r} \log(x_{ri} - \alpha_r) - \lambda \sum_{i=1}^{n_r} x_{ri} + n_r(\lambda\alpha_r - \log\Gamma\left(\theta_r\right) + \theta_r \log\lambda)\right\}\right]; \quad \alpha_r \leq \min\left(x_{r1}, x_{r2}, \ldots, x_{rn_r}\right), \tag{6.8}$$

where $x_{r1}, x_{r2}, \ldots, x_{rn_r}$ are sequences of independent and identical observations on the path, link or OD flow of interest, the results of Arnold et al. (1993, 1996) and Castillo et al. (1996) suggest the following conjugate family as a general conjugate family with active[4]

[4] A hyperparameter is said active if its updated posterior value depends on the sample. Otherwise it is said inactive.

hyperparameters:

$$q(\boldsymbol{\theta},\boldsymbol{\alpha},\lambda|\boldsymbol{\eta},\boldsymbol{\xi},\boldsymbol{\rho}) \doteq \exp\left[\sum_{r\in\mathcal{R}}\left\{(\theta_r-1)\sum_{s=1}^{\eta_r}\log(\xi_{rs}-\alpha_r) - \lambda\rho_r + \eta_r(\lambda\alpha_r - \log\Gamma(\theta_r) + \theta_r\log\lambda)\right\}\right],$$
$$\alpha_r \le \min(\xi_{rs},\dots,\xi_{r\eta_r}), \tag{6.9}$$

where the posterior active hyperparameters $(\hat{\boldsymbol{\eta}},\hat{\boldsymbol{\xi}},\hat{\boldsymbol{\rho}})$, denoted by the hat, become

$$\hat{\eta}_r = \eta_r + n_r; \quad \hat{\xi}_{rs} = \begin{cases} x_{rs} & \text{if} \quad s \le \hat{\eta}_r \\ 0 & \text{otherwise} \end{cases}, \quad \hat{\rho}_r = \rho_r + \sum_{i=1}^{n_r} x_{ri}, \tag{6.10}$$

where η_r and ρ_r are the prior values, n_r is the sample size and $\{x_{r\ell} : \ell = 1,2,\dots,\hat{\eta}_r\}$ is the sample corresponding to route r, including the virtual sample used for prior assessment (see Section 6.6.1). Note that Expression (6.10) is the hyperparameter updating formula for this model.

Note that our prior does not produce a conjugate family in the usual sense, since the parameter vector ξ increases in dimension with the number of observations n_r, but we can consider an asymptotic family with an infinite number of parameters including all of them.

Since $\sum_{\ell=1}^{n}\log(x_\ell - \alpha)$ and $\sum_{\ell=1}^{n} x_\ell$ tend to ∞ as $n \to \infty$, the effect of prior information $(\boldsymbol{\eta},\boldsymbol{\xi},\boldsymbol{\rho})$ could vanish as $n \to \infty$. However, we must take some care, because if $\xi_{rs} = x^0_{rs} < \alpha_{r,true}$ (where $\alpha_{r,true}$ indicates the true value of the shift parameter for route r) then the support of the posterior distribution of α_r will never include the true value. This issue can be dealt with by selecting cautious, large values of $x^0_{rs}(s = 1,2,\dots,\eta_r)$ but this will introduce bias into point estimates for finite sample sizes.

Due to its complexity (exponential and gamma functions appear in it), the mean of the gamma conjugate distribution (6.9) cannot be obtained in closed form. Thus, as an alternative to the mean, we use the mode of (6.9) to estimate the Gamma parameters, i.e., we maximize (6.9), with respect to all parameters, to estimate them.

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) should not be considered to be a worse criterion, especially because the maximum likelihood criterion is generally accepted in Statistics.

Even though the gamma Bayesian model is well known by statisticians, there are some implementation details, which are not so well established and without them, one can face important numerical problems. Some of the ideas, which can be useful to people wishing to implement these models, are the following:

- Maximize the logarithm of (6.9) instead of (6.9) 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 $\chi_r^2 = \theta_r$ to guarantee non-negativity of the parameters.

- From a practical point of view it is convenient to replace the parameter α_r by the expression:
$$\alpha_r = \min(x_{r1}, x_{r2}, \dots, x_{r\eta}) - \beta_r^2, \tag{6.11}$$
with $\beta_r \in \mathbb{R}$. This permits ignoring the constraints
$$0 \leq \alpha_r \leq \min(x_{r1}, x_{r2}, \dots, x_{r\eta}).$$

- For the initial θ-estimates, which are required by any non-linear maximization procedure, one can use moment estimators.

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. However, this is not a problem, because if scanners have been installed, since all recordings are automatic, one could in theory collect as many observations as required. Some practical method for the prior assessment is given in Section 6.6.

6.6 Practical implementation of the proposed method

In this section we explain how the proposed traffic model can be implemented in practice. First, we point out that since T_i, F_r, V_a and W_s are $\mathcal{H}(\alpha, \theta, \lambda)$ random variables, we can use the Bayesian approach with any of them[5]. However, to apply this approach, we need to observe a sample of the corresponding random variable. In fact, we observe directly only the W_s and some link variables V_a, and calculate or estimate from them T_i, F_r and the remaining V_a variables. If the calculation is possible, this is equivalent to a direct observation, and the Bayesian approach is valid. Otherwise, the sample is only an estimate and then the Bayesian estimate is only approximate. This method is practically illustrated in Sections 6.7 and 6.8 by its application to the Nguyen-Dupuis and the Cuenca networks.

In the following section when we refer to a sample, we refer indistinctly to a sample of T_i, F_r, V_a or W_s, depending on the random variable we are dealing with. Thus, the proposed methods are valid for any of those flows.

[5] Nevertheless, if we estimate or reconstruct the path flows F_r, then all other variables T_i, V_a and W_s can be calculated from these.

6.6.1 Prior assessment in practice

For the proposed method it is more convenient to have informative priors for traffic flows in order to obtain good results when the data is not sufficient for a good reproduction of the traffic flows. This is a consequence of the under-specification of the traffic problem.

The prior assessment can be done by the following methods:

1. *Assessment of priors from a virtual sample of path flows.* Following Klieter (1992), by means of an imaginary sample, i.e., we ask the human expert to provide virtual samples of sizes n_{r0} as the most representative of his/her knowledge. Once this sample is known, we use (6.10) to get the prior hyperparameters:

$$\begin{aligned} \eta_r &= n_{r0} && (6.12)\\ \xi_{rs} &= x^0_{rs}, \quad 1 \le s \le n_0, && (6.13)\\ \rho_r &= \sum_{s=1}^{\eta_r} x^0_{rs}, && (6.14) \end{aligned}$$

   where x^0_{rs} is the s element in the sample, and the superindex 0 refers to virtual prior sample.

   Note that the prior values have negligible effect for large n.

2. *Assessment of priors based on an out-of-date route information.* Another possibility consists in using a prior or out-of-date route information x_r. Then the prior hyperparameters are also given by (6.12) to (6.14).

6.6.2 Sampling traffic flows

In Bayesian methods we need a sample of the random variable being involved. As indicated in this chapter, we consider plate scanning as the main sampling technique. This means that we directly obtain w_r for $r \in \mathcal{R}$, the set of feasible combinations of scanned links, and the corresponding links. However, the information about routes, OD-pairs and the unobserved links can be obtained exactly or approximated as follows.

For estimating the path flows based on plate scan of a subset of link flows, we use the model presented in Section 6.3, with objective function (6.2) subject to (6.3) and (6.5)[6], or with objective function (6.6) subject to (6.3) and (6.5). Likewise, in the reconstruction problem, once the path flows have been estimated, $\hat{f}_r$, the OD flows t_i and the unobserved link flows v_a can be obtained by means of equations (6.7).

We note that in order for the Bayesian techniques to be fully valid, the route, OD-pair and links flows involved must be uniquely determined. To this end it is interesting to obtain which sets of flows satisfy this property when insufficient number of links have

[6] Remember that constraints (6.4) for scanned links are redundant given constraints (6.3).

been scanned. This can be done using the techniques in Castillo et al. (2008a,b). Two examples of these techniques are illustrated in Sections 6.7 and 6.8. If the flows involved in the Bayesian method are not uniquely determined, we can use the estimates, but in this case the results must be interpreted with care, because the uncertainties in these values have not been taken into account.

6.6.3 Obtaining posteriors

Once the sample of the random variable being considered has been obtained, the posterior $\hat{\eta}$ hyperparameters can be calculated in terms of the prior η parameters as

$$\begin{aligned} \hat{\eta}_r &= \eta_r + n_r && (6.15)\\ \hat{\xi}_{rs} &= x_{rs}, \quad 1 \le s \le \hat{\eta}_r, && (6.16)\\ \hat{\rho}_r &= \rho_r + \sum_{i=1}^{n_r} x_{ri} && (6.17) \end{aligned}$$

where x_{rs} are the sample elements of the random variable, that is, T_i, F_r, V_a or W_s, including the virtual sample used for prior assessment.

Note that $\eta_r = n_0$, which is the size of the virtual prior sample, measures the relative weight of the human expert information with respect to the information contained in a real sample of size n. For example, if $n_{r0} = n_r$, they have the same associated amount of 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 on the prior is not recommended, unless the plate scanned information is complete or almost complete.

6.6.4 Bayesian estimates of flows

The proposed model for traffic flow estimation uses the family of conjugate distributions for the $\mathcal{H}(\alpha, \theta, \lambda)$ family associated with the corresponding random variables T_i, F_r, V_a and W_s, taking into account that we select the posterior mode for estimating the parameters, the estimation problem can be stated as:

$$\underset{\boldsymbol{\theta},\boldsymbol{\alpha},\lambda}{\text{Maximize}} \left[\sum_{r\in\mathcal{R}} \left\{ (\theta_r - 1) \sum_{s=1}^{\hat{\eta}_r} \log(\hat{\xi}_{rs} - \alpha_r) - \lambda\hat{\rho}_r + \hat{\eta}_r(\lambda\alpha_r - \log\Gamma(\theta_r) + \theta_r \log\lambda) \right\} \right], \quad (6.18)$$

where we have assumed a prior with independent components for the traffic flow parameters, which can be recognized as the maximum likelihood estimate based on the extended sample (prior plus real sample).

Finally to clarify, we explicitly indicate that the outputs of the proposed model are the posterior distributions, from which one immediately can obtain point estimates (the modes) and/or credible intervals. If point estimates are used, they provide the α, θ and λ parameters of the $\mathcal{H}(\alpha, \theta, \lambda)$ corresponding densities, and then probability intervals for the flows can be obtained. As an alternative, we can also use the Bayesian distribution (5.9).

6.6.5 Variability of parameter estimates and predictions

The posterior distributions can also be used for deriving estimates of the parameter estimates variability. One of the weak points in traditional traffic assignment models is that they produce a point prediction for the traffic flow vector. This is often interpreted as some kind of estimate of the mean flow pattern. However, from a system performance perspective it is of paramount importance. Estimates of variability are therefore highly desirable. One of the advantages of a statistical approach is the availability of such estimates (e.g. OD flow credible intervals). Using the posterior distributions of the parameter estimates one can obtain the credible intervals for the parameter estimates, an advantage with respect to other traffic flow prediction or estimation methods. However, this is valid only for flows uniquely determined by the scanned links. Otherwise, the credible intervals will be unconservative (too narrow) because they ignore a source of variation (uncertainty associated with the unobserved flows).

When the number of scanned links is sufficient to cover all routes of the network, the problem is not simply to estimate the population mean by the sample mean $\alpha + \theta/\lambda$ using link scanned data, because it is a Bayesian approach, but to provide the posterior distributions, from which the Bayesian distributions can be calculated, and credible intervals too.

6.6.6 Proposed algorithm

The following algorithm is proposed for reproducing and estimating the $\mathcal{H}(\alpha, \theta, \lambda)$ model in the sense of Hazelton (2001). Though it is described for path flows F_r, the model is directly applicable to T_i, V_a or W_s flows.

Step 0: Initialization: Obtain a set of links to be scanned and identify the feasible set of observed scanned link combinations. This set can be obtained from a previous set of cameras in the network, or the optimal set of links to be scanned can be obtained using the methods in Castillo et al. (2008c).

Next, the set of all observable scanned link combinations is determined by the intersection of the set of scanned links $\mathcal{SC}$ with the set of links in each route. If no intersection set is repeated when considering all routes, the corresponding set of observed links will allow us to identify all path flows; otherwise, the path flows with identical intersection sets will be confounded.

Step 1: Obtain the virtual prior sample of F_r^0. We assume that we have an out-of-date trip matrix t_i^0, which is used to solve the SUE problem to obtain the corresponding reference virtual prior sample of f_r^0. Once the flows in $\{f_r^0; r \in \mathcal{R}\}$ are known, the link or node flow prior samples can be obtained from them.

Step 2: Identify the sets of route, OD and link flows uniquely determined. This can be done using the techniques in Castillo et al. (2008a,b) or Tables such as 6.1. Then, we know to which flows the proposed Bayesian technique can be applied and for which flows this technique provides only approximate estimates.

Step 3: Obtain the F_r priors. To assess the prior hyperparameters, we use (6.12)-(6.14) for the $\mathcal{H}(\alpha, \theta, \lambda)$ variables F_r with the virtual sample in Step 1, replicating n_0 times each flow[7]. Of course, we can also assign different n_0 values for each path flow.

Step 4: Get the plate scanned observations and reconstruct path flows. The observations $\hat{w}_r$ and the associated subset of link flows $\hat{v}_a$ are directly obtained, that is, we directly get the two types of data: scanned data and count data. Since the path flows are not directly observed, we solve the problem (6.2)-(6.3) and (6.5) (or the problem (6.6) subjected to (6.3) and (6.5)) using the prior values f_r^0 and the observations $\hat{w}_r$, to obtain the path flows f_r. Finally, using (6.7) we obtain the unobserved flows v_a and the OD flows t_i. Note that this step corresponds to the reconstruction problem in the sense of Hazelton (2001).

Step 5: Obtain the F_r posteriors. We use Equations (6.15) to (6.17) to obtain the F_r posteriors.

Step 6: Obtain the mode posteriors of F_r. The model parameters are estimated by solving the problem (6.18), that is, obtaining the mode posteriors.

Step 7: Use of the $\mathcal{H}(\alpha, \theta, \lambda)$ model. Finally, we can use the $\mathcal{H}(\alpha, \theta, \lambda)$ model with the posterior modes as parameters, or the Bayesian distribution to represent the traffic flows. In particular, we can use these distributions to obtain probability intervals for the corresponding random variables. Alternatively, for uniquely determined flows we could obtain the corresponding credible intervals of the Bayesian distribution (mixture of $\mathcal{H}(\alpha, \theta, \lambda)$ distributions using the posterior as weight function).

[7] The value of n_0 must be selected depending on the weight one wants to give to the prior information.

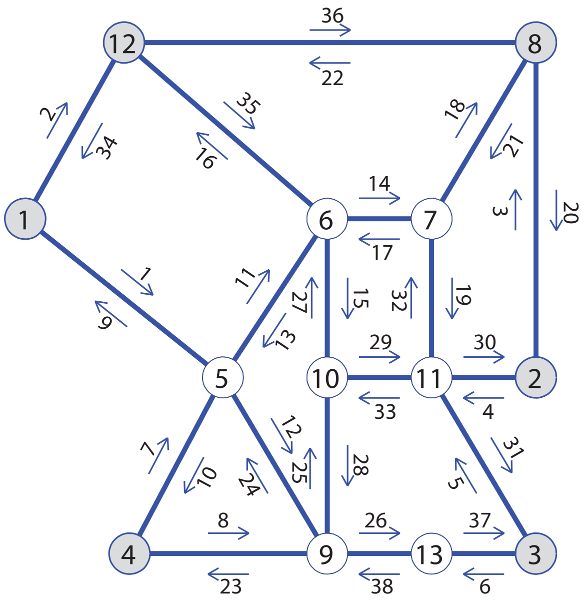

Figure 6.2: The Nguyen Dupuis network used for illustrative purposes, showing the nodes, links and the set of scanned links (with outlined arrows). The origin-destination nodes have been shown shaded.

6.7 Example: The Nguyen-Dupuis network

With the purpose of illustrating the proposed methods, they are applied to the Nguyen-Dupuis network topology, which consists of 13 nodes and 38 links. We assume the existence of bidirectional flow, i.e. any pair of nodes i and j is connected in both directions (see Figure 6.2). The $\mathcal{H}(\alpha, \theta, \lambda)$ model was already justified in Section 6.5, Assumption 1. To test the proposed methods, six origin-destination nodes have been assumed (they are shown shaded in Figure 6.2), together with 18 OD-pairs (all possible combinations between the three left nodes $\{12, 1, 4\}$ and the three right nodes $\{8, 2, 3\}$).

Given the nature of the investigation, it is crucial that the specification of OD pairs and routes is comprehensive, or at least clearly objective. Otherwise there is a real danger that the limited choice of OD/routes invalidates the findings of the study. So, in practical cases this must be done with care. We consider the already described six origin-destination nodes and OD-pairs; however any set of OD-pairs could be considered too.

In order to know how the demand is distributed by the network, we use a path-based SUE assignment model, with the multinomial logit assignment model with parameter $\theta = 0.5$ for the stochastic loading, because it gives a unique set of path flows. However, with the aim of getting a simple example, the paths have been generated by means of an objective process: UE assignment has been applied with the demands shown in Table 6.4

OD-pair	r	Paths
1-2	1	1 5 6 7 8 2
1-2	2	1 12 6 7 8 2
1-2	3	1 12 8 2
1-3	4	1 5 6 7 11 3
1-3	5	1 5 6 10 11 3
1-3	6	1 5 9 10 11 3
1-3	7	1 5 9 13 3
1-3	8	1 12 6 7 11 3
1-3	9	1 12 6 10 11 3
1-8	10	1 5 6 7 8
1-8	11	1 12 6 7 8
1-8	12	1 12 8
2-1	13	2 8 7 6 5 1
2-1	14	2 8 7 6 12 1
2-1	15	2 8 12 1
2-4	16	2 8 7 6 5 4
2-4	17	2 8 7 11 10 9 4
2-4	18	2 11 10 9 4
2-12	19	2 8 7 6 12
2-12	20	2 8 12
3-1	21	3 11 7 6 5 1
3-1	22	3 11 7 6 12 1
3-1	23	3 11 10 6 5 1
3-1	24	3 11 10 6 12 1
3-1	25	3 11 10 9 5 1
3-1	26	3 13 9 5 1
3-4	27	3 11 10 9 4
3-4	28	3 13 9 4
3-12	29	3 11 7 6 12
3-12	30	3 11 10 6 12
4-2	31	4 5 6 7 8 2
4-2	32	4 9 10 11 2
4-2	33	4 9 10 11 7 8 2
4-3	34	4 9 10 11 3
4-3	35	4 9 13 3
4-8	36	4 5 6 7 8
4-8	37	4 9 10 11 7 8
8-1	38	8 7 6 5 1
8-1	39	8 7 6 12 1
8-1	40	8 12 1
8-4	41	8 7 6 5 4
8-4	42	8 7 11 10 9 4
8-12	43	8 7 6 12
8-12	44	8 12
12-2	45	12 6 7 8 2
12-2	46	12 8 2
12-3	47	12 6 7 11 3
12-3	48	12 6 10 11 3
12-8	49	12 6 7 8
12-8	50	12 8

Table 6.2: Set of 18 OD-pairs and 50 routes (defined by nodes) considered in the Nguyen-Dupuis example.

(true mean OD flows). In this way, we obtain a specific restricted set of paths for each OD pair, and we are able to show the results in the form of tables and figures of reasonable size.

Once the UE assignment has been applied with the demands of Table 6.4 (column 4), 50 minimum cost routes are obtained, which are given in Table 6.2. We have used the Bureau of Public Roads (BPR) cost function

$$c_a(v_a) = c_{a0}\left[1+\beta_a\left(\frac{v_a}{q_a}\right)^{\gamma}\right],$$

where for a given link a, c_{a0} is the cost associated with free flow conditions, q_a is a constant measuring the flow producing congestion, and α_a and γ are constants defining how the cost increases with traffic flow. The network data used in this example are shown in Table

Link	c_a	q_a	Link	c_a	q_a	Link	c_a	q_a	Link	c_a	q_a
1 -5	7	700	5 -6	12	420	8 -7	9	700	11-3	8	560
1 -12	9	560	5 -9	9	420	8 -12	14	560	11-7	4	700
2 -8	9	700	6 -5	12	420	9 -4	5	375	11-10	4	700
2 -11	9	280	6 -7	5	700	9 -5	9	420	12-1	9	560
3 -11	9	560	6 -10	4	280	9 -10	5	280	12-6	7	140
3 -13	11	560	6 -12	6	140	9 -13	9	280	12-8	14	560
4 -5	12	560	7 -6	5	700	10-6	4	280	13-3	11	560
4 -9	5	375	7 -8	9	700	10-9	4	280	13-9	9	280
5 -1	7	700	7 -11	4	700	10-11	4	700			
5 -4	12	560	8 -2	9	700	11-2	9	280			

Table 6.3: Network parameters of the Nguyen-Dupuis network.

OD	O	D	True mean flow	Prior mean flow	OD	O	D	True mean flow	Prior mean flow
1	1	2	210.00	247.38	10	4	2	320.00	387.45
2	1	3	430.00	498.54	11	4	3	110.00	123.14
3	1	8	320.00	385.01	12	4	8	210.00	226.16
4	2	1	210.00	218.78	13	8	1	320.00	341.64
5	2	4	320.00	370.36	14	8	4	210.00	236.61
6	2	12	50.00	64.33	15	8	12	60.00	69.87
7	3	1	430.00	561.33	16	12	2	50.00	65.64
8	3	4	110.00	124.85	17	12	3	40.00	47.60
9	3	12	40.00	42.07	18	12	8	60.00	76.62

Table 6.4: OD pairs and corresponding true and prior mean flows.

6.3, where the c_a and q_a cost constants used for every link a are shown, being $\beta_a = 1$ and $\gamma = 4$ for every one.

With these 50 routes, we can then apply the SUE assignment model with the true mean OD-pair flows given in Table 6.4 in order to get the true mean link flows (see Table 6.5, columns 3 and 7).

As we have mentioned previously, we have selected this simple example in order to be able to show the results in form of tables and figures of reasonable size. But, due to the good properties of the quadratic objective function in (6.2) and the linear constraints in (6.3), we do not see special problems in dealing with much larger problems, though there is a limit imposed mainly by the number of routes.

Link	Cost	True mean flow	Prior mean flow
1- 5	9.09	422.84	517.34
1-12	21.97	537.16	613.59
2- 8	9.71	309.50	370.85
2-11	18.34	270.50	282.62
3-11	11.11	300.14	389.63
3-13	12.47	279.86	338.62
4- 5	12.58	201.64	262.65
4- 9	17.77	438.36	474.10
5- 1	9.11	425.75	518.41
5- 4	12.54	196.33	258.28
5- 6	35.27	412.60	495.64
5- 9	10.89	211.89	284.36
6- 5	32.56	401.83	480.52
6- 7	6.54	422.03	521.64
6-10	4.31	139.43	148.25
6-12	21.65	154.41	177.92
7- 6	6.34	422.71	503.53
7- 8	10.81	362.40	468.90
7-11	4.02	181.85	179.60
8- 2	10.06	310.70	410.30
8- 7	10.14	369.66	417.28
8-12	32.66	529.84	601.69
9- 4	17.71	443.67	473.53
9- 5	11.23	220.25	296.17
9-10	29.80	368.70	417.87
9-13	28.70	281.54	340.59
10- 6	4.37	133.54	154.91
10- 9	26.47	384.06	431.09
10-11	5.71	508.13	566.12
11- 2	19.38	269.30	290.17
11- 3	8.95	298.46	328.69
11- 7	4.03	175.26	213.11
11-10	5.96	517.60	586.00
12- 1	21.13	534.25	603.33
12- 6	23.80	148.87	174.25
12- 8	36.31	538.30	629.19
13- 3	12.51	281.54	340.59
13- 9	28.25	279.86	338.62

Table 6.5: Links, link costs and true and prior link mean flows.

6.7.1 Reconstructing and estimating traffic flows

Following the ideas of Hazelton (2001), we distinguish between reconstruction and estimation, i.e., we reconstruct all traffic flows and use them for estimation purposes. To this end, the steps indicated in the algorithm in Section 6.6.6 are used. This algorithm has been implemented in GAMS (*General Algebraic Modeling System*) (see http://www.gams.com/).

In order to test the performance of the proposed method, we have assumed the true mean OD flows t_i^{true} shown in Table 6.4 column 4, and assigned them to the network by SUE, to obtain the assumed true mean link and path flows (see Table 6.5 columns 3 and 7, and Table 6.6 column 4). The resulting OD, path and link flows are taken as the true values. Though we apply the proposed Gamma method to path flows, it can be applied to OD or link flows in a similar way.

Step 0: Initialization. Obtain a set of links to be scanned and identify the feasible set of observed scanned link combinations. In this example we have considered that we do not have a previous set of cameras installed. Therefore, we are free to choose the optimal set of scanned links for full observability, which have been obtained using the method explained in Castillo et al. (2008c). This method takes into account the topology of the network and routes, and looks for a set of scanned links such that given any two different routes we have at least one observed link which is in one of the routes and not in the other. In other words, a subset of scanned links able to identify all path flows. The resulting set was:

$$\mathcal{SC} \equiv \{1, 2, 3, 5, 8, 9, 11, 13, 18, 20, 21, 22, 23, 29, 31, 33, 34, 36\}, \tag{6.19}$$

which is shown in Figure 6.2 (see the links with continuous arrows).

Note that this set is only optimal for the 50 paths selected in Table 6.2. If we had more paths, the optimal set $\mathcal{SC}$ (6.19) would be larger.

The set of all scanned link feasible observations is determined by intersection of the set of scanned links $\mathcal{SC}$ with the set of links in each route (see Table 6.6, column 6).

Step 1: Obtain the virtual prior sample of F_r^0. In order to illustrate the goodness of the proposed method, we have assumed random prior flows sufficiently apart from the true flows. To this end, the prior OD matrix is generated randomly from the true OD flows t_i^{true} in Table 6.4 column 4, as $t_i^0 = \alpha_i t_i^{true}$, where the α_i factors are assumed to be independent normal $N(1.2, 0.1^2)$ random variables (see Table 6.4, column 5). Next, assigning them to the network by SUE, the prior virtual sample of link and path flows f_r^0 is obtained (see Table 6.5 columns 4 and 8, and Table 6.6 column 3). This provides virtual samples for the OD, path and link flows.

Step 2: Identify the sets of route, OD and link flows uniquely determined. As an example, using the first 14 scanned links (see Table 6.6 and the vertical line separating these links from the rest of scanned links), we identify the following 28 routes:

$$\{1, 2, 3, 4, 5, 6, 7, 9, 10, 11, 13, 16, 17, 25, 26, 27, 31, 33, 35, 36, 37, 38, 41, 42, 45, 46, 48, 49\},$$

the following 23 total link flows:

$$\{1, 2, 3, 5, 7, 8, 9, 10, 11, 12, 13, 15, 16, 18, 20, 21, 22, 23, 24, 25, 26, 29, 37\}$$

and the 4 OD-pairs: $\{1, 12, 14, 16\}$ (see Table 6.7 row 14(a)). We can apply the proposed $\mathcal{H}(\alpha, \theta, \lambda)$ Bayesian model to these flows, but applying this model to undetermined flows by using their estimates as true values implies no consideration of the associated uncertainty, and then the analysis must be done with care (it is only an approximation), and Bayesian credible intervals are no longer valid. Nevertheless, in these cases the mean estimates are still useful in practice.

Step 3: Obtain the F_r priors. Based on the fictitious sample f_r^0 replicated n_0 times, the prior hyperparameters of F_r are obtained using (6.12)-(6.14).

Step 4: Get the plate scanned observations and reconstruct traffic flows. Since we have no observed flows w_r^0, we have simulated a sample of size n of the path flows using the corresponding $\mathcal{H}(\alpha, \theta, \lambda)$ distributions and the true means, and evaluated the scanned flows w_r^0. Next, the problem (6.6) subject to (6.3) and (6.5) with the prior values f_r^0 and the observations w_r^0 is solved to obtain the path flow estimates f_r. Finally the corresponding unobserved flows v_a and the OD flows t_i are reconstructed using (6.7).

Step 5: Obtain the F_r posteriors. Once the sample of the path flows has been obtained, the posterior $\hat{\eta}$ hyperparameters can be calculated in terms of the prior η parameters using equations (6.15) and (6.17), in order to obtain the F_r posteriors.

Step 6: Obtain the posterior modes of F_r. The posterior modes of F_r are obtained by solving the problem (6.18). Since the mode, the resulting mean posterior values are calculated and shown in Table 6.6 (column 5) for $n = n_0 = 10$.

Step 7: Use of the Gamma model. Finally, the Gamma model can be used with the posterior modes as parameters, or the Bayesian distribution to represent the traffic flows, i.e. OD-pair and link flows. Do not forget that we could obtain the corresponding intervals of the Bayesian distribution (mixture of $\mathcal{H}(\alpha, \theta, \lambda)$ distributions using the posterior as weight function).

To illustrate the power and the performance of the proposed method based on plate scanning and to facilitate the comparison with the standard method based on link observations, we have estimated the OD-pair flows by two methods:

Case (a) Based on the information supplied by the scanned links.

Case (b) Based on the information contained in the counted links.

The results of both (a) and (b) cases are shown by means of graphs and tables. Table 6.6 shows the estimated $\hat{\alpha} + \hat{\theta}/\lambda$ path flows together with the set of scanned link feasible observations.

The discussion below is separated in two subsections, one related to reconstruction results, and one to estimation results.

6.7.2 Discussion of the reconstruction results

In this section we use the proposed methods for reconstructing the traffic flows, i.e., we find a solution compatible with the observations.

Figure 6.3 shows the reconstructed links flows for the cases of: (a) no observations (largest black circles), (b) 10 observations (small gray circles), and (c) 18 observations (small black circles). The true link flows are indicated by the largest red circles. The

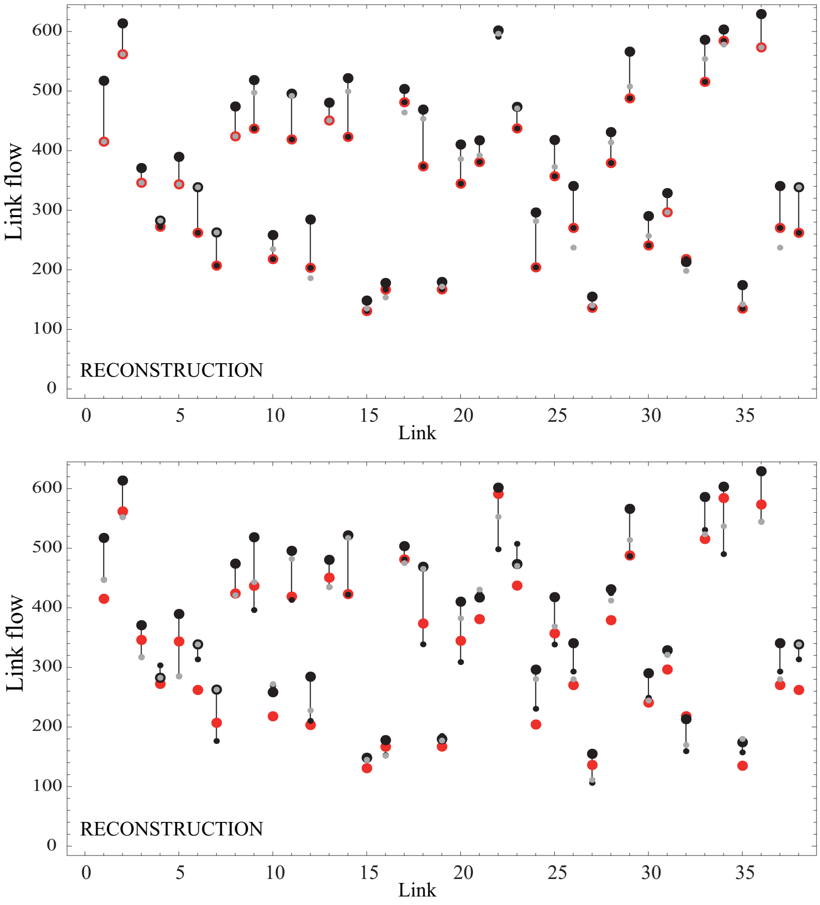

Figure 6.3: Reconstructed link flows for the cases of no observations (largest black circles), 10 (small gray circles) and 18 (small black circles) observations are shown. The true link flows are indicated by the red circles. Upper figure refers to plate scanning, and lower figure to the case of the standard total counted link flow observations.

upper figure refers to plate scanning, and the lower figure to standard counted links. From it, we can conclude that the prior information of some link flows is far from the true flow (a typical situation in practice). However, the link flow estimates after observing 18 links by plate scanning is exact (see upper figure). This is not the case for the standard method of using counted link flows (lower figure).

Figure 6.4 shows the reconstructed OD flows for the cases of: (a) no observations (largest black circles), (b) 10 observations (small gray circles), and (c) 18 observations (small black circles). The true OD flows are indicated by the largest red circles. The upper figure refers to plate scanning, and the lower figure to the case of the standard total link flow observations. Similar conclusions to those drawn from Figure 6.3 can be obtained for the OD estimates. Both Figures 6.3 and 6.4 permit us to conclude that the

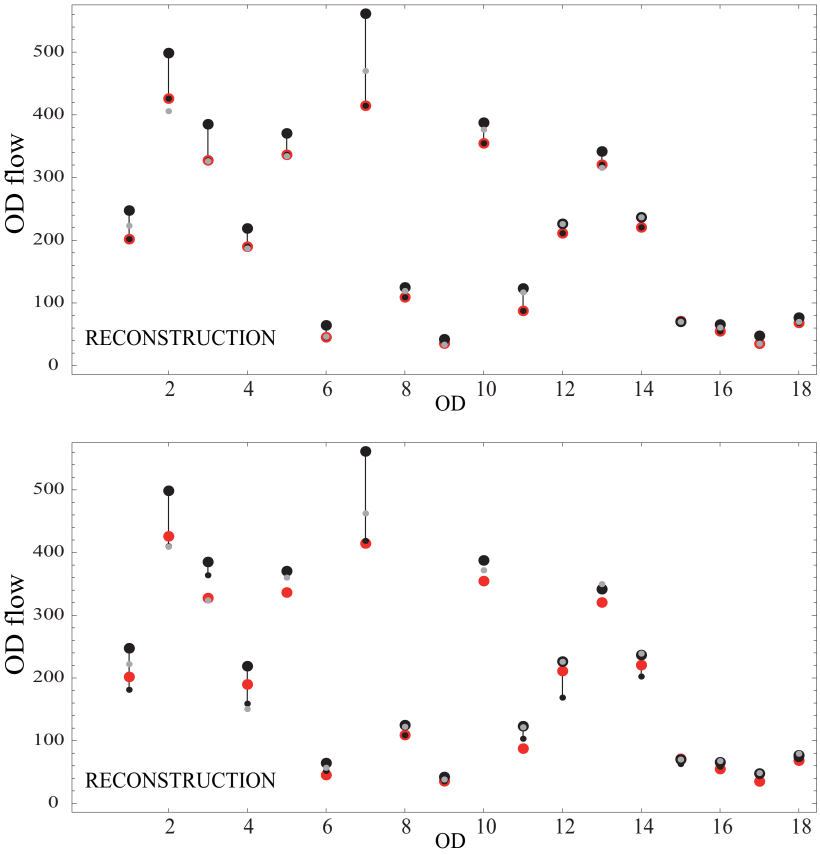

Figure 6.4: Reconstructed OD flows for the cases of no observations (largest black circles), 10 (small gray circles) and 18 (small black circles) observations are shown. The true OD flows are indicated by the largest red circles. Upper figure refers to plate scanning, and lower figure to the case of the standard total counted link flow observations.

proposed method, based on plate scanning, clearly outperforms the standard method of using counted link flows for reconstructing purposes.

Apart from illustrating the particular case above, we have analyzed the effect of the prior on the final parameter estimates by simulation. To this end, we have kept the same true and prior flows and simulated the actual flows assuming $\mathcal{H}(\alpha, \theta, \lambda)$ distributions for the path flows. For each of these cases, the OD flows have been reconstructed and the corresponding errors evaluated. The results have all been produced based on 100 replications, and are given in Table 6.7, where the predictions (expected values) for the OD-pair flows and their corresponding root mean square errors (RMSE) are shown for several cases. The first column refers to the number of observed links, which are scanned links for rows labelled (a), and counted link flows for rows with label (b). It is assumed that the observed links are those in (6.19), which have been introduced in the following

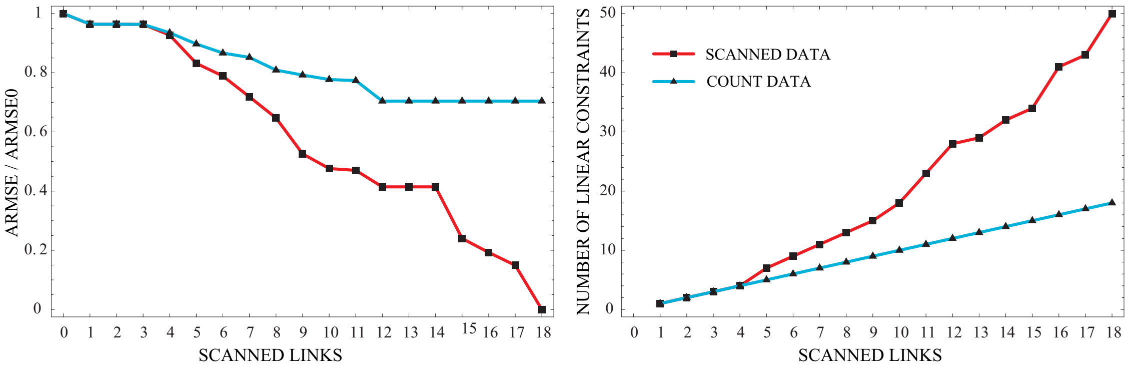

Figure 6.5: Left graph: Relative gain in terms of the average RMSE for OD flows, using scanned links with respect to the total link observations (count data), and right graph: Number of independent linear equality constraints for scanned and count data.

way

$$\mathcal{SC} = \{8, 3, 2, 31, 13, 5, 1, 36, 34, 33, 21, 22, 18, 11, 20, 23, 29, 9\}$$

in order to avoid redundant information for the count data case, i.e. to avoid link flows which are combinations of previous link flows, and therefore they do not add new information.

In each group of two rows, the first contains the expected values of the OD flow estimates and the second the root mean square error (RMSE), which have been calculated for all OD-pairs. In the table, the exact estimates have been shown in bold. Note that, as the number of scanned links decreases, the number of exact estimates reduces.

The case of no observations (0 observed links) contains the prior means, since for no information, one gets the prior values.

From this table, the following conclusions can be drawn:

1. The OD-pair flow estimates based on the information supplied only by the scanned links (case (a)) is exact for 18 scanned links for all OD-pairs, and still remains exact for some OD-pairs when the number of scanned links decreases (see the boldfaced values). When this number decreases, the OD-pair flows are closely estimated.

2. The OD-pair flow estimates based on the information contained in counted links (case (b)), together with the prior information is reasonable good, but this is more due to the important weight given to the prior information, compensating the reduced information contained in the observations. For no information at all, the predictions coincide with the prior values, as expected.

3. The proposed method, based on scanned links gives much better results than the one based on total link flows only, because it supplies more information than counted

links. In addition, it is much less influenced by the prior flow information. In fact, when the number of scanned links reaches the optimum value, this influence is null.

Figure 6.5 compares graphically the two methods: scanned links and counted links. The left graph in Figure 6.5 shows the relative gain in terms of the normalized average RMSE (ARMSE) of OD flows for each observed case (depending on the number of observed links), using the scanned links compared with the total link observations (count data). This figure shows the ratio $ARMSE/ARMSE_0$, where $ARMSE_0$ is the average RMSE for the case of no observations, versus the number of observed links. One can see that the ratio is one when we have no observations in both cases, and decreases with the number of observations, reaching the value zero (exact values) for the scanned data, whereas for count data, the ratio is 0.6 for 18 scanned links. Note that, the differences between scanned links and counted links appear only after the first four scanned links are reached. This means that the information provided by the first scanned links is the same as that provided by link counts. However, after four scanned links, the gain due to scanned data is clear.

The right graph in Figure 6.5 shows the number of independent linear constraints provided for every number of scanned links, illustrating the information provided in both scanned and counted data cases. We can observe that the number of linear equations is equal to both cases for the first four observed links. However, after this, the number of linear equations for scanned data increases faster than for count data. So, 50 linear constraints are produced from the scanned data while only 18 linear constraints from the count data when we observe 18 links. This clearly explains the average RMSE coincidence for the first 3 observed links in Figure 6.5, the decrease error for scanned data, and the higher quantity of information supplied by scanned data than by count data.

Finally, Figure 6.6 shows the RMSE/mean ratios for the reconstruction of the path flows in the cases of counted links (lower figure) and scanned links (upper figure) for different number of observed links (0 to 18). The continuous lines correspond to 18 observed links. They have been obtained by solving the problems (6.6)-(6.4)-(6.5) and (6.6)-(6.3)-(6.5) for counted and scanned links, respectively. Note that (a) the errors for a reduced number of observed links (scanned or counted) are exactly the same, (b) for an intermediate number of observed links, the errors are smaller for the case of scanned links than for counted links, and (c) the errors are null for 18 scanned links.

6.7.3 Discussion of estimation results

First, we deal with the case of full observability, that is, the case in which we have observed all the scanned links. Figure 6.7 shows the estimated $\hat{\alpha} + \hat{\theta}/\lambda$ values of the gamma $\mathcal{H}(\alpha, \theta, \lambda)$ path flows. The first figure is for a very informative prior and a small sample, the second one corresponds to an equilibrium between the prior and the sample, and the third figure is for an uninformative prior and a large sample. The true values of $\alpha + \theta/\lambda$ are shown in red, and the estimates as black small circles. This figure shows how the parameter

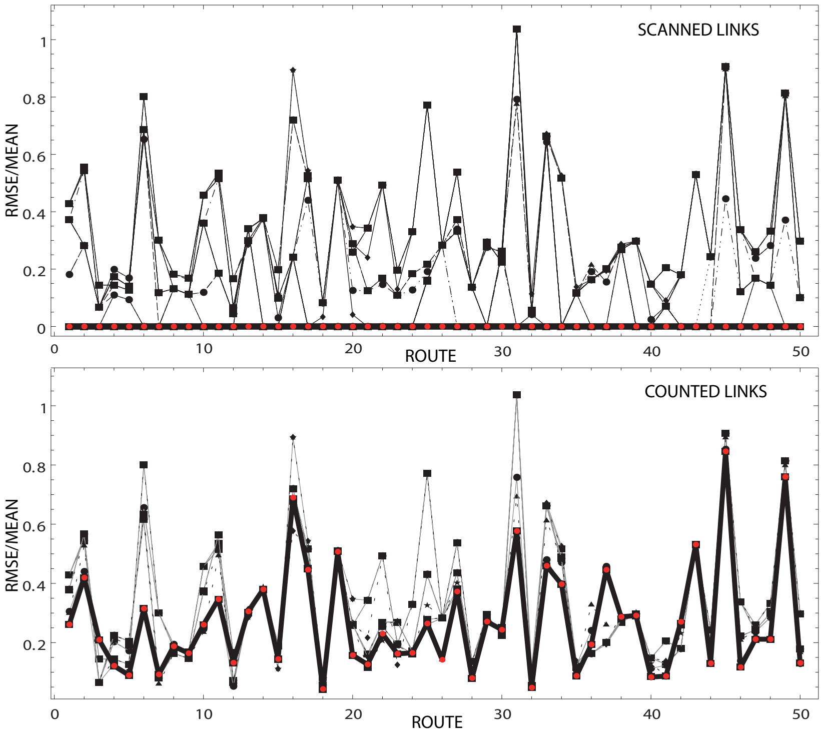

Figure 6.6: RMSE/mean ratios for the predictions of the path flows in the cases of counted (upper figure) and scanned (lower figure) links for different number of observed links (0 to 18). The continuous lines correspond to 18 observed links.

estimates (posterior modes) improve with increasing the sample size n compared with the informative content of the prior n_0. Note that the parameter estimates converge to the true values with increasing n.

In Table 6.6 column 5 we give the parameter estimates of the route variables F_r ($\hat{\alpha}+\hat{\theta}/\lambda$ values) for the case of $n = n_0 = 10$ (similar information of the sample and the prior). From these values we can immediately calculate the parameter values of the $\mathcal{H}(\alpha, \theta, \lambda)$ variables T_i, V_a and W_s.

Next, we deal with partial observability. In Figure 6.8 we show the RMSE/mean ratios for the posterior mode estimates of the path flows, for the cases of $m = 10; n = 10$ (upper figure), and $m = 1; n = 10$ (lower figure). Note how the RMSE errors decrease with the weight of the sample size, that is, when the prior has less weight, and how, for a low number of scanned links, the RMSE coincides with that for the same number of counted links (see Figure 6.6).

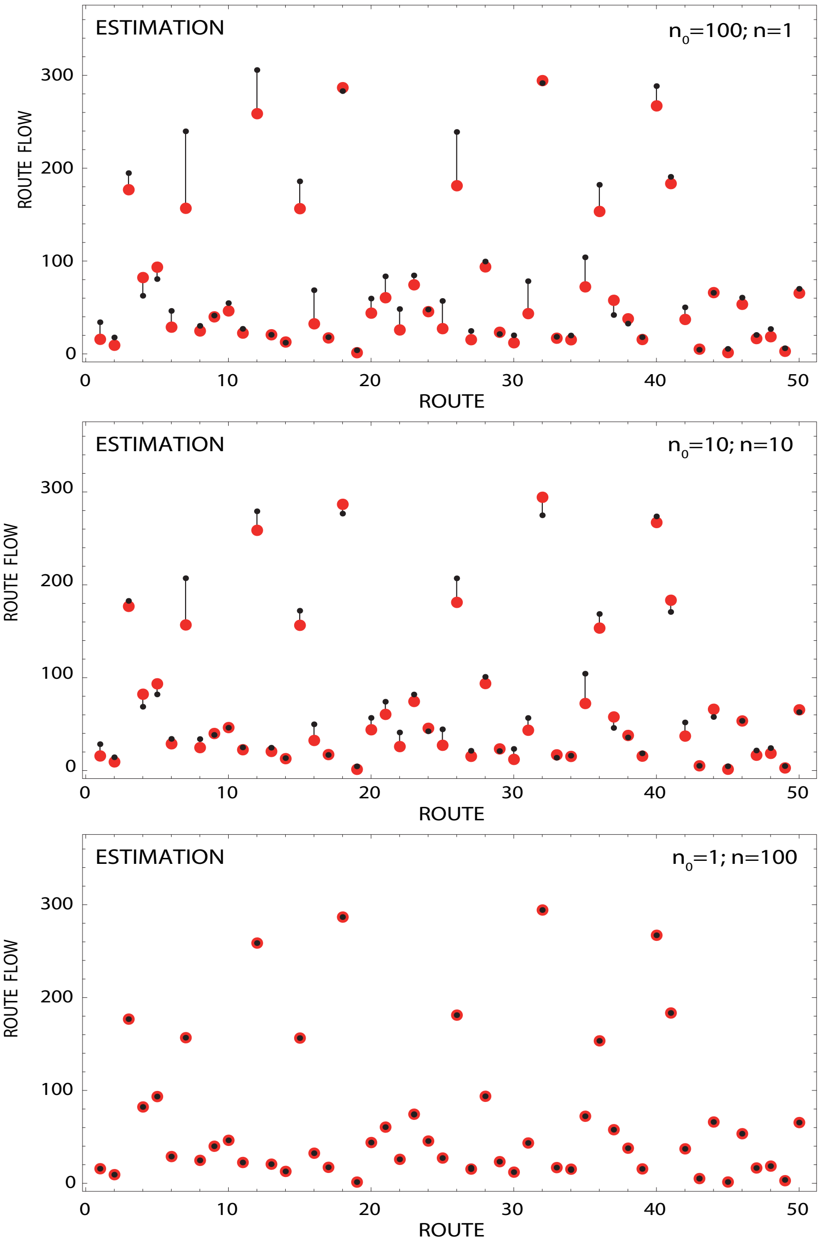

Figure 6.7: Estimated $\hat{\alpha} + \hat{\theta}/\lambda$ parameters of the $\mathcal{H}(\alpha, \theta, \lambda)$ path flows. The first figure is for a very informative prior and a small sample, the second one corresponds to an equilibrium between the prior and the sample, and the third figure is for an uninformative prior and a large sample. The true values of $\alpha + \theta/\lambda$ are shown in red, and the estimates as black small circles.

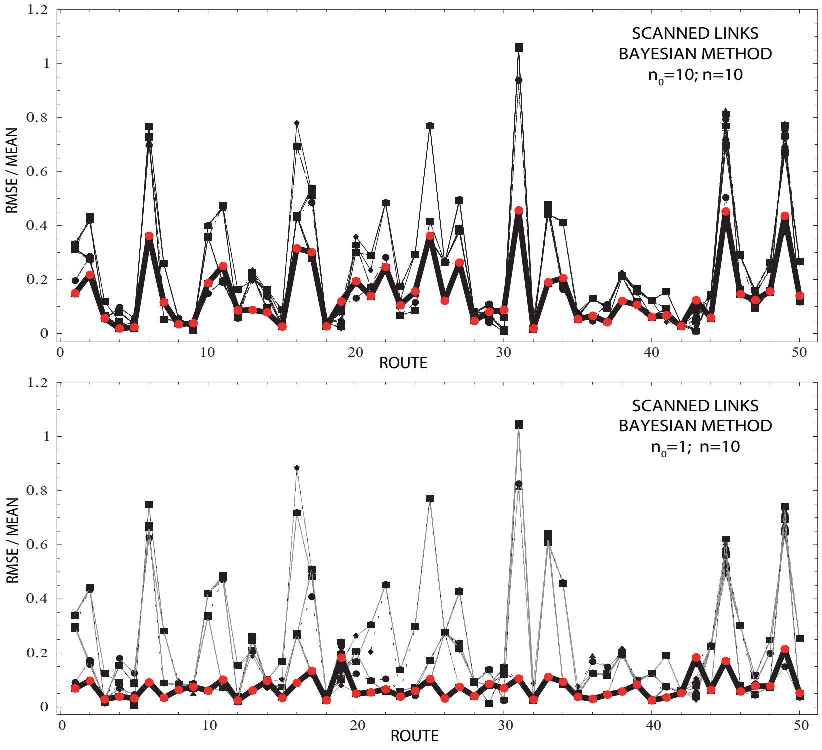

Figure 6.8: RMSE/mean ratios for the posterior mode estimates of the path flows, for the cases of $n_0 = 10; n = 10$ (upper figure), and $n_0 = 1; n = 10$ (lower figure).

6.7.4 Final comments

It is interesting to point out here the importance and the differences between counted links and what we are observing (scanned links). The information $\hat{w}_r$ in Table 6.6 contains 50 values resulting from the 18 scanned links. This means that instead of observing 18 numerical values (the total flows at the scanned links), we observe 50 numerical values, i.e. a lot more information, which explains the great improvement of the proposed method with respect to that of observing the link flows, even if a prior information is not used, and the perfect identification of the corresponding OD-pairs flows based on 18 scanned links.

In fact, observing the link flows alone (counted link flows) provides a poor information because the number of compatible solutions for the OD-pair flows is very large, and this must be complemented with other information, as for example previous or prior link flows. The number of observed links (18 in our example) could be very small compared with the high number of degrees of freedom one large network can have. In summary, the cases with scanned links provide better reconstruction and in consequence, better estimates than the case with link counts alone, if the set of scanned links is sufficiently large.

		w_r			Scanned links and their intersections with route links																			
OD	r	prior	true	$\hat{\alpha}+\hat{\theta}/\hat{\lambda}$	1	2	3	5	8	9	11	13	18	20	21	22	23	29	31	33	34	36	$n_1(r, \mathcal{SC}_1)$	$n_1(r, \mathcal{SC}_2)$
1	1	35	25	27	X						X		X	X									49	49
1	2	17	12	13		X							X	X									49	49
1	3	195	173	175		X								X								X	49	49
2	4	62	68	68	X						X								X				49	49
2	5	81	80	82	X						X							X	X				49	49
2	6	47	27	33	X													X	X				49	49
2	7	238	185	198	X																		49	49
2	8	31	33	33		X													X				48	49
2	9	40	38	38		X												X	X				49	49
3	10	55	38	43	X						X		X										49	49
3	11	27	19	21		X							X										49	49
3	12	304	263	280		X																X	48	49
4	13	21	27	24			X			X		X			X								49	49
4	14	12	14	14			X								X						X		48	49
4	15	186	170	175			X									X					X		48	49
5	16	70	37	46			X					X			X								49	49
5	17	18	12	14			X								X		X			X			49	49
5	18	283	271	274													X			X			48	49
6	19	4	4	3			X								X								48	49
6	20	60	46	52			X									X							48	49
7	21	83	64	69				X		X		X											48	49
7	22	48	33	39				X													X		46	49
7	23	85	74	80				X		X		X								X			48	49
7	24	49	38	41				X												X	X		46	49
7	25	58	34	41				X		X										X			49	49
7	26	238	187	202						X													49	49
8	27	24	17	20				X									X			X			49	49
8	28	100	93	96													X						48	49
9	29	21	19	19				X															46	49
9	30	21	21	22				X												X			46	49
10	31	79	39	54							X		X	X									49	49
10	32	290	269	273					X									X					48	49
10	33	18	11	14					X				X	X				X					49	49
11	34	20	14	16					X									X	X				48	49
11	35	103	96	102					X														49	49
12	36	184	162	170							X		X										49	49
12	37	43	48	45					X				X					X					49	49
13	38	33	41	39						X		X			X								49	49
13	39	19	21	20											X						X		48	49
13	40	290	258	268												X					X		48	49
14	41	188	159	166								X			X								49	49
14	42	48	51	52											X		X			X			49	49
15	43	4	5	4											X								48	49
15	44	66	55	57												X							48	49
16	45	5	3	3									X	X									49	49
16	46	60	47	51										X								X	49	49
17	47	21	18	20															X				0	49
17	48	27	22	24														X	X				49	49
18	49	6	4	5									X										49	49
18	50	70	56	62																		X	0	49

Table 6.6: The set of scanned link feasible observations, observed prior and true mean flows, reconstruct and estimated w_r flows, and associated values of r. The $\hat{\theta}$ estimates are for $n_0 = 10$ and $n = 20$. $\mathcal{SC}_1$ is the set of the first 14 scanned links, and $\mathcal{SC}_2$ is the set of all 18 scanned links.

Observed	OD																	
links	$t_{1,2}$	$t_{1,3}$	$t_{1,8}$	$t_{2,1}$	$t_{2,4}$	$t_{2,12}$	$t_{3,1}$	$t_{3,4}$	$t_{3,12}$	$t_{4,2}$	$t_{4,3}$	$t_{4,8}$	$t_{8,1}$	$t_{8,4}$	$t_{8,12}$	$t_{12,2}$	$t_{12,3}$	$t_{12,8}$
	PRIOR																	
0(a)	247.4	498.5	385.0	218.8	370.4	64.3	561.3	124.8	42.1	387.5	123.1	226.2	341.6	236.6	69.9	65.6	47.6	76.6
	42.7	74.5	69.7	20.6	55.2	17.0	137.0	18.0	7.8	70.3	19.4	28.6	34.4	34.8	14.9	19.1	10.7	19.0
(b)	247.4	498.5	385.0	218.8	370.4	64.3	561.3	124.8	42.1	387.4	123.1	226.2	341.6	236.6	69.9	65.6	47.6	76.6
	42.7	74.5	69.7	20.6	55.2	17.0	137.0	18.0	7.8	70.3	19.4	28.6	34.4	34.8	14.9	19.1	10.7	19.0
	Predictions and RMSE																	
1(a)	247.4	498.5	385.0	218.8	370.4	64.3	561.3	124.8	42.1	356.3	119.1	225.5	341.6	236.6	69.9	65.6	47.6	76.6
	42.7	74.5	69.7	20.6	55.2	17.0	137.0	18.0	7.8	37.9	15.8	28.1	34.4	34.8	14.9	19.1	10.7	19.0
(b)	247.4	498.5	385.0	218.8	370.4	64.3	561.3	124.8	42.1	356.3	119.1	225.5	341.6	236.6	69.9	65.6	47.6	76.6
	42.7	74.5	69.7	20.6	55.2	17.0	137.0	18.0	7.8	37.9	15.8	28.1	34.4	34.8	14.9	19.1	10.7	19.0
2(a)	247.4	498.5	385.0	170.7	363.2	59.3	561.3	124.8	42.1	356.3	119.1	225.5	341.6	236.6	69.9	65.6	47.6	76.6
	42.7	74.5	69.7	40.9	48.8	12.7	137.0	18.0	7.8	37.9	15.8	28.1	34.4	34.8	14.9	19.1	10.7	19.0
(b)	247.4	498.5	385.0	170.7	363.2	59.3	561.3	124.8	42.1	356.3	119.1	225.5	341.6	236.6	69.9	65.6	47.6	76.6
	42.7	74.5	69.7	40.9	48.8	12.7	137.0	18.0	7.8	37.9	15.8	28.1	34.4	34.8	14.9	19.1	10.7	19.0
3(a)	225.5	497.1	332.1	170.7	363.2	59.3	561.3	124.8	42.1	356.3	119.1	225.5	341.6	236.6	69.9	65.6	47.6	76.6
	23.1	73.2	23.1	40.9	48.8	12.7	137.0	18.0	7.8	37.9	15.8	28.1	34.4	34.8	14.9	19.1	10.7	19.0
(b)	225.5	497.1	332.1	170.7	363.2	59.3	561.3	124.8	42.1	356.3	119.1	225.5	341.6	236.6	69.9	65.6	47.6	76.6
	23.1	73.2	23.1	40.9	48.8	12.7	137.0	18.0	7.8	37.9	15.8	28.1	34.4	34.8	14.9	19.1	10.7	19.0
4(a)	224.9	478.1	330.6	170.7	363.2	59.3	561.3	124.8	42.1	361.9	113.4	225.6	341.6	236.6	69.9	65.6	45.7	76.6
	22.6	52.3	20.4	40.9	48.8	12.7	137.0	18.0	7.8	42.8	12.3	28.2	34.4	34.8	14.9	19.1	9.1	19.0
(b)	226.7	472.3	335.0	170.7	363.2	59.3	561.3	124.8	42.1	356.9	118.5	225.5	341.6	236.6	69.9	65.6	45.7	76.6
	23.9	47.2	24.8	40.9	48.8	12.7	137.0	18.0	7.8	38.4	15.4	28.1	34.4	34.8	14.9	19.1	9.4	19.0
5(a)	224.9	478.1	330.6	185.8	346.4	61.1	545.4	124.8	42.1	361.9	113.4	225.6	340.4	196.8	69.9	65.6	45.7	76.6
	22.6	52.3	20.4	26.9	34.9	13.9	121.0	18.0	7.8	42.8	12.3	28.2	33.5	19.6	14.9	19.1	9.1	19.0
(b)	226.7	472.3	335.0	175.9	357.4	59.9	542.0	124.8	42.1	356.9	118.5	225.5	340.2	188.3	69.9	65.6	45.7	76.6
	23.9	47.2	24.8	35.9	43.8	13.1	117.8	18.0	7.8	38.4	15.4	28.1	33.3	25.5	14.9	19.1	9.4	19.0
6(a)	224.9	478.1	330.6	185.8	346.4	61.1	477.0	121.0	36.4	361.9	113.4	225.6	340.9	213.5	69.9	65.6	45.7	76.6
	22.6	52.3	20.4	26.9	34.9	13.9	53.7	15.4	7.1	42.8	12.3	28.2	33.9	13.2	14.9	19.1	9.1	19.0
(b)	226.7	472.3	335.0	172.2	361.5	59.5	473.1	122.6	38.8	356.9	118.5	225.5	341.2	222.0	69.9	65.6	45.7	76.6
	23.9	47.2	24.8	39.4	47.1	12.8	50.3	16.4	6.9	38.4	15.4	28.1	34.1	22.4	14.9	19.1	9.4	19.0
7(a)	223.3	408.7	326.7	185.8	346.4	61.1	477.0	121.0	36.4	361.9	113.4	225.6	340.9	213.5	69.9	65.6	**40.8**	76.6
	21.5	23.4	18.5	26.9	34.9	13.9	53.7	15.4	7.1	42.8	12.3	28.2	33.9	13.2	14.9	19.1	**0.0**	19.0
(b)	224.6	404.2	329.9	172.2	361.5	59.5	473.1	122.6	38.8	356.6	118.8	225.5	341.2	222.0	69.9	65.6	46.7	76.6
	22.3	29.2	21.7	39.4	47.1	12.8	50.3	16.4	6.9	38.2	15.6	28.1	34.1	22.4	14.9	19.1	9.9	19.0
8(a)	223.3	408.7	326.7	185.8	346.4	61.1	477.0	121.0	36.4	361.9	113.4	225.6	340.9	213.5	69.9	53.4	**40.8**	60.0
	21.5	23.4	18.5	26.9	34.9	13.9	53.7	15.4	7.1	42.8	12.3	28.2	33.9	13.2	14.9	8.1	**0.0**	6.8
(b)	223.7	407.5	327.6	172.2	361.5	59.5	473.1	122.6	38.8	356.7	118.7	225.5	341.2	222.0	69.9	59.6	46.4	68.4
	21.6	26.1	20.6	39.4	47.1	12.8	50.3	16.4	6.9	38.2	15.6	28.1	34.1	22.4	14.9	13.6	9.7	11.2
9(a)	223.3	408.7	326.7	202.3	345.0	45.8	478.6	120.4	35.5	361.9	113.4	225.6	307.3	213.5	69.9	53.4	**40.8**	60.0
	21.5	23.4	18.5	9.9	33.9	6.1	55.0	15.1	6.7	42.8	12.3	28.2	11.5	13.2	14.9	8.1	**0.0**	6.8
(b)	223.7	407.5	327.6	171.6	361.8	59.8	473.0	122.6	38.8	356.7	118.7	225.5	332.6	221.5	69.9	59.6	46.4	68.4
	21.6	26.1	20.6	39.9	47.4	12.9	50.3	16.5	6.9	38.2	15.6	28.1	22.1	21.9	14.9	13.6	9.7	11.2
10(a)	223.3	408.7	326.7	202.3	330.6	**50.1**	477.9	120.3	36.4	361.9	113.4	225.6	307.3	213.2	69.9	53.4	**40.8**	60.0
	21.5	23.4	18.5	9.9	13.7	**0.0**	54.3	15.1	5.6	42.8	12.3	28.2	11.5	13.0	14.9	8.1	**0.0**	6.8
(b)	223.7	407.5	327.6	171.7	346.8	59.8	473.1	122.6	38.8	356.7	118.7	225.5	332.5	220.8	69.9	59.6	46.4	68.4
	21.6	26.1	20.6	39.8	29.9	13.0	50.3	16.4	6.9	38.2	15.6	28.1	22.0	21.4	14.9	13.6	9.7	11.2
11(a)	223.3	408.7	326.7	202.3	328.6	**50.1**	477.9	120.3	36.4	361.9	113.4	225.6	307.3	215.2	69.8	53.4	**40.8**	60.0
	21.5	23.4	18.5	9.9	9.9	**0.0**	54.3	15.1	5.6	42.8	12.3	28.2	11.5	11.5	15.0	8.1	**0.0**	6.8
(b)	223.7	407.5	327.6	174.2	354.5	59.8	475.5	121.6	37.4	356.7	118.7	225.5	336.8	200.8	69.8	59.6	46.4	68.4
	21.6	26.1	20.6	37.3	36.8	13.0	52.4	15.8	6.9	38.2	15.6	28.1	25.5	15.0	14.8	13.6	9.7	11.2
12(a)	223.3	408.7	326.7	202.3	328.6	**50.1**	477.9	120.3	36.4	361.9	113.4	225.6	307.3	215.2	**58.7**	53.4	**40.8**	60.0
	21.5	23.4	18.5	9.9	9.9	**0.0**	54.3	15.1	5.6	42.8	12.3	28.2	11.5	11.5	**0.0**	8.1	**0.0**	6.8
(b)	223.7	407.5	327.6	180.9	359.9	51.8	477.0	121.0	36.4	356.7	118.7	225.5	324.0	198.5	58.8	59.6	46.4	68.4
	21.6	26.1	20.6	30.9	41.6	8.2	53.8	15.4	7.1	38.2	15.6	28.1	16.1	15.7	8.1	13.6	9.7	11.2
13(a)	217.2	**429.5**	312.0	202.3	328.6	**50.1**	477.9	120.3	36.4	352.8	113.6	166.1	307.3	215.2	**58.7**	53.4	**40.8**	59.9
	17.7	**0.0**	17.7	9.9	9.9	**0.0**	54.3	15.0	5.6	33.7	12.3	40.6	11.5	11.5	**0.0**	8.1	**0.0**	6.8
(b)	221.0	416.6	321.1	180.9	359.9	51.8	477.0	121.0	36.4	346.0	119.2	144.6	324.0	198.5	58.8	58.7	46.2	67.2
	19.8	18.4	18.9	30.9	41.6	8.2	53.8	15.4	7.1	28.9	15.9	62.4	16.1	15.7	8.1	12.9	9.6	10.4
14(a)	217.2	**429.5**	312.0	202.3	328.6	**50.1**	477.9	120.3	36.4	353.5	113.6	169.8	307.3	215.2	**58.7**	51.6	**40.8**	57.5
	17.7	**0.0**	17.7	9.9	9.9	**0.0**	54.3	15.1	5.6	34.3	12.3	36.9	11.5	11.5	**0.0**	7.0	**0.0**	7.0
(b)	223.3	408.8	326.6	180.9	359.9	51.8	477.0	121.0	36.4	359.8	119.2	148.4	324.0	198.5	58.8	53.5	41.4	60.1
	21.4	24.5	19.7	30.9	41.6	8.2	53.8	15.4	7.1	40.7	15.8	58.6	16.1	15.7	8.1	9.2	7.5	8.1
15(a)	**209.0**	**429.5**	**320.2**	202.3	328.6	**50.1**	477.9	120.3	36.4	317.7	113.6	**205.6**	307.3	215.2	**58.7**	**48.5**	**40.8**	**60.6**
	0.0	**0.0**	**0.0**	9.9	9.9	**0.0**	54.3	15.0	5.6	12.3	12.3	**0.0**	11.5	11.5	**0.0**	**0.0**	**0.0**	**0.0**
(b)	183.1	409.3	366.3	180.9	359.9	51.8	477.0	121.0	36.4	351.8	119.2	157.2	324.0	198.5	58.8	50.3	41.4	62.8
	27.7	24.0	48.2	30.9	41.6	8.2	53.8	15.4	7.1	33.2	15.8	49.9	16.1	15.7	8.1	7.5	7.5	8.2
16(a)	**209.0**	**429.5**	**320.2**	202.3	328.6	**50.1**	480.7	**111.9**	36.7	317.7	113.6	**205.6**	307.3	215.2	**58.7**	**48.5**	**40.8**	**60.6**
	0.0	**0.0**	**0.0**	9.9	9.9	**0.0**	56.5	**0.0**	5.4	12.3	12.3	**0.0**	11.5	11.5	**0.0**	**0.0**	**0.0**	**0.0**
(b)	183.1	409.3	366.3	181.0	355.8	51.6	477.4	111.7	36.4	351.8	119.2	157.2	324.2	198.5	58.7	50.3	41.4	62.8
	27.7	24.0	48.2	30.8	37.2	8.1	54.2	8.6	7.1	33.2	15.8	49.9	16.2	15.7	8.0	7.5	7.5	8.2
17(a)	**209.0**	**429.5**	**320.2**	202.3	328.6	**50.1**	480.7	**111.9**	36.7	**322.0**	**109.3**	**205.6**	307.3	215.2	**58.7**	**48.5**	**40.8**	**60.6**
	0.0	**0.0**	**0.0**	9.9	9.9	**0.0**	56.5	**0.0**	5.4	**0.0**	**0.0**	**0.0**	11.5	11.5	**0.0**	**0.0**	**0.0**	**0.0**
(b)	183.1	409.3	366.3	181.0	355.8	51.6	477.4	111.7	36.4	351.8	119.2	157.2	324.2	198.5	58.7	50.3	41.4	62.8
	27.7	24.0	48.2	30.8	37.2	8.1	54.2	8.6	7.1	32.9	13.5	49.9	16.2	15.7	8.0	7.5	7.5	8.2
18(a)	**209.0**	**429.5**	**320.2**	**210.1**	**320.8**	**50.1**	**427.4**	**111.9**	**38.9**	**322.0**	**109.3**	**205.6**	**315.6**	**207.0**	**58.7**	**48.5**	**40.8**	**60.6**
	0.0	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**	**0.0**
(b)	183.1	409.3	366.3	180.6	356.4	52.0	449.5	111.6	36.6	351.8	119.2	157.2	323.1	199.2	59.0	50.3	41.4	62.8
	27.7	24.0	48.2	31.1	37.7	8.2	25.8	8.6	7.0	32.9	13.5	49.9	15.4	15.2	8.0	7.5	7.5	8.2

Table 6.7: Mean values and RMSEs obtained based on 100 replications for the cases (a) scanned plates, and (b) the standard link data. Model $\mathcal{H}(\alpha, \theta, \lambda)$.

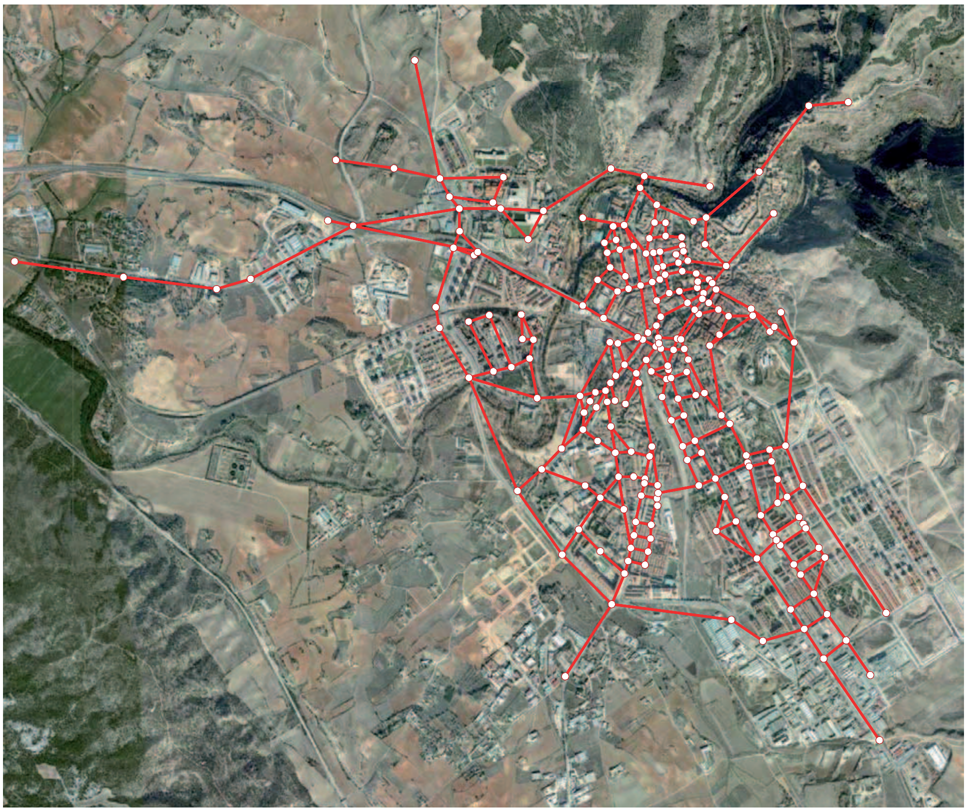

Figure 6.9: The Cuenca network showing the nodes and links.

6.8 The Cuenca network

In this section we show that the proposed method can be applied to real size networks, as the Cuenca network (see Figure 6.9[8]), which consists of 672 links, 232 nodes and 219 routes.

Unfortunately, we have had no financial support for installing the cameras. So, we have used simulated data for this network. We proceeded as follows:

1. To assume the parameters of our models $\mathcal{H}(\alpha, \theta, \lambda)$ for all the route, OD-pairs, links and scanned flows in a consistent way, we have assumed the true mean OD flows t_i^{true}, and we have assigned them to the network by SUE. The resulting flows were the assumed true mean flows.

2. Since 78 scanned links are sufficient for full observability of this network, we have studied a sequence of these links, ranging from no links to all of them, to illustrate how the results improve with increasing the number of scanned links.

[8] Image is captured from http://maps.google.es/maps

3. To generate the priors, we generate a random set of OD-pair flows as $t_i^0 = \alpha_i t_i^{true}$, where the α_i factors were assumed to be independent normal $N(1.2, 0.1^2)$ random variables. Next, assigning them to the network by SUE, the prior virtual samples of all flows were obtained.

4. The set of flows to which the proposed Bayesian technique can be applied and for which this technique provides only approximate estimates was determined (see Table 6.8).

5. The prior hyperparameters are determined using (6.12)-(6.13), replicating $n_0 = 10$ times each flow.

6. The true flows were generated from the $\mathcal{H}(\alpha, \theta, \lambda)$ model, from which the sample flows were simulated ($n = 10$). A similar weight for the prior and the sample was considered ($n_0 = n = 10$). Note that the sample size n required for the sample to dominate the estimation depends on the value of the sample size n_0 of the virtual sample prior, i.e. on the ratio n/n_0. For $n/n_0 > 1$ the sample dominates.

7. We solved the problem (6.6) subject to (6.3) and (6.5) using the scanned link flows to obtain the path flow f_r, from which all other flows were calculated, and obtained the corresponding RMSE. One example is given in the left part of Figure 6.10 for the OD-pair flows (continuous line).

8. We solved the problem (6.6) subject to (6.4) and (6.5) using the counted link flows to obtain the path flow f_r, from which all other flows were calculated, and obtained the corresponding RMSE. One example is given in the left part of Figure 6.10 for the OD-pair flows (dashed line). Figure 6.10 contains the information in Figure 6.5 for this case, where it can be seen that the amount of information increases exponentially with the number of scanned links. We note that the RMSE have been calculated based on the assumed true flow values.

9. For the flows which are fully observable we obtained the posteriors using Equations (6.15) and (6.16), and solved problem (6.18) to obtain the mode posteriors.

It is impossible to show all the results in this chapter, but we can say that they are similar to those shown for the Nguyen Dupuis network.

When the method described in Section 6.6 was applied to this network, the execution time for one run with 78 scanned links was 0.03 seconds on a Dell Optiplex 755 desktop computer.

To illustrate how the reduction in the number of scanned links influences the number of fully observable link, OD and path flows, we give Table 6.8, which shows how the percentages of fully observed links, OD and routes change with the number of scanners used. Finally, Figure 6.10 shows how the number of linear constraints and the prediction

Number	Percentages		
of scanned links	link flows	path flows	OD flows
78	100	100	100
70	78.82	76.71	74.37
60	57.64	49.77	48.24
50	36.46	33.78	31.15
40	27.88	23.74	20.10

Table 6.8: Percentages of fully observed link, route and OD flows for different number of scanned links.

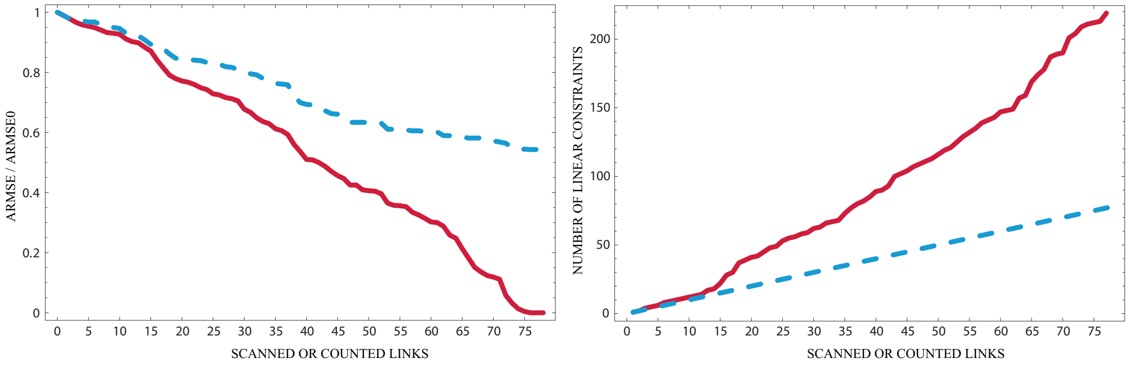

Figure 6.10: Results for the Cuenca network. Left figure: relative gain in terms of the average RMSE values for OD flows using scanned links (continuous line) with respect to the counted link observations (dashed line). Right figure: number of independent linear equality constraints for scanned (continuous line) and count (dashed line) data relative gain in terms of the average.

errors change with the number of scanners, going from no scanners to the number of scanners leading to full observability, so all range of possibilities are studied.

6.9 Conclusions

The main conclusions that can be drawn from this chapter are:

1. A new model based on the family $\mathcal{F} \equiv \mathcal{H}(\alpha,\theta,\lambda) \equiv \{\alpha+\Gamma(\theta,\lambda); \alpha \geq 0, \theta > 0, \lambda > 0\}$, which is a triparametric family of positive, reproductive, infinitely divisible and with positive skewness of distributions has been proposed. The model has more degrees of freedom than other models, such as the Poisson, for example, because the mean $E[X] = \alpha + \theta/\lambda$ and the variance $Var[X] = \theta/\lambda^2$ are independent parameters.

2. The family $\mathcal{F} \equiv \mathcal{H}(\alpha,\theta,\lambda) \equiv \{\alpha+\Gamma(\theta,\lambda); \alpha \geq 0, \theta > 0\}$with common λ for all flows

is statistically consistent to model route, OD-pair, link, node, scanned flows, etc. In this way, the same family is valid to model all of them.

3. Since evaluation of posterior means is difficult, we have suggested to use the posterior modes instead, which can be calculated by solving simple optimization problems.

4. The proposed methods allow us to use sampling techniques which cannot be used by other methods, such as plate scanning information. These techniques lead to very good estimates of all traffic flows due to the large amount of information contained in them.

5. Link count data can be sufficient to estimate (uniquely) OD (and route) flows given repeated sampling (see Vardi (1996) and Hazelton (2000)). However, these methods require some reasonably strong assumptions on the traffic generation process. Without these assumptions, link count information is insufficient to identify origin, destinations or routes. On the contrary, plate scanning makes this estimation possible.

6. A comparison of the information contained in count data with that contained in scanned data indicates that using the traditional link flow information to estimate traffic flows can lead to poor results compared with the latter. The reason is simply that plate scanned data supply more information than link counts only.

7. Application of the proposed method to the Nguyen-Dupuis network example has shown the applicability and interest of the proposed method, and pointed out that the proposed estimation method leads to good estimates.

8. Though the proposed Bayesian method is valid only for uniquely determined (route, OD and link) flow samples, application to incomplete samples seems to give good results for point estimates, as the Nguyen-Dupuis and Cuenca examples have shown. Figures 6.5 and 6.10 show how the number of linear constraints and the prediction errors change with the number of scanners, going from no scanners to the number of scanned leading to full observability.

9. Due to the low cpu cost, the proposed methods can be applied to large networks.

Chapter 7

FIFO Rule consistent dynamic network model

Contents

7.1 Notation

α_a	free-flow travel time of link a.
$\bar{\alpha}_{pi}$	vector of parameters associated with path p.
β_a	parameter of the travel time function associated with link a.
δ_a	parameter of the travel time function associated with link a.

γ	parameter of the travel time function.
μ_{pi}	mean parameter of the ith component of path p.
σ_{pi}	standard deviation parameter of the ith component of path p.
$\tau_a^{-1}(t)$	entry time of a user that exits link a at time t.
$\tau_a(t)$	link traversal time for a vehicle entering the link a at time t.
$\theta_{a_i}^p(t)$	departure time from the origin of path p of a user who exits the ith link a_i of path p at time t.
$D_a(t)$	lower bound of link travel time for vehicles entering link a at time t.
$e_a(t)$	inflow rate for link a at time t.
$f_{N(\mu,\sigma)}(t)$	probability density function (pdf) of the normal distribution with mean μ and standard deviation σ.
$G_a(t)$	cumulative flow associated with the exit of link a by time t.
$g_a(t)$	outflow rate for link a at time t.
$g_a^p(t)$	outflow rate of link a at time t due to path p.
$H_p(t)$	cumulative inflow rate at the origin of path p at time t.
$h_p(t)$	inflow rate at the origin of path p at time t.
h_{pi}	coefficients of the linear combination to generate $h_p(t)$.
n_p	number of function components.
n_{links}	number of network links.
p	path.
$q(t;\bar{\alpha}_{pi})$	parametric family of probability density functions with parameter $\bar{\alpha}_{pi}$.
Q_{ak}	queue dissipation time at link a at time t_k.
$r_n(t)$	flow intensity at node n and time t.
$s_a(t)$	link a congestion ratio.
t_k	set of discrete times to be considered for $k = 1, \dots, n$.
t_a^{out}	link exit time of a user who enters link a at time t.
$x_a(t)$	number of vehicles on link a at time t (link traffic volume).
x_a^{max}	number of vehicles on link a leading to a travel time $\alpha_a(1+\beta_a)$.
$\mathcal{A}$	set of links of a traffic network.
$\mathcal{A}(n)$	set of links entering node n.
$\mathcal{D}$	set of origin-destination pairs of nodes.
$\mathcal{N}$	set of nodes of a traffic network.
$\mathcal{P}(a)$	set of paths containing link a.
$\mathcal{S}(a)$	set of all links downstream link a in all its paths.

7.2 Introduction

In Chapter 4 we have introduced dynamic models, that is, models describing the traffic evolution not only in space but also in time.

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, called *continuous models*, 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. More precisely, models can be classified as:

1. *Discrete models.* In this case both space and time are discretized by means of a net $\mathcal{N} \equiv \{(x_i, t_j)|i = 1, 2, \ldots, n; j = 1, 2, \ldots, m\}$ and the traffic evolution is calculated only at the nodes of this net. This simplifies the treatment of the problem, but the prize we need to pay is that in order to know traffic variables in other points we need some kind of interpolation. One example of these models is the well known cell transmission model proposed by Daganzo (1992).

2. *Continuous models.* We say that we have a continuous model when both variables, space and time, are treated as continuous. In this case, any traffic variable y is given as a continuous function $y = g(x, t)$. Though these models are those reproducing better the real situation of traffic, they are difficult to deal with and require a lot of resources. Examples of these models are the hydrodynamic-based models.

3. *Mixed models.* These are models that combine discrete and continuous solutions either in time or space. Thus, two models can be considered in this group:

   (a) *Discrete time and continuous space.* If we model the traffic at a discrete set of times and use continuous functions for space, we have one of such a models.

   (b) *Discrete space and continuous time.* One example of this group is the case of a network in which only the traffic evolution at the nodes or the link ends is modeled by continuous functions.

4. *Smoothed discrete models.* These are models of discrete time or/and space such that they have made continuous by using interpolating bidimensional functions such as the Gordon-Coons. In this case we end with a continuous model that has been built based on discrete data.

When interpolating values it is very important to make an adequate selection of: (a) the basic points (the net), and (b) the interpolating function.

As it was explained in Chapter 4, in dynamic models we can distinguish two phases: an assignment stage and a loading stage. In the assignment stage we analyze how the users choose their destinations or transportation modes, and also how they decide among all possible routes of their OD. In the loading stage, we analyze how the traffic network behaves when the users compite for space and congestion appears based on the assumed demand functions resulting from the assignment stage.

In addition to the previous classification, traffic problems can be grouped as:

1. *Deterministic models*: in these type of models we assume that the users make decisions in a systematic way based on a measurable criterion, such as travel time, cost, etc. In addition the network characteristics are assumed to be fixed and known to the user and no changes, such as climatological incidents, accidents, etc., are considered. Consequently, both the resulting demand and the travel times (costs) are of a systematic type, i.e., with no uncertainty. This is why these models are called deterministic.

2. *Stochastic models*: in these models we assume that: (i) the users decide in a random manner, or (ii) they decide in a systematic way but the network characteristics are uncertain. This implies random demands and/or travel times. If the users are responsible for uncertainty, we are led to a stochastic demand that will produce stochastic travel time functions. However, stochastic travel time functions can also be due to random network behavior (network capacity, climatology, etc.).

In this chapter we present a deterministic dynamic traffic model which is of smooth type above described in time but discrete in space. More precisely, we assume given path demand time functions, that leads to "systematic" travel time functions, and after selecting a net based on nodes, we smooth our discrete time values by means of a polynomial interpolation.

Based on the time demand functions at their origin nodes, we study how the different traffic flows progresses and compite for space throughout the network until they reach their destinations. To this end we need to analyze the level of congestion at all points in the network. Once these levels have been determined, the travel times of all links and paths can be evaluated. Since we are looking for realistic predictions, we need to consider the satisfaction of some important properties such as the link FIFO rule (see Chapter 4), in such a way that any user arriving first to any link also reaches its end in the first place. We note that link FIFO satisfaction implies path FIFO satisfaction.

Since we need to choose a link travel time function, reflecting in a realistic way all variables determining this travel time, satisfaction of the FIFO rule is not trivial. For example, Ran et al. (1996) assumes that the link traversal time, $D_a(t)$, at time t can be determined by a function of the link volume $x_a(t)$ and the entering $e_a(t)$ and $g_a(t)$ exiting flow rates,

$$D_a(t) = f(e_a(t), g_a(t), x_a(t)). \tag{7.1}$$

However, as Nie and Zhang (2005b) explain, such an assumption was found problematic. For instance, Daganzo (1995a) shows that (for the above delay function), unless the right hand side is reduced to $f(x_a(t))$, some inflow profiles can always be found to cause FIFO violations. Since the formulation of the delay-function model requires a FIFO-consistent delay function, the type of functions as 7.1 in general cannot be used in a delay-function model. Nevertheless, Daganzo not only demonstrates this, but states that the dependence on $x_a(t)$ must be constant of linear (see the different formulations of the FIFO rule in Chapter 4). Examples of authors using these linear relationships are Friesz et al. (1993a), Astarita (1996) and Celikoglu (2007). Some important results on conditions for non-linear to be valid are given by Xu et al. (1999) and Zhu and Marcotte (2000). However, the practical implication is that these conditions can fail in real cases.

On the other hand, continuous dynamic traffic models raise, as indicated above, the problem of selecting: (a) the set of basic points (discrete times) on which the interpolation is to be based, and (b) the polynomial used in such a interpolation. Some relevant papers dealing with this topic are Xu et al. (1998), Wu et al. (1998) or Rubio-Ardanaz et al. (2003). However, there are still some problems to be solved, such as:

1. These algorithms try to identify the times at which some important events occur, which are used as the basic times for a polynomial interpolation. These events are the arrivals of the first users of each path to the beginning of each link. Unfortunately, these times are irregularly distributed on the period of study, and can be too many (an excessive number to deal with in real situations). Moreover, these time events are not necessarily determinant in the analysis, because usually the beginning of a path flow corresponds to very low flow, which is compatible with the assumption of null initial flow. On the contrary, identifying significant changes of flow would be much more relevant.

2. The number of fitted interpolating polynomials is number of routes × number of links, which is very high.

3. The use of Lagrangian polynomials does not guarantee some important properties of the flow. For example, if the basic points show an increasing trend, the fitted polynomial could not keep this property, and in fact it fails to do so in some cases.

The proposed model considers that the link travel time is in fact a non-linear function of the link volumes, but we make some corrections in order to satisfy the FIFO rule. The idea consists in adding the minimum delay for such a condition to be satisfied (this is equivalent to considering a queue of vehicles due to the congestion and its later dissipation), which is complemented by the use of the cubic monotone Hermite interpolation (see Fritsch and Carlson (1980)) that guarantees the satisfaction of the FIFO rule at the interpolated points.

The chapter is structured as follows. In Section 7.3, we define the path flow wave at the origin node, i.e., the flow intensity functions at the origin of each path by means of some time-dependent functions. In Section 7.4, we determine how these individual path flow waves propagate throughout their paths. In Section 7.5, we analyze the path flow wave interactions (in nodes and links) and, in Section 7.6, the congestion due to these interactions. In Section 7.7, we explain the approximation of the $\tau_a^{-1}(t)$ function

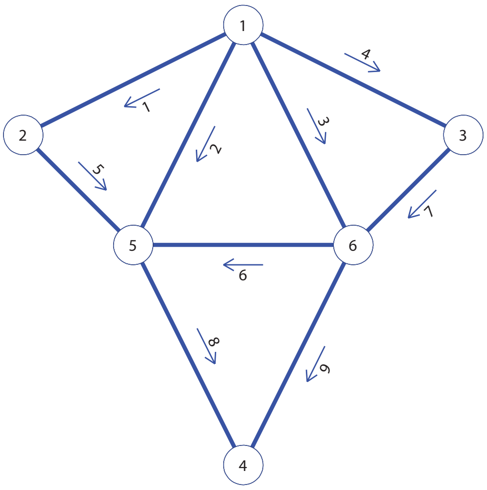

Figure 7.1: The elementary example network used for illustrative purposes, showing the nodes and links.

as the basis of the model. The Section 7.8 describes the proposed methodology and the algorithm implemented in Matlab code. In the next Section 8.4 we provide some examples of application. In Section 7.10, we compare the proposed model with other existing models and finally, in Section 7.11, we draw some conclusions.

To facilitate the understanding of all the concepts managed, 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 the subset $\mathcal{D}$, of origin-destination pairs of nodes, and we use the simple illustrative example shown in Figure 7.1. It consists of 6 nodes, 9 links and 9 paths as indicated in Table 7.1, where its three columns contain the OD-pairs defined by their origins and destinations, the path number and the path links, respectively.

OD	Path number (p)	Links
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 7.1: Set of 3 OD–pairs and 9 paths (defined by its and nodes and links) in the example.

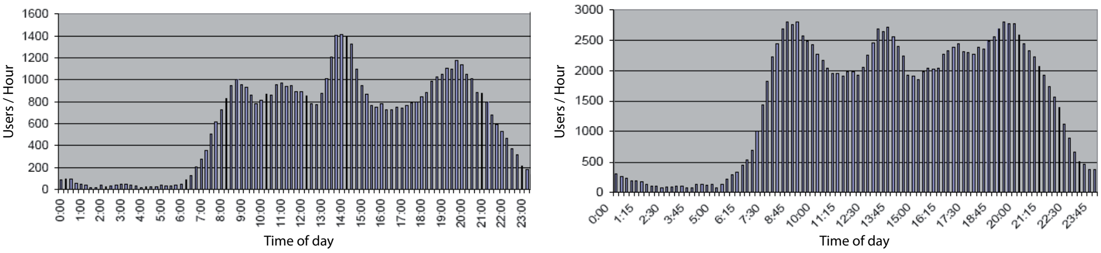

Figure 7.2: Observed time flow evolution at two given sites in the city of Cuenca (Spain).

7.3 Path flow wave at the origin node

In this section we expose the path flow intensity functions at the origin of each path by means of some time-dependent functions, i.e., the time evolution of the number of users entering the network.

If we observe the time evolution of traffic flow at a given location, we obtain curves such as those indicated in Figure 7.2, which correspond to a given section of the city of Cuenca (Spain). We note that practically all the data we have observed have this pattern, which can be considered as a sum of several simpler individual waves, with relative maxima at the rush-hours and very low values at night hours.

Since a function has infinite degrees of freedom, arbitrary functions cannot be dealt with, so, 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. We propose path flow intensity functions of the form

$$h_p(t) = \sum_{i=1}^{n_p} h_{pi} q(t; \bar{\alpha}_{pi}), \tag{7.2}$$

where $h_p(t)$ is the inflow traffic intensity (veh/hour) associated with path p at its origin and time t, n_p is the number of function components, $q(t; \bar{\alpha}_{pi})$ is a parametric family of probability density functions[1], with vector parameter $\bar{\alpha}_{pi}$, and $h_{pi} > 0$ are the coefficients of the linear combination (the path flow associated with the i component).

In summary, path intensity functions are assumed linear combinations of a basic set of probability density functions (pdf).

Example 7 (Illustrative example) *Let us assume an industrial area network. The path flows in this area present two maxima at rush-hours, one in the morning, the other*

[1] Note that we have used probability density functions in order to have a total flow h_{pi} associated with each component.

Path	μ_{p1}	σ_{p1}	h_{p1}	μ_{p2}	σ_{p2}	h_{p2}	Path flow
1	8.25	2.07	8214	15.55	1.63	7883	16096
2	7.59	1.81	7925	15.21	1.85	8668	16592
3	7.58	1.92	8831	14.56	2.06	7584	16414
4	7.96	1.93	8304	15.61	1.75	8489	16792
5	7.51	1.94	8636	15.23	2.31	7929	16565
6	8.48	1.51	8216	15.16	1.81	7603	15819
7	8.07	2.00	7648	14.50	2.20	8609	16257
8	8.00	1.88	7836	15.77	2.00	8056	15892
9	7.99	1.80	8829	15.57	2.45	8974	17804

Table 7.2: Parameters used in path flows of the illustrative example.

one, early afternoon, and the night flow is very low. Accordingly, we can reproduce with sufficient precision the traffic intensity of each path by a linear combination of normal densities (waves), whose total area is the total path flow, that is:

$$h_p(t) = \sum_{i=1}^{n_p} h_{pi} f_{N(\mu_{pi},\sigma_{pi})}(t), \tag{7.3}$$

where n_p is the number of components (two in this case), $h_p(t)$ is the traffic intensity (veh/hour) associated with path p at time t, $f_{N(\mu_{pi},\sigma_{pi})}(t)$ is the probability density function (pdf) of the normal distribution, h_{pi} is the daily flow associated with the normal density $N(\mu_{pi},\sigma_{pi})$, and μ_{pi} and σ_{pi} are the time associated with the maximum intensity and the corresponding standard deviation σ_i, which measures the traffic spread of wave i of path p, respectively[2].

For illustrative purposes, in Table 7.2 and Figure 7.3 we present one example of the assumed path flow functions for the illustrative example in Figure 7.1. Note that all of them exhibit two relative maxima around 8.00 *and* 15.00 *hours, which are the peak hours, though different shapes modeled by using different standard deviation parameters $\sigma_i; i = 1, 2, \dots, n_p$.*

As shown in Table 7.2, the assumed maxima intensities of the two traffic waves of all paths takes place close to 8.00 *and* 15.00 *hours, and the corresponding values of σ_{pi} have been simulated randomly. Similarly, the total path flow has been assumed random and it has been randomly distributed between both components.*

[2] The family of functions in (7.3) is one of the many possible alternatives, so that it can be replaced by other family without any problem.

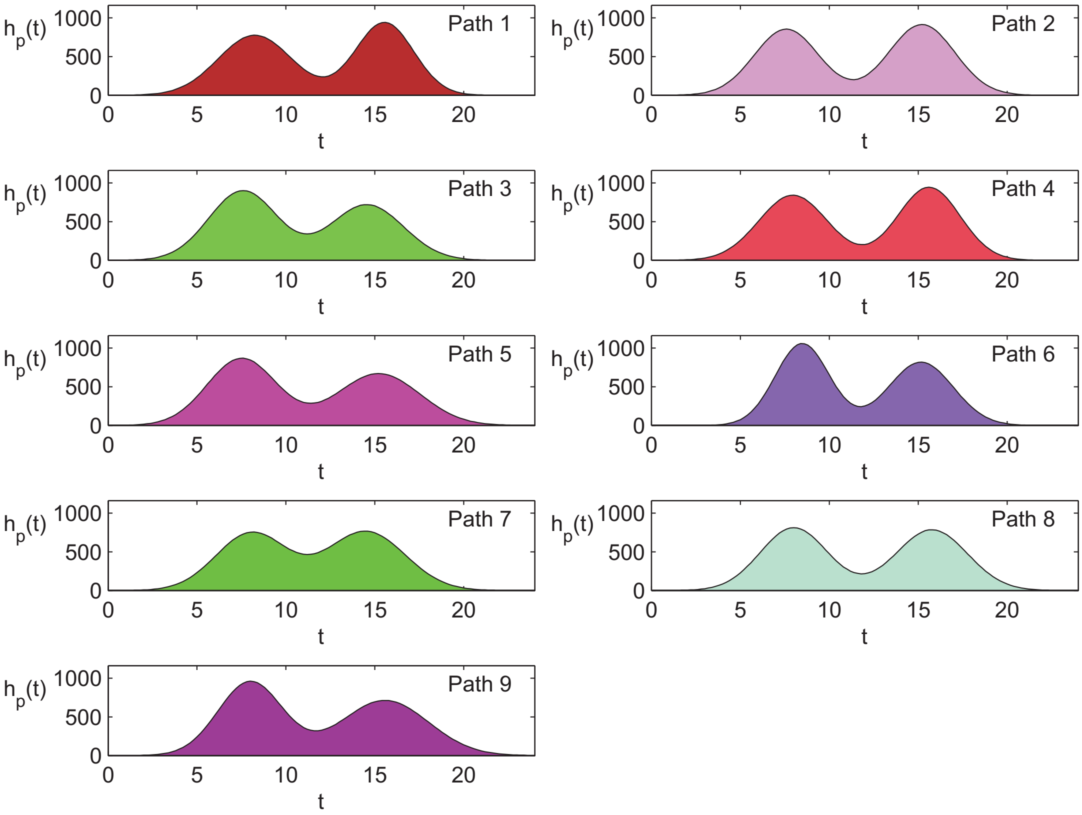

Figure 7.3: Path flow intensities at the path origins as a function of time for the illustrative example.

Finally, we define the cumulative inflow rate at the origin of path p at time t as

$$H_p(t) = \int_0^t h_p(t)dt = \sum_{p=1}^{n_p} h_{pi} \int_0^t q(t; \bar{\alpha}_{pi})dt. \tag{7.4}$$

7.4 Path flow wave propagation

In this section we analyze how the individual path flow waves progress throughout the path and evolve with time. As indicated in the introductory chapters, the traffic flow intensities decrease and the travel times increase with the increasing congestion. More precisely, once each path flow enters the network, it progresses as a wave that stretches or enlarges, depending on the degree of congestion of the traversed links.

Figure 7.4 shows how the path wave progresses throughout the path and evolves with time. The upper plot corresponds to the traffic intensities at the origin node, and the

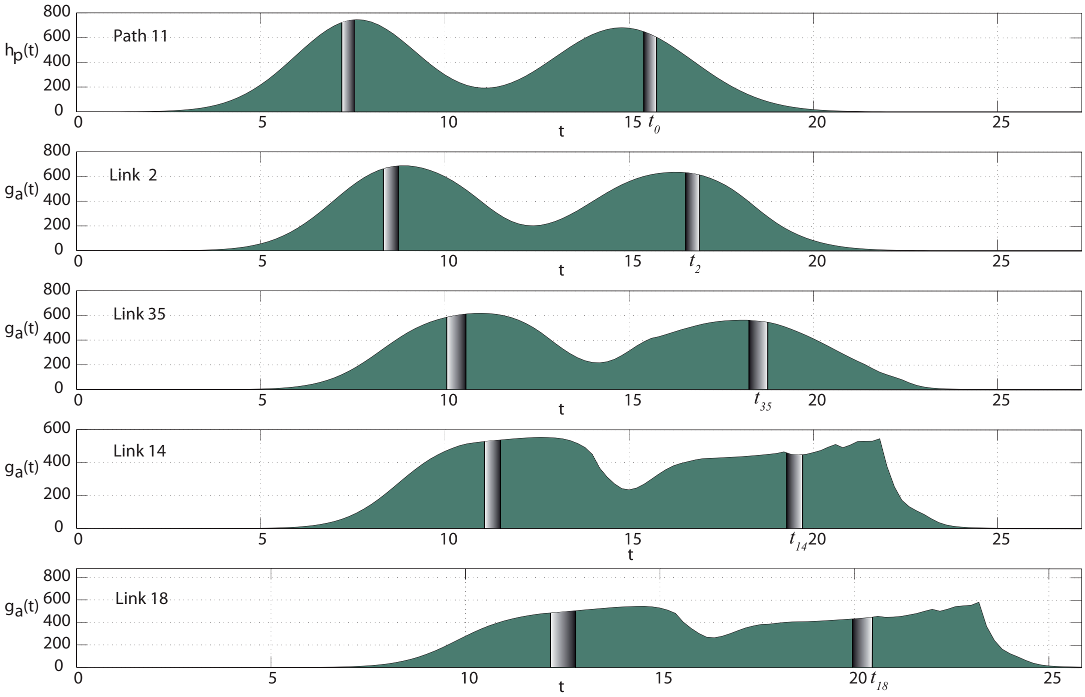

Figure 7.4: Evolution of the wave flow intensity at the origin node and at the ends of all links of path 6 together with the corresponding intervals associated with two arbitrary selected departure time periods.

other four plots indicate how this wave moves with time and deforms due to the increasing congestion. Note that at earlier times in the day the path congestion is nonexistent or low, but later it increases due to the presence of users of other paths, producing a decrease in the traffic intensities, that is, an enlargement of the traffic wave (intermediate and lower plots in Figure 7.4).

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.

In order to see how the individual path waves progress throughout the path, we need to apply the conservation law. In Figure 7.5 we show the traffic intensity wave at two given locations: (a) the path origin and (b) the end of link a of path p, and we show the associated time intervals corresponding to the same users.

If $g^p_{a_i}(t)$ is the outflow rate of link a_i at time t due to path p, since the area associated with a time interval and below the intensity function is the number of users that pass through that location, the flow propagation principle leads to the following equation that expresses the coincidence of the number of users in the time intervals $(t_0, t_0 + dt_0)$ and $(\theta^p_{a_i}(t_0), \theta^p_{a_i}(t_0 + dt_0))$, where $\theta^p_{a_i}(t)$ is the path p departure time of a user who exits the

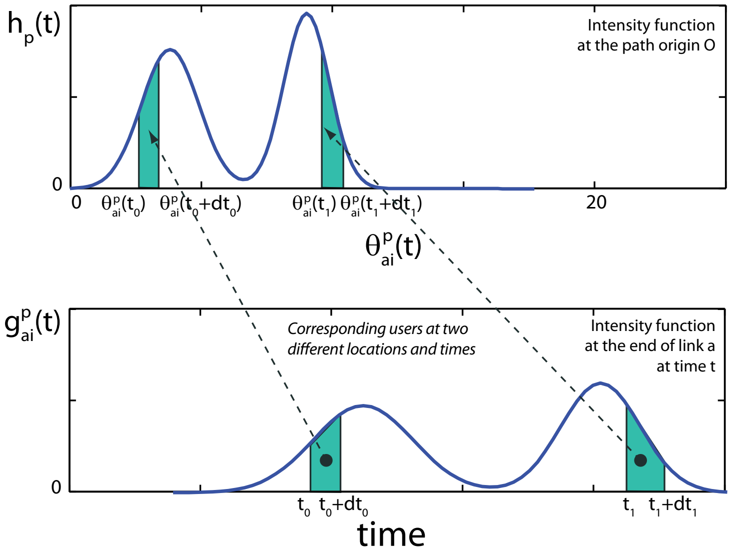

Figure 7.5: Illustration of how the traffic wave satisfies the flow propagation principle and the function $\theta^p_a(t)$.

ith link a_i of path p at time t (see the graph in Figure 7.5 and Friesz et al. (2001)):

$$g^p_{a_i}(t) = h_p(\theta^p_{a_i}(t)) \frac{d\theta^p_{a_i}(t)}{dt}. \tag{7.5}$$

Note that this expression is based on the FIFO rule and that the argument $\theta^p_{a_i}(t)$ of function $h_p(\cdot)$ models the local stretching or enlarging of the wave, and that $\dfrac{d\theta^p_{a_i}(t)}{dt}$ modifies the traffic intensity (wave height) accordingly.

We point out that there exists a different $\theta^p_{a_i}(t)$ function for each link a_i of each path p.

Consequently, the progression of the individual path intensity wave throughout their paths can be calculated as soon as the function $\theta^p_{a_i}(t)$ is known and the resulting wave form is given by Equation (7.5).

Equation (7.5) guarantees that the global mass balance constraint holds, that is,

$$\int_0^t g^p_{a_i}(t)dt = \int_{\theta^p_{a_i}(0)}^{\theta^p_{a_i}(t)} h_p(\theta^p_{a_i}(t))d\theta^p_{a_i}(t), \tag{7.6}$$

which can be used to derive $\theta^p_{a_i}(t)$ when $h_p(t)$ is given.

In summary, the important equation (7.5) allows us to obtain the link flows at time t, if the path flow functions at their origins $h_p(t)$ and the $\theta^p_{a_i}(t)$ function are known, where a is the link. In a following section we indicate how the last function can be obtained.

7.5 Path flow wave interactions

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. This unit combines all individual path flow intensities to obtain the link and node flow intensities.

7.5.1 Link flow intensities

The traffic flow intensity at the exit of link a is given by adding the flow intensities of the corresponding paths (flow conservation principle), that is,

$$g_a(t) = \sum_{p \in \mathcal{P}(a)} g^p_{a_i}(t) = \sum_{p \in \mathcal{P}(a)} h_p\left(\theta^p_{a_i}(t)\right) \frac{d\theta^p_{a_i}(t)}{dt}, \tag{7.7}$$

where $\mathcal{P}(a)$ is the set of paths containing link a.

Note that in Expression (7.7) we use the $\theta^p_{a_i}(t)$ function, which must be known before calculating $g_a(t)$. This fact forces to an iterative approach, as we explain later.

Example 8 (Illustrative example) *Figure 7.6 illustrates this formula showing the flow intensity curves for all links at the end of the iterative process, 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*, the peaks disappear due to the congestion.*

7.5.2 Node flow intensities

Finally, the node flow intensities (excluding the path origin nodes) can be calculated by adding the flow intensities of the corresponding paths (flow conservation principle), that is, by means of the formula

$$r_n(t) = \sum_{a \in \mathcal{A}(n)} g_a(t), \tag{7.8}$$

where $r_n(t)$ is the flow intensity at node n and time t and $\mathcal{A}(n)$ is the set of links entering node n.

Example 9 (Illustrative example) *Figure 7.7 shows the traffic intensity entering the different nodes at the end of the iterative process, that is, after convergence, but excluding*

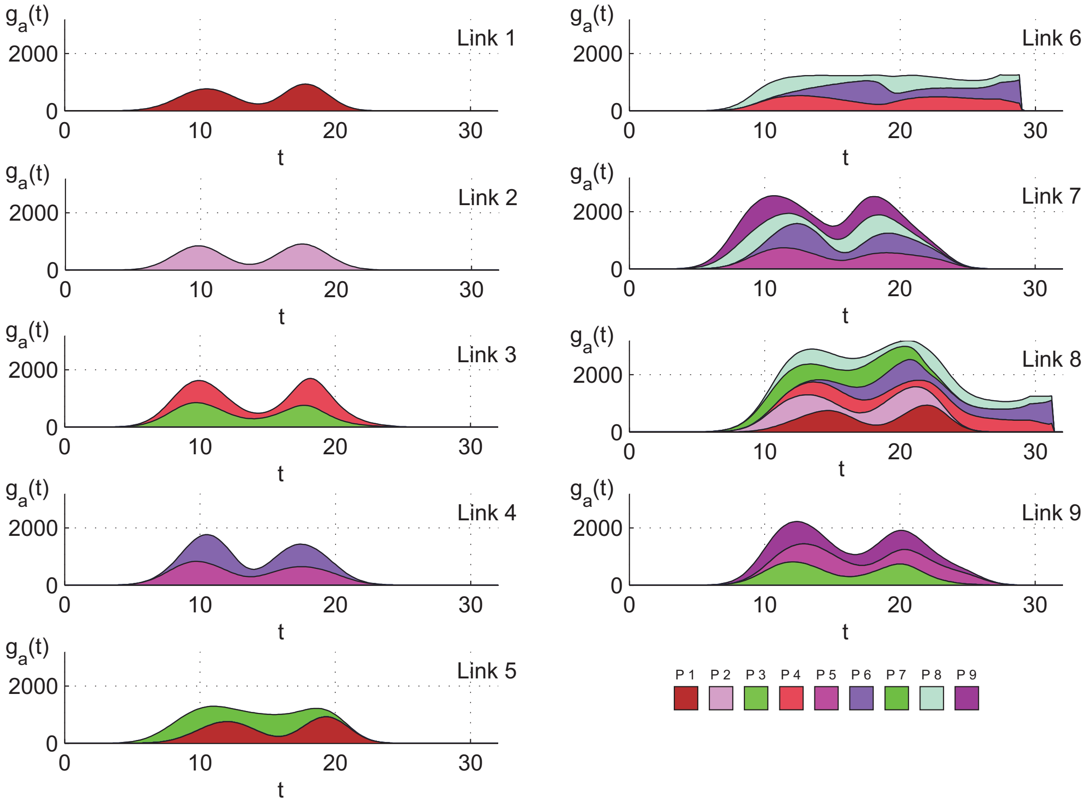

Figure 7.6: Link flow intensity curves showing the corresponding path components for the case of the illustrative example.

the path flows at their origins, 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.

7.6 Congestion analysis

In this section we analyze the congestion in order to determine how the link volume varies with time. To this end, we define $D_a(t)$ as the link travel time for vehicles entering link a at time t. In fact, $D_a(t)$ is a lower bound of the link travel time (see Equation (7.16)), which is active only when no queue exists.

To evaluate the $D_a(t)$ 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 the larger the travel time. In this model, the travel time $D_a(t)$ of a vehicle that enters link a

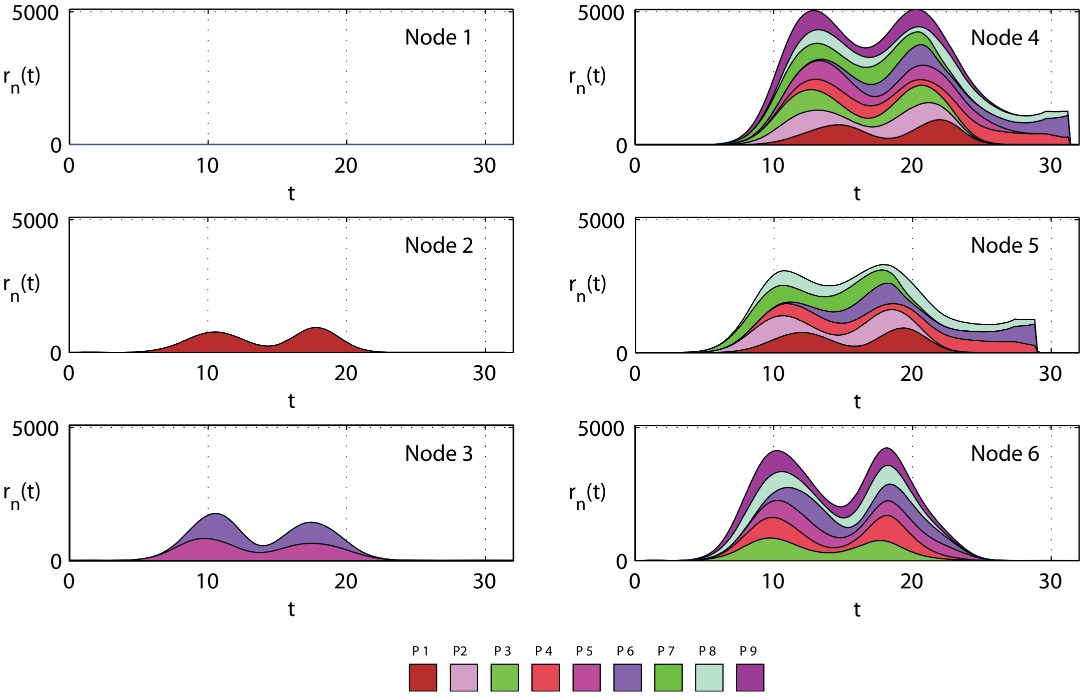

Figure 7.7: Node flow evolution in the illustrative example.

at time t is assumed to be a function of the number of vehicles $x_a(t)$ (link volume) at link a and time t. But, furthermore, in order to reproduce the upstream shock wave due to the congestion, we assume that it depends on the immediate downstream path link volumes. More precisely,

$$D_a(t) = \alpha_a \left[1 + \beta_a \left(s_a(t)\right)^\gamma + \delta_a \max_{b \in \mathcal{S}(a)} \left(s_b(t)\right)^\gamma\right], \tag{7.9}$$

where α_a is the free travel time and β_a, γ are parameters of the travel time function associated with link a, δ_a is a parameter used to take into account the congestion ahead of the link a being considered, $\mathcal{S}(a)$ is the set of all links downstream link a in all its paths, and $s_a(t)$ is a dimensionless ratio measuring the degree of link congestion,

$$s_a(t) = \frac{x_a(t)}{x_a^{max}}, \tag{7.10}$$

where x_a^{max} is a constant measuring the flow producing a given congestion level in the link a (it can be associated with a given service level).

We point out the similitude between Equation (7.9) and the widely-used Bureau of

Public Roads (BPR) function[3]

$$D_a = \alpha_a \left[1 + \beta_a \left(\frac{x_a}{x_a^{max}} \right)^{\gamma} \right]. \qquad (7.11)$$

The BPR values usually are $\beta_a = 1$ and $\gamma = 4$ but, obviously, it depends on the studied case.

Equation (7.9) adds a third term to Equation (7.11) in order to consider the shock wave effect, going back from the congested area to the studied link.

We note that $\delta_a << \beta_a$ because the influence of the degrees of saturation of downstream links on the a link travel time must be smaller than the influence of its own degree of saturation.

The travel time function (7.9) is an important part of the overall model. The term headed by β_a penalizes the number of cars inside link a at entry time t, and the term headed by δ_a penalizes high congestions on any path immediately ahead of link a and allows us to propagate congestion upstream of link a.

Figure 7.8 shows the influence of the D_a parameters, namely, β_a, γ, and δ_a, and the maxima downstream saturation $max(s_b)$. The magenta line, as reference, shows the $\beta_a = 1$, $\gamma = 4$, $\delta_a = 0.1$ and $max(s_b) = 1$ case. We see that higher β_a values involve higher travel times, and this increase is larger for larger s_a values. Moreover, for high γ values, the travel time increases faster when s_a is close to 1, but not for small values of s_a. Also note the back effect of the shock wave due to the δ_a and $max(s_b)$ terms when a downstream link is congested: when δ_a is around β_a, the shock wave influence is important (it can duplicate the travel time), mainly when the studied link is low congested. The graphic of the effect of the downstream link saturation shows that $max(s_b) \leq 1$ values hardly influence D_a, but when $max(s_b) > 1$, a constant delay is added to the link travel time. Accordingly, we want to point out that the model calibration is fundamental.

The number of vehicles on link a at time t that appears in (7.10) and is used in (7.9), can be calculated by:

$$x_a(t) = G_a(t_a^{out}) - G_a(t) \quad \forall a, \forall t, \qquad (7.12)$$

i.e., the difference of the link cumulative outflows between the link entry time t and the link exit time t_a^{out}. The cumulative number of vehicles that has left the link a until time t is given by

$$G_a(t) = \sum_p H_p\left(\theta_a^p(t)\right), \qquad (7.13)$$

where $H_p(\cdot)$ is the expression in (7.4).

[3] The advantage of the BPR formula is that uses flows that are not restricted by link capacity, so we can know the network behavior. Moreover, it is a function easy to compute and evaluate. To use this function it is necessary to evaluate the parameter values for each particular network.

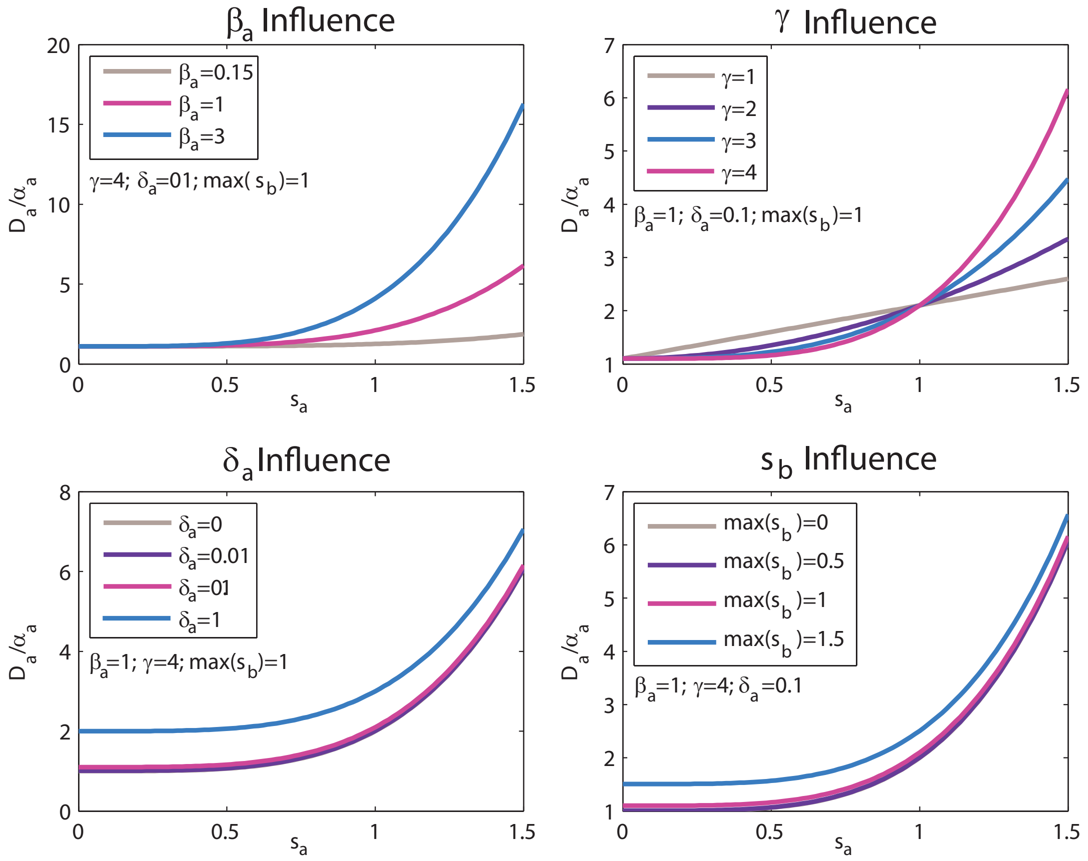

Figure 7.8: Influence of β_a, γ, δ_a y $max(s_b)$ in the link travel time function.

If we define $\tau_a^{-1}(t)$ as the entry time of a user that exits link a at time t, $\theta_a^p(t)$ can be expressed as follows:

$$\theta_a^p(t) = \tau_{a_n}^{-1}(\tau_{a_{n-1}}^{-1}(\dots \tau_{a_1}^{-1}(t))), \tag{7.14}$$

where $a_1, a_2, \dots, a_n$ are the links of path p. In this model we propose to interpolate the $\tau_a^{-1}(t); \forall a$ functions in order to evaluate (7.14). We note that spline functions have been used before by other authors to approximate travel time functions, but for reproducing path travel times and not necessarily monotone (see Rubio-Ardanaz et al. (2003)).

We point out 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 time functions alone.

The main problem is that initially we do not know the number of vehicles $x_a(t)$ inside link a, and we need to make an initial guess to later iterate until convergence. This is explained in Section 7.8

7.7 Approximation of the $\tau_a^{-1}(t)$ function

As we have indicated, the model aims to approximate the continuous $\tau_a^{-1}(t)$ functions by mean monotonic splines. We approximate the inverse of the link travel time functions instead of the link travel time functions because we need it to evaluate the path travel times and we can use this function directly without inverting it.

On the paragraphs below, we explain this approximation and why this interpolation is adequate.

7.7.1 Interpolation by cubic monotonic Hermite splines

A spline, $S^{(m)}(x)$, is a m-order curve defined by $n-1$ pieces in the interval $[a,b]$ defined by means of polynomials, such as:

$$\begin{aligned} S^{(m)}(x) &= p_1(x) && x_1 \leq x < x_2, \\ S^{(m)}(x) &= p_2(x) && x_2 \leq x < x_3, \\ &\vdots \\ S^{(m)}(x) &= p_{n-1}(x) && x_{n-1} \leq x \leq x_n, \end{aligned}$$

where $\{a = x_1, x_1, x_2 ..., x_n = b\}$ are the known or basic points.

They are usually used because they yield similar results than the original functions but requiring low-degree polynomials. So, we avoid the oscillations when high-degree polynomial interpolation are used.

The widely-used interpolation is the cubic one. If we assume a bidimensional space, where the polynomials $p_i(x)$ form a 3-order spline $S^{(3)}(x)$ in the interval $[a,b]$, then, for a set of n basic points, there exist $n-1$ polynomials of degree 3 to fit, then $4(n-1)$ unknown values. We have to impose the following fitting conditions:

- each polynomial $p_i(x)$ stars and finishes at two basic points of the set $\{a, x_1, x_2 ..., b\}$ ($2(n-1)$ conditions).
- the first-derivatives of two adjacent polynomials $p_i(x)$ at a basic point are the same value ($(n-2)$ conditions).
- the second-derivatives of two adjacent polynomials $p_i(x)$ at a basic point are the same value ($(n-2)$ conditions).

However, two more conditions need to be specified, which can be two of the following ones:

- the second-derivative of $S^{(3)}(x)$ at the endpoints, a and b, are some known values (2 conditions). As particular case, zero curvature can be assumed.
- the first-derivative of $S^{(3)}(x)$ at the endpoints, a and b, are the same unknown value (1 condition).

- the second-derivative of $S^{(3)}(x)$ at the endpoints, a and b, are the same unknown value (1 condition).

Due to the fact that each solved curvature is only related to the curvature of its two adjacent neighbors, the solution matrix for a cubic spline is always tridiagonal.

Most of the linear algebra algorithms require much less computer cost when tridiagonal matrices are used, thus, this interpolation is significatively more efficient.

On the other hand, the Hermite interpolation is a method to obtain the lowest order polynomial function, $P^{(m)}(x)$, that fits the basic points, the function itself and some of its derivatives. It consists in calculating the successive derivatives in the basic points, so that the polynomial[4] is closer to the real function. Moreover, the Hermite interpolate function keeps other imposed requisites as the monotonicity, concavity, etc.

In the proposed model we use the interpolation by cubic monotonic Hermite splines with the following conditions:

- on each interval, $P^{(3)}(x)$ is fitted in the basic points and certain slopes at the two endpoints.

- the first derivative of $P^{(3)}(x)$ is continuous. However, the second derivative is probably not continuous and there may be jumps at the second derivative.

- The slopes of $P^{(3)}(x)$ at the basic points are chosen in such a way that $P^{(3)}(x)$ preserves the shape of the data and respects monotonicity. This means that, on intervals where the basic data are monotonic, so is $P^{(3)}(x)$; at points where the data has a local extremum, so does $P^{(3)}(x)$[5].

The difference between the standard splines interpolation and the Hermite splines interpolation is that, in the first case, the slopes are chosen in such a way that the second derivative is continuous. This produces a more accurate result if the data consists of values of a smooth function. However, the Hermite splines have no overshoots and less oscillation if the data are not smooth. The two are equally expensive to evaluate.

The link entry time functions, $\tau^{-1}(t)$, are approximated by cubic monotonic Hermite splines in order to satisfy the FIFO rule. The monotonic Hermite splines guaranty that, if the basic points have a increasing character, the whole spline is increasing, i.e., the FIFO rule is satisfied.

7.7.2 Selection of the basic points

The basic points for the Hermite interpolation are chosen as follows:

1. We divide the studied time interval into $n-1$ disjoint and exhaustive intervals $(t_k, t_{k+1}); k = 1, 2, \dots, n-1$, where $t_1 = 0$ and t_n is the end time.

[4] This polynomial exists and is unique.

[5] In section 7.7.3 we explain that the basic points are corrected in order to force the monotonic trend.

2. We analyze users who enter the links at times $t_1, t_2, \dots, t_n$ and calculate the number of link users $x_a(t)$ at the time of arrival to the link.

3. With this information, we use the flow propagation principle and the congestion equation (7.9) to obtain the times $D_a(t_k)$ required for these users to travel each path link. Accordingly, we calculate t_{ak}^{out}, i.e., the link a exit time when a user enters at the discrete time t_k.

4. The set of points $\{(t_{ak}^{out}, t_k); \forall k\}$ is used to fit the cubic Hermite splines to approximate $\tau^{-1}(t)$ and thus, the travel time functions for each link a $(D_a(t))$.

7.7.3 The FIFO rule consistency

In order to satisfy the FIFO rule, we work with monotone functions $\tau_a^{-1}(t)$, i.e., the necessary and sufficient condition for the FIFO condition to hold. The $\tau_a^{-1}(t)$ functions are monotone splines, and the monotonicity is guaranteed by means of two conditions: (a) a set of monotone increasing basic points $\{(t_{ak}^{out}, t_k) | k = 1, 2, \dots, n\}$ used to fit the splines, and (b) the use of monotone Hermite splines, preserving its increasing character at all points.

In other words, we approximate the $\tau_a^{-1}(t)$ function by means of

$$\tau_a^{-1}(t) \approx spline_a(t; (\mathbf{t_{ak}^{out}}, \mathbf{t_k})), \tag{7.15}$$

where $\mathbf{t_{ak}^{out}}= (t_{a1}^{out}, t_{a2}^{out}, \dots, t_{an}^{out})$ and $\mathbf{t_k}= (t_1, t_2, \dots, t_n)$ are sets of discrete entry times where the link a exit times are evaluated. Note that identical entry times are selected for all links.

In order to respect the FIFO rule, we update t_{ak}^{out} as follows:

$$t_{ak}^{out} = \max \left\{ t_k + D_a(t_k), t_{ak-1}^{out} + Q_{ak} \right\}, \tag{7.16}$$

where Q_{ak}, is the queue dissipation time at link a at time t added in order to include the traffic jam effect, as follows,

$$Q_{ak} = \left[G_a(t_{ak}^{out}) - G_a(t_{ak-1}^{out}) \right] \frac{\alpha_a(1 + \beta_a)}{x_a^{max}}. \tag{7.17}$$

Note that preventing the overtaking means queueing, and this is what expression (7.16) does.

Figure 7.9 shows two users A and B and their positions at four different instants t_{k-1}, t_k, t_{ak-1}^{out} and t_{ak}^{out} in the neighborhood of link a, where the first two correspond to the entering times of these users, and the last two are the corresponding exit times. Since $D_a(t_k)$ is a lower bound of the link travel time of user B, we have

$$t_{ak}^{out} \geq t_k + D_a(t_k). \tag{7.18}$$

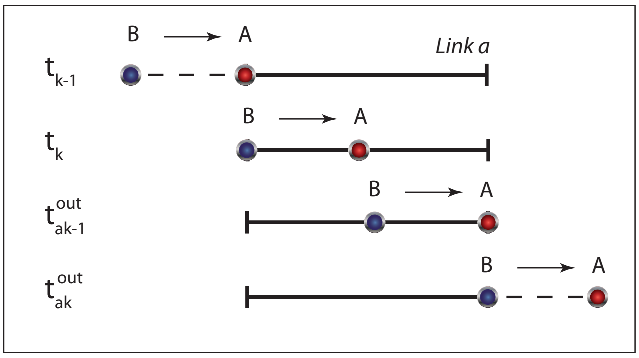

Figure 7.9: Illustration of the different terms in Expressions (7.16) and (7.17).

On the other hand, since $G_a(t^{out}_{ak}) - G_a(t^{out}_{ak-1})$ is the number of users traveling between users A and B, if we assume a degree of saturation $s_a(t_k) = 1$, the time required for user B to exit the link is Q_{ak} and then

$$t^{out}_{ak} \geq t^{out}_{ak-1} + Q_{ak}. \tag{7.19}$$

Then, from (7.18) and (7.19), we get (7.16).

The expression (7.17) is based in the proportionality rule, in such a way that if there is $G_a(t^{out}_{ak}) - G_a(t^{out}_{ak-1})$ vehicles, since the saturation is $x_a/x^{max}_a = 1$, the link traversal time required is $\alpha_a(1 + \beta_a)$[6].

If we assume a strictly increasing $G_a(t)$ function, (7.17) provides a $Q_{ak} > 0$ if $t^{out}_{ak} > t^{out}_{ak-1}$ (this condition is forced in the iterative procedure to be described in Section 7.8), and then (7.16) guarantees that the set of points $\{(t^{out}_{ak}, t_k)|k = 1, 2, \ldots, m\}$ has an increasing trend.

Note that, in fact, (7.16) can imply an increase of the link travel time if the second term (a queuing term) $t^{out}_{ak-1} + Q_{ak}$ is greater than the first term $t_k + D_a(t_k)$.

Example 10 (Illustrative example) *To illustrate, Figure 7.10 shows the described process. Path* 6 *has been chosen, which has four links (*4*,* 7*,* 6 *and* 8*).*

The parameters used in the congestion function (7.9) for the network are indicated in Table 7.3. We have assumed a free flow speed of 100 *Km/hour and, in order to analyze the upstream congestion propagation, a value* x^{max}_6 *that is a* 25% *of the values for the remaining links.*

The left figure 7.10 shows the link travel time functions during 32 *hours obtained. Note the high congestion in link* 6 *due to the assumed low value of* x^{max}_6*. An associated light upstream congestion propagation can be observed at link* 7*.*

[6] Note that the usual value $\beta_a = 1$, when the saturation is 100%, implies that the link travel time is twice as much as the free-flow link travel time.

link	$\alpha_a(hours)$	β_a	δ_a	γ	x_a^{max} (veh.)
1	2.24	1.00	1/3	2.00	22360.68
2	2.24	1.00	1/3	2.00	22360.68
3	2.24	1.00	1/3	2.00	22360.68
4	2.24	1.00	1/3	2.00	22360.68
5	1.41	1.00	1/3	2.00	14142.14
6	2.00	1.00	1/3	2.00	5000.00
7	1.41	1.00	1/3	2.00	14142.14
8	2.24	1.00	1/3	2.00	22360.68
9	2.24	1.00	1/3	2.00	22360.68

Table 7.3: Parameters used in the congestion function.

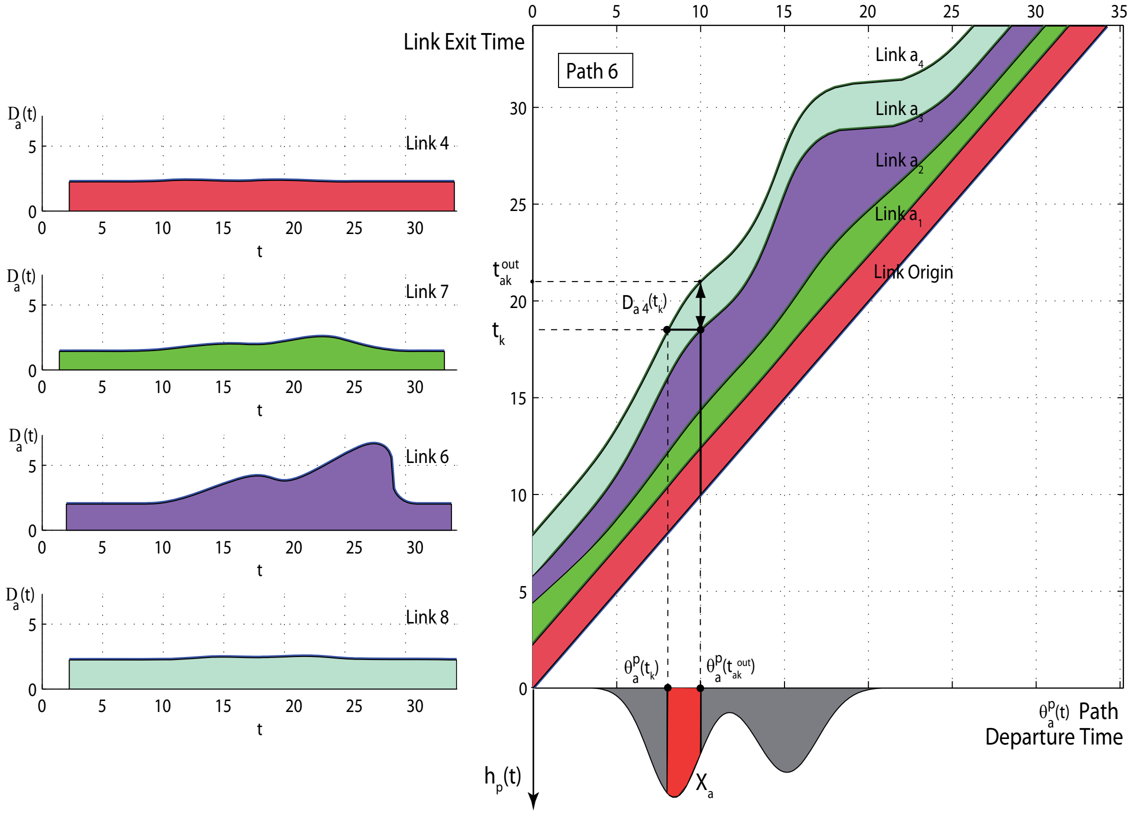

Figure 7.10: Left figure: Travel time functions of all links of path 6. Right figure: link exit time function for path 6 and illustration of how the link flow $x_{a_4}(t)$ is evaluated.

The right figure 7.10 shows the flow intensity function at the origin and the link exit time curves, in order to obtain the link travel time $D_a(t_k)$. The link exit times are function of the corresponding path departure times. Since path 6 *has four links, we show a set of five curves, including the path origin curve, that is, the identity function $f(t) = t$. These curves provide the exit times of all the links of a user starting the trip at time t obtained from the link travel time functions shown on the left part of the figure.*

To approximate the link travel time curves by cubic Hermite splines, we select a discrete set of points $\{(t_{ak}^{out}, t_k); k = 1, 2, \dots, n\}$, where $t_k = (k-1)\delta$ with $n = 32/\delta + 1$ 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. Figure 7.10 illustrates how the values of $x_a(t)$ and $D_a(t)$ are calculated at a generic point t_k of this set of points based on the actual approximation (splines) of the $D_a(t)$ functions. Given a pair (t_{ak}^{out}, t_k) of exit-entry times to link a, which is obtained from the spline approximation, we obtain the corresponding path departure times $\theta_a^p(t_k)$ and $\theta_a^p(t_{ak}^{out})$, which allows us to obtain the values of $G_a(t_k)$ and $G_a(t_{ak}^{out})$, using expression (7.13), and the value of $x_a(t_k)$ by means of (7.12)*

$$x_a(t_3) = \int\limits_{\theta_a^p(t_k)}^{\theta_a^p(t_{ak}^{out})} h_p(t)dt.$$

This allow us to evaluate $D_a(t)$ using Expression (7.9).

We note that the above process requires the knowledge of $\theta_a^p(t)$, as the inverse exit link time functions. There are two possible options to deal with these functions:

1. Use cubic splines to fit these functions. This option requires fitting a high number of splines (number of routes × number of links per route), which is memory and time consuming.

2. Build these functions based on the $\tau_a^{-1}(t)$ functions. This process requires only to fit as many functions as links, but has the inconvenience that each evaluation is more costly (travel times of all links in a path need to be evaluated).

The second is the one used in the examples and in the MatLab computer implementation that has been developed for the examples.

The organigram in Figure 7.11 shows how to build the exit link time functions $\theta_{a_i}^p(t)$ based on the $\tau_{a_i}^{-1}(t)$ functions.

Example 11 (Illustrative example) *Figure 7.12 shows the $\theta_{a_i}^p(t)$ functions obtained after using the Hermite spline interpolation between the selected discretized times with the units being hours. The number of curves coincides with the number of links per path. The link exit-entry time functions have been obtained by adding the departure time function*

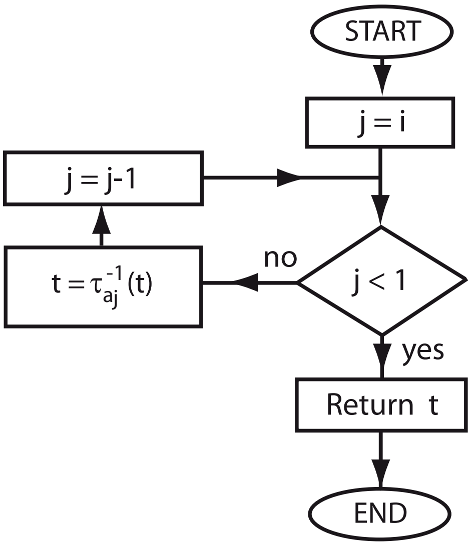

Figure 7.11: Organigram: How to obtain the $\theta^p_{a_i}(t)$ function based on the link exit-entry time function $\tau^{-1}_{a_i}(t)$. We enter with the link exit time t and the resulting value t is the desired path starting time.

$f(t) = t$ and the corresponding link travel time functions for all paths with the corresponding link. If we pay attention to path 6, it is easy to identify when is the worst time to start the path travel. In this case (see Figure 7.13) the segments t_4, t_7, t_6 and t_8 show the time needed for this traveler to cover the different links belonging to this path (links 4, 7, 6 and 8). At the same time it is interesting to realize 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.

Note that each $\theta^p_a(t)$ time function allows us to identify the instant at which a traveler reaches the end of every link during the journey.

7.8 Inference engine

Since, as indicated previously, the $x_a(t)$ and $D_a(t)$ 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 initially need to assume a 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 following algorithm shows how the iterative process works and is the core of the proposed method. Note that we have a triple control of the iterations:

Number of iterations: when it exceeds the number of iterations $iter_{max}$ implies that the

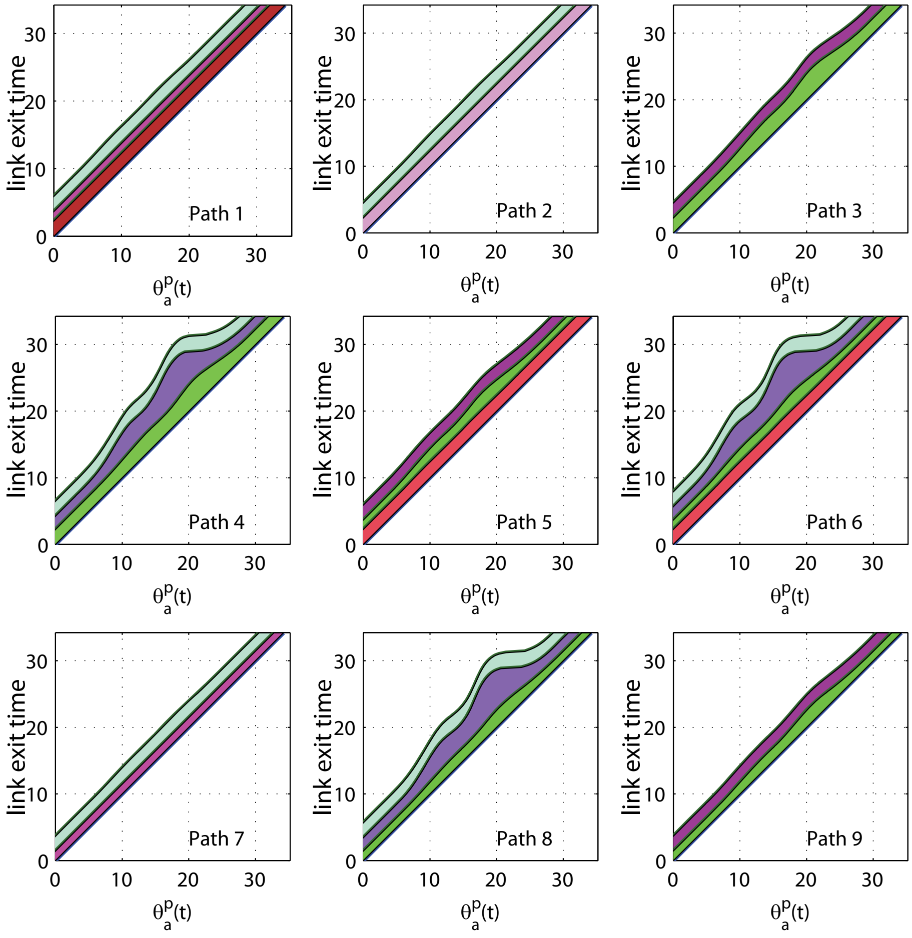

Figure 7.12: Illustrative example. Link exit time functions obtained by adding the departure time function $f(t) = t$ and the corresponding link travel time functions for all paths with the corresponding link.

problem does not converge. In this case, the program has to stop.

Tolerance: when the error between consecutive iterations is less than a permitted tolerance *tol*, it is assumed that the problem has correctly converged, and the iterative process finishes.

Maximum link saturation: the fact that the maximum saturation exceeds a given threshold sat^0_{max} is an indication that the flow cannot be supported by the network and it becomes blocked by traffic. In that case, we stop the iterative process.

Algorithm 2 (Evaluation of the $\tau_a^{-1}(t)$ functions.) [7]

[7] To understand this algorithm it is convenient to refer to Figure 7.10.

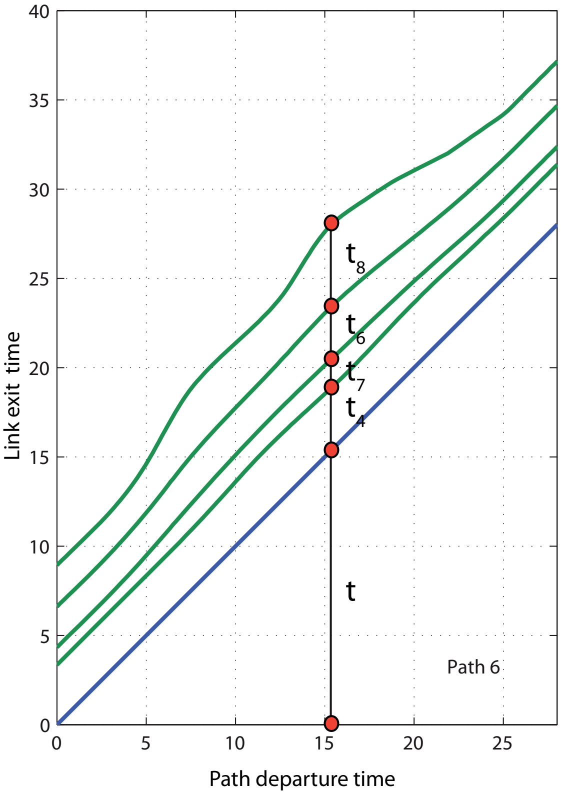

Figure 7.13: $\theta_a^p(t)$ curves showing the largest congestion time for path 6 and the corresponding departure time and leaving times of all links of path 6.

INPUT: *The network topology, the set of intervals* $(t_k); k = 1, 2, \ldots, n$, *the free travel times of all links* α_a, *the link parameters* $\beta_a, \delta_a, x_a^{max}$ *and* γ, *the tolerance tol, the maximum allowed saturation* sat^0_{max} *and the maximum number of iterations* $iter_{max}$.

OUTPUT: *The set of functions* $\{\tau_a^{-1}(t)|a \in \mathcal{A}\}$.

1. **Initiation.** *Let the initial basic set of points for the* $\tau_a^{-1}(t)$ *spline function be* (t_{ak}^{out}, t_k), *where* $t_{ak}^{out} = t_k + \alpha_a$, *and let error* $= 2$ *tol, iter* $= 1$ *and* $sat_{max} = 0$.

2. **Step 1. Check convergence or out of limit conditions.** *While error* $>$ *tol and iter* $< iter_{max}$ *and* $sat_{max} < sat^0_{max}$ *repeat Steps 2 to 6. Otherwise, go to Step 11.*

3. **Step 2. Consider all points** t_k **of the discrete set of times.** *For* $k = 1$ *to* n *repeat Steps 3 to 9.*

4. **Step 3. Consider all links.** *For $a = 1$ to n_{links} repeat Steps 4 to 9.*

5. **Step 4. Consider all paths with link a.** *For all paths with link a repeat Steps 5 to 9.*

6. **Step 5. Obtain associated departure times.** *For each paths with link a calculate departure times $\theta_a^p(t_k)$ and $\theta_a^p(t_{ak}^{out})$ associated with the pair of values (t_{ak}^{out}, t_k), where the $\theta_a^p(t)$ are evaluated using the actual spline functions and the formula (7.14):*

$$\theta_a^p(t) = \tau_{a_n}^{-1}(\tau_{a_{n-1}}^{-1}(\dots \tau_{a_1}^{-1}(t))).$$

7. **Step 6. Evaluate cumulative flows associated with each path.** *For each of the above paths, evaluate the cumulative flow associated with $\theta_a^p(t_k)$ and $\theta_a^p(t_{ak}^{out})$, and accumulate it using Expression (7.13):*

$$G_a(t) = \sum_p H_p\left(\theta_a^p(t)\right).$$

8. **Step 7. Calculate $x_a(t_k)$.** *Calculate $x_a(t_k)$ using the formula in Expression (7.12):*

$$x_a(t_k) = G_a(t_{ak}^{out}) - G_a(t_k).$$

9. **Step 8. Calculate $D_a(t_k)$.** *Calculate $D_a(t_k)$ using Expressions in (7.10) and (7.9), that is,*

$$D_a(t_k) = \alpha_a \left[1 + \beta_a \left(s_a(t_k)\right)^\gamma + \delta_a \max_{b \in \mathcal{S}(a)} \left(s_b(t_k)\right)^\gamma\right],$$

where

$$s_a(t_k) = \frac{x_a(t_k)}{x_a^{max}}$$

and update the maximum saturation as

$$sat_{max} = \max\left(sat_{max}, s_a(t_k)\right).$$

10. **Step 9. Update t_{ak}^{out} values to t_{ak}^{out}.** *Update the basic exit and entry times (t_{ak}^{out}, t_k) by means of Expressions (7.17) and (7.16):*

$$Q_{ak} = \left[G_a(t_{ak}^{out}) - G_a(t_{k-1}^{out})\right] \frac{\alpha_a(1 + \beta_a)}{x_a^{max}}$$

and

$$t_{ak}^{out} = \max\left\{t_k + D_a(t_k), t_{ak-1}^{out} + Q_{ak}\right\}.$$

11. **Step 10. Update splines.** *The splines are updated by means of (7.15)*

$$\tau_a^{-1}(t) \approx spline_a(t; (\mathbf{t}_a^{out}, \mathbf{t}))$$

and go to Step 1.

12. **Step 11. End of algorithm.** *If $iter = iter_{max}$ inform the user about a convergence problem. If $sat_{max} \geq sat^0_{max}$ inform the user about a saturation problem (network blocked by an excessive flow). Otherwise, return actual splines as the final ones and used them to evaluate all flow functions.*

With respect to convergence we have experienced no problems. In fact, we can use a relaxation factor to avoid oscillations, so that the iterations converge monotonically to the solution. We start from a free flow solution and iterate in such a way that congestion increases monotonically. The convergence is very fast (a few iterations).

This model satisfies important properties, such as causality and link and route FIFO properties, but not the OD FIFO property, because users have not complete information about travel times of all paths. However, the differentiability with respect to path flows does not hold due to the non-differentiability of Equations (7.9) and (7.16).

To finish, in Appendix 10 we show the Matlab implementation of the exposed model and, in order to clarify, a diagram of the procedure dependence.

Example 12 (Illustrative example) *To conclude the illustrative example, figure 7.14 shows the path* 6 *flow evolution at the end of all its links.*

In this case, we see an important distortion due to the congestion produced at link 6 *when the link intensity is higher than x_a^{max}. More precisely, it maintains its shape (wave length and height) at links* 4 *and* 7 *due to lack of congestion. At link* 6 *the wave suffers two different processes: (a) an elongation, due to congestion, roughly between* 22 *and* 26 *hours, and (b) a wave stretching (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, we point out that the cpu time required to solve the illustrative problem on a portable DELL Latitude E6500 computer, was 47.00 *sec.*

7.9 Examples of application

In this section we apply the proposed method to more complicated examples to show that it is applicable in real practice.

7.9.1 The Nguyen-Dupuis example

In this section we use the simple Nguyen-Dupuis type network, with 13 nodes and 38 links in Figure 6.2, to illustrate the proposed methods. The paths considered are those in Table 7.4.

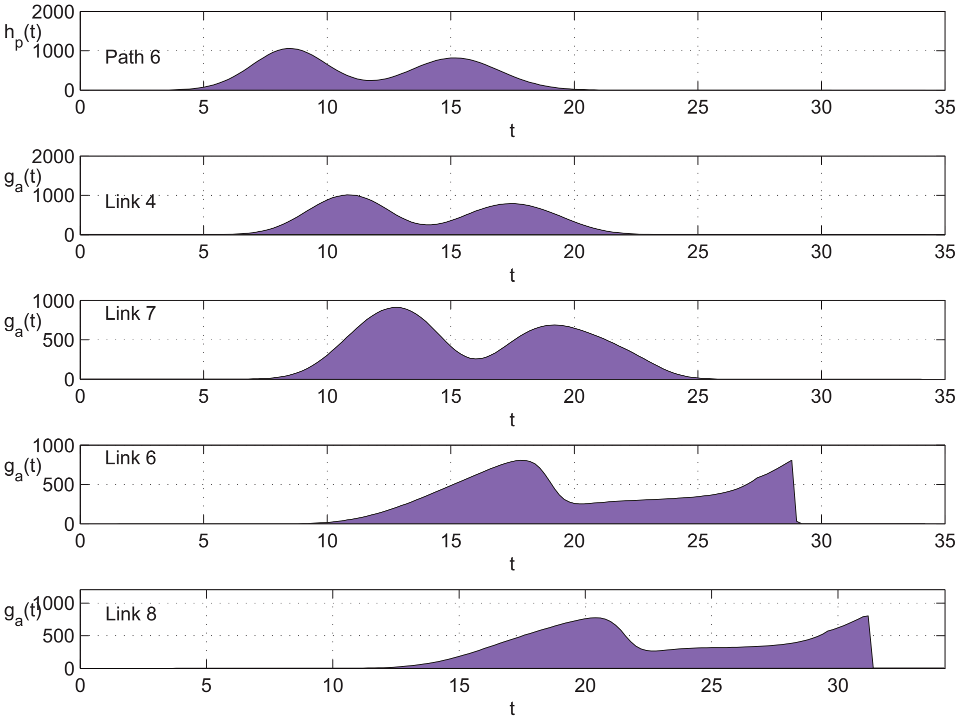

Figure 7.14: Illustrative example. Path 6 flow evolution at the end of all its links.

Four of the assumed path intensity functions at their origins are shown in Figure 7.15. Note that they consist of linear combinations of two normal densities with maxima around 8.00 and 15.00 hours, and different spreads. A complete description of the assumed path flow intensities is given in Table 7.5.

We have used the following parameter values: $\beta_a = 1$, $\delta_a = 1/3$ and $\gamma = 3$, the α_a values have been obtained using a free flow speed of 100 Km/hour, and the $x_a^{max}; \forall a$ correspond to 100 vehicles/Km.

Once these path flows have been propagated through the network, we can see the resulting link intensity functions. Some are shown in Figure 7.16. Note that they are combinations of the path waves with different delays.

The link travel time functions are shown in Figure 7.17, where we can see how traffic congestion in the different links varies with time.

The node flow intensity functions appear in Figure 7.18, where it can be seen that they are linear combinations of path flows, and more complex than link functions because they involve more paths.

It is interesting to see the upstream congestion caused by link 14 on links 11 and 35, where the induced same trends of the link travel time function can be observed.

Path links						
Path	Links					
1	1	11	14	18	20	
2	2	35	14	18	20	
3	2	36	20			
4	1	11	14	19	31	
5	1	11	15	29	31	
6	1	12	25	29	31	
7	1	12	26	37		
8	2	35	14	19	31	
9	2	35	15	29	31	
10	1	11	14	18		
11	2	35	14	18		
12	2	36				
13	3	21	17	13	9	
14	3	21	17	16	34	
15	3	22	34			
16	3	21	17	13	10	
17	3	21	19	33	28	23
18	4	33	28	23		
19	3	21	17	16		
20	3	22				
21	5	32	17	13	9	
22	5	32	17	16	34	
23	5	33	27	13	9	
24	5	33	27	16	34	
25	5	33	28	24	9	
26	6	38	24	9		
27	5	33	28	23		
28	6	38	23			
29	5	32	17	16		
30	5	33	27	16		

Path links						
Path	Links					
31	7	11	14	18	20	
32	8	25	29	30		
33	8	25	29	32	18	20
34	8	25	29	31		
35	8	26	37			
36	7	11	14	18		
37	8	25	29	32	18	
38	21	17	13	9		
39	21	17	16	34		
40	22	34				
41	21	17	13	10		
42	21	19	33	28	23	
43	21	17	16			
44	22					
45	35	14	18	20		
46	36	20				
47	35	14	19	31		
48	35	15	29	31		
49	35	14	18			
50	36					
51	37					
52	38					
53	34					
54	1					
55	2					
56	3					
57	5					
58	33	27				
59	33					
60	29					

Table 7.4: Set of paths with the corresponding links considered for the Nguyen-Dupuis network example.

Finally, the resulting $(\tau^p)_a^{-1}(t)$ functions after convergence of the iterative process appear in Figure 7.19.

Figures 7.16, 7.17, 7.18, 7.19 and 7.20 have been built with the same criteria as in the case of our simple example described above. In this case, due to the high number of paths (60), only paths 1, 4, 11 and 47 have been considered, but link and node flows show all paths passing throughout them. It is particularly interesting to see what the evolution of flow corresponding to path 11 is with important delays in the arrival times (see Figure

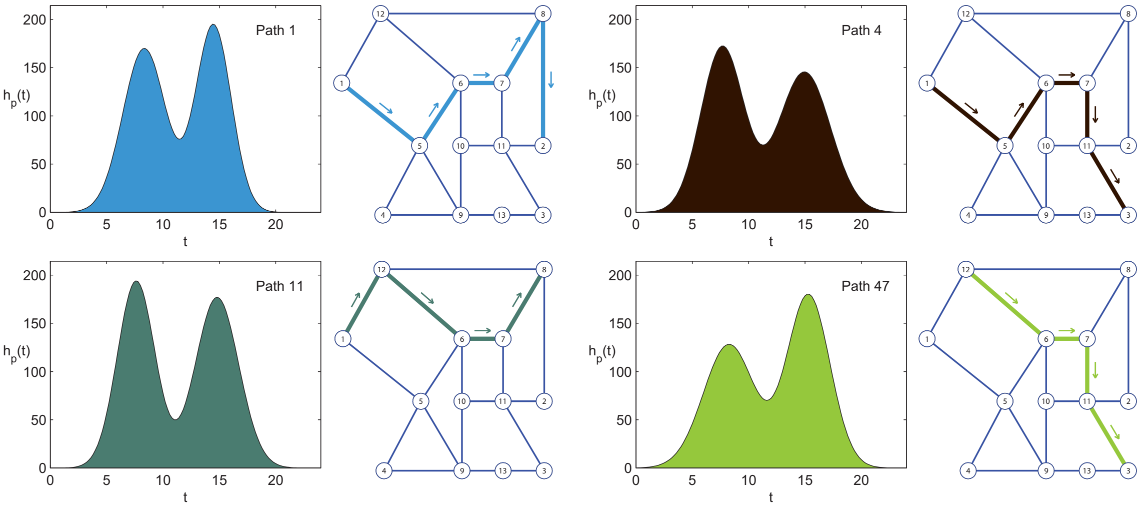

Figure 7.15: The Nguyen-Dupuis network showing the path intensity functions at their origins.

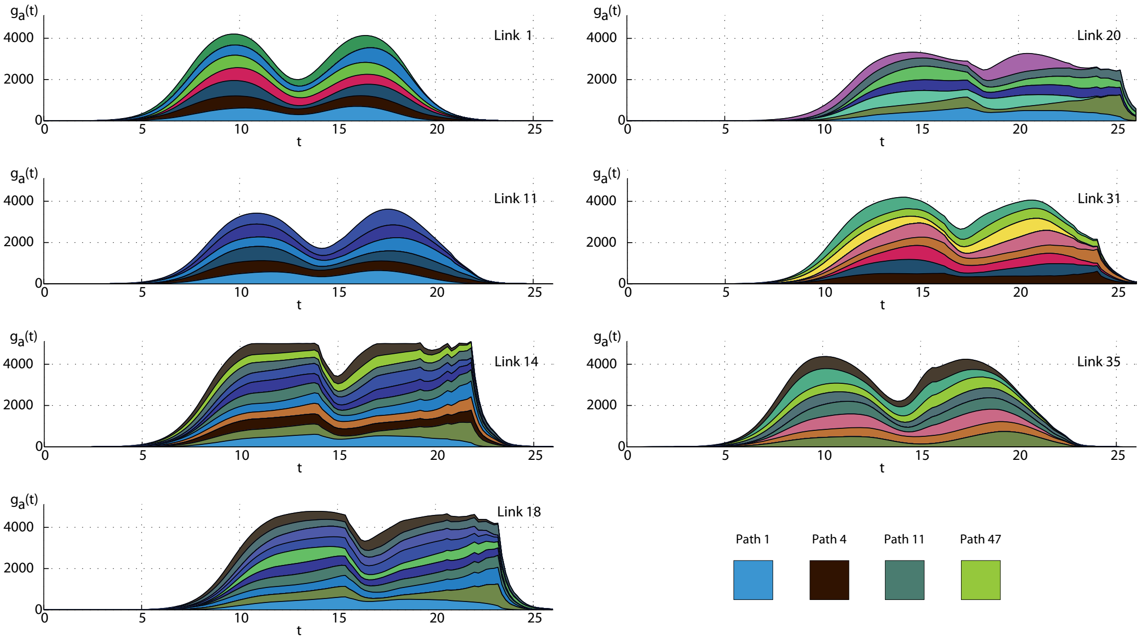

Figure 7.16: The Nguyen-Dupuis network showing the link intensity functions at their exits.

7.19). This path shows a significant evolution of its second wave in links 35, 14 and 18 with an increase of its crest.

Path	μ_{r1}	σ_{r1}	h_{p1}	μ_{r2}	σ_{r2}	h_{p2}	Path flow	Path	μ_{r1}	σ_{r1}	h_{p1}	μ_{r2}	σ_{r2}	h_{p2}	Path flow
1	8.34	1.90	3004	14.48	1.58	2905	5909	31	8.29	1.77	2972	15.16	1.72	2959	5931
2	7.77	2.03	2917	15.78	1.55	3140	6058	32	8.32	2.45	2907	14.58	1.59	3199	6106
3	8.10	1.88	3189	14.17	1.80	2815	6004	33	7.54	1.71	3158	15.18	2.41	2971	6129
4	7.67	1.91	3031	14.96	2.31	3087	6118	34	8.02	1.96	2965	15.41	1.61	3045	6010
5	7.97	1.65	3131	15.46	1.90	2919	6050	35	8.16	2.40	2864	14.44	2.49	3212	6076
6	8.46	1.88	3005	14.62	2.19	2821	5826	36	8.44	2.19	3037	14.87	1.56	3215	6252
7	7.65	1.70	2834	14.39	2.10	3123	5957	37	7.95	2.22	3092	14.31	1.84	2782	5874
8	7.94	2.38	2891	15.16	2.09	2957	5848	38	8.34	1.91	2783	14.09	1.56	3226	6009
9	7.71	1.68	3189	14.56	1.61	3232	6421	39	7.97	2.13	3053	15.37	1.83	3226	6279
10	8.00	2.47	3106	15.53	1.76	3217	6323	40	7.85	1.70	2941	15.33	2.02	3205	6146
11	7.61	1.67	3005	14.80	1.92	3111	6116	41	8.23	2.43	2837	15.37	1.74	3174	6011
12	8.42	2.09	3127	14.47	2.29	2772	5899	42	8.46	2.06	2829	15.08	2.43	2866	5695
13	8.02	2.13	2748	14.22	1.77	3168	5916	43	8.39	2.12	3206	15.12	1.92	2732	5938
14	7.60	2.29	2867	15.34	2.36	3125	5992	44	8.24	1.75	3119	14.98	1.54	3005	6124
15	8.38	2.14	2930	14.61	1.80	2889	5819	45	7.99	1.92	2915	15.86	2.28	3093	6008
16	8.48	2.08	2743	15.12	2.07	2996	5739	46	7.90	2.23	2865	15.55	1.50	3230	6095
17	8.09	2.33	2924	15.32	1.51	2748	5672	47	8.26	2.27	2804	15.31	1.90	3114	5918
18	7.65	1.71	2955	14.69	2.04	3001	5957	48	7.81	1.56	3222	14.14	2.25	3160	6382
19	7.81	2.46	2946	14.95	2.45	3134	6080	49	8.14	1.89	3055	14.63	1.58	2791	5846
20	7.60	2.16	2930	15.63	1.61	3045	5975	50	8.15	1.72	3007	14.95	1.66	2762	5770
21	7.70	2.38	3226	14.87	2.46	2836	6063	51	8.45	1.51	3123	15.18	1.63	3012	6135
22	7.50	2.12	2705	14.54	1.62	2780	5485	52	7.72	1.54	3215	15.11	2.31	2727	5943
23	8.31	1.88	2939	14.74	1.80	2973	5913	53	7.69	2.04	2717	14.19	1.59	2848	5565
24	7.71	1.83	3091	15.54	2.43	3137	6228	54	7.63	2.03	2927	14.08	1.65	2922	5849
25	8.03	1.81	3230	15.12	1.83	3162	6392	55	8.28	2.23	3120	14.70	1.91	2904	6023
26	7.80	1.58	2708	14.81	1.60	3235	5943	56	7.68	2.06	3066	14.98	2.34	2854	5920
27	7.75	1.83	2789	14.99	1.80	2869	5659	57	8.16	1.85	3020	14.37	2.02	2961	5981
28	7.79	1.83	3005	14.78	2.20	2719	5724	58	7.83	1.76	2875	15.89	2.47	3090	5965
29	8.11	1.89	2972	15.04	2.17	2934	5905	59	8.42	2.16	3127	14.94	1.98	2927	6055
30	8.18	1.55	2738	15.29	2.02	3078	5817	60	8.18	2.45	2990	15.53	2.02	2745	5735

Table 7.5: Parameters used in path flows of the Nguyen-Dupuis example.

Finally, we point out that the cpu required time for these calculations on a portable was 544 sec.

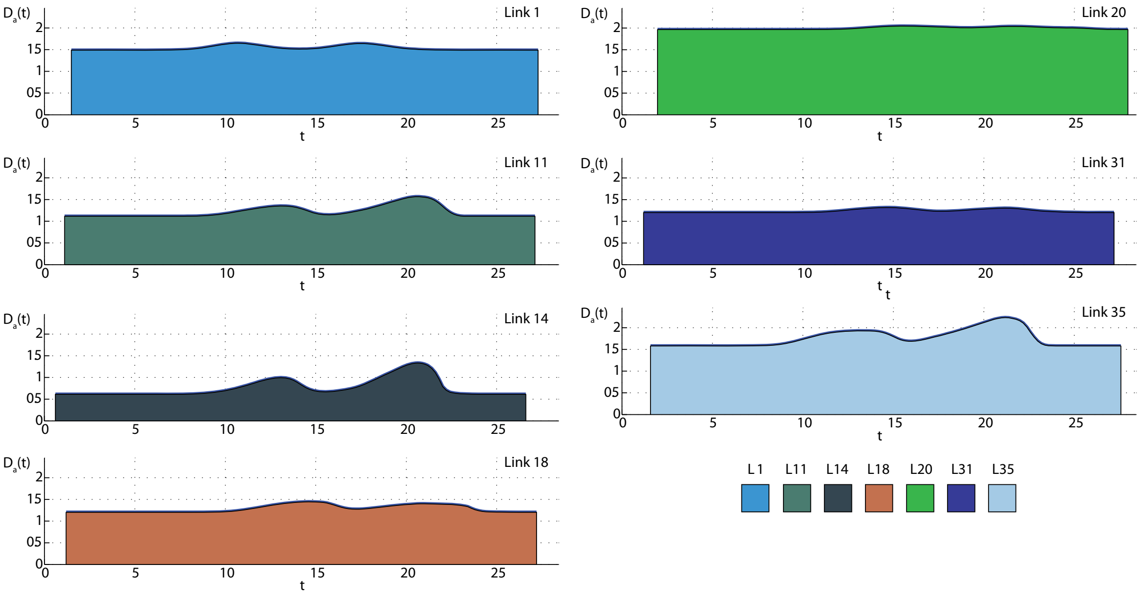

Figure 7.17: The link travel time $D_a(t)$ functions for the Nguyen-Dupuis example.

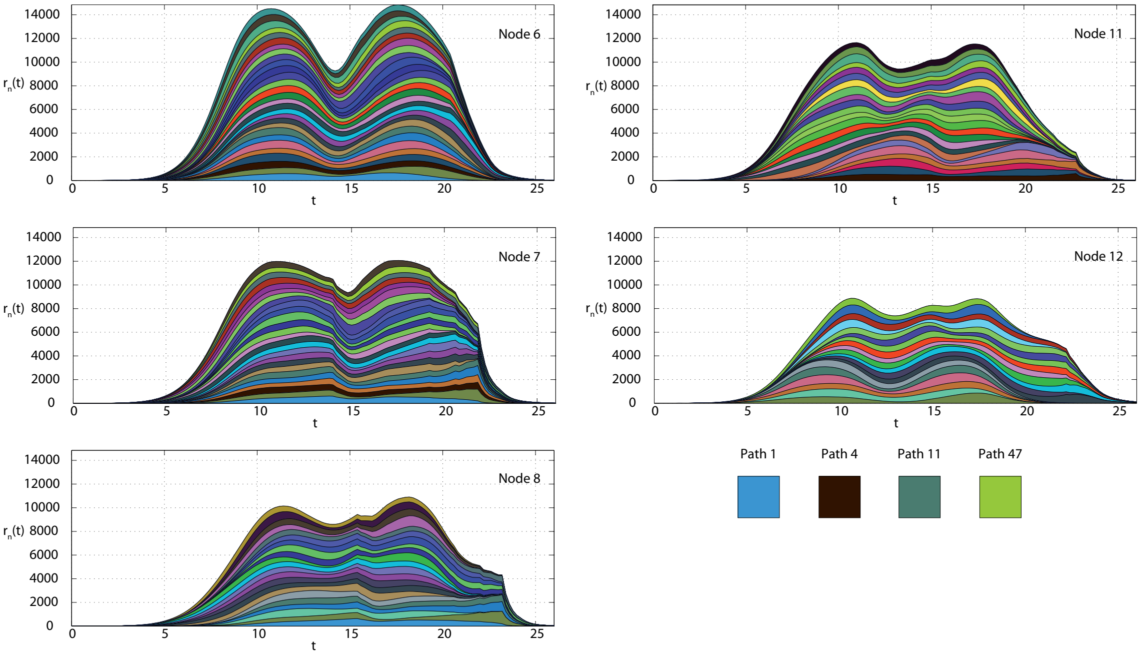

Figure 7.18: The Nguyen-Dupuis network showing the node intensity functions.

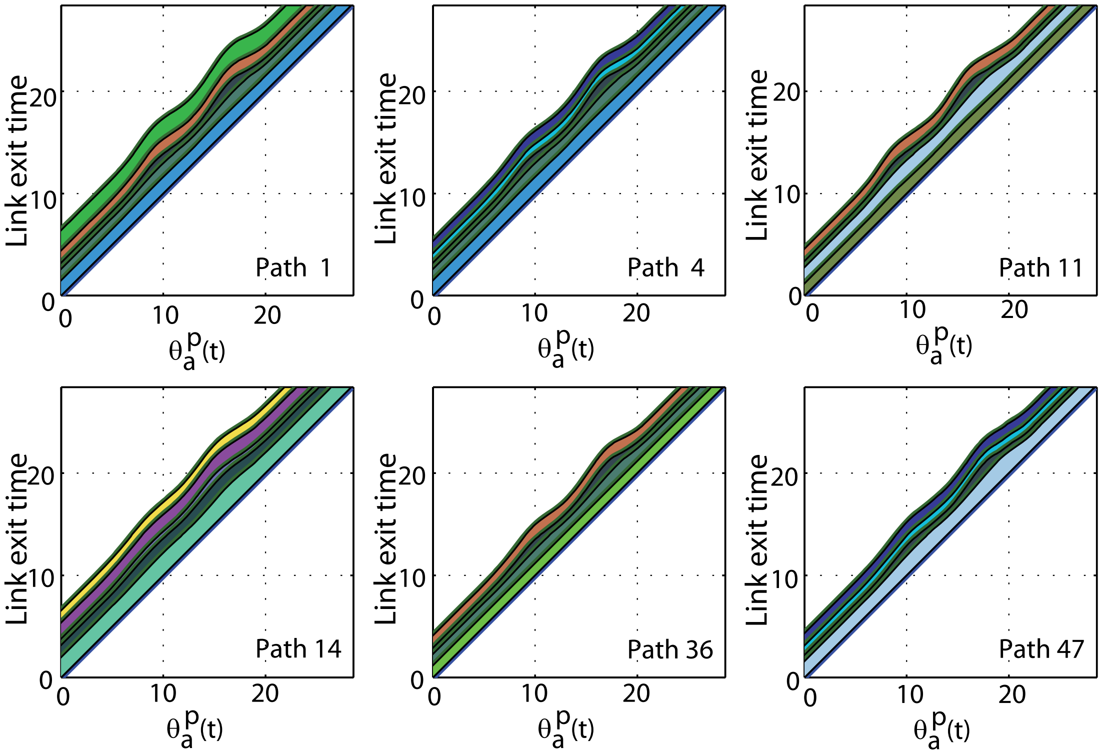

Figure 7.19: The Nguyen-Dupuis network showing the $\theta_a^p(t)$ functions.

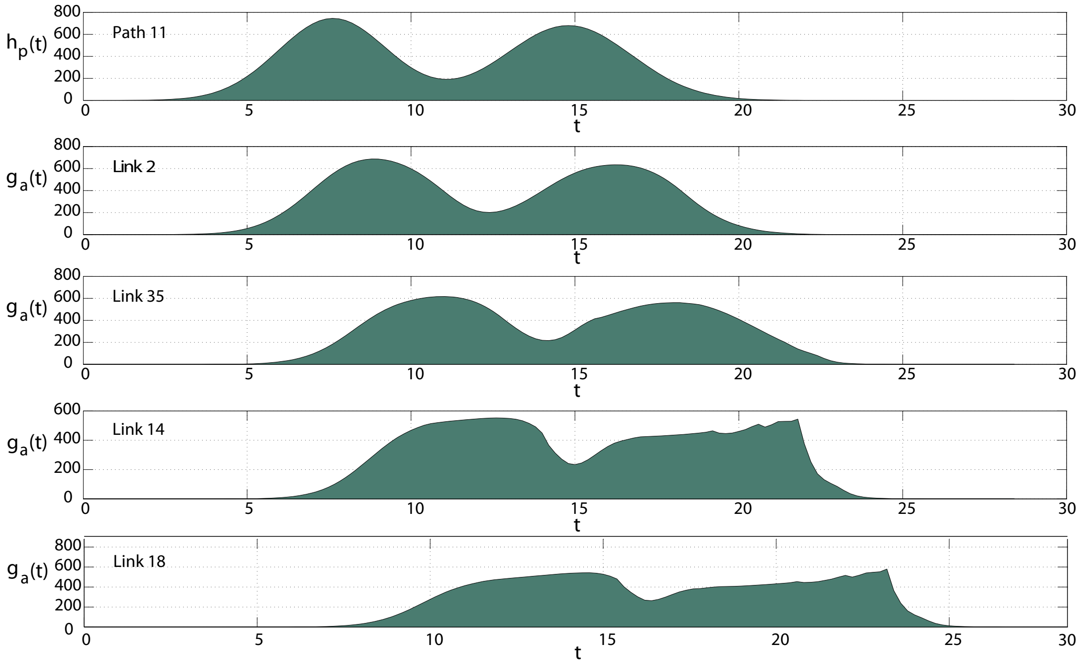

Figure 7.20: The Nguyen-Dupuis example. Evolution of path 11 when passing throughout links 2, 35, 14, and 18.

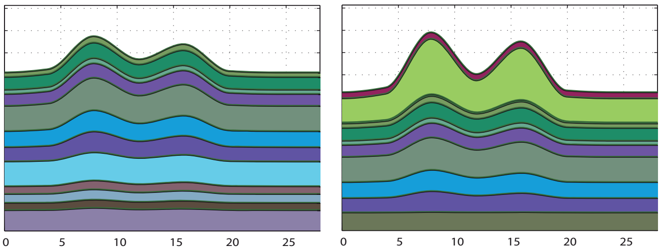

Figure 7.21: Examples of the cumulated link travel times curves showing link congestion of two paths of the Cuenca network.

7.9.2 The Cuenca network

In this section we aim at testing the proposed methods with a real network and checking that the computation times are reasonable. With this aim in mind, the methods were applied to the city of Cuenca network, with 232 nodes, 672 links and 219 paths (see Figure 6.9).

We have used the following parameter values: $\beta_a = 1$, $\delta_a = 1/3$ and $\gamma = 1$, the α_a values have been obtained using a free flow speed of 30 Km/hour, and the $x_a^{max}; \forall a$ correspond to 100 vehicles/Km.

The observed flow trends corresponding to this example were very similar to those presented before for the small networks[8]. Alternatively, in Figure 7.21 we have accumulated the link travel times for two paths. The following interesting features can be observed from this figure:

- In the case of the path shown in the left graph in Figure 7.21 we can see that the first and last few links have no or small congestion, thus showing almost parallel trends. However, some intermediate links have some congestion at peak hours that lead to bands with width changes leading to a general trend showing relative maxima and minima at peak hours, while an almost horizontal and parallel trend appears at early and late hours in the day.

- The path of the graph on the right-hand side of Figure 7.21 is composed by links with congestion because this path comes from a very generative zone and goes to a very attractive one. See that the second last link of this path has a huge increase of link travel time in both peak hours. Again the traffic is fluid at early and late hour, showing the typical parallel trend.

[8] Note that since this is a medium-size city network the path travel time is less than half hour. This is the reason why we have not represented the $\theta_a^p(t)$ curves, which present a very narrow trend and are difficult to visualize.

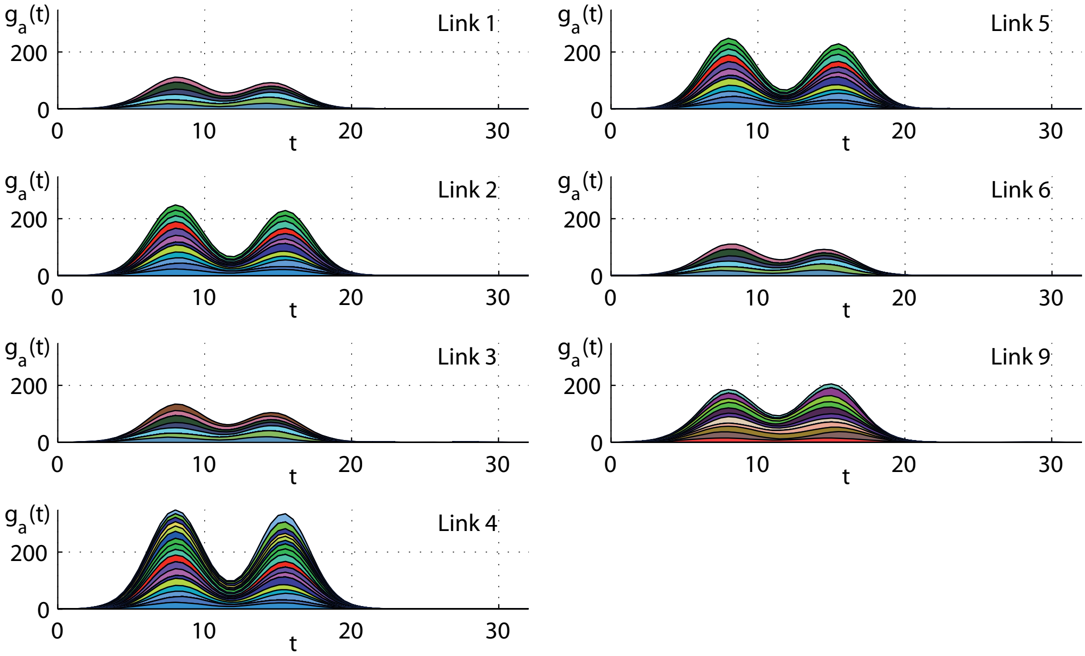

Figure 7.22: Some examples of link flow intensity curves of the Cuenca network with the corresponding path flow components.

Figure 7.22 shows some examples of link flow intensity curves of the Cuenca network with the corresponding path flow components. The difference with previous graphs here is that the congestion is almost simultaneous in all links and path flows.

Finally, we note that the number of iterations required for convergence varies from a single iteration when there is no saturation at all (reduced traffic flow), to 3 to 8 iterations when large congestion is present.

The resulting cpu time was 1215 sec. for the case of no congestion.

7.10 Comparison with other models

Though with some limitations, the exposed model has important advantages. The originality of the model lies mainly on proposing a FIFO consistent (see Section 7.7) dynamic traffic model which includes non-linear delay functions of link volumes, but also lies in the following convenient features, some of them shared with other existing methods:

1. We deal with a smooth continuous model for real traffic networks, which seems more convenient than a discretized version of the real continuous problem.

2. Though time functions involve infinite degrees of freedom, we approximate these

functions by cubic splines. This implies that the number of parameters used to define the models remains finite and small. To this end we determine the link travel times of some users at a discrete set of fixed times, and these data are used to directly approximate the link travel time functions by monotone cubic Hermite splines (continuous functions). This is an efficient way of dealing with such a complex problem and satisfying the FIFO rule.

3. Instead of inverse path link exit time functions, which are a very large number of functions (number of paths×mean number of links per path), we work with link exit-entry time functions, which are a number of functions equal to the number of links. This implies an important reduction in the number of implied functions and a reduction in memory and spline function estimations on the one hand, and avoids redundancies, on the other.

4. The inverse path-link exit time functions are evaluated in terms of the link travel time functions by a simple algorithm without the need to build the corresponding path approximations (splines or other approximations).

5. We use an iterative scheme. At a given iteration, we first determine the link travel times associated with a carefully selected set of users based on a previous iteration cubic-spline approximation of link travel time functions and later we update the cubic splines, fitting them to the updated travel times. The process is iterated until convergence. Our experience shows that this process is rapidly convergent.

6. 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.

7. The required memory and cpu requirements for the proposed model remain moderate.

7.11 Conclusions

The main conclusion that we draw from the material in his chapter are the following.

1. Although existing dynamic traffic models based on the hydraulic analogy explain important features of traffic flow, they are very high time consuming and become prohibitive to be used in intermediate or large networks. Thus, alternative models become necessary to make them useful in the engineering practice.

2. The proposed method is a continuous method in which the link exit-entry time functions $\tau_a^{-1}(t)$ are approximated by monotone cubic Hermite splines and path origin flow functions are assumed to be linear combinations of a basic set of functions.

Both approximations can be improved as much as desired by increasing the set of basic points and the set of basic functions, respectively.

3. The proposed method is a possible alternative that implies a reasonable computation time and on the other hand seems to reproduce the real traffic in a sufficiently close way. The path traffic waves propagate through the network by stretching and enlarging according to the degree of congestion of the corresponding links and joining with other path waves at the network nodes. Both the stretching-enlarging and the merging processes satisfy the FIFO rule within and among different path flows.

4. The proposed linear combination of normal densities to reproduce the time varying path flows seems to be a reasonable simplification which is close to the real values. We note that the combination of these functions leads to complex flow intensity evolution curves with a trend similar to the observed curves in reality.

5. The model assumes that the link travel time is a non-linear function of the link volume and also of the immediate downstream path link volumes, but some corrections (queue considerations) are made in order to satisfy the FIFO rule at the basic set. Further the use of monotone cubic splines preserving monotonicity, guarantees that the FIFO rule is satisfied at all points.

6. 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 and no more than 15 iterations when some congestion is present.

7. Due to the 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 and propagation laws are satisfied over all the network.

Chapter 8

Generalized Beta-Gaussian Bayesian network model

Contents

8.1 Notation

α	shape parameter of the Beta distribution.
$\beta(\alpha, \theta)$	Beta function with parameters α and θ.
$\Gamma(\alpha)$	Gamma function with parameter α.
μ	mean.
Φ	cumulative distribution function of the standard normal random variable.

σ^2	variance.
Σ_{XX}	variance-covariance matrix of X.
θ	shape parameter of the Beta distribution.
a	link.
b	lower end of the generalized Beta variable.
$\hat{b}_{at}$	estimated Beta parameter of the random variable X_{at} of link a at time t.
c	upper end of the generalized Beta variable.
$\hat{c}_{at}$	estimated Beta parameter of the random variable X_{at} of link a at time t.
$GBeta(\alpha, \theta; b, c)$	generalized Beta function with parameters α, θ, b and c.
k	observation.
S	subset of variables.
t	time.
t_0	time of prediction.
t_m	discrete time value.
t_s	predicted time.
u_{at}	random value from a standard $Beta(\alpha, \theta)$ variable.
$\hat{X}$	estimate of variable X.
X_{at}	generalized Beta random variable associated with link a and time t.
x_{kat}	observation k of the random variable x of link a at time t.
Y_{at}	standard normal random variable associated with link a and time t.
y_{kat}	standard normal variable value corresponding to observation k in link a at time t.
$\mathcal{A}$	set of links of a traffic network.
$\mathcal{B}_0$	set of link-time pairs.
$\mathcal{D}$	set of origin-destination pairs of nodes.
$\mathcal{N}$	set of nodes of a traffic network.

8.2 Introduction

In Chapter 7 we have discussed the loading problem in dynamic models, that is, given a deterministic traffic demand evolution function, we were interested in knowing how the users compete for space and the traffic wave propagates throughout the network by stretching or enlarging, depending on the congestion level. However, in reality the demand is not deterministic but random, and changes from day to day. In this chapter, we deal with the stochastic nature of demand and analyze how these random changes can be predicted based on previous information, that is, the traffic situation in different locations at times before the prediction time.

Several models have been used in the existing literature to reproduce the stochastic character of traffic flow, such as time series models (see Lee and Fambro (1999), William (1999)), neural network models (see Hall and Mars (1998), E. Castillo and Pruneda. (1998)), Bayesian networks (see Sun et al. (2006), Queen and Albers (2009)), etc. We concentrate on Bayesian network models that allow us to reproduce both the stochastic character of flows together with its dynamic nature. Nevertheless, Bayesian networks are more appropriate for traffic models because:

1. Contrary to neural network models, Bayesian network models provide not only mean values of predictions but the corresponding variabilities. In fact, they provide density functions. We note that this is much more than the confidence intervals provided by time series models.

2. Bayesian network models are specially designed to make predictions when new information is available and to perform the corresponding prediction updating. Note that this is exactly the problem we have in traffic prediction because the available information is changing with time and we need to update predictions.

3. Apart from theoretical reasons, there is clear evidence in favor of Bayesian networks. For example, Sun et al. (2006) compare Bayesian network models with other existing models such as neural network and time series models and conclude that Bayesian networks are clearly superior in performance.

Bayesian networks are very useful to represent the statistical relationships among multivariate random variables in a simple and easily implementable way and have been used for traffic prediction in the past. In particular, Sun et al. (2006), Castillo et al. (2008d,e); Sánchez-Cambronero et al. (2011) have applied Bayesian networks to steady-state or total traffic flow problems.

Sun and Zhang (2005) present a spatio-temporal Bayesian network predictor that incorporates the spatial and temporal information available by selecting variables using the Pearson correlation coefficients. The joint density is assumed to be a mixture of Gaussian distributions. Sun et al. (2006) propose a Bayesian network to model the traffic flow rates of adjacent road links in a transportation network, so that link flow time changes can be predicted in terms of other link flow rate time changes. The joint probability distribution of the variables involved is assumed to be a mixture of Gaussian distributions whose parameters are estimated using the well-known expectation maximization (EM) method. The Bayesian network structure is advantageously used to make predictions when partial information is available. Principal component analysis is also used for selecting the set of relevant variables required to predict a given link flow.

Yu and Cho (2008) use a Gaussian mixture Bayesian network to predict a given link flow based on both upstream and downstream link data and its own historical data.

Reference	**Bayesian network distribution**	**Estimation method**	**Variable selection method**	**Spatio-temporal**
Castillo et al. (2008d,e) Sanchez-Cambronero et al. (2011)	Gaussian	Maximum likelihood	Traffic network structure	NO
Sun et al. (2006)	Gaussian mixture	Expectation-Maximization	Principal components	NO
Queen and Albers (2009)	Multiregression	-	External intervention	NO
Sun and Zhang (2005)	Gaussian mixture	Maximum likelihood	Pearson correlation	YES
Yu and Cho (2008)	Gaussian mixture	Expectation-Maximization	Latest adjacent links	NO
Proposed method	Generalized beta-Gaussian	Maximum likelihood	Conditional expectations variances	YES

Table 8.1: Some existing Bayesian network models.

Queen and Albers (2009) propose a multivariate Bayesian dynamic model called the multiregression dynamic model (MDM). In particular, they deal with the prediction of traffic when sudden changes occur using the technique of external intervention that allows identifying the relevant variables based on the Bayesian network structure. Table 8.1 gives a summary of previous models.

Most Bayesian network models used in stochastic dynamic models are Gaussian or mixtures of Gaussian distributions. This means that the random variables involved have unlimited range $(-\infty, \infty)$, and that all individual variables are mixtures of normals with the same weights. Although the model is powerful, this could be an important limitation in practice. In order to provide a more realistic and a positive range for the variables and extra degrees of freedom, in this chapter we propose a Generalized Beta-Gaussian Bayesian network, the marginals of which are generalized Beta variables, that is, variables with range $[b_i, c_i]$ instead of the $[0, 1]$ range of standard Beta variables. This allows us to define the marginal distribution of each individual variable independently, according to its characteristics. In addition, the covariance structure is modeled by means of a multivariate normal distribution.

The large number of parameters of the proposed model is very easy to estimate, as shown in the following sections. There seems to be agreement in using the maximum likelihood as estimation method, though in some cases with the help of the expectation maximization (EM) algorithm. The variable selection methods are based on Pearson correlation coefficients, principal components or conditional variances. Finally, some models are spatio-temporal, that is, they include times and locations.

This chapter is organized as follows. In Section 8.4, we describe the dynamic model defined as Generalized Beta-Gaussian Bayesian network (GBGN) model for stochastic demand. In Section 8.6, we explain some implementation problems we have faced in the implementation of the Gaussian mixture Bayesian network models. In Section 8.5, we give some examples of application to illustrate the performance of the GBGN proposed method. In Section 8.6, we compare the model proposed in this chapter, the normal

Bayesian network, and the Gaussian mixture model. Finally, in Section 8.7 we give some conclusions.

8.3 The generalized Beta distribution

As indicated in section 5.3.2, the Beta random variable is useful for modeling experimental data with range limited to the interval $[0, 1]$. This fact does not restrict the use of the Beta random variable because if Y is a random variable defined on the interval $[b, c]$, then

$$X = \frac{Y - b}{c - b}$$

defines a new variable such that $0 \leq X \leq 1$. Therefore, the Beta density function can be applied to a random variable defined on the interval $[b, c]$ by translation and a change of scale, that is,

$$\begin{aligned} E[Y] &= b + (c - b)E[X] && (8.1) \\ \sigma_Y^2 &= (c - b)^2 \sigma_X^2. && (8.2) \end{aligned}$$

In this chapter we consider this generalized Beta distribution that is denoted $GBeta(\alpha, \theta; b, c)$. Since it is important to relate the statistical parameters to traffic parameters, we provide this relation below:

b is the minimum possible value of the random variable being dealt with (link travel time, link flow, etc.).

c is the maximum possible value of the random variable being dealt with.

α, θ are two parameters that permit the shape of the density of the random variable being dealt with to be defined. If $\alpha = \theta$ we obtain symmetric densities. If $\alpha > \theta$ we get densities concentrated on the largest values (negative skewness) and if $\theta > \alpha$ we get densities concentrated on the lowest values (positive skewness) (see Figure 5.1).

8.4 The Generalized Beta-Gaussian Bayesian network model

In this section, we propose what we denominate the Generalized Beta-Gaussian Bayesian network model (GBGN). Let X_{at} be the random variable associated with link a and time t. This variable can be the link travel time, the link flow, the link flow density or any other variable we are interested in. To facilitate the understanding of our description we use here as X_{at} the link travel time associated with a user entering link a at time t, where $t \in \{t_0, t_1, \ldots, t_m\}$ is a discrete time. Nevertheless, in other examples we use X_{at} as the flow rate, where a refers to a station.

8.4.1 Model assumptions

In our model we make the following assumptions:

1. We assume that $X_{at} \sim GBeta(\alpha_{at}, \theta_{at}; b_{at}, c_{at})$ (approximately), that is, X_{at} can be approximated by a generalized Beta variable, whose parameters depend on the link a and time t being considered.

   This assumption provides us with sufficient degrees of freedom (four location, scale and shape parameters) to reproduce the marginal X_{at} random variables.

2. Since in the next assumption we are going to assume Gaussian Bayesian networks, we need to work with normal marginals Y_{at}. To this end, we transform the X_{at} generalized Beta random variables to standard normals $N(0,1)$ by means of the transformation:[1]

   $$Y_{at} = \Phi^{-1}(F_{GBeta(\alpha_{at},\theta_{at};b_{at},c_{at})}(X_{at})), \tag{8.3}$$

   where $\Phi(x)$ is the cdf of the standard normal random variable and $F_{GBeta(\alpha,\theta;b,c)}(x)$ is the cdf of the generalized Beta distribution.

3. We assume that the random variables $Y_{at}; a \in \mathcal{A}; t \in \{t_0, t_1, \dots, t_m\}$ can be reproduced by a normal Bayesian network, where the parents of Y_{at} are in $\cup_{k\in\{0,1,2,\dots,s\}}\{Y_{a_0t-k}|a_0 \in \mathcal{A}\}$, that is, future flow properties depend only on present and past properties.

   This assumption permits us to reproduce the variance-covariance structure of all the variables.

Our GBGN proposed model depends on a large number of parameters that include the four generalized Beta parameters per variable, already discussed, plus the parameters involved in our Bayesian normal model, i.e., $(n^2 + 11n)/2$ parameters, where n is the number of variables. The covariance parameters include the information on the correlation of the different pairs of traffic variables, which is very important to decide which of the traffic variables have information on other traffic variables and how this changes when new information becomes available.

8.4.2 Learning the Generalized Beta-Gaussian Bayesian network

In order to learn the GBGN parameters we need some data. To this end, we have two alternatives: (a) measure real data in the network where the model is to be applied, or (b) use simulations to obtain the data.

[1] If we transform any random variable using as transformation function its cdf, we obtain a standard uniform $U(0,1)$ variable. Performing this transformation to the $GBeta(\alpha, \theta; b, c)$ random variable X_{at} and the standard normal $N(0,1)$ random variable Y_{at} we get $\Phi(Y_{at}) = F_{GBeta(\alpha_{at},\theta_{at};b_{at},c_{at})}(X_{at})$, from which (8.11) can be immediately obtained.

In either case, we assume that we know the set

$$\{x_{kat}; a = 1, 2, \ldots, n; t = 1, 2, \ldots, m; k = 1, 2, \ldots, s\}, \tag{8.4}$$

where a, t and k refers to link, time and replication, the lower case letters x_{kat} refer to the sample values of X_{at}^k, and n, m and s are the number of links, the number of time instants considered and the sample size, respectively.

The learning process has two main steps: (a) learning the generalized Beta parameters and (b) learning the mean and covariance matrix of the standard normal Bayesian network.

Estimating the generalized Beta parameters

In order to estimate the generalized Beta parameters we have selected two alternatives:

1. *The maximum likelihood method.* We estimate the parameters as follows:

   (a) The parameter b_{at} is estimated using the expression

   $$\hat{b}_{at} = 1.05 \min_k(x_{kat}) - 0.05 \max_k(x_{kat}). \tag{8.5}$$

   Although the maximum likelihood estimate is $\min_k(x_{kat})$, we use the above expression to reduce the probability of future data to be out of range.

   (b) Similarly, though the maximum likelihood estimate is $\max_k(x_{kat})$, the parameter c_{at} is estimated using the expression

   $$\hat{c}_{at} = 1.05 \max_k(x_{kat}) - 0.05 \min_k(x_{kat}). \tag{8.6}$$

   (c) The α_{at}, θ_{at} parameters are estimated by maximum likelihood of the standard Beta random variable using the sample

   $$u_{at} = (x_{kat} - \hat{b}_{at})/(\hat{c}_{at} - \hat{b}_{at}); \quad k = 1, 2, \ldots, N, \tag{8.7}$$

   where the u_{at} are random values from a standard $Beta(\alpha, \theta)$ variable.

   Thus, we use transformation :

   $$Y_{at} = \Phi^{-1}(F_{GBeta(\alpha_{at},\theta_{at};b_{at},c_{at})}(X_{at})), \tag{8.8}$$

   to convert the generalized Beta random values in the set

   $$\{x_{at}; a \in \mathcal{A}; t \in \{t_0, t_1, \ldots, t_k\}\}$$

   into a set of normal values $\{y_{at}; a \in \mathcal{A}; t \in \{t_0, t_1, \ldots, t_k\}\}$.

2. *The least squares method.* Since we aim at transforming the generalized Beta variables to standard Gaussians, we incorporate two more parameters, one location μ_{at} and one scale σ_{at}, and select them in such a way that the transformed variables are as close as possible to the standard normal variables, that is, to estimate $\alpha_{at}, \beta_{at}, \mu_{at}$ and σ_{at} we minimize the sum of squares:

$$\sum_{i=1}^{n} \left[r_{in} - \frac{\Phi^{-1}\left(F_{B(\alpha_{at},\beta_{at})}(\frac{x_{at}-b_{at}}{c_{at}})\right) - \mu_{at}}{\sigma_{at}} \right]^2 \tag{8.9}$$

with respect to $\alpha_{at}, \beta_{at}, \mu_{at}$ and σ_{at}, where $F_{B(\alpha_{at},\beta_{at})}(x)$ is the cdf of the Beta distribution and

$$r_{in} = \Phi^{-1}\left(\frac{i}{n+1}\right). \tag{8.10}$$

Thus, in this case our transformation becomes:

$$Y_{at} = \frac{\Phi^{-1}(F_{GBeta(\alpha_{at},\theta_{at};b_{at},c_{at})}(X_{at})) - \mu_{at}}{\sigma_{at}}. \tag{8.11}$$

Estimating the mean and covariance matrix

We use the maximum likelihood method. The mean matrix of the normal variables, that should be close to the null matrix, is learnt by calculating the mean of the transformed sample values Y_{at}:

$$E[\mathbf{Y}_{at}] = \frac{\sum\limits_{k=1}^{N} \mathbf{y}_{kat}}{N} \tag{8.12}$$

and the covariance matrix of the standard Normal Bayesian network is estimated by the covariance matrix of the same transformed sample, that is,

$$Cov(\mathbf{Y}_{at}, \mathbf{Y}_{at}) = \frac{\sum\limits_{k=1}^{N} (\mathbf{y}_{kat} - E[\mathbf{Y}_{at}])(\mathbf{y}_{kat} - E[\mathbf{Y}_{at}])^T}{N}, \tag{8.13}$$

where N is the sample size.

It is worth mentioning that the learning process must be dealt with only once and can be done in advance and off line. The model parameters can be stored in a database and used later for successive predictions. This is the data basis on which the flow evolution predictions will be based. This process can also be updated from time to time in order to follow the long time variation of traffic, which is not stationary. In addition, we want to emphasize that the computational power required for predicting all links in a network is linear (proportional) to the number of links because the number of variables involved in a link prediction for close times depends only on the adjacent links. In other words, the computational power to predict a link flow rate is not dependent on the network size. Finally, the database required for local predictions includes only local information (adjacent links). These are important facts for practical applications.

8.4.3 Making predictions

To make a prediction of X_{at_s} at a given time $t = t_0$ for a future time t_s, where the values of t_0 and t_s are in $\{t_0, t_1, \ldots, t_k\}$, we proceed as follows:

1. We select the subset of variables $S = \{Y_{a_0 t_k}; (a_0, t_k) \in \mathcal{B}_0\}$ where the set $\mathcal{B}_0$ is the set of link-time pairs (a_0, t_k) such that the correlation coefficient of Y_{at_s} and $Y_{a_0 t_k}$ is larger than a given threshold value *threshold*, say 0.05, and consider it as the observed set.

2. We calculate the conditional expectations $\mu_{at_s} = E[Y_{at_s} | Y_{a_0 t_k}; (a_0, t_k) \in \mathcal{B}_0]$ and the conditional variances $Var(Y_{at_s} | Y_{a_0 t_k}; (a_0, t_k) \in \mathcal{B}_0)$ using (5.15) and (5.16).

3. We calculate the confidence intervals for the prediction as

$$\hat{Y}_{at_s} \equiv \left(E[Y_{at_s} | Y_{a_0 t_k}; (a_0, t_k) \in \mathcal{B}_0] \pm 2\sqrt{Var(Y_{at_s} | Y_{a_0 t_k}; (a_0, t_k) \in \mathcal{B}_0)} \right). \qquad (8.14)$$

4. We transform the random variable Y_{at_s} and the corresponding confidence intervals to the initial variable X_{at_s} by means of the transformation (inverse of transformation (8.11)):

$$\hat{X}_{at_s} = F^{-1}_{GBeta(\alpha_{at_s}, \theta_{at_s}; b_{at_s}, c_{at_s})}(\Phi(\hat{Y}_{at_s}))). \qquad (8.15)$$

5. We obtain the density $f_{X_{at_s}}(x)$ of $X_{at_s} | X_{a_0 t_k}; (a_0, t_k) \in \mathcal{B}_0$ by means of the following expression[2]

$$f_{N(\mu_{at_s}, \sigma_{at_s})}(\Phi^{-1}(F_{GBeta(\alpha_{at_s}, \theta_{at_s}; b_{at_s}, c_{at_s})}(x))) \times \frac{f_{GBeta(\alpha_{at_s}, \theta_{at_s}; b_{at_s}, c_{at_s})}(x)}{f_{N(0,1)}(F_{GBeta(\alpha_{at_s}, \theta_{at_s}; b_{at_s}, c_{at_s})}(x))}, \qquad (8.16)$$

where $f_{GBeta(\alpha_{at_s}, \theta_{at_s}; b_{at_s}, c_{at_s})}(x)$, $f_{N(\mu_{at_s}, \sigma_{at_s})}(x)$ are the pdfs of the $GBeta(\alpha_{at_s}, \theta_{at_s}; b_{at_s}, c_{at_s})$ and normal $N(\mu_{at_s}, \sigma_{at_s})$ distributions, respectively.

We point out that by evaluating the correlation of a particular traffic variable with the rest of the traffic variables and following the above procedure, the relevant traffic variables (to predict the target variables) can be selected without the need to have any knowledge of the network structure or paths.

8.5 Examples of application

Once we have presented our GBGN model, one question arises: is the GBGN a practical tool for real applications based on large networks? In this section, we show that the proposed model is applicable to very large networks because:

[2] This expression has been obtained using the change of variable method based on (8.15).

1. For short or medium ahead prediction periods, the informative variables are local, that is, link flows depend only on adjacent link flows, that is, the size of the network is irrelevant to the complexity of the required calculations.

2. For long-term predictions, the set of influential links increases in size, but this increase has nothing or practically nothing to do with the network size, but with the difference between the prediction time and the time to be predicted.

3. The proposed methods use matrix products and inverses, that is operations that have been optimized and do not involve very large sizes.

In this section we present three illustrative examples, two fictitious and one real. There is an important difference between the first two examples and the third one. In the first two, we have no data and we need to simulate them first. On the contrary, in the last example, we have real data available.

8.5.1 Nguyen Dupuis network example

To facilitate the understanding of all the concepts introduced in the previous sections, 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 the subset $\mathcal{D}$, of origin-destination pairs of nodes, and we use the Nguyen-Dupuis type network, with 13 nodes and 38 links in Figure 6.2, to illustrate the proposed methods. The paths considered are those in Table 7.4.

Since no data is available from direct measurements, we have simulated 100 days of random incoming traffic in this network and we have utilized one of the physical-queue representation deterministic dynamic models based on the travel time function approach (see Szeto and Lo (2005) and Castillo et al. (2011)) to obtain the link travel time evolution curves which are the required input to our problem. Sixty paths have been considered.

Figure 8.1 shows the plots of 100 realizations of the link travel time evolutions corresponding to 100 days for some selected links of the illustrative example, together with their mean values (dashed green line). The red line (continuous thick line) corresponds to the link travel times of one day to be predicted in the present example and the circle on the abscissas axis indicates the time when predictions are made. The plots corresponding to other links are similar.

The flow evolutions at the path origins were considered as the sum of two normal densities scaled to random total volumes, and random mean time locations and standard deviations, that is, two daily peak values were assumed.

We can see that all daily link travel times show one morning and one evening maxima, but the trends are changing with the day. In addition, we can see how congestion varies on a daily basis. This information is extremely valuable to predict the future behavior of the network.

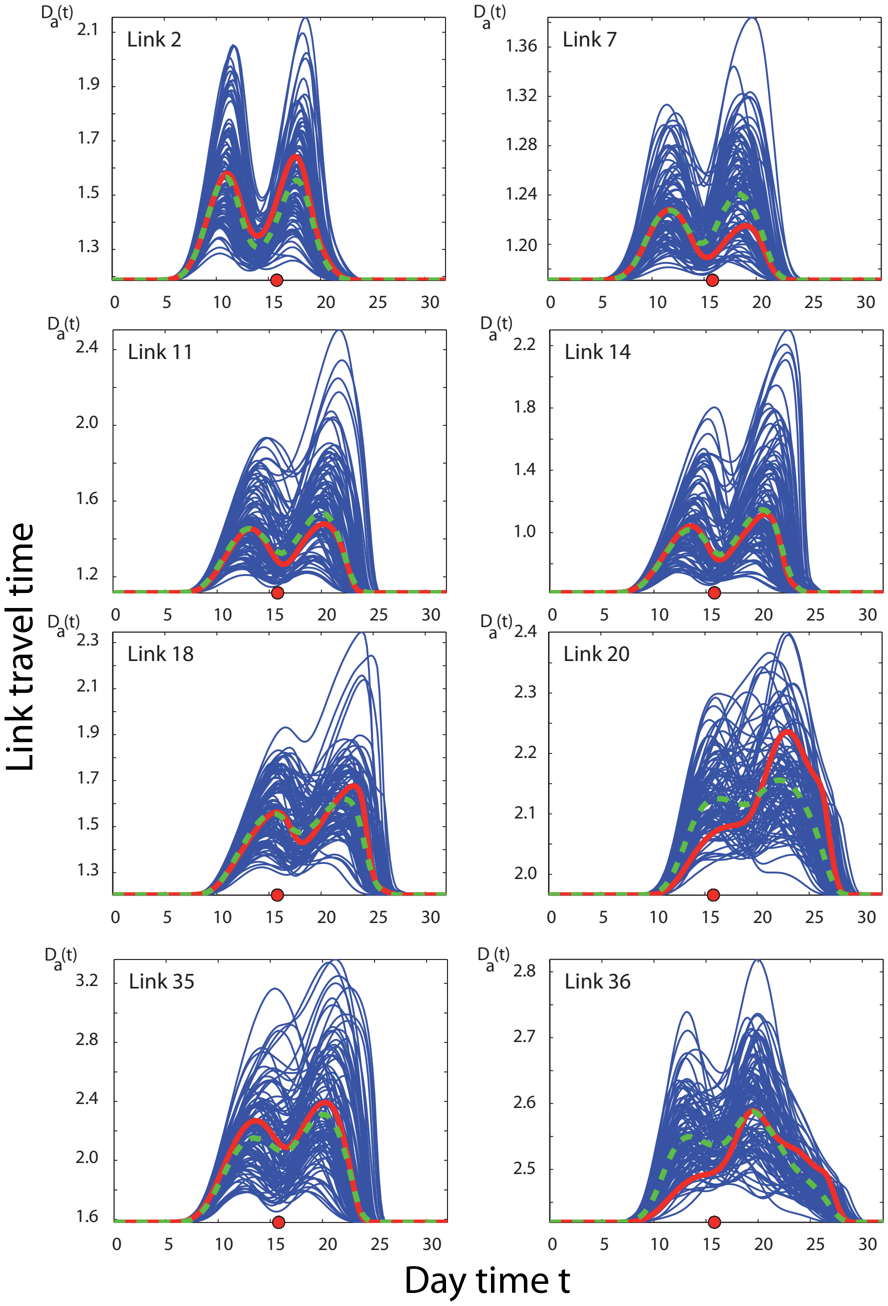

Figure 8.1: Nguyen-Dupuis: Observed daily link travel time functions (in hours) for some representative selected links. The dashed lines show the mean travel times and the thick continuous line is one particular realization (one particular day).

In this example, we want to predict at time 16:00, the future link travel times when we know the actual and the previous link travel times for all relevant links.

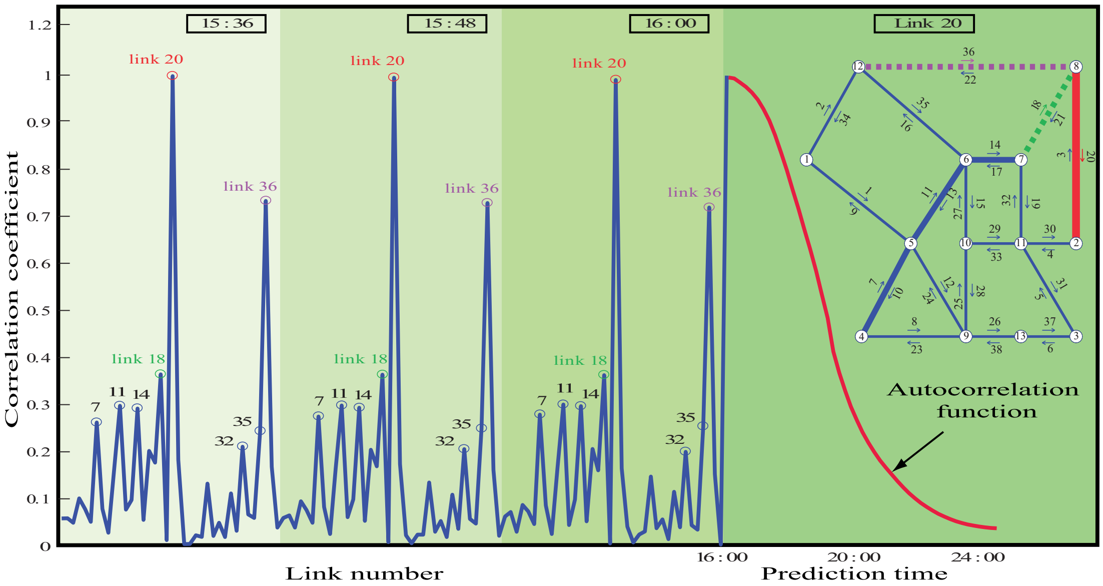

Figure 8.2: Nguyen-Dupuis: Correlation coefficients of travel time for link 20 at time 16:00 with the travel times of all links at times 16:00, 15:48 and 15:36.

Figure 8.2 shows the correlation coefficients of travel time for link 20 at time 16:00 with the travel times of all links at times 16:00, 15:48 and 15:36. We can see that the maximum correlations correspond to the same link 20 for previous times 15:48 and 15:36, and links 36 and 18 (the links connected to link 20) for the three times. We note that these correlations permit us to select the more informative links on which we can base our predictions, as indicated in Section 8.4.3. The right hand side of Figure 8.2 shows the correlation coefficients of the link 20 travel times at time 16:00 with the link 20 travel times for future times. As expected, the correlation coefficient decreases when we move further apart from the 16:00 prediction time.

Next, we have estimated the generalized Beta parameters for all links every 12 minutes using the methods described in Section 8.4 and formulas (8.5) to (8.9). Figure 8.3 shows the estimated (continuous line) generalized Beta cumulative distribution functions and the corresponding empirical (data points) distribution functions for some selected links at time $t =$ 16:00. Similar results are obtained for other links and times. It can be seen that the theoretical model reproduces the observed data fairly well, showing that the generalized Beta provides sufficient degrees of freedom to that purpose.

One important step of the suggested procedure is the variable selection method in which we try first to identify and later to avoid redundant information. Table 8.2 illustrates how this identification and elimination of redundant variables is made using the method proposed in Section 8.4.3. To this end, we show the order in which variables are selected

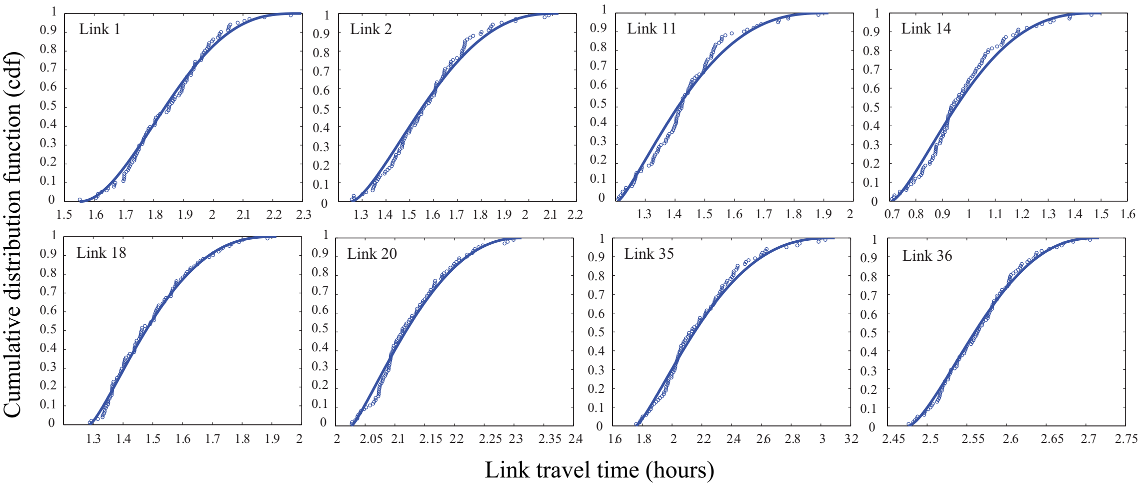

Figure 8.3: Nguyen-Dupuis: Estimated generalized Beta cdf and empirical cdf of link travel times for some selected links at time $t = 16{:}00$.

for different prediction times and time of prediction 16:00 for the cases of complete data (upper table) and missing data $\{18, 20, 36\}$ (lower table). If we want to predict the travel time of link 20 at time 16:12, the more informative data corresponds to travel time of the same link at the previous times 16:00, 15:48 and 15:36. However, once the information at time 16:00 is known, the additional information reported by the other two times is negligible, as shown in Table 8.2, where we can see that the initial variance of the travel time to be predicted (link 20 at time 16:12) of 1.007 reduces to 0.002. Predictions of the travel time of link 20 at time 17:00 have the same behavior. However, for the prediction of the travel time of link 20 at time 18:00, the variance reduces to 0.329, and then knowledge of the travel time of link 20 at time 15:48 implies an extra reduction to 0.153. It is then when links 18, 5 and 16 allow extra reductions to 0.102, 0.092 and 0.087. Finally, further predictions, say of the travel time of link 20 at time 21:00, involve information of links 36, 7 and 35, but with much smaller reductions in variance (from 1.005 to 0.877, 0.716 and 0.630, respectively).

The effect of missing data is illustrated in the lower part of Table 8.2 where the corresponding information is given as in the upper part. Note that if the information of the most relevant links $(20, 18$ and $36)$ is not available, other links (11 and 21) take its place, but then the variance reduction is much smaller (from 1.007 to 0.915 and 0.863) and more links must be used to reduce the variance further, though the lowest value can be limited not necessarily to zero. Consequently, the effect of missing information can be important if, as in this case, relevant links are included in the missing information set, but this is small if only partial information is missed. Note that the variables contain a lot of redundant information, which allows easily replacing the missing data links with others.

Complete data																		
	16:12			17:00			18:00			19:00			20:00			21:00		
step	link	time	var	link	time	var	link	time	var	link	time	var	link	time	var	link	time	var
1	**20**	**16:00**	**1.007**	**20**	**16:00**	**1.006**	**20**	**16:00**	**1.007**	36	16:00	1.003	18	16:00	1.003	**36**	**16:00**	**1.005**
2	**20**	**15:48**	**0.002**	**20**	**15:48**	**0.087**	**20**	**15:48**	**0.329**	18	16:00	0.617	36	16:00	0.784	**36**	**15:48**	**0.877**
3	**20**	**15:36**	**0.000**	**20**	**15:36**	**0.019**	**20**	**15:36**	**0.153**	36	15:48	0.436	36	15:48	0.640	**7**	**16:00**	**0.716**
4	35	15:36	0.000	21	16:00	0.011	18	16:00	**0.102**	18	15:36	0.344	18	15:48	0.521	**35**	**15:48**	**0.630**
5	7	16:00	0.000	36	15:48	0.010	5	15:36	**0.092**	20	16:00	0.320	16	15:36	0.471	7	15:48	0.590
6	36	15:36	0.000	18	15:48	0.010	16	16:00	**0.087**	20	15:48	0.300	7	16:00	0.452	7	15:36	0.556
7	36	16:00	0.000	32	15:36	0.009	13	15:36	0.085	20	15:36	0.248	7	15:48	0.437	16	15:36	0.524
8	36	15:48	0.000	7	16:00	0.009	36	16:00	0.083	36	15:36	0.218	7	15:36	0.417	20	15:36	0.506
9	4	16:00	0.000	32	16:00	0.009	36	15:48	0.081	16	15:36	0.206	4	15:48	0.388	18	16:00	0.487
10	21	15:36	0.000	32	15:48	0.009	28	16:00	0.073	32	15:36	0.197	19	15:36	0.378	20	16:00	0.464

Missing data $\{18, 20, 36\}$																		
	16:12			17:00			18:00			19:00			20:00			21:00		
step	link	time	var	link	time	var	link	time	var	link	time	var	link	time	var	link	time	var
1	**11**	**15:36**	**1.007**	11	15:36	1.006	14	15:36	1.007	14	15:36	1.003	14	15:36	1.003	11	16:00	1.005
2	**21**	**16:00**	**0.915**	21	16:00	0.899	16	15:36	0.856	2	15:36	0.815	2	15:36	0.843	2	15:36	0.887
3	**21**	**15:48**	**0.863**	21	15:48	0.865	5	16:00	0.837	2	15:48	0.791	7	16:00	0.798	7	16:00	0.838
4	4	16:00	0.838	16	15:36	0.839	28	15:36	0.818	35	16:00	0.770	7	15:48	0.771	7	15:48	0.806
5	16	15:36	0.819	7	16:00	0.824	14	15:48	0.794	35	15:48	0.735	33	15:36	0.743	7	15:36	0.763
6	7	16:00	0.803	14	15:36	0.815	14	16:00	0.772	14	15:48	0.681	11	16:00	0.725	2	15:48	0.742
7	14	15:36	0.793	14	15:48	0.796	35	16:00	0.750	14	16:00	0.671	35	16:00	0.711	2	16:00	0.706
8	14	15:48	0.771	14	16:00	0.770	35	15:48	0.742	35	15:36	0.642	2	15:48	0.693	14	15:36	0.689
9	14	16:00	0.755	35	16:00	0.747	35	15:36	0.733	28	16:00	0.632	7	15:36	0.665	35	16:00	0.676
10	35	16:00	0.726	32	15:36	0.732	2	15:36	0.728	19	16:00	0.625	2	16:00	0.632	16	16:00	0.639

Table 8.2: Nguyen-Dupuis: Order in which variables are selected for a prediction made at 16:00 of the travel time in link 20 at different times for the cases of complete data (upper table) and missing data $\{18, 20, 36\}$ (lower table).

Next, at time 16:00 we predict the future link travel times. To this end, we use the proposed Bayesian network and use only the link travel time information available at 16:00. Figure 8.4 provides in blue (darkest continuous line) the predictions of the link travel times for the different links made at time 16:00 and the corresponding 0.95 confidence intervals (as dashed lines) for the selected links. For reference, the actual flows are indicated as a red (lightest continuous) line. Note that the confidence intervals increase when we move apart from the prediction time, but decreases further when we approach the final day hours because the link travel time variances decrease for this time of the day.

Figure 8.5 shows the densities of link travel time predictions (in hours) at different future times made at time $t = 16{:}00$ for some representative selected links of the Nguyen Dupuis network. Note that the proposed model supplies not only confidence intervals but full densities, which have been calculated using (8.16).

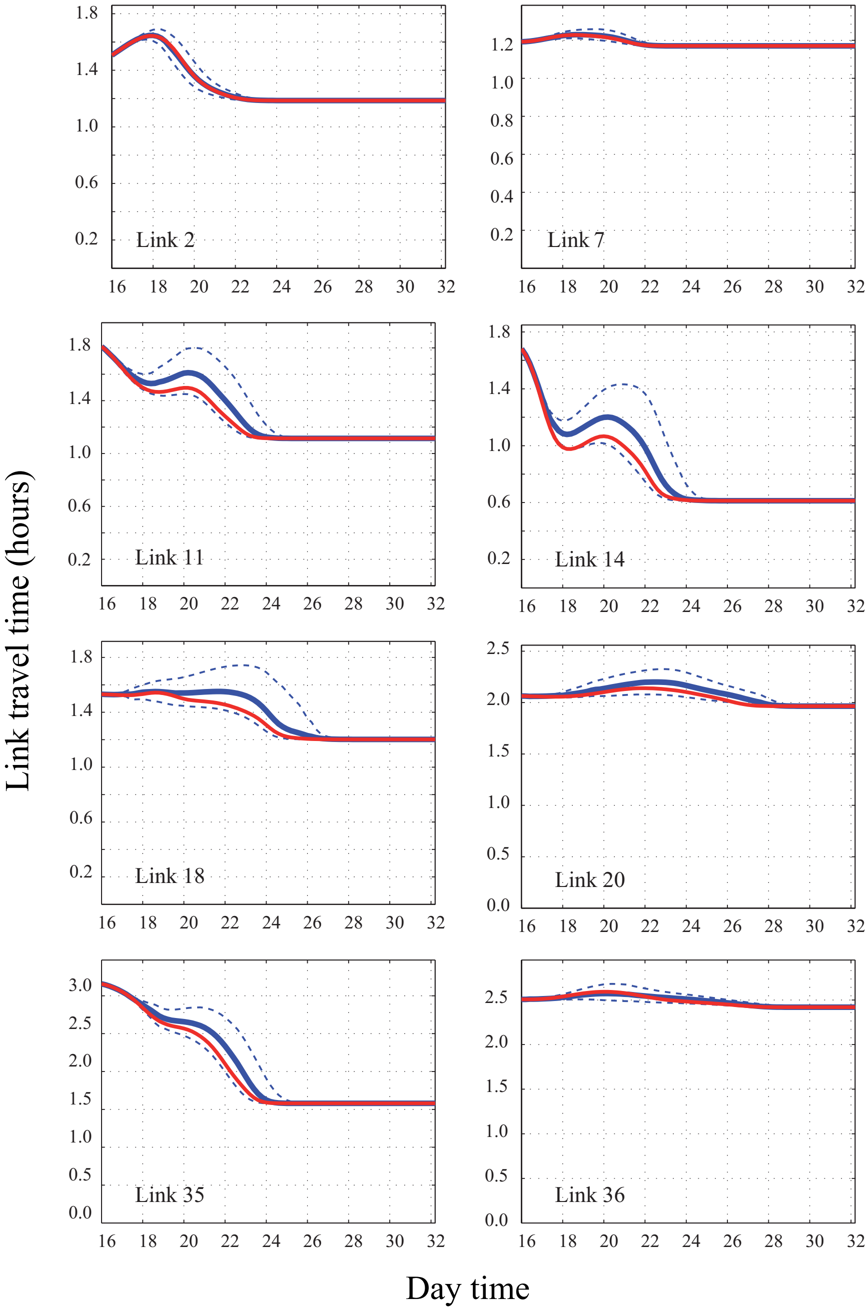

Figure 8.4: Nguyen-Dupuis: Predicted link travel time made at time t = 16:00 and 0.95 confidence intervals for some representative selected links.

The cpu time used for a single prediction of all link travel times was 0.34 seconds using a non-optimized computer program written in Matlab and run on a Dell Optiplex 755 computer with 4 Gb of memory and a processor Intel Core 2 Quad Q6700 (2.66 GHz,

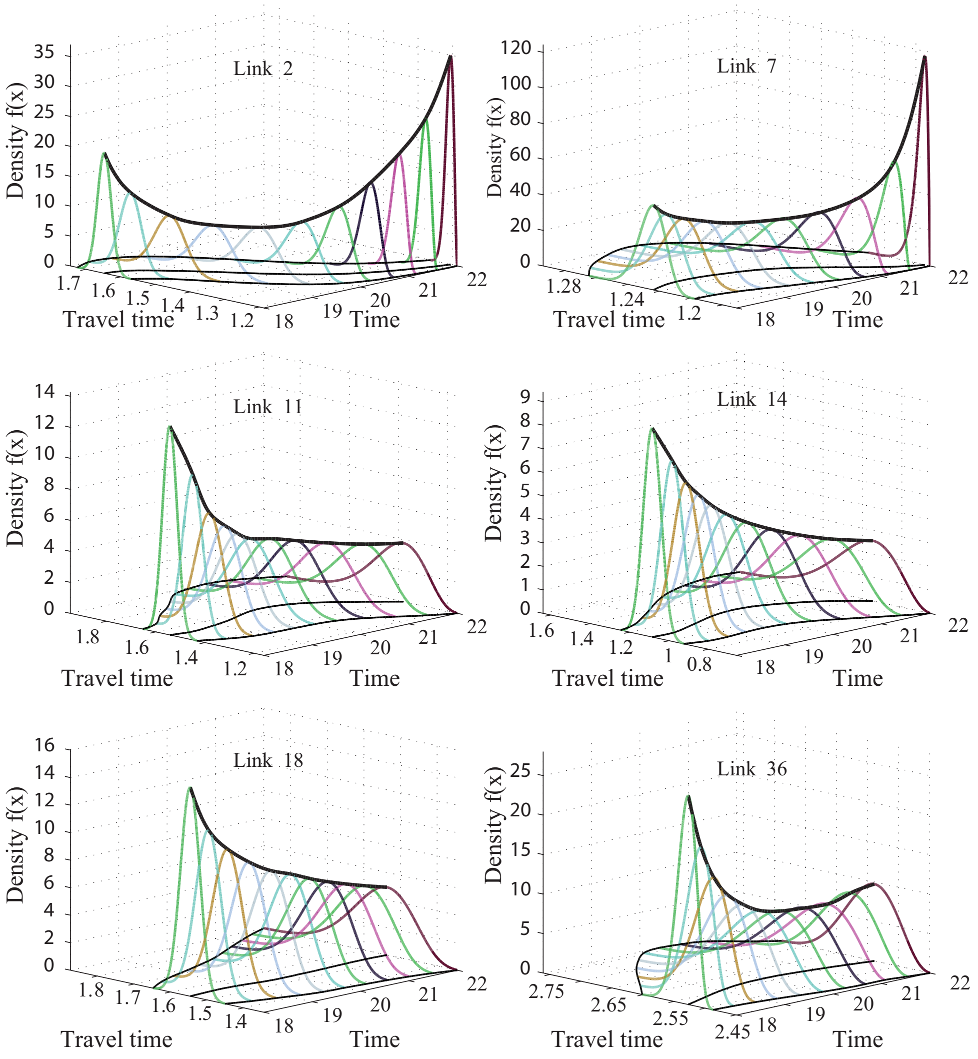

Figure 8.5: Nguyen-Dupuis: Densities of link travel time predictions (in hours) at time $t = $ 16:00 for some representative selected links at different times.

8 MB de cache L2, 1066 MHz FSB). We note that most of the cpu time required by Equations (5.15) and (5.16) is the inversion of matrix Σ_{ZZ}, which if we use the same Z, it is common for all link predictions, so that the difference in predicting a few and all link travel times is small.

Table 8.3 shows the RMSE of the link travel time predictions made at time 16:00 for all links and several times. We can see that the RMSE increases with time but decreases later due to the small variances of link travel times at the end of the day (night time).

In order to see the effect of missing information we present in Table 8.3, the cases of

Prediction times														
	Complete data							Missing data $\{18, 20, 36\}$						
link	16	18	20	22	24	26	28	16	18	20	22	24	26	28
1	0.0000	0.0033	0.0076	0.0023	0.0001	0.0000	0.0000	0.0000	0.0033	0.0076	0.0023	0.0001	0.0000	0.0000
2	0.0000	0.0027	0.0068	0.0016	0.0001	0.0000	0.0000	0.0000	0.0028	0.0070	0.0016	0.0001	0.0000	0.0000
3	0.0000	0.0061	0.0198	0.0213	0.0040	0.0000	0.0000	0.0000	0.0061	0.0198	0.0213	0.0040	0.0000	0.0000
4	0.0000	0.0009	0.0018	0.0004	0.0001	0.0000	0.0000	0.0000	0.0010	0.0020	0.0005	0.0001	0.0000	0.0000
5	0.0000	0.0050	0.0163	0.0033	0.0001	0.0000	0.0000	0.0000	0.0050	0.0163	0.0033	0.0001	0.0000	0.0000
6	0.0000	0.0001	0.0001	0.0000	0.0000	0.0000	0.0000	0.0000	0.0001	0.0001	0.0000	0.0000	0.0000	0.0000
7	0.0000	0.0014	0.0023	0.0010	0.0000	0.0000	0.0000	0.0000	0.0015	0.0025	0.0011	0.0000	0.0000	0.0000
8	0.0000	0.0023	0.0023	0.0004	0.0000	0.0000	0.0000	0.0000	0.0023	0.0023	0.0004	0.0000	0.0000	0.0000
9	0.0000	0.0008	0.0024	0.0046	0.0052	0.0023	0.0001	0.0000	0.0008	0.0024	0.0046	0.0052	0.0023	0.0001
10	0.0000	0.0004	0.0005	0.0004	0.0006	0.0004	0.0000	0.0000	0.0004	0.0005	0.0005	0.0006	0.0004	0.0000
11	0.0000	0.0036	0.0109	0.0173	0.0116	0.0000	0.0000	0.0000	0.0038	0.0115	0.0183	0.0121	0.0000	0.0000
12	0.0000	0.0015	0.0022	0.0005	0.0000	0.0000	0.0000	0.0000	0.0015	0.0023	0.0005	0.0000	0.0000	0.0000
13	0.0000	0.0013	0.0030	0.0044	0.0037	0.0004	0.0000	0.0000	0.0013	0.0031	0.0045	0.0038	0.0004	0.0000
14	0.0000	0.0055	0.0131	0.0257	0.0198	0.0002	0.0000	0.0000	0.0057	0.0134	0.0262	0.0201	0.0003	0.0000
15	0.0000	0.0020	0.0045	0.0051	0.0007	0.0000	0.0000	0.0000	0.0020	0.0045	0.0051	0.0007	0.0000	0.0000
16	0.0000	0.0020	0.0058	0.0092	0.0103	0.0040	0.0001	0.0000	0.0021	0.0060	0.0095	0.0108	0.0051	0.0002
17	0.0000	0.0019	0.0059	0.0110	0.0026	0.0001	0.0000	0.0000	0.0023	0.0066	0.0126	0.0031	0.0001	0.0000
18	0.0000	0.0033	0.0078	0.0137	0.0209	0.0100	0.0003	0.0000	0.0082	0.0097	0.0150	0.0223	0.0111	0.0003
19	0.0000	0.0023	0.0054	0.0051	0.0035	0.0005	0.0000	0.0000	0.0024	0.0058	0.0054	0.0039	0.0005	0.0000
20	0.0000	0.0018	0.0056	0.0087	0.0092	0.0062	0.0032	0.0000	0.0083	0.0096	0.0111	0.0106	0.0067	0.0034
21	0.0000	0.0052	0.0160	0.0208	0.0063	0.0001	0.0000	0.0000	0.0053	0.0163	0.0213	0.0062	0.0001	0.0000
22	0.0000	0.0022	0.0053	0.0062	0.0041	0.0018	0.0005	0.0000	0.0022	0.0053	0.0062	0.0041	0.0018	0.0005
23	0.0000	0.0009	0.0022	0.0027	0.0018	0.0005	0.0001	0.0000	0.0009	0.0022	0.0027	0.0018	0.0006	0.0001
24	0.0000	0.0006	0.0013	0.0017	0.0015	0.0004	0.0000	0.0000	0.0006	0.0013	0.0017	0.0015	0.0004	0.0000
25	0.0000	0.0027	0.0066	0.0059	0.0006	0.0000	0.0000	0.0000	0.0027	0.0066	0.0059	0.0006	0.0000	0.0000
26	0.0000	0.0010	0.0007	0.0002	0.0000	0.0000	0.0000	0.0000	0.0010	0.0007	0.0002	0.0000	0.0000	0.0000
27	0.0000	0.0007	0.0012	0.0013	0.0011	0.0002	0.0000	0.0000	0.0008	0.0013	0.0014	0.0012	0.0002	0.0000
28	0.0000	0.0009	0.0023	0.0027	0.0010	0.0004	0.0000	0.0000	0.0009	0.0023	0.0028	0.0011	0.0004	0.0000
29	0.0000	0.0030	0.0075	0.0096	0.0025	0.0003	0.0000	0.0000	0.0030	0.0075	0.0096	0.0025	0.0003	0.0000
30	0.0000	0.0001	0.0002	0.0001	0.0000	0.0000	0.0000	0.0000	0.0001	0.0002	0.0001	0.0000	0.0000	0.0000
31	0.0000	0.0032	0.0080	0.0137	0.0174	0.0040	0.0000	0.0000	0.0033	0.0080	0.0137	0.0175	0.0040	0.0000
32	0.0000	0.0027	0.0057	0.0073	0.0048	0.0008	0.0000	0.0000	0.0027	0.0058	0.0079	0.0051	0.0009	0.0000
33	0.0000	0.0024	0.0068	0.0017	0.0003	0.0000	0.0000	0.0000	0.0024	0.0068	0.0017	0.0003	0.0000	0.0000
34	0.0000	0.0019	0.0050	0.0059	0.0045	0.0021	0.0003	0.0000	0.0018	0.0050	0.0059	0.0045	0.0021	0.0003
35	0.0000	0.0056	0.0126	0.0247	0.0271	0.0002	0.0000	0.0000	0.0062	0.0138	0.0253	0.0273	0.0002	0.0000
36	0.0000	0.0020	0.0061	0.0058	0.0037	0.0025	0.0015	0.0000	0.0071	0.0101	0.0071	0.0043	0.0028	0.0016
37	0.0000	0.0010	0.0007	0.0002	0.0000	0.0000	0.0000	0.0000	0.0011	0.0007	0.0002	0.0000	0.0000	0.0000
38	0.0000	0.0003	0.0004	0.0004	0.0002	0.0001	0.0000	0.0000	0.0003	0.0004	0.0004	0.0002	0.0001	0.0000

Table 8.3: Nguyen-Dupuis: RMSE of link travel time predictions for different links at different times done at time 16:00. The cases of complete data and with some missing data are considered.

missing information in links 18, 20 and 36. We can see that missing information increases the RMSEs. The errors are induced in the proper missing links and those adjacent to them in the path directions. An important result is that the degradation due to missing data is not important, as long as the number and associated information of missing data is small. This is due to the large correlations of the data for different links, which contain a part of the information contained in the missing links.

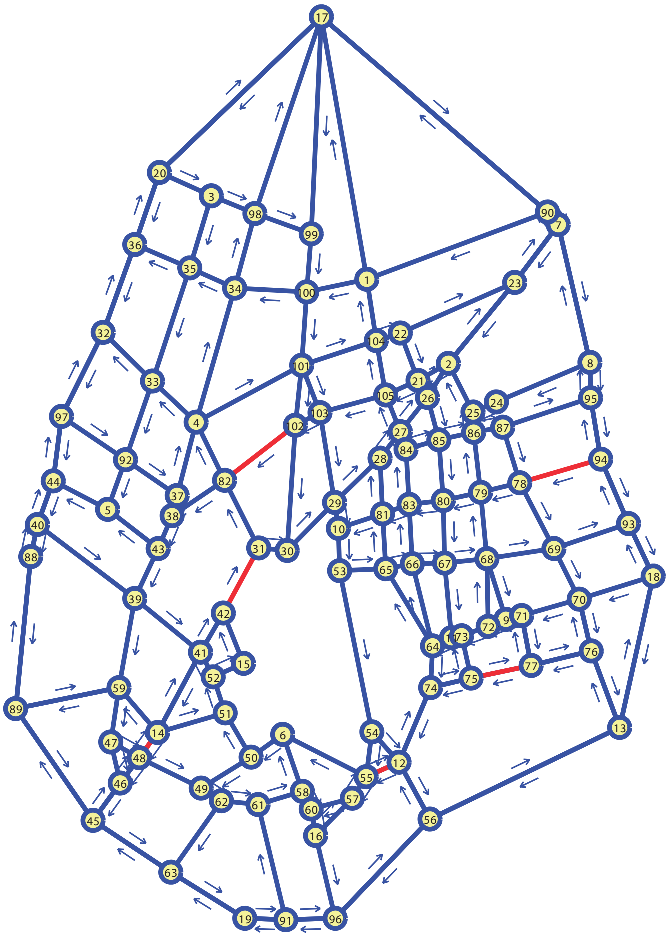

Figure 8.6: Ciudad Real network showing their nodes and one and two-ways links.

8.5.2 The Ciudad Real network

To demonstrate that the proposed methods can be applied to real size networks, we consider the Ciudad Real network with 219 links and 590 paths, which is shown in Figure 8.6, where all streets (one way and two ways) are indicated.

Since we had no real data, as in the previous example, we have simulated 16 minutes ranging from 8:00 to 8:16 for 100 different days of random incoming traffic in this network, and measured traffic every 15 seconds. Next, using the resulting traffic flows as input, we have utilized one of the physical-queue representation deterministic dynamic models based on the travel time function approach (see Szeto and Lo (2005) and Castillo et al. (2011)) to obtain the required travel times. The results are similar to those in Figure 8.1 and we do not include them due to lack of space.

Figure 8.7 shows in red (the lightest continuous line) the selected day travel time

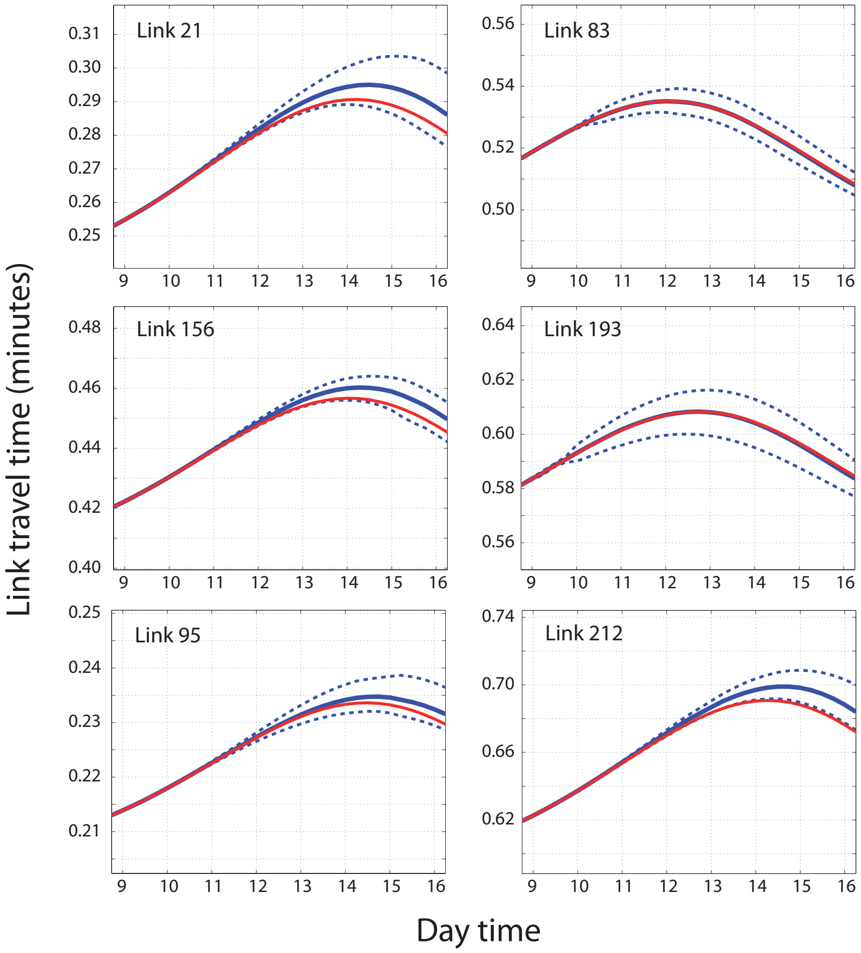

Figure 8.7: Ciudad Real network: Predicted link travel time for different day times made at time $t =$ 8:08 and 0.95 associated confidence intervals for some representative selected links.

evolution in the corresponding links, in blue (the darkest continuous line) the predicted link travel times made at 8:08, and in blue dashed lines the 0.95 confidence intervals of these predictions.

The time required to make these predictions is very low. However, what takes time is to simulate the 100 days of flow or to measure it in reality. Fortunately, this must be done only once and the information used for predictions during many days. Nevertheless, this information needs to be updated from time to time to reproduce the long time variations of link travel times.

Due to the high number of links in this network and to avoid being very repetitive, we cannot show all the data and resulting predictions and densities, but they are similar to those already presented for the Nguyen-Dupuis example.

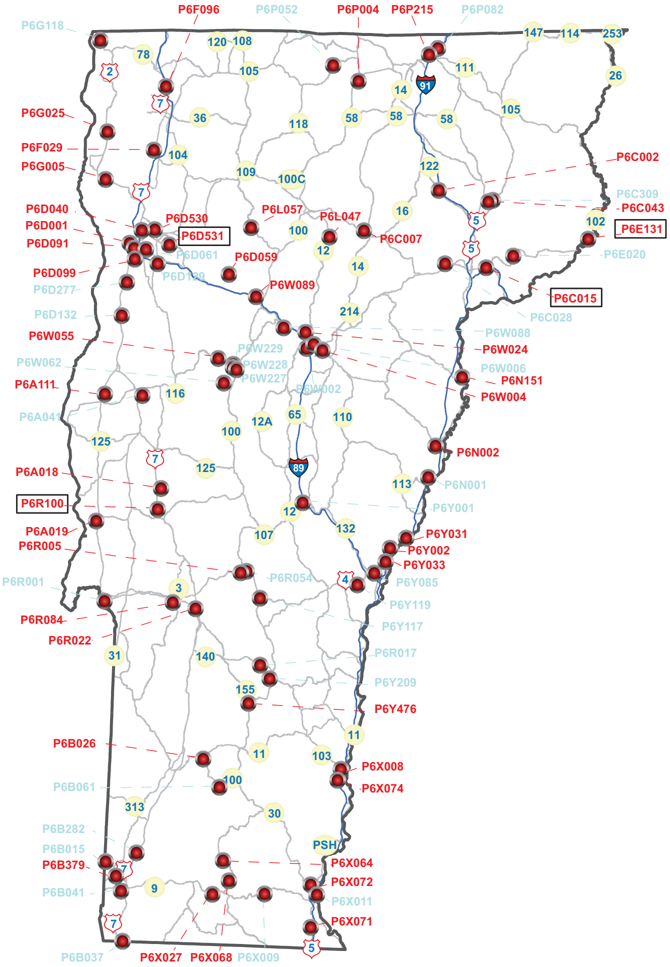

Figure 8.8: Station locations of the Vermont-State data.

8.5.3 The Vermont-state case example

In this section we present an example with real data, which come from the Continuous Traffic Count (CTC) Program of the Vermont Agency of Transportation (VTrans). There are 85 stations, as indicated in Figure 8.8, where their exact locations and associated names are shown. These data are available at http://www.aot.state.vt.us.

We have used the daily counts from September 2008 to July 2010 (two years) obtained by the permanent traffic recorder stations of VTrans.

Since there is a significant different behavior of traffic in weekdays and weekends, we have initially selected for our example only Monday-Thursday traffic flows, and we have removed outliers, which correspond to holidays or days where some special events have occurred. In order to detect singular days, we have evaluated the mean traffic flow at each location-hour pair and calculated the number of standard deviations the actual traffic flow differs from the mean. If this number exceeds the value 2.8 (corresponding to a probability of 0.0025) and the maximum daily deviation in a given location is larger than the 40% of the daily mean flow, we have removed that day considering it as an outlier. The exterior dark (black) bands in the four graphs in Figure 8.9 represent the criterion used to detect outliers[3]. The removed observations appear as dashed lines (in red color) that have some points outside the previous bands. A simple observation to these days reveals that they show a completely different trend. The remaining data appear in light (green) color and occupy a central region limited by the bands. In addition, the mean curves, before removing outliers and after removing them, appear in blue and yellow (central area lines), respectively. Note that the differences are very small due to the relatively low number of outliers. Finally, we have selected 46 stations (their labels appear in Figure 8.8 boldfaced) and 66 days such that the $46 \times 24 = 1104$ station-time pairs have complete data (for all 66 days).

In this example we apply our GBGN model to traffic counts.

With the selected data we have estimated the generalized Beta parameters b, c, α and θ for each location-time pair. The graphs in Figure 8.10 show the estimated generalized Beta cumulative distribution functions and the corresponding empirical distribution functions for some selected station-time pairs (their labels appear with a rectangular frame in Figure 8.8). Similar results are obtained for other stations and times. As can be seen, the fitting is very reasonable, showing that the generalized Beta model reproduces the observed data fairly well. In addition, since the data used cover many different stations and both congested and uncongested periods, this proves that the generalized Beta model used is adequate and reasonable in both cases.

We have used the proposed model to predict traffic flows at all stations from 8:00 to 24:00. To this end, we have used information of traffic at 8:00 and before at all stations, and utilized the process described in subsection 8.4.3. This means that we have selected the most informative variables and calculated the conditional means and covariance matrices and applied the formula (8.14) to determine the 0.95 confidence intervals for each location-time pair. The Figure 8.11 show some representative examples of predicted (in blue) and actual (in red) traffic flows together with their corresponding 0.95 confidence bands. They correspond to the same stations as those in Figure 8.9. It demonstrates that the traffic predictions are good and very close to the real ones. It is important to note that we provide confidence bands instead of point estimates and that the width of the confidence bands is very small compared with the dispersion observed in the upper four graphs of Figure 8.9.

In order to measure the general performance of the model when predicting flows, we have considered the case of a prediction made at 8:00 of the traffic flow for the rest of the

[3] We could have used another criteria, but for illustrative purposes, we consider this a reasonable one.

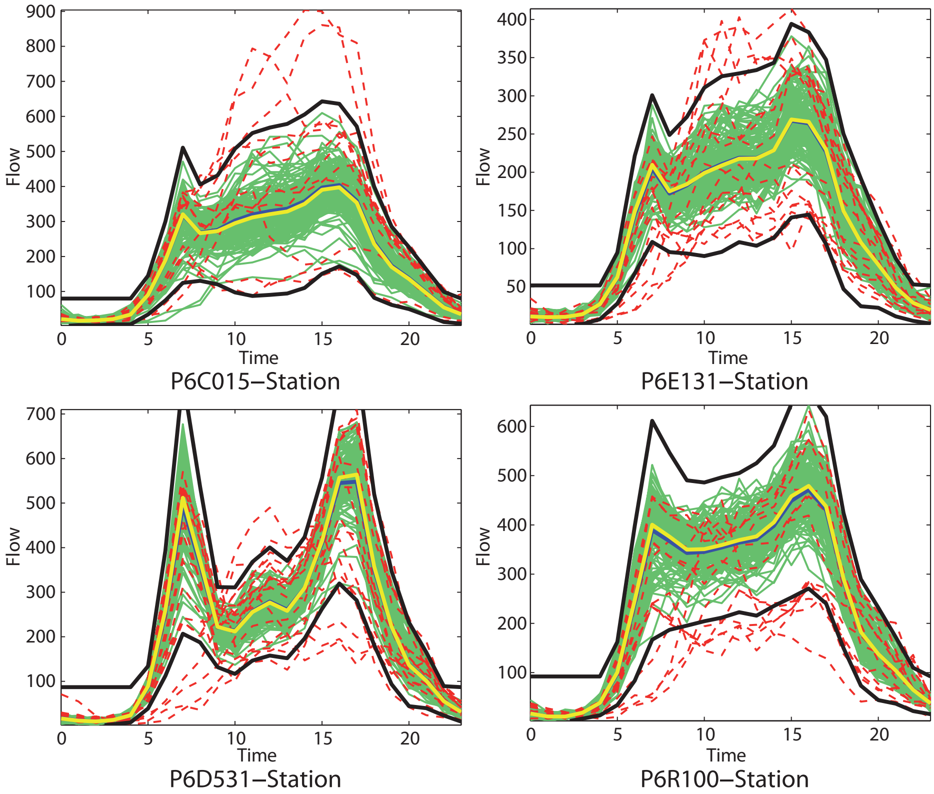

Figure 8.9: Some illustrative examples of daily traffic volume evolution in the Vermont-State example, showing the outlier detection bands (in black), the thick continuous lines of mean values before and after removing outliers (in blue and yellow, respectively), the discarded days (in dashed red) and the accepted data (in green).

day. We have measured the percent RMSE/mean ratios of flow predictions for all stations from 8:00 to 24:00. The results appear in Table 8.4. The maximum daily percent is 13.83% with only four stations with a maximum above 10%, and 25 of a total of 46 stations with maxima less than 5%. Since the percent RMSE/mean ratios of flow predictions are low, this proves that the proposed method provides a practical tool to deal with real cases.

8.6 A comparison among several Bayesian network models

In this section we compare the Gaussian mixture (GM) model proposed by Sun et al. (2006); Yu and Cho (2008), the normal Bayesian network (NBN) and the model proposed in this chapter (GBGN).

Previously, we comment on some implementation problems we have faced in the implementation of the Gaussian mixture Bayesian network models and we indicate how them

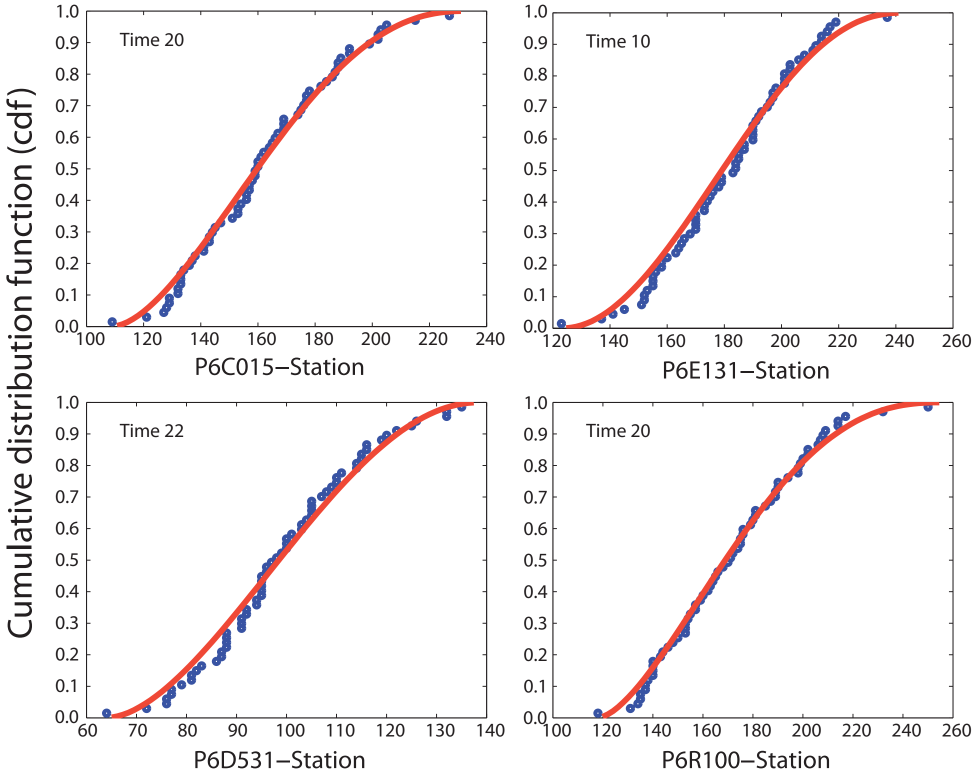

Figure 8.10: The Vermont-State example. Estimated generalized Beta cumulative distribution functions and the corresponding empirical distribution functions for some selected station-time pairs.

have been solved.

8.6.1 Implementation problems of the GM models

Any experienced programmer knows that implementing a given method in a computer program usually faces many numerical problems that need to be solved. In the implementation of this models we have encountered the following problems:

1. *Dealing with singular or almost singular covariance matrices.* If the matrix is singular, it has no inverse, but if it is almost singular, its inverse cannot be calculated with precision. To this end, it is very important to identify singular or almost singular matrices. This can be done with the help of the *condition number of the matrix* and its *determinant*. The problem is that the number of variables must be taken into account, because the determinant depends on the matrix dimension and the variable units being used.

   The condition number of a function with respect to an argument measures the

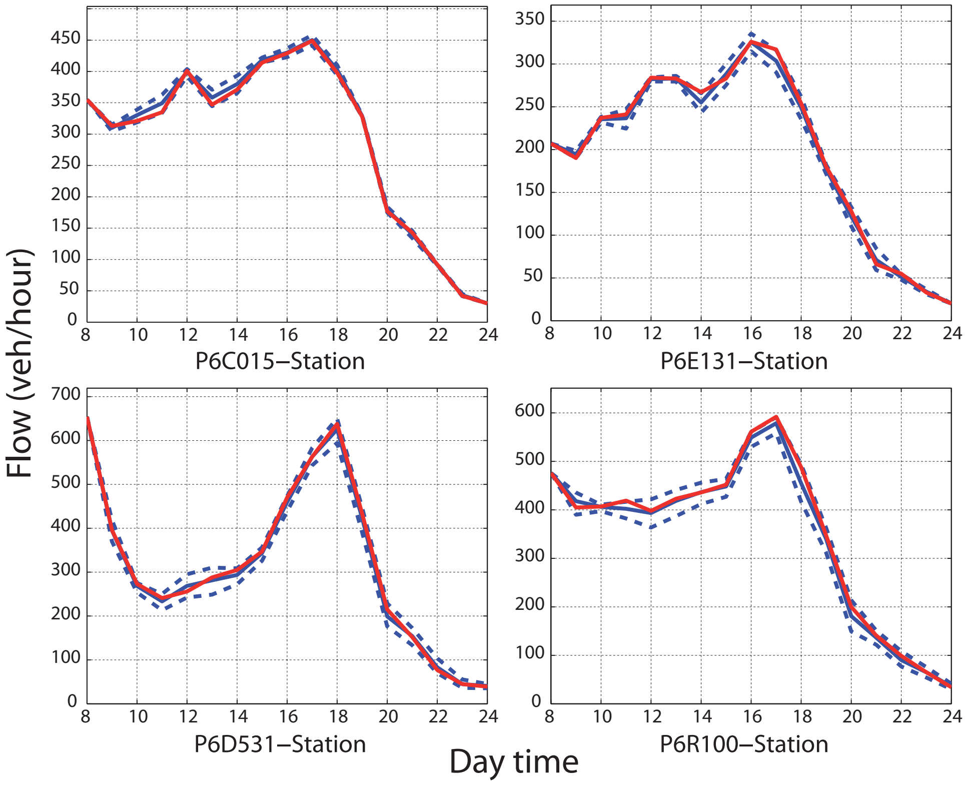

Figure 8.11: The Vermont-State example. Some representative examples of predicted (in blue) and actual (in red) traffic flows together with their corresponding 0.95 confidence bands (dashed lines).

asymptotically worst case of how much the image of the function can change in proportion to small changes in its arguments. A problem with a low condition number is said to be *well-conditioned*, while a problem with a high condition number is said to be *ill-conditioned*.

The condition number of a matrix is defined as the product of the norm of A and the norm of its inverse, that is,

$$\kappa(\mathbf{A}) = ||\mathbf{A}||||\mathbf{A}||^{-1}. \tag{8.17}$$

When using the determinant it must be taken into account the scale of the variables involved, because a change in each element by a factor k implies a change in the determinant by a factor k^n, where n is the dimension of the matrix.

2. *Identifying rows in the covariance matrix that are linear combinations or almost linear combinations of other rows.* To this end, it is convenient to sort the variables

	Prediction time								
Station name	8	10	12	14	16	18	20	22	24
P6A018	0.00	0.47	0.81	0.09	0.69	0.29	1.86	2.34	1.34
P6A019	0.00	4.57	3.38	3.86	4.75	5.58	13.83	9.12	8.54
P6A111	0.00	2.06	0.27	1.83	0.44	0.30	0.47	1.80	0.05
P6B026	0.00	1.24	1.35	1.77	1.37	1.08	3.43	3.87	0.07
P6B379	0.00	0.74	1.90	1.20	0.56	0.25	1.17	2.38	0.11
P6C002	0.00	1.26	0.22	3.34	1.25	0.66	2.47	4.16	0.05
P6C007	0.00	2.41	1.29	1.95	2.30	2.05	4.98	3.92	8.07
P6C015	0.00	1.96	2.66	2.13	0.86	0.95	1.11	0.44	0.03
P6C043	0.00	3.20	3.11	3.48	3.79	2.22	4.92	7.11	7.24
P6D001	0.00	0.73	0.67	0.62	0.00	0.19	0.58	0.46	0.12
P6D040	0.00	0.56	1.26	0.97	0.93	0.30	1.84	1.26	0.08
P6D059	0.00	0.05	2.85	1.24	2.70	0.22	1.19	5.15	0.07
P6D091	0.00	1.27	1.10	0.16	1.40	1.32	1.46	0.57	0.01
P6D099	0.00	0.32	0.34	0.34	0.46	0.50	0.59	0.68	0.01
P6D530	0.00	0.18	0.75	0.87	0.81	0.48	1.85	0.99	0.03
P6D531	0.00	3.66	3.48	3.74	1.86	2.09	3.87	6.16	5.20
P6E131	0.00	2.12	1.40	1.94	1.67	2.28	4.60	2.43	0.04
P6F029	0.00	1.84	3.68	1.86	1.09	2.39	3.88	5.87	3.01
P6F096	0.00	1.49	1.09	0.85	1.13	0.92	0.79	2.14	0.03
P6G005	0.00	0.84	1.20	1.11	0.58	1.40	3.65	3.53	0.02
P6G025	0.00	3.17	2.31	2.31	0.31	0.78	2.47	0.92	0.04
P6L047	0.00	0.02	2.60	0.90	2.14	0.55	0.14	5.06	0.67
P6L057	0.00	7.77	4.38	1.73	3.92	4.25	10.30	3.86	0.07
P6N002	0.00	0.43	0.08	0.84	1.60	2.32	1.92	1.97	1.62
P6N151	0.00	0.49	1.80	1.65	1.27	0.82	1.25	2.03	0.02
P6P004	0.00	3.84	3.71	4.18	4.40	3.36	5.42	11.08	11.16
P6P215	0.00	0.35	0.40	0.64	0.09	1.58	0.60	0.43	0.35
P6R005	0.00	1.69	1.18	0.78	0.69	1.44	2.73	2.40	0.03
P6R022	0.00	0.48	0.84	1.28	0.79	0.73	1.06	1.18	0.04
P6R084	0.00	0.34	0.32	0.70	0.47	1.16	1.01	0.66	0.03
P6R100	0.00	1.70	3.08	2.23	2.40	2.81	6.63	6.10	6.04
P6W004	0.00	1.55	1.97	2.64	1.82	1.22	3.44	4.04	3.96
P6W024	0.00	2.12	1.22	1.47	1.23	1.88	3.25	5.18	3.71
P6W055	0.00	4.39	4.60	6.00	3.42	1.68	4.50	6.11	1.21
P6W089	0.00	1.18	0.99	1.37	1.07	1.00	0.74	1.34	0.02
P6X008	0.00	1.36	1.17	1.71	0.59	0.79	1.29	2.91	0.04
P6X027	0.00	3.16	4.94	3.24	3.11	2.70	3.86	5.82	0.04
P6X064	0.00	2.32	1.83	2.27	2.60	2.29	2.89	4.63	0.06
P6X068	0.00	3.10	3.09	2.21	2.86	3.27	7.12	8.97	0.10
P6X071	0.00	3.53	3.87	3.65	3.36	3.03	4.31	5.85	5.28
P6X072	0.00	0.57	0.57	0.56	0.22	0.54	0.83	0.83	0.02
P6X074	0.00	1.56	1.75	1.35	1.14	1.13	2.90	1.89	1.78
P6Y002	0.00	0.87	0.39	0.46	0.89	0.55	0.25	1.28	0.43
P6Y031	0.00	1.12	1.32	2.18	1.19	1.05	1.60	3.25	0.05
P6Y033	0.00	0.42	0.86	1.63	1.04	0.35	0.32	1.07	0.02
P6Y476	0.00	3.57	1.70	4.68	4.71	1.80	2.45	3.56	0.16

Table 8.4: The Vermont example. Percent of RMSE of flow predictions for different stations at different times done at time 8:00.

by its importance (for example, its coefficient of correlation with the variable to be predicted) in decreasing order and maintain the variables (rows) occupying the first places. It is also important to eliminate the same columns as rows in order to get the covariance matrix of the most important variables.

3. *The evaluation of the multi-normal density* $f(\mathbf{x}; \boldsymbol{\mu}_{aX}, \boldsymbol{\Sigma}_{a\mathbf{XX}})$ *in (5.17) for large number of variables.* There are three main problems here: (a) to evaluate the exponential of a very large number in formula (5.17), (b) to evaluate with a certain precision the determinant $|\boldsymbol{\Sigma}_a|$, and (c) to calculate the square root of the determinant of the covariance matrix, that due to precision problems can be null or negative (small).

In all cases it is convenient to use normalized variables (with mean zero and variance one), which must be later transformed to their original values.

8.6.2 Comparison

To compare, we have used 30 stations of the Vermont state data set that have no missing data in 131 common days and we have fitted the above three models in order to make predictions at 8:00 of the traffic flows for each following hour in the day. In Figure 8.13 we show the percent relative RMSE of the predictions corresponding to each of these models for Station $P6L057$. We note that the results for the remaining stations are similar.

The three lines in the upper graph in Figure 8.13 compare the percent relative RMSE of the flow predictions made at 08:00 for all hours in the range 09:00–24:00 for the GM model with 3 normal components and 20 evidential variables[4], and the NBN and the GBGN using 120 evidential variables. These variables have been selected as those with the largest correlation with the variable being predicted, no matter whether they are colinear or not. The corresponding cpu times are shown in the legend of the graphs and correspond to the cpu required to predict at 08:00 hours the flow of the 131 days for all hours in the indicated range. It is interesting to compare the high cpu time (2852 seconds) required by the GM model when compared with the other two methods NBN and GBGN (546 and 560 seconds, respectively), even though the number of evidencial variables used is much higher (120 instead of 20 variables). A comparison between the NBN and the GBGN models indicates that they are practically equivalent in terms of RMSE and that for some times the NBN is better than the GBGN, but for other times the GBGN produces the best results. An investigation of the causes for these differences led to the conclusion that they are due to the fact that data sets are symmetric in some cases but are not in other cases. For the sake of illustration, we show in Figure 8.12 the pdf of the observed flow in the $P6L057$ station for times 13:00 and 16:00, which are practically symmetric, and for times 19:00 and 22:00, which are not. These four times have been indicated in the upper graph of Figure 8.13 by the corresponding ellipses. According to the results in

[4] Here the evidential variables are selected station-time pairs.

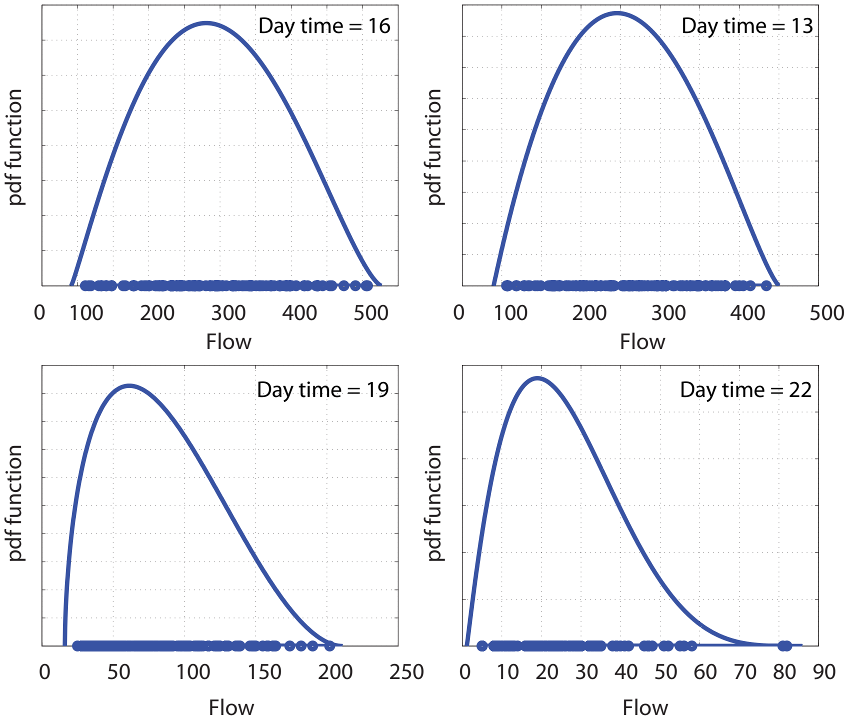

Figure 8.12: A comparison of the pdf densities of the flows at different times for station $P6L057$. Note that the two upper curves are more symmetric than the lower two curves.

this figure, it appears that the NBN behaves better than the GBGN in the presence of symmetric data, while the GBGN is the best for asymmetric data. This explains partially why in the upper graph in Figure 8.13 the NBN model gives lower percent RMSE than the GBGN model for times 13:00 and 16:00, and why the GBGN gives smaller percent RMSE for times 19:00 and 22:00.

The intermediate graph in Figure 8.13 analyzes how the GBGN model improves when the number of evidential variables increases. A comparison of the cases of 20, 60 and 120 evidential variables shows that increasing the number of them leads a very important improvement in terms of RMSE and without no substantial penalization in terms of cpu time.

The lowest graph in Figure 8.13 shows the effect of the number of normal components used in the GM model on the RMSE. After a comparison of this graph with the intermediate one, it is clear that an increase in the number of components is much more costly in terms of cpu than an increase in the number of evidential variables.

In summary, an analysis of the results indicates that:

1. In terms of RMSE, the three methods are satisfactory because they provide suffi-

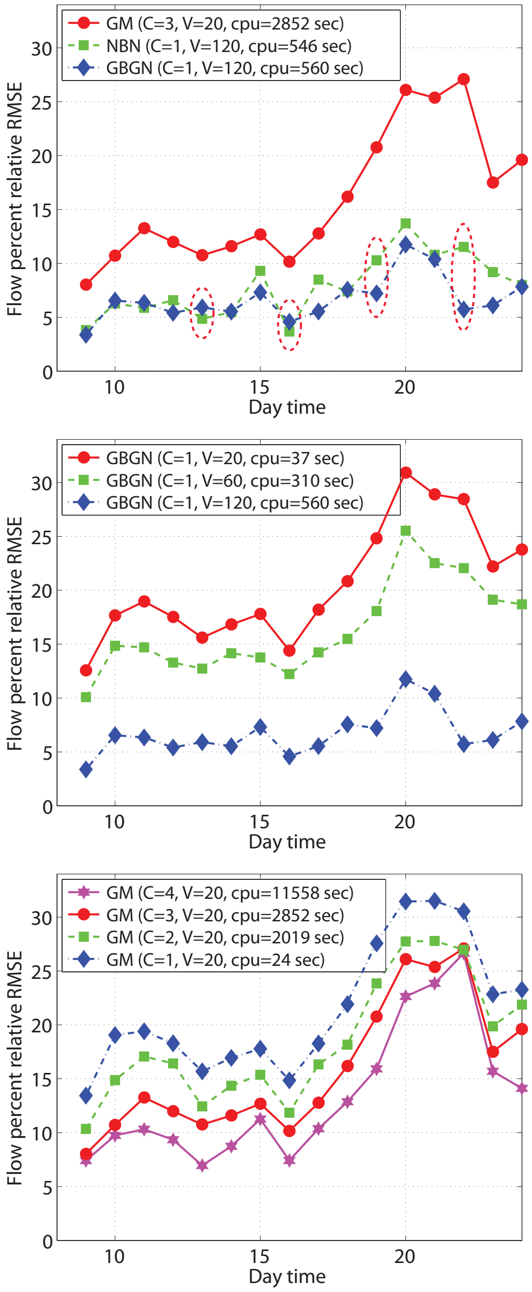

Figure 8.13: A comparison of the flow percent relative RMSE for the Gaussian mixture (GM) model, the normal Bayesian network and the GBGN models.

ciently small values. However, the convenience of each one depends on the data: if the data are unimodal and symmetric, the NBN model provides good results; if the data are unimodal asymmetric, the GBGN model is the most appropriate; finally, for multimodal data, the GM model is the best, because it has more degrees of freedom and so, it has more possibilities to fit any type of data.

2. In terms of computation cost, the NBN and GBGN models are equivalent. However, the GM model demands much more time, because its EM learning algorithm is more complicated and involves iterations. Thus, we recommend the GM model only when very small errors are required. In this case, the other two models cannot compite.

3. As expected, an increase in the number of components in the GM model or an increase in the number of evidential variables, in any model, implies an improvement of the predictions, but this increment is limited by the proper variance of the stochastic traffic process.

4. The implementations of the NBN and GBGN models are much simpler than the implementation of the GM model. Its main difficulties include learning the normal components and weights, some numerical problems that need to be handled in the presence of a high number of evidential variables and colinearities among them.

8.7 Conclusions

The main conclusions drawn from this chapter are:

1. The generalized Beta variables reproduce sufficiently well the marginal densities of the link travel times at different day times (see Figures 8.3 and 8.10). The parameters used to fit different times and links (stations) are independent.

2. The generalized Beta variables reproduce the traffic parameters at any link (station) and time independently of its congestion level (see Figures 8.3 and 8.10).

3. The Beta-Gaussian Bayesian network supplies very good predictions, as shown by the resulting RMSE values in the examples.

4. The model complexity does not depend on the network size, because only the traffic in adjacent links (stations) is relevant to predictions. Consequently, the model can be applied to very large networks and to subnetworks, too.

5. Missing link (station) data does not have a great influence on predictions, because the missing information is contained in other links (stations).

6. Since the required cpu times to make predictions are small, the method can be applied to large networks.

7. A comparison of the proposed GBGN model with other existing Bayesian network models reveals its nice performance. As the NBN model it has a low computational

cost and small RMSE when a large number of evidential variables is used. It is clearly superior to the NBN model when the data have asymmetries. However, when a great precision is needed, the GM model with a high number of components must be used, but then the associated computational cost is prohibitive.

Part IV

Conclusions and Future Work

Chapter 9

Conclusions, future work and publications

Contents

9.1 Conclusions

In order to organize better this section we divide the conclusions in three parts, which correspond to Chapters 6, 7 and 8.

9.1.1 Conclusions with respect to Bayesian methods

1. We have proposed a Bayesian model that assumes the route flows to belong to the shifted-gamma family $\mathcal{F} \equiv \mathcal{H}(\alpha, \theta, \lambda) \equiv \{\alpha + \Gamma(\theta, \lambda); \alpha \geq 0, \theta > 0, \lambda > 0\}$, where the λ parameter is the same for all routes. This is a family that has convenient properties, such as: (a) the variables are positive, (b), it is reproductive and (c) it has positive skewness. In addition, the mean $E[X] = \alpha + \theta/\lambda$ and the variance $Var[X] = \theta/\lambda^2$ are independent parameters. So, the model has more degrees of freedom than other models, as the Poisson, for example, which mean and variance are dependent. In addition it shares all these nice properties, with the exception of the positive skewness, with normal models.

2. As a consequence of the assumed model $\mathcal{F} \equiv \mathcal{H}(\alpha, \theta, \lambda) \equiv \{\alpha + \Gamma(\theta, \lambda); \alpha \geq 0, \theta > 0\}$ for routes, all random flows can be modeled by the same family, no matter whether they are route, OD-pair, link, node or scanned flows.

3. To facilitate the use of the model we have assumed conjugate distributions, that is, the prior and posterior families are coincident. This facilitates the updating of parameters when new information becomes available because very simple and closed formulas for this updating are given.

4. The difficulties in calculating the posterior means is avoided by considering posterior modes instead, which can be easily calculated by standard optimization procedures.

5. An strictly rigorous application of Bayesian methods requires complete samples from route, OD-pair, link or node flows, depending on the flow being estimated. Thus, in the case of route or OD-pair flow estimation, the scanned link technique needs to be applied, because other alternatives such as standard counting does not supply sufficient information to obtain the corresponding route or OD-pair flow samples. These procedures have been shown to lead to extremely good estimates of all traffic flows due to the large amount of information associated with plate scanning.

6. Though from a theoretical point of view, the proposed Bayesian method is valid only for uniquely determined route, OD and link flows samples, application to other flows seems to give good results for point estimates, as the Nguyen-Dupuis and Cuenca examples have shown.

7. In spite of all this, link count data can be sufficient to estimate (uniquely) OD and route flows given repeated sampling. However, these methods require some strong assumptions on the traffic generation process. Without these assumptions, link count information is insufficient to identify origin, destinations or routes. On the contrary, plate scanning makes this estimation possible.

8. Finally, the very low execution times required for the practical examples in this thesis seem to indicate that this method could be very useful even for large networks.

9.1.2 Conclusions with respect to the loading dynamic network model

1. Although existing dynamic traffic models based on the hydraulic analogy explain important features of traffic flow, they become prohibitive for intermediate or large networks. Thus, alternative models become necessary to make them useful in engineering practice.

2. As an alternative, we have proposed a continuous method that implies a reasonable computation time and on the other hand, seems to reproduce the real traffic

in a sufficiently close way. We assume path origin flow functions that are linear combinations of a basic set of time functions. In particular, we can assume linear combinations of normal densities to reproduce the time varying path flows as a reasonable simplification close to the real flows. We note that the combination of these functions leads to complex flow intensity evolution curves with a trend similar to those observed in reality.

3. The path traffic waves propagate through the network by stretching and enlarging according to the degree of congestion of the corresponding links and joining with other path waves at the network nodes. Both the stretching-enlarging and the merging processes are forced to satisfy the FIFO rule within and among different path flows.

4. The link exit-entry time functions $\tau_a^{-1}(t)$ are approximated by monotone cubic Hermite splines and the quality of the approximations can be improved as much as desired by increasing the set of basic points. The model assumes that the link travel time is a non-linear function of the actual and the immediate downstream path link volumes, but some corrections (queue considerations) are made in order to satisfy the FIFO rule at the basic set. Further the use of monotone cubic splines preserving monotonicity, guarantees that the FIFO rule is satisfied at all points.

5. The treatment of traffic congestion can be easily done by means of an iterative process: first, traffic congestion based on the actual link travel time functions is evaluated, and later, the link travel time functions based on congestion are updated. The process can be started by assuming no congestion in the first iteration, and correcting this in successive iterations. The convergence properties of the process are good. We have obtained convergence in 3 or 4 iterations and no more than 15 iterations when high congestion is present.

6. Since the 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 and propagation laws are satisfied over all the network.

9.1.3 Conclusions with respect to stochastic demand dynamic models

1. A generalized-beta Bayesian network has been proposed to reproduce the stochastic dynamic traffic behavior in which the nodes represent traffic variables (link travel time, traffic flow, etc.) at given location-time pairs.

2. The marginals are assumed generalized Beta variables that seem to provide enough freedom to reproduce the densities of the traffic variables at different location-time pairs. This model has been tested with real and simulated data.

3. The model seems to be adequate to reproduce the traffic parameters, no matter if it corresponds to a congested or uncongested period. In addition, the proposed model appears to give very reasonable predictions, as is shown by the resulting values of the RMSE of the examples provided in this thesis.

4. An interesting property of this model is that the complexity of predicting the near future traffic conditions in a link (station) does not depend on the network size, because only the traffic in adjacent links (stations) are relevant to such a prediction. This has very important practical implications because the proposed model can be applied to very large networks.

5. The proposed Bayesian network model is applicable to any sub-network. To this end, we need to identify which are the relevant links for the given sub-network and then observe those links to learn the parameters of the corresponding Bayesian network.

6. The Bayesian network approach presented shows a very good behavior with respect to missing data, as long as the number of affected links is small. The high correlations of the link data due to common paths seems to be the main reason for this behavior. In other words, missing node information can be easily replaced by other link information.

7. The observed cpu times required to make link travel time predictions are very small, which makes the proposed method applicable to real time prediction in very large networks.

8. The proposed Generalized Beta-Gaussian Bayesian network model compares well with other existing Bayesian network models. With the normal Bayesian network (NBN) model shares its low computational cost and small RMSE when using a large number of evidential variables. It is specially recommendable and superior to the NBN model when asymmetries of the data are present. It is only outperformed by the GM model with a high number of components, but in this case, the associated computational cost is almost prohibitive.

9.2 Future work

There are many possibilities for future work. In particular, we can mention the following lines:

1. With respect to the proposed Bayesian method based on the shifted-gamma model, it would be interesting to test it based on real measurements and samples and comparing it with other alternative models.

2. In the models described in this thesis we have paid a special attention to the FIFO rule. However, in reality, different users circulate at different speeds and overtaken is common, specially in uncongested networks. Thus, it would be interesting to consider different classes of users circulating to different mean speeds.

3. Though the use of monotonic splines in the proposed FIFO consistent model guarantees satisfaction of the FIFO rule, it would be convenient to derive the mathematical and statistical properties of the proposed procedure.

4. Future research is needed to reproduce the link congestion and calibrate the model.

5. Finally, we end indicating that the computer programs have been implemented in MatLab without a sufficient optimization effort to reduce the cpu time. Thus, the reported times can be substantially reduced if this optimization and specially other computer languages, such as C, are used. Another alternative consists of using parallel computing, which can be easily implemented.

9.3 Publications from this thesis

The results of this thesis have been published or accepted for publication in the following International Journals:

- Castillo, E., Jiménez, P., Menéndez, J. M., and Nogal, M. (Revision). A Bayesian method for estimating traffic flows based on plate scanning. *Transportation*.

- Castillo, E., Menéndez, J. M., Nogal, M., Sánchez-Cambronero, S., and Jiménez, P. (2011). FIFO rule consistent model for the continuous dynamic network loading problem. *IEEE Transactions on Intelligent Transportation Systems*, In press (DOI:10.1109/TITS.2011.2169668).

- Castillo, E., Menéndez, J. M., Nogal, M., Sánchez-Cambronero, S., and Jiménez, P. (2011). Stochastic demand dynamic traffic models using generalized Beta-Gaussian Bayesian networks. *IEEE Transactions on Intelligent Transportation Systems*, In press (DOI:10.1109/TITS.2011.2173933).

Part V

Appendix

Chapter 10

Programs codes

Contents

10.1 Code for the Bayesian estimation model

This section deals with the Gams implementation of the Bayesian estimation based on plate scanning model which has been exposed in chapter 6.

```
$Title GammaPlateScanning
```

```
file out/GammaPlateScanning.out/;
file out1/GammaPlateScanning1.out/;
file out2/GammaPlateScanningMathematica.out/;
file out3/GammaPlateScanningTablas.out/;
file out4/GammaPlateScanning2.out/;
put out;

OPTION nlp=CONOPT;

$onempty
$offlisting
$offupper
$offsymlist
$offsymxref
$offuellist
$offuelxref
Option limrow=0, limcol=0, solprint=off, sysout=off

SETS
LINK /1*38/
OD /1*18/
I                set of nodes /1*13/
NODES(I)         set of nodes
END              link edges /1*2/
J                maximum number of links in a path /1*6/
K                maximum number of paths per origin-destination pair /1*6/
NNN              sample size /1*20/
MMM              prior sample size /1*10/
itersue          maximum number of iterations for SUE /1*800/
NSC              scanned links/1*18/
NSCA             scanned links +1/1*19/
SNSC(NSC)        subset of scanned links
OBSLINKS(LINK)   subset of observed links
KK               terms of gamma function/1*6/
CC               feasible values of r/1*51/
simulation       number of simulations/1*2/
cases /a,b/
validcases(cases) /a,b/
;

ALIAS(I,I1);
ALIAS(LINK,LINK1);
ALIAS(OD,OD1,OD2);
ALIAS(K,K1,K2);
ALIAS(CC,CC1)

PARAMETER
ORIGIN(OD)       origin of OD
```

```
DESTINATION(OD)  destination of OD
OBSERVEDLINK(LINK)
thetasim(OD,K)
eta(OD,K)        prior hyperparameter
rho(OD,K)        prior hyperparameter
etapost(OD,K)    posterior hyperparameter
posteriorrho(OD,K) posterior hyperparameter
BITSP(CC,NSC)
Val(OD,NSCA,cases,simulation)
mean(OD,cases)   mean value of OD flows
RMSE(OD,cases)   mean squared error of OD flows
COEFF(KK)        Gamma funcion parameters
/1       76.18009172947146
2        -86.50532032941677
3        24.01409824083091
4        -1.231739572450155
5        0.001208650973866179
6        -0.000005395239384953/
VVOBSERVED(NSCA,CC)
VV(CC)
VVOBSERVEDSIM(NSCA,CC,simulation)
MEANCC(CC)
RMSECC(NSCA)
TOBS0(OD,K,NSCA,simulation)
TOBS(OD,K,NSCA,simulation)
TOBS1(OD,K,NSCA,simulation)
RMSET(OD,K,NSCA)
RMSET0(OD,K,NSCA)
RMSET1(OD,K,NSCA)
PREC1(OD,K,OD1,K1) precision matrix
NP(CC)
N(OD)
U1(I)            abscisa of node I
/1 0
2 5
3 5
4 1
5 2
6 3
7 4
8 5
9 3
10 3
11 4
12 1
13 4/
V1(I)            ordinate of node I
/1 4
2 2
```

```
3 0
4 0
5 2
6 4
7 4
8 6
9 0
10 2
11 2
12 6
13 0/

TMEAN(OD,K)      simulated path flows with uniform random variable
VMEAN(LINK)      simulated link flow with uniform random variable
TOOSIMUL(OD,cases,simulation)
TTRUE(OD,K,NNN)  route flows
TTRUEME3(OD,K,NNN) route flows
TTPRIOR(OD,K,MMM)
TPRIOR(OD,K)     OD flows used in ME3 and ME4 models
VPRIOR(LINK)
VTRUE(LINK)      true link flows
VVLINKS(LINK,NSCA,cases)
VOBSERVED(CC)
VOBSERVED1(CC)
VOBSERVED2(CC)
RVAL(CC)
R(OD,K)
BIT(NSC)
BITO(OD,K,NSC)
SCANNED(NSC)
TO(OD)
;

SCALAR
PrintOption /0/
lambda0 /0.4/
kkk/0.3/
epsilon1/0.000000000001/
vvv
cont
ppp
xx
mm
vm
thetapar0
mean0
std
nOD
add
```

```
theta            logit parameter/0.5/
errorsue /1/
tolsue /0.001/
rhosue           relaxation factor for SUE/0.97/
iterationsue
MAXCC            maximum number of combinations of observed links
nscanned         number of scanned links
nscmin           minimum number of scanned links /0/
nscmax           maximum number of scanned links
aux
indic
nlinks
aaa/1/
;

TABLE
LINKEND(LINK,END)
       1    2
    1  1    5
    2  1    12
    3  2    8
    4  2    11
    5  3    11
    6  3    13
    7  4    5
    8  4    9
    9  5    1
   10  5    4
   11  5    6
   12  5    9
   13  6    5
   14  6    7
   15  6    10
   16  6    12
   17  7    6
   18  7    8
   19  7    11
   20  8    2
   21  8    7
   22  8    12
   23  9    4
   24  9    5
   25  9    10
   26  9    13
   27  10   6
   28  10   9
   29  10   11
   30  11   2
   31  11   3
   32  11   7
   33  11   10
   34  12   1
   35  12   6
   36  12   8
   37  13   3
   38  13   9
;

*****************************************************************************************
*  Nodes are obtained from links
*****************************************************************************************
NODES(I)=no;
loop((LINK,END,I),if(LINKEND(LINK,END)=ord(I),NODES(I)=yes;  ););
put "NODES"/;
loop(NODES(I),put I.tl:3:0;);put /;
*****************************************************************************************
PARAMETERS
PATHFLOW(OD,K)
V0(LINK)
T00(OD)
T000(OD) OD flows
/
    1   210.00
    2   430.00
    3   320.00
    4   210.00
    5   320.00
    6    50.00
    7   430.00
    8   110.00
    9    40.00
   10   320.00
   11   110.00
   12   210.00
   13   320.00
   14   210.00
   15    60.00
   16    50.00
   17    40.00
   18    60.00
/
TTRUE00(OD)
Ca(LINK)  free flow cost (BPR parameter)
/
    1       7.000
    2       9.000
    3       9.000
    4       9.000
    5       9.000
    6      11.000
    7      12.000
    8       5.000
    9       7.000
   10      12.000
   11      12.000
   12       9.000
   13      12.000
   14       5.000
   15       4.000
   16       6.000
   17       5.000
   18       9.000
   19       4.000
   20       9.000
   21       9.000
   22      14.000
   23       5.000
   24       9.000
   25       5.000
   26       9.000
   27       4.000
   28       4.000
   29       4.000
   30       9.000
   31       8.000
   32       4.000
   33       4.000
   34       9.000
   35       7.000
   36      14.000
   37      11.000
   38       9.000
/
alpha(LINK)  (BPR parameter)
/
    1       1.000
    2       1.000
    3       1.000
    4       1.000
    5       1.000
    6       1.000
    7       1.000
    8       1.000
    9       1.000
   10       1.000
   11       1.000
   12       1.000
   13       1.000
   14       1.000
   15       1.000
   16       1.000
   17       1.000
   18       1.000
   19       1.000
   20       1.000
   21       1.000
   22       1.000
   23       1.000
   24       1.000
   25       1.000
   26       1.000
   27       1.000
   28       1.000
   29       1.000
   30       1.000
   31       1.000
   32       1.000
   33       1.000
   34       1.000
   35       1.000
   36       1.000
   37       1.000
   38       1.000
/
Qa(LINK)  Link capacity (BPR parameter)
/
    1     700.000
    2     560.000
    3     700.000
    4     280.000
    5     560.000
    6     560.000
    7     560.000
    8     375.000
    9     700.000
   10     560.000
   11     420.000
   12     420.000
   13     420.000
   14     700.000
   15     280.000
   16     140.000
   17     700.000
   18     700.000
   19     700.000
   20     700.000
   21      700.000
   22      560.000
   23      375.000
   24      420.000
   25      280.000
   26      280.000
   27      280.000
   28      280.000
   29      700.000
   30      280.000
   31      560.000
   32      700.000
   33      700.000
   34      560.000
   35      140.000
   36      560.000
   37      560.000
   38      280.000
/
gamma(LINK) (BPR parameter)
/
    1        4.000
    2        4.000
    3        4.000
    4        4.000
    5        4.000
    6        4.000
    7        4.000
    8        4.000
    9        4.000
   10        4.000
   11        4.000
   12        4.000
   13        4.000
   14        4.000
   15        4.000
   16        4.000
   17        4.000
   18        4.000
   19        4.000
   20        4.000
   21        4.000
   22        4.000
   23        4.000
   24        4.000
   25        4.000
   26        4.000
   27        4.000
   28        4.000
   29       4.000
   30       4.000
   31       4.000
   32       4.000
   33       4.000
   34       4.000
   35       4.000
   36       4.000
   37       4.000
   38       4.000
/;

TABLE
PATHLINKS(OD,K,J) routes defined by links
            1    2    3    4    5    6
    1.1     1   11   14   18   20    0
    1.2     2   35   14   18   20    0
    1.3     2   36   20    0    0    0
    2.1     1   11   14   19   31    0
    2.2     1   11   15   29   31    0
    2.3     1   12   25   29   31    0
    2.4     1   12   26   37    0    0
    2.5     2   35   14   19   31    0
    2.6     2   35   15   29   31    0
    3.1     1   11   14   18    0    0
    3.2     2   35   14   18    0    0
    3.3     2   36    0    0    0    0
    4.1     3   21   17   13    9    0
    4.2     3   21   17   16   34    0
    4.3     3   22   34    0    0    0
    5.1     3   21   17   13   10    0
    5.2     3   21   19   33   28   23
    5.3     4   33   28   23    0    0
    6.1     3   21   17   16    0    0
    6.2     3   22    0    0    0    0
    7.1     5   32   17   13    9    0
    7.2     5   32   17   16   34    0
    7.3     5   33   27   13    9    0
    7.4     5   33   27   16   34    0
    7.5     5   33   28   24    9    0
    7.6     6   38   24    9    0    0
    8.1     5   33   28   23    0    0
    8.2     6   38   23    0    0    0
    9.1     5   32   17   16    0    0
    9.2     5   33   27   16    0    0
   10.1     7   11   14   18   20    0
   10.2     8   25   29   30    0    0
   10.3     8   25   29   32   18   20
   11.1     8   25   29   31    0    0
   11.2     8   26   37    0    0    0
   12.1     7   11   14   18    0    0
   12.2     8   25   29   32   18    0
   13.1    21   17   13    9    0    0
   13.2    21   17   16   34    0    0
   13.3    22   34    0    0    0    0
   14.1    21   17   13   10    0    0
   14.2    21   19   33   28   23    0
   15.1    21   17   16    0    0    0
   15.2    22    0    0    0    0    0
   16.1    35   14   18   20    0    0
   16.2    36   20    0    0    0    0
   17.1    35   14   19   31    0    0
   17.2    35   15   29   31    0    0
   18.1    35   14   18    0    0    0
   18.2    36    0    0    0    0    0
;

put "PATHLINKS"/;
loop((OD,K)$(PATHLINKS(OD,K,'1')<>0), loop(J,if(PATHLINKS(OD,K,J)<>0,
             put PATHLINKS(OD,K,J):3:0;););put /;);

TOO(OD)=TOOO(OD);
put "Origin destinations flows"/;
loop(OD,put TOO(OD):6:0);put /;


PARAMETER
PATHFLOW(OD,K)
Kdelta(I,I) Kronecker's delta
DELTA(LINK,K,OD) link-route incidence matrix
PPRIOR(OD,K) prior route choice probability
P1(OD,K) route choice probability
P(OD,K)
D(OD,K,OD1,K1,LINK) one if LINK discriminates route OD-K fromn route OD1-K1 (used in model
                                                                                     SCANN)
;

*******************************************************************************************
*  Kronecker's delta is defined
*******************************************************************************************
kdelta(I,I1)=0;
kdelta(I,I1)$(ord(I)=ord(I1))=1;

*******************************************************************************************
* CALCULATING DELTA MATRIX
*******************************************************************************************
DELTA(LINK,K,OD)$(PATHLINKS(OD,K,'1')<>0)=SUM(J$(ord(LINK)=PATHLINKS(OD,K,J)),1);
*******************************************************************************************
```

```
put "links:   ";
loop(LINK,put LINK.tl:4:0;);put /;
put "delta"/;
loop((OD,K)$(PATHLINKS(OD,K,’1’)<>0),loop(LINK,put DELTA(LINK,K,OD):2:0);  put /; );
******************************************************************************************

******************************************************************************************
* origins and destinations of OD are obtained
******************************************************************************************
DESTINATION(OD)=0;
loop(OD,loop(K$(PATHLINKS(OD,K,’1’)<>0),if(ord(K)=1,ORIGIN(OD)=PATHLINKS(OD,K,’1’););
        loop(J,if(DESTINATION(OD)+PATHLINKS(OD,K,J)=0,DESTINATION(OD)=PATHLINKS(OD,K,J-1));
); ); );
put "ORIGINS and DESTINATIONS"/;
loop(OD,put OD.tl:3:0, ORIGIN(OD):4:0,DESTINATION(OD):4:0/;);
ORIGIN(OD)=SUM(LINK1$(ord(LINK1)=ORIGIN(OD)),LINKEND(LINK1,’1’));
DESTINATION(OD)=SUM(LINK1$(ord(LINK1)=DESTINATION(OD)),LINKEND(LINK1,’2’));
******************************************************************************************
* origins and destinations of routes are checked
******************************************************************************************
put "CHECKING PATHS"/;
loop(OD,loop(K$(PATHLINKS(OD,K,’1’)<>0),
if(SUM(LINK$(PATHLINKS(OD,K,’1’)=ord(LINK)),LINKEND(LINK,’1’))<>ORIGIN(OD),put "ERRONEOUS
                                     ORIGIN IN PATH ",ord(K):4:0," OF OD ",ord(OD):4:0/;);
aux=0;
loop(J$(aux=0 and ord(J)>1), if(PATHLINKS(OD,K,J)=0,aux=1;
          if(SUM(LINK$(PATHLINKS(OD,K,J-1)=ord(LINK)),LINKEND(LINK,’2’))<>DESTINATION(OD),
                 put "ERRONEOUS DESTINATION IN PATH ",ord(K):4:0," OF OD ",ord(OD):4:0/;);
else
if(SUM(LINK$(PATHLINKS(OD,K,J)=ord(LINK)),LINKEND(LINK,’1’))<>SUM(LINK$(PATHLINKS(OD,K,J-1)
          =ord(LINK)),LINKEND(LINK,’2’)),put "ERRONEOUS LINK ",PATHLINKS(OD,K,J):4:0,
          " IN PATH ",ord(K):4:0," OF OD ",ord(OD):4:0/;);
);  );  ););
put /;
******************************************************************************************

BINARY VARIABLES
U(LINK);

VARIABLES
Z;

POSITIVE VARIABLES
V(LINK)
C(K,OD)
TKK(OD,K)
TB(OD,K)
ALPHAA(OD,K)
```

```
lambda;
lambda.lo=0.001;

V.lo(LINK)=0.00000000001;
V.up(LINK)=5000;

EQUATIONS
z1def
z2def
z3def
VDEF
VDEF2
Cdef(K,OD)
CCC1
CCC2
zGBay
;

z1def..z=e=SUM(OD,TOO(OD)*log(SUM(K$(PATHLINKS(OD,K,'1')<>0),exp(-theta*C(K,OD)))))/theta+
            SUM(LINK,V(LINK)*(Ca(LINK)*(1+ALPHA(LINK)*(V(LINK)/Qa(LINK))**GAMMA(LINK)))-
            Ca(LINK)*(V(LINK)+ALPHA(LINK)*Qa(LINK)*(V(LINK)/Qa(LINK))**(GAMMA(LINK)+1)/
            (GAMMA(LINK)+1)));
Cdef(K,OD)..C(K,OD)$(PATHLINKS(OD,K,'1')<>0)=e=SUM(LINK,(Ca(LINK)*(1+ALPHA(LINK)*(V(LINK)/
            Qa(LINK))**GAMMA(LINK)))*DELTA(LINK,K,OD));
MODEL SUE/z1def,Cdef/;

z2def..z=e=SUM(LINK,U(LINK));
CCC1(OD,K,OD1,K1)$(((ord(OD)<>ord(OD1)) or (ord(K)<>ord(K1))) and PATHLINKS(OD,K,'1')<>0
            and PATHLINKS(OD1,K1,'1')<>0 and (SUM(LINK,DELTA(LINK,K,OD)*DELTA(LINK,K1,OD1))
            >0))..SUM(LINK,U(LINK)*(DELTA(LINK,K,OD)+DELTA(LINK,K1,OD1))*(1-
            DELTA(LINK,K,OD)*DELTA(LINK,K1,OD1)))=g=aaa;
CCC2(OD,K)$(PATHLINKS(OD,K,'1')<>0)..SUM(LINK,U(LINK)*DELTA(LINK,K,OD))=g=1;
MODEL SCANN/z2def,CCC1,CCC2/;

z3def..z=e=SUM((OD,K)$(PATHLINKS(OD,K,'1')<>0),sqr((TKK(OD,K)-TPRIOR(OD,K))/TPRIOR(OD,K)));
VDEF(CC)$(ord(CC)<=MAXCC)..VV(CC)-SUM((OD,K)$(PATHLINKS(OD,K,'1')<>0 and RVAL(CC)=R(OD,K)),
            TKK(OD,K))=e=0;
VDEF2(OBSLINKS(LINK))..OBSERVEDLINK(OBSLINKS)-SUM((OD,K)$(PATHLINKS(OD,K,'1')<>0),
            TKK(OD,K)*DELTA(LINK,K,OD))=e=0;
MODEL ME3/z3def,VDEF/;
MODEL ME4/z3def,VDEF2/;

zGBay..z=e=SUM((OD,K)$(PATHLINKS(OD,K,'1')<>0),(TB(OD,K)-1)*(SUM(MMM,log(max(
            TTPRIOR(OD,K,MMM),epsilon1)-ALPHAA(OD,K)))+SUM(NNN,log(max(TTRUEME3(OD,K,NNN),
            epsilon1)-ALPHAA(OD,K))))-lambda*posteriorrho(OD,K)+etapost(OD,K)*(lambda*
            ALPHAA(OD,K)+TB(OD,K)*log(lambda)-(-(TB(OD,K) + 5.5) + (TB(OD,K) + 0.5)*
            log(TB(OD,K) + 5.5)+log(2.506628274631*(1.00000000019 + Sum(kk,Coeff(kk)/
            (TB(OD,K) + ord(kk))))/max(TB(OD,K),epsilon1)))));
MODEL GammaBayes/zGBay/;
```

```
******************************************************************************************
* THE SUBSET OF LINKS TO BE SCANNED IS DETERMINED
******************************************************************************************

* D matrix is generated *****************************************************************
D(OD,K,OD1,K1,LINK)$(PATHLINKS(OD,K,'1')<>0 and PATHLINKS(OD1,K1,'1')<>0)=0;
D(OD,K,OD1,K1,LINK)$(PATHLINKS(OD,K,'1')<>0 and PATHLINKS(OD1,K1,'1')<>0 and
                                                DELTA(LINK,K,OD)<>DELTA(LINK,K1,OD1))=1;
******************************************************************************************

OPTION mip=BDMLP;
OPTION OPTCR=1.0E-10;
* Scann Model refers to Equation (6.1)
SOLVE SCANN USING mip MINIMIZING z;
if(SCANN.modelstat<>2 or SCANN.solvestat<>1, put "SCANN=",z.l:12:2," modelstat=",
                SCANN.modelstat," solvestat=",SCANN.solvestat," resusd=",SCANN.resusd/; );
put "SUGGESTED LINKS TO BE SCANNED"/;
cont=0;
loop(LINK,if(U.l(LINK)=1,cont=cont+1;put cont:5:0,"  ",LINK.tl:3:0/;);); 

* Selected links are scanned ************************************************************
cont=0;
SCANNED(NSC)=0;
loop(LINK,if(U.l(LINK)=1,cont=cont+1;SCANNED(NSC)$(ord(NSC)=cont)= ord(LINK););); 
******************************************************************************************

* OPTIMAL SET OF SCANNED LINKS TO COUNT DATA
SCANNED(NSC)=0;
SCANNED('1')=1;
SCANNED('2')=2;
SCANNED('3')=3;
SCANNED('4')=5;
SCANNED('5')=8;
SCANNED('6')=9;
SCANNED('7')=11;
SCANNED('8')=13;
SCANNED('9')=18;
SCANNED('10')=20;
SCANNED('11')=21;
SCANNED('12')=22;
SCANNED('13')=23;
SCANNED('14')=29;
SCANNED('15')=31;
SCANNED('16')=33;
SCANNED('17')=34;
SCANNED('18')=36;

loop(NSC,put SCANNED(NSC):3:0/;);
```

```
******************************************************************************************
* SOLVES THE SUE PROBLEM BY TWO METHODS
******************************************************************************************
V.l(LINK)=100;
******************************************************************************************
* 1 - SOLVES THE SUE PROBLEM BY ITERATIONS
* looking for the solution to Equation (3.92)
******************************************************************************************
T00(OD)=T000(OD);
errorsue=1;
if(PrintOption=1,put "SOLVING BY ITERATIONS"/; );
iterationsue=0;
loop(itersue$(errorsue>tolsue),
  iterationsue=iterationsue+1;
  C.l(K,OD)$(PATHLINKS(OD,K,'1')<>0)=SUM(LINK,(Ca(LINK)*(1+ALPHA(LINK)*(V.l(LINK)/Qa(LINK))
                                 **GAMMA(LINK)))*DELTA(LINK,K,OD));

  loop((OD,K)$(PATHLINKS(OD,K,'1')<>0),if(SUM(K1$(PATHLINKS(OD,K1,'1')<>0),exp(-C.l(K1,OD)*
                                  theta))=0,
  put "TEST ",OD.tl:4:0,K.tl:4:0,(SUM(K1$(PATHLINKS(OD,K1,'1')<>0),exp(-C.l(K1,OD)*theta)-
                                  C.l(K,OD))):12:8,C.l(K,OD):12:3/;); );

  P(OD,K)$(PATHLINKS(OD,K,'1')<>0)=exp(-C.l(K,OD)*theta)/SUM(K1$(PATHLINKS(OD,K1,'1')<>0),
                                  exp(-C.l(K1,OD)*theta));

  V0(LINK)=SUM((OD,K)$(PATHLINKS(OD,K,'1')<>0),P(OD,K)*T00(OD)*DELTA(LINK,K,OD));
  errorsue=SUM(LINK,abs(V.l(LINK)-V0(LINK)));
  V.l(LINK)=rhosue*V.l(LINK)+(1-rhosue)*V0(LINK);
);

put "errorsue=",errorsue:16:8, " iterationsue=",iterationsue:7:0/;
if(iterationsue>=card(itersue),put out2;put "errorsue=",errorsue:16:8," Iterations=",
                                  card(itersue)/;); put out;

* Prints results *************************************************************************
if(PrintOption=1,
put "RESULTS AFTER SUE ITERATIONS"/;
loop(LINK,put LINK.tl:4:0,V.l(LINK):10:2,(SUM((OD,K)$(PATHLINKS(OD,K,'1')<>0),P(OD,K)*
                                                      T00(OD)*DELTA(LINK,K,OD))):10:2/;);
put /;
put "P matrix"/;
loop(OD,loop(K$(PATHLINKS(OD,K,'1')<>0), put P(OD,K):7:2;); put /;);
put "Costs"/;
loop(OD,loop(K$(PATHLINKS(OD,K,'1')<>0), put C.l(K,OD):7:2;); put /;);
put /;
);

* Generates TMEAN VALUES *****************************************************************
```

```
TMEAN(OD,K)$(PATHLINKS(OD,K,’1’)<>0)=T00(OD)*P(OD,K);
VMEAN(LINK)=V.l(LINK);

if(PrintOption=1,
put "TMEAN"/;
loop(OD,loop(K$(PATHLINKS(OD,K,’1’)<>0),put TMEAN(OD,K):12:4;);put /; );

put "VMEAN"/;
loop(LINK,put VMEAN(LINK):12:4;);
put /;
);

******************************************************************************************
* 2 - SIMULATES PRIOR values by SUE model
*     using Equation (3.91)
******************************************************************************************
* We generate randomly the prior values OD matrix from the true OD flows where
* Each OD flows_i= OD flows * N(1.2, 0.1)
T00(OD)=T000(OD)*(1+normal(0.2,0.1));
* SUE Model refers to Equation (3.91)
SOLVE SUE USING nlp MINIMIZING z;
if(SUE.modelstat<>2 or SUE.solvestat<>1,put "Direct SUE 2 =",z.l:12:2," modelstat=",
          SUE.modelstat," solvestat=",SUE.solvestat,SUE.modelstat," resusd=",SUE.resusd/;);
T0(OD)=SUM(K$(PATHLINKS(OD,K,’1’)<>0),exp(-C.l(K,OD)*theta));
P(OD,K)$(PATHLINKS(OD,K,’1’)<>0)=exp(-C.l(K,OD)*theta)/T0(OD);
TPRIOR(OD,K)$(PATHLINKS(OD,K,’1’)<>0)=T00(OD)*P(OD,K);
VPRIOR(LINK)=SUM((OD,K)$(PATHLINKS(OD,K,’1’)>0),DELTA(LINK,K,OD)*TPRIOR(OD,K));

put "SUE OPTIMIZACION: LINK  V.l(LINK)  VPRIOR(LINK)"/;
loop(LINK,put LINK.tl:4:0,V.l(LINK):7:2,VPRIOR(LINK):8:2/;);put /;

******************************************************************************************
* SOLVES SUE BY ITERATIONS
******************************************************************************************
errorsue=1;
if(PrintOption=1,put "SOLVING BY ITERATIONS"/; );
iterationsue=0;
loop(itersue$(errorsue>tolsue),
  iterationsue=iterationsue+1;
  C.l(K,OD)$(PATHLINKS(OD,K,’1’)<>0)=SUM(LINK,(Ca(LINK)*(1+ALPHA(LINK)*(V.l(LINK)/Qa(LINK))
                         **GAMMA(LINK)))*DELTA(LINK,K,OD));

  loop((OD,K)$(PATHLINKS(OD,K,’1’)<>0),if(SUM(K1$(PATHLINKS(OD,K1,’1’)<>0),exp(-C.l(K1,OD)*
                          theta))=0,
  put "TEST ",OD.tl:4:0,K.tl:4:0,(SUM(K1$(PATHLINKS(OD,K1,’1’)<>0),exp(-C.l(K1,OD)*theta)-
                          C.l(K,OD))):12:8,C.l(K,OD):12:3/;); );

  P(OD,K)$(PATHLINKS(OD,K,’1’)<>0)=exp(-C.l(K,OD)*theta)/SUM(K1$(PATHLINKS(OD,K1,’1’)<>0),
                          exp(-C.l(K1,OD)*theta));
  V0(LINK)=SUM((OD,K)$(PATHLINKS(OD,K,'1')<>0),P(OD,K)*T00(OD)*DELTA(LINK,K,OD));
  errorsue=SUM(LINK,abs(V.l(LINK)-V0(LINK)));
  V.l(LINK)=rhosue*V.l(LINK)+(1-rhosue)*V0(LINK);
);

put "errorsue=",errorsue:16:8, " iterationsue=",iterationsue:7:0/;
if(iterationsue>=card(itersue),put out2;put "errorsue=",errorsue:16:8," Iterations=",
                          card(itersue)/;); put out;
put "SUE ITERATIONS: LINK  V.l(LINK)  VPRIOR(LINK)"/;
loop(LINK,put LINK.tl:4:0,V.l(LINK):7:2,VPRIOR(LINK):8:2/;);put /;
T0(OD)=SUM(K$(PATHLINKS(OD,K,'1')<>0),exp(-C.l(K,OD)*theta));
P(OD,K)$(PATHLINKS(OD,K,'1')<>0)=exp(-C.l(K,OD)*theta)/T0(OD);
TPRIOR(OD,K)$(PATHLINKS(OD,K,'1')<>0)=T00(OD)*P(OD,K);
VPRIOR(LINK)=SUM((OD,K)$(PATHLINKS(OD,K,'1')>0),DELTA(LINK,K,OD)*TPRIOR(OD,K));

* Initialization of precision matrix *****************************************************
PREC1(OD,K,OD1,K1)$(PATHLINKS(OD,K,'1')<>0 and PATHLINKS(OD1,K1,'1')<>0)=0;
PREC1(OD,K,OD,K)$(PATHLINKS(OD,K,'1')<>0)=1/sqr(TPRIOR(OD,K));


nscmax=card(NSC);
* Values initialization
RMSET0(OD,K,NSCA)$(PATHLINKS(OD,K,'1')<>0)=0;
RMSET(OD,K,NSCA)$(PATHLINKS(OD,K,'1')<>0)=0;
RMSET1(OD,K,NSCA)$(PATHLINKS(OD,K,'1')<>0)=0;
******************************************************************************************
* Starts loop simulation
******************************************************************************************
loop(simulation,
put "Simulation=",simulation.tl:9:0/;
T00(OD)=T000(OD);

* The reference route flows are simulated from the real ones using a Gamma(TMEAN(OD,K),1)
******************************************************************************************
if(ORD(simulation)=1,
         lambda0=1/(1+kkk);
put "lambda0=",lambda0:12.4," kkk=",kkk:12.4/;
loop((OD,K)$(PATHLINKS(OD,K,'1')<>0),
thetapar0=lambda0*TPRIOR(OD,K);
thetasim(OD,K)$(PATHLINKS(OD,K,'1')<>0)=thetapar0;
mean0=thetapar0/lambda0;
std=sqrt(thetapar0)/lambda0;

vm=(thetapar0+5*sqrt(thetapar0))/lambda0;
loop(MMM,
         ppp=0;
         while(uniform(0,1)>ppp,
         xx=uniform(0,vm);
         ppp=exp((thetapar0-1)*log((lambda0*xx)/(thetapar0-1))-lambda0*xx-1+thetapar0);
         if(ppp>1,put "ERROR ppp>1 ",ppp:12:5, " thetapar0=", thetapar0:12:5, " lambda0=",
                      lambda0:12:5, " xx= ",xx:12:5, " ppp= ", ppp:12:5/; put 1/(xx-xx)/;);
         abort$(ppp>1) "ppp>1", ppp;
 );
TTPRIOR(OD,K,MMM)=xx;
if(TTPRIOR(OD,K,MMM)<0,put "ERROR ppp>1 ",ppp:12:5, " thetapar0=", thetapar0:12:5,
                      " lambda0=", lambda0:12:5, " xx= ",xx:12:5, " ppp= ", ppp:12:5,
                      " TMEAN=",TMEAN(OD,K):12:5/;);
);
loop(MMM,if(TTPRIOR(OD,K,MMM)/TMEAN(OD,K)>3,put OD.tl:3:0,K.tl:3:0,MMM.tl:5:0,
                      TTPRIOR(OD,K,MMM):12:5, TMEAN(OD,K):12:5/;));
vvv=smax(MMM,TTPRIOR(OD,K,MMM)/TMEAN(OD,K));
abort$(vvv>10) "vvv>10  vvv=",vvv;
);
);

loop((OD,K)$(PATHLINKS(OD,K,'1')<>0),
thetapar0=lambda0*TMEAN(OD,K);
thetasim(OD,K)$(PATHLINKS(OD,K,'1')<>0)=thetapar0;
mean0=thetapar0/lambda0;
std=sqrt(thetapar0)/lambda0;

vm=(thetapar0+5*sqrt(thetapar0))/lambda0;
loop(NNN,
         ppp=0;
         while(uniform(0,1)>ppp,
         xx=uniform(0,vm);
         ppp=exp((thetapar0-1)*log((lambda0*xx)/(thetapar0-1))-lambda0*xx-1+thetapar0);
         if(ppp>1,put "ERROR ppp>1 ",ppp:12:5, " thetapar0=", thetapar0:12:5,
                       " lambda0=", lambda0:12:5, " xx= ",xx:12:5, " ppp= ", ppp:12:5/;
                       put 1/(xx-xx)/;);
         abort$(ppp>1) "ppp>1", ppp;
 );
TTRUE(OD,K,NNN)=xx;
if(TTRUE(OD,K,NNN)<0,put "ERROR ppp>1 ",ppp:12:5, " thetapar0=", thetapar0:12:5,
                       " lambda0=", lambda0:12:5, " xx= ",xx:12:5, " ppp= ", ppp:12:5,
                       " TMEAN=",TMEAN(OD,K):12:5/;);
);
loop(NNN,if(TTRUE(OD,K,NNN)/TMEAN(OD,K)>3,put OD.tl:3:0,K.tl:3:0,NNN.tl:5:0,
                       TTRUE(OD,K,NNN):12:5, TMEAN(OD,K):12:5/;));
vvv=smax(NNN,TTRUE(OD,K,NNN)/TMEAN(OD,K));
abort$(vvv>10) "vvv>10  vvv=",vvv;
);

TTRUE00(OD)=SUM(K$(PATHLINKS(OD,K,'1')>0),TTRUE(OD,K,'1'));

put out1;
put "    TMEAN   TPRIOR   TTRUE "/;
loop((OD,K)$(PATHLINKS(OD,K,'1')>0),put TMEAN(OD,K):9:2,TPRIOR(OD,K):9:2,
                       TTRUE(OD,K,'1'):9:2/;);put /;


* LINK flows are stored ******************************************************************
VTRUE(LINK)=SUM((OD,K)$(PATHLINKS(OD,K,'1')>0),DELTA(LINK,K,OD)*TTRUE(OD,K,'1'));
loop(LINK,OBSERVEDLINK(LINK)=VTRUE(LINK););

put out2;
put "TTRUE=  {";
loop((OD,K)$(PATHLINKS(OD,K,'1')<>0), put TTRUE(OD,K,'1'):7:1/;) ;put"};"/;

put "VTRUE={";
loop(LINK,if(ord(LINK)=1,put VTRUE(LINK):7:1; else  put ",",VTRUE(LINK):7:1/;););
                       put "};"/;

put "TTRUE00={";
loop(OD,if(ord(OD)=1,put TTRUE00(OD):7:1; else  put ",",TTRUE00(OD):7:1;);); put "};"/;
put out;

put out3;
put "\begin{table}"/;
put "\centering"/;
put "\begin{tabular}{|c|c|cccccc|}"/;
put "\hline"/;
put " OD & path & \multicolumn{6}{|c|}{Links}\\"/;
put "\hline"/;
loop((OD,K)$(PATHLINKS(OD,K,'1')>0),put ord(OD):4:0," & ",ord(K):4:0;loop(J,
                        if(PATHLINKS(OD,K,J)>0,put " & ",PATHLINKS(OD,K,J):4:0;
                        else put " & ";););put " \\"/;);
put "\hline"/;
put "\end{tabular}"/;
put "\caption{\label{t400}Assumed paths.}"/;
put "\end{table}"//;
put out;

put out3;
put "\begin{table}"/;
put "\centering"/;
put "\begin{tabular}{|c|c|c|c|c|}"/;
put "\hline"/;
put " OD & Origin & Destination & True mean flow & Prior mean flow\\"/;
put "\hline"/;
loop(OD,put ord(OD):4:0," & ",ORIGIN(OD):4:0," & ",DESTINATION(OD):4:0," & ", T00(OD):7:2,
                        " & ",SUM(K$(PATHLINKS(OD,K,'1')<>0),TPRIOR(OD,K)):7:2, " \\"/;);
put "\hline"/;
put "\end{tabular}"/;
put "\caption{\label{t400}OD pairs and corresponding flows used in the SUE model.}"/;
put "\end{table}"//;
put out;

put out3;
put "\begin{table}"/;
put "\centering"/;
put "\begin{tabular}{|c|c|c|c|}"/;
put "\hline"/;
put "Link & Cost &  True mean flow  & Prior mean flow \\"/;
put "\hline"/;
loop(LINK,put LINKEND(LINK,'1'):2:0,"-",LINKEND(LINK,'2'):2:0;put " & ",(Ca(LINK)*
                        (1+ALPHA(LINK)*(VPRIOR(LINK)/Qa(LINK))**GAMMA(LINK))):7:2," & ",
                         put VMEAN(LINK):7:2," & ", VPRIOR(LINK):7:2"\\"/;);
put "\hline"/;
put "\end{tabular}"/;
put "\caption{\label{tall2}Links, link costs and link total flows resulting from the SUE
                         model.}"/;
put "\end{table}"//;
put out;
******************************************************************************************

* Starts loop cases  *********************************************************************
loop(validcases(cases),
******************************************************************************************
* STARTS ESTIMATION OF OD FLOWS WITH DIFFERENT OBSERVATION LEVELS
******************************************************************************************
loop(NSCA$(ord(NSCA)>=nscmin+1 and ord(NSCA)<=nscmax+1),
nscanned=ord(NSCA)-1;
if(PrintOption=1,
put "nscanned=",nscanned:5:0/;
);
put out2;

put out;
SNSC(NSC)=no;
SNSC(NSC)$(ord(NSC)<=nscanned)=yes;
OBSLINKS(LINK)=no;
loop(SNSC,OBSLINKS(LINK)$(ord(LINK)=SCANNED(SNSC))=yes;);

******************************************************************************************
* CALCULATING RVAL(CC) and R(OD,K) VALUES
* Scanning tecnique
******************************************************************************************
RVAL(CC)=0;

MAXCC=0;
loop((OD,K)$(PATHLINKS(OD,K,'1')<>0), BIT(SNSC)=0;loop((J,SNSC)$(PATHLINKS(OD,K,J)=
                        SCANNED(SNSC)), BIT(SNSC)=1;
);
R(OD,K)=SUM(SNSC(NSC),BIT(SNSC)*power(2,ord(NSC)-1));
NP(CC)$(ord(CC)=MAXCC+1)=SUM(SNSC(NSC),BIT(SNSC));
BITSP(CC,SNSC)$(ord(CC)=MAXCC+1)=BIT(SNSC);

loop(SNSC,
);

add=1;
loop(CC$(ord(CC)<=MAXCC),if(RVAL(CC)=R(OD,K),add=0); );
if(add=1 and R(OD,K)<>0,MAXCC=MAXCC+1;RVAL(CC)$(ord(CC)=MAXCC)=R(OD,K););

BITO(OD,K,NSC)=BIT(NSC);
);

loop(CC$(ord(CC)<=MAXCC),
);

loop(CC$(ord(CC)<=MAXCC),
);
put "nscanned=",nscanned:4:0," MAXCC=",MAXCC:6:0/;
*****************************************************************************************


*****************************************************************************************
* STORES OBSERVED SCANNED AND LINK VALUES
*****************************************************************************************
VOBSERVED(CC)=0;
VOBSERVED(CC)$(ord(CC)<=MAXCC)=SUM((OD,K)$(PATHLINKS(OD,K,'1')<>0 and RVAL(CC)=R(OD,K)),
                        TTRUE(OD,K,'1'));

VOBSERVED1(CC)=VOBSERVED(CC);
if(PrintOption=1,
put "VOBSERVED"/;
loop(CC$(ord(CC)<=MAXCC),
put CC.tl:9:0,RVAL(CC):10:0,VOBSERVED1 (CC):9:3/;
put "OBSERVED LINK FLOWS"/;
loop(OBSLINKS,
put OBSERVEDLINK(OBSLINKS):12:3;
);
PUT /;
*****************************************************************************************
put "MAXCC=",MAXCC:8:0/;
);
);

loop(LINK,
);

VV(CC)$(ord(CC)<=MAXCC)=VOBSERVED(CC);
```

```
if(PrintOption=1,
put "OBSLINKS"/;
loop(OBSLINKS, put OBSERVEDLINK(OBSLINKS):12:3/;);
put "VV(CC)"/;
loop(CC$(ord(CC)<=MAXCC),put VV(CC):12:3/;);
);
********************************************************************************************

if(ord(cases)=1,
********************************************************************************************
* PRIOR Hyperparameters are calculated
********************************************************************************************

eta(OD,K)$(PATHLINKS(OD,K,'1')<>0)=card(MMM);
rho(OD,K)$(PATHLINKS(OD,K,'1')<>0)=SUM(MMM,TTPRIOR(OD,K,MMM));
if(PrintOption=1,
put "PRIOR Hyperparameters"/;
loop((OD,K)$(PATHLINKS(OD,K,'1')<>0),put eta(OD,K):12:4,rho(OD,K):12:4/;);
);

TKK.l(OD,K)$(PATHLINKS(OD,K,'1')<>0)=TPRIOR(OD,K);
SOLVE ME3 USING nlp MINIMIZING z;
if(ME3.modelstat<>2 or ME3.solvestat<>1, put "ME3=",z.l:12:2," modelstat=",ME3.modelstat,
                           " solvestat=",ME3.solvestat," resusd=",ME3.resusd/; put out; );
TOBSO(OD,K,NSCA,simulation)=TKK.l(OD,K);
Val(OD,NSCA,cases,simulation)=SUM(K$(PATHLINKS(OD,K,'1')<>0),TKK.l(OD,K));
if(PrintOption=1,
put "ROUTE flows after ME3  TKK   TTRUE    TPRIOR  nscanned=",nscanned:5:0/;
loop((OD,K)$(PATHLINKS(OD,K,'1')<>0),if(TKK.l(OD,K)<>0 or TTRUE(OD,K,'1')<>0 or
                           TPRIOR(OD,K), put TKK.l(OD,K):12:3,TTRUE(OD,K,'1'):12:3,
                           TPRIOR(OD,K):12:3/;););
put /;
);

PATHFLOW(OD,K)=TKK.l(OD,K);
put out4;
put out;
loop(NNN,
VV(CC)$(ord(CC)<=MAXCC)=SUM((OD,K)$(PATHLINKS(OD,K,'1')<>0 and RVAL(CC)=R(OD,K)),
                           TTRUE(OD,K,NNN));
TKK.l(OD,K)=TPRIOR(OD,K);
SOLVE ME3 USING nlp MINIMIZING z;
if(ME3.modelstat<>2 or ME3.solvestat<>1, put "ME3=",z.l:12:2," modelstat=",
                           ME3.modelstat," solvestat=",ME3.solvestat," resusd=",
                           ME3.resusd/; put out; );
TTRUEME3(OD,K,NNN)=TKK.l(OD,K);
put out4;
put out;
```

```
);

*****************************************************************************************
* POSTERIOR
*****************************************************************************************
etapost(OD,K)$(PATHLINKS(OD,K,'1')<>0)=eta(OD,K)+card(NNN);
posteriorrho(OD,K)$(PATHLINKS(OD,K,'1')<>0)=rho(OD,K)+SUM(NNN,TTRUEME3(OD,K,NNN));
*****************************************************************************************

put "GammaBayes"/;
lambda.l=1;

put out4;
put "RESULTS GammaBayes"/;
        TB.l(OD,K)=thetasim(OD,K);
        ALPHAA.up(OD,K)=min(smin(MMM,TTPRIOR(OD,K,MMM)),smin(NNN,TTRUEME3(OD,K,NNN)))-
                          epsilon1;
*       GammaBayes Model refers to Equation(6.18)
        SOLVE GammaBayes USING dnlp MAXIMIZING z;
        if((GammaBayes.modelstat<>1 and GammaBayes.modelstat<>2)or GammaBayes.solvestat<>1,
                         put "GammaBayes z=",z.l:15:8," modelstat=",GammaBayes.modelstat,
                         " solvestat=", GammaBayes.solvestat" resusd=",
                         GammaBayes.resusd:12:8/;);
put out;
put "lambda.l=",lambda.l:12:4/;
TOBS(OD,K,NSCA,simulation)=ALPHAA.l(OD,K)+TB.l(OD,K)/lambda.l;

VOBSERVED(CC)=SUM((OD,K)$(PATHLINKS(OD,K,'1')<>0 and RVAL(CC)=R(OD,K)),ALPHAA.l(OD,K)+
                         TB.l(OD,K)/lambda.l);
if(PrintOption=1,
loop((OD,K)$(PATHLINKS(OD,K,'1')<>0),
put OD.tl:5:0,K.tl:3:0," e=",etapost(OD,K):8:3," TB=",TB.l(OD,K):9:3," VOBSERVED=",
                        (ALPHAA.l(OD,K)+TB.l(OD,K)/lambda.l):9:3,ALPHAA.l(OD,K):9:3/;
);
);

VOBSERVED2(CC)=VOBSERVED(CC);

put out2;
if(ord(NSCA)=nscmax+1,
);

put out;
if(PrintOption=1,
put "VOBSERVED AFTER GammaBayes"/;
loop(CC$(ord(CC)<=MAXCC),
put CC.tl:9:0,RVAL(CC):10:0,VOBSERVED(CC):9:3/;
);
```

```
);
VVOBSERVED(NSCA,CC)$(ord(CC)<=MAXCC)=VOBSERVED(CC);
VVOBSERVEDSIM(NSCA,CC,simulation)$(ord(CC)<=MAXCC)=VOBSERVED(CC);
);

if(ord(cases)=2,
SOLVE ME4 USING nlp MINIMIZING z;
if(ME4.modelstat<>2 or ME4.solvestat<>1,  put "ME4=",z.l:12:2," modelstat=",
                          ME4.modelstat," solvestat=",ME4.solvestat," resusd=",ME4.resusd/;  );
 TOBS1(OD,K,NSCA,simulation)=TKK.l(OD,K);
Val(OD,NSCA,cases,simulation)=SUM(K$(PATHLINKS(OD,K,'1')<>0),TKK.l(OD,K));
);

V.lo(OBSLINKS)=0;
V.up(OBSLINKS)=5000;

if(PrintOption=1,
put "Origin destinations flows after ME3 or ME4 cases=", cases.tl:3:0, nscanned:7:0/;
put "TKK and TTRUE00(OD)"/;
loop(OD,
put SUM(K$(PATHLINKS(OD,K,'1')<>0),TKK.l(OD,K)):8:2,TTRUE00(OD):8:2/;
);
put /;
put "Origin destinations target  flows"/;
loop(OD,
put SUM(K$(PATHLINKS(OD,K,'1')<>0),TPRIOR(OD,K)):8:2;
);
put /;
);

VVLINKS(LINK,NSCA,cases)$(ord(NSCA)=nscanned+1)=SUM((OD,K)$(PATHLINKS(OD,K,'1')<>0),
                            TKK.l(OD,K)*DELTA(LINK,K,OD));

if(PrintOption=1,
         put "CHECK   Val TTRUE00 "/;
         put "NSCA=",NSCA.tl:3:0," cases=",cases.tl:3:0," simulation=",simulation.tl:3:0/;
         loop(OD,put Val(OD,NSCA,cases,simulation):9:3,TTRUE00(OD):9:3/);
);
*******************************************************************************************

nscanned=nscanned+1;
);

T0OSIMUL(OD,cases,simulation)=TTRUE00(OD);
);
* end loop cases **************************************************************************

RMSETO(OD,K,NSCA)$(PATHLINKS(OD,K,'1')<>0)=RMSETO(OD,K,NSCA)+sqr(TOBS0(OD,K,NSCA,
                               simulation)-TTRUE(OD,K,'1'));
```

```
RMSET(OD,K,NSCA)$(PATHLINKS(OD,K,’1’)<>0)=RMSET(OD,K,NSCA)+sqr(TOBS(OD,K,NSCA,simulation)-
                               TMEAN(OD,K));
RMSET1(OD,K,NSCA)$(PATHLINKS(OD,K,’1’)<>0)=RMSET1(OD,K,NSCA)+sqr(TOBS1(OD,K,NSCA,simulation
                               )-TTRUE(OD,K,’1’));

);
* end loop simulation *****************************************************************

RMSET0(OD,K,NSCA)$(PATHLINKS(OD,K,’1’)<>0)=sqrt(RMSET0(OD,K,NSCA)/card(simulation));
RMSET(OD,K,NSCA)$(PATHLINKS(OD,K,’1’)<>0)=sqrt(RMSET(OD,K,NSCA)/card(simulation));
RMSET1(OD,K,NSCA)$(PATHLINKS(OD,K,’1’)<>0)=sqrt(RMSET1(OD,K,NSCA)/card(simulation));

put out4;
put "max RMSE/MEAN=", smax((OD,K,NSCA)$(PATHLINKS(OD,K,’1’)<>0),sqrt(RMSET(OD,K,NSCA)/
                               card(simulation))/TMEAN(OD,K)):12:5/;

MEANCC(CC)$(ord(CC)<=MAXCC)=SUM((OD,K)$(PATHLINKS(OD,K,’1’)<>0 and RVAL(CC)=R(OD,K)),
                               TMEAN(OD,K));
RMSECC(NSCA)=sqrt(SUM(CC$(ord(CC)<=MAXCC),SUM(simulation,SQR(VVOBSERVEDSIM(NSCA,CC,
                               simulation)-MEANCC(CC))))/(card(simulation)*
                               sum(CC$(ord(CC)<=MAXCC),1)));

put "SIMULATIONS"/;
put out3;
put "\begin{table}"/;
put "\centering"/;
put "\renewcommand{\tabcolsep}{3pt}"/;
put "{\tiny"/;
put "\begin{tabular}{|c|"; loop(OD,put "c";);put "|}"/;
put "\hline"/;
put " $n$"; loop(OD,put "& $t_{",ORIGIN(OD):3:0,",",DESTINATION(OD):3:0,"}$"/;);put " \\"/;
put "\hline"/;
loop(NSCA,put (ord(NSCA)-1):3:0;
mean(OD,validcases(cases))=SUM(simulation,Val(OD,NSCA,cases,simulation))/card(simulation);
RMSE(OD,validcases(cases))=sqrt(SUM(simulation,SQR(Val(OD,NSCA,cases,simulation)-
                              TOOSIMUL(OD,cases,simulation)))/card(simulation));
loop(validcases(cases),put "(",cases.tl:1:0,")";
loop(OD,if(abs(RMSE(OD,cases))>0.01,put "&", mean(OD,cases):6:1; else put "&{\bf ",
                              mean(OD,cases):6:1,"}";););  put"\\" /;
put loop(OD,if(abs(RMSE(OD,cases))<0.01,put "&{\bf ",RMSE(OD,cases):5:1,"}"; else put "&",
                              RMSE(OD,cases):5:1; );); put "\\"/;put "\hline"/;
);
);
put "\end{tabular}"/;
put "}"/;
put "\caption{\label{t501aa} Simulations.}"/;
put "\end{table}"//;

put out2;
```

```
nOD=SUM(OD,1);
put "MATHEMATICA TABLE"/;
put "SIMULATIONS={"/;
loop(NSCA,put "{",(ord(NSCA)-1):3:0;
mean(OD,validcases(cases))=SUM(simulation,Val(OD,NSCA,cases,simulation))/card(simulation);
RMSE(OD,validcases(cases))=sqrt(SUM(simulation,SQR(Val(OD,NSCA,cases,simulation)-
                              TOOSIMUL(OD,cases,simulation)))/card(simulation));
loop(validcases(cases),put ",{"; cont=0;
loop(OD,cont=cont+1;if (cont=nOD,put mean(OD,cases):7:2,"},{"; else put mean(OD,cases):7:2,
                             ",";););  put /;
cont=0; loop(OD,cont=cont+1; if(cont=nOD,put RMSE(OD,cases):7:2,"}";
                             else put RMSE(OD,cases):7:2,",";);); put /;
);if(ord(NSCA)=card(NSCA),put "}}"/; else put "},"/;);
);

put "RMSECC={";
loop(NSCA, if(ord(NSCA)=1,put RMSECC(NSCA):10:4/; else put ",",RMSECC(NSCA):10:4/;););
put "};"/;

put out;

if(validcases('a'),
put out3;
put "\begin{table}"/;
put "\centering"/;
put "\renewcommand{\tabcolsep}{3pt}"/;
put "{\scriptsize"/;
put "\begin{tabular}{|c|c|c|c|c|"; loop(SNSC(NSC),put "c";);put "|}"/;
put "\hline"/;
put "&& \multicolumn{3}{|c|}{$w_r$}&\multicolumn{",card(NSC):3:0,"}{|c|}{Scanned links}\\"/;
put "\hline"/;
put "$OD$ & $r$ & prior & true & $\hat \alpha + \hat \theta/hat \lambda$ ";
loop(SNSC(NSC),put " & ",SUM(LINK$(ord(LINK)=SCANNED(NSC)),ord(LINK)):3:0;);put " \\"/;
put "\hline"/;
aux=0;
loop(OD,loop(K$(PATHLINKS(OD,K,'1')<>0),aux=aux+1;put OD.tl:3:0," & ", aux:7:0," & ",
                                 floor(TPRIOR(OD,K)+0.5):5:0," & ",
                                 floor(TMEAN(OD,K)+0.5):5:0," & ",
                                 SUM(CC$(ord(CC)=aux),floor(VOBSERVED2(CC)+0.5)):5:0;
 loop(SNSC(NSC),put " & ";if(BITO(OD,K,NSC)=1, put "X"; else put " ";););put "\\"/;
);
put "\hline"/;
);
put "\end{tabular}"/;
put "}"/;
put "\caption{\label{t502}The set of scanned link feasible observations, observed prior and
                                 true $w_r$ flows, and associated values of $r$. The
                                 $\hat \theta$ estimates are for $n=",card(NNN):4:0,"
                                 $$m=",card(MMM):4:0,"$.}"/;
```

```
put "\end{table}"//;

put out;
);

** Simulations results
*****************************************************************************************

** Scanned link feasible observations with reconstruc path flows
*****************************************************************************************
if(cases('a'),
put out3;
put "\begin{table}"/;
put "\centering"/;
put "\renewcommand{\tabcolsep}{3pt}"/;
put "{\scriptsize"/;
put "\begin{tabular}{|c|c|c|c|c|c|"; loop(SNSC(NSC),put "c";);put "|}"/;
put "\hline"/;
put "&&\multicolumn{4}{|c|}{$w_r$}&\multicolumn{",card(NSC):3:0,"}{|c|}{Scanned links}\\"/;
put "\hline"/;
put "$OD$ & $r$ & mean prior & mean true & $f_r$& $\hat \theta$ "; loop(SNSC(NSC),put" & ",
                                 SUM(LINK$(ord(LINK)=SCANNED(NSC)), ord(LINK)):3:0;);
                                 put "\\"/;
put "\hline"/;
aux=0;
loop(OD,loop(K$(PATHLINKS(OD,K,'1')<>0),aux=aux+1;put ORIGIN(OD):4:0,"-",
                                 DESTINATION(OD):4:0," & ", aux:7:0," & ",
                                 floor(TPRIOR(OD,K)+0.5):5:0," & ",
                                 floor(TMEAN(OD,K)+0.5):5:0," & ",PATHFLOW(OD,K):5:0," & ",
                                 SUM(CC$(ord(CC)=aux),floor(VOBSERVED2(CC)+0.5)):5:0;
loop(SNSC(NSC),put " & ";if(BITO(OD,K,NSC)=1, put "X"; else put " ";););put "\\"/;
);
put "\hline"/;
);
put "\end{tabular}"/;
put "}"/;
put "\caption{\label{t502}The set of scanned link feasible observations, observed prior and
                                 true $w_r$ flows, and associated values of $r$. The
                                 $\hat \theta$ estimates are for $n=",card(NNN):4:0,"$$m=",
                                 card(MMM):4:0,"$.}"/;
put "\end{table}"//;

put out;
);

*****************************************************************************************
* DIBUJO RED CON MATHEMATICA
*****************************************************************************************
put out2;
put "Arrowpos = 0.4;"/;
put "ArrowSize = 0.025;"/;
put "FontSize1 = 12;"/;
put "Radious = 0.06;"/;
put "Points = {";
loop(I,if(ord(I)<card(I),put "{",U1(I):7:3,",",V1(I):7:3,"},"/; else put "{",U1(I):7:3,",",
                                V1(I):7:3,"}};"/;););
put /;
cont=0;
nlinks=SUM(LINK,1);
put "LinkNodes = {";

loop(LINK,cont=cont+1;if(cont<nlinks, put "{",LINKEND(LINK,'1'):3:0,",",
                                LINKEND(LINK,'2'):3:0,"},"/;  else put "{",
                                LINKEND(LINK,'1'):3:0,",",LINKEND(LINK,'2'):3:0,"}}"/;););

put "bb = NetworkPlot[Points, LinkNodes, Arrowpos, ArrowSize, FontSize1, Radious];"/;

put "Show[bb, ImageSize -> 360, Axes -> False, AxesLabel -> None, AxesStyle -> False];"/;

mm=0;

Loop(OD,
put "OriginNode=",ORIGIN(OD):5:0,";"/;
put "DestinationNode=",DESTINATION(OD):5:0,";"/;
mm=mm+1;
cont=0;
indic=0;
nlinks=card(LINK);

loop(LINK,if(cont=0,indic=1;put "Network3DAll[Points, Potentials,LinkNodes, LinkNodes1,",
                                mm:5:0,",",ORIGIN(OD):5:0,",",DESTINATION(OD):5:0,"]"/;
                                put "LinkNodes1= {";);cont=cont+1;if(cont<nlinks,
                                put"{",LINKEND(LINK,'1'):3:0,",",LINKEND(LINK,'2'):3:0,"},
                                "/;  else put "{",LINKEND(LINK,'1'):3:0,",",LINKEND(LINK,
                                '2'):3:0,"}};"/; ););

if(indic=1,put "a[",mm:5:0,"]=NetworkPlot1[Points, LinkNodes, LinkNodes1, Arrowpos,
                                ArrowSize, FontSize1, Radious];"/;
);
);

put "Show[GraphicsArray[{{a[1], a[2]}, {a[3], a[4]}}], ImageSize -> 800,Axes -> False,
                                AxesLabel -> None, AxesStyle -> False];"/;

put "TT={";
loop(NSCA,if(ord(NSCA)=1,put "{"; else put ",{"; );
         loop(validcases(cases),if(ord(cases)=1,put "{"; else put ",{";);
        loop(OD,if(ord(OD)=1,put Val(OD,NSCA,cases,'1'):6:1; else put ",",Val(OD,NSCA,
                                cases,'1'):6:1);
               );
        put "}"/;
        );
        put "}"/;
);
put "};"/;

aux=0;
put "VVLINKS={";
loop(NSCA,if(ord(NSCA)=1,put "{"; else put ",{"; );
        loop(validcases(cases),if(ord(cases)=1,put "{"; else put ",{";);
        loop(LINK,if(aux=15,put /;aux=0;);aux=aux+1;if(ord(LINK)=1,put VVLINKS(LINK,NSCA,
                                cases):6:1; else put ",",VVLINKS(LINK,NSCA,cases):6:1);
               );
        put "}"/;
        );
        put "}"/;
);
put "};"/;

put "MAXCC=",MAXCC:5:0/;
put "mmm=",card(MMM):8:0/;
put "nnn=",card(NNN):8:0/;
put "VVOBSERVED={";
loop(NSCA,if(ord(NSCA)=1,put "{"; else put ",{"; );
        loop(CC$(ord(CC)<=MAXCC),if(aux=15,put /;aux=0;);aux=aux+1;if(ord(CC)=1,put
                                VVOBSERVED(NSCA,CC):6:1; else put ",",
                                VVOBSERVED(NSCA,CC):6:1);
               );
        put "}"/;
);
put "};"/;

put "RMSET0={";
loop(NSCA,
put "{";
cont=0;
loop((OD,K)$(PATHLINKS(OD,K,'1')<>0),cont=cont+1;if(cont=1,put RMSET0(OD,K,NSCA):9:3;
                                else put ",",RMSET0(OD,K,NSCA):9:3/;););
                                if(ord(NSCA)=card(NSCA),put "}};"/; else put "},"/; );
);

put "RMSET={";
loop(NSCA,
put "{";
cont=0;
loop((OD,K)$(PATHLINKS(OD,K,'1')<>0),cont=cont+1;if(cont=1,put RMSET(OD,K,NSCA):9:3;
                                 else put ",",RMSET(OD,K,NSCA):9:3/;);); 
                                 if(ord(NSCA)=card(NSCA),put "}};"/; else put "},"/; );
);

put "RMSET1={";
loop(NSCA,
put "{";
cont=0;
loop((OD,K)$(PATHLINKS(OD,K,'1')<>0),cont=cont+1;if(cont=1,put RMSET1(OD,K,NSCA):9:3;
                                 else put ",",RMSET1(OD,K,NSCA):9:3/;););
                                 if(ord(NSCA)=card(NSCA),put "}};"/; else put "},"/; );
);

put "RouteMean={";
cont=0;
loop((OD,K)$(PATHLINKS(OD,K,'1')<>0),cont=cont+1;if(cont=1,put TMEAN(OD,K):9:3;
                                 else put ",",TMEAN(OD,K):9:3/;););
put "};"/;

put out;
*****************************************************************************************
* END DIBUJO RED CON MATHEMATICA
*****************************************************************************************

put "END OF PROCESS"/;
putclose out;putclose out1;putclose out2;putclose out3; putclose out4;
```

10.2 Code for the FIFO rule consistent dynamic model

This section deals with the Matlab implementation of the FIFO rule consistent dynamic model which has been exposed in chapter 7.

In order to clarify the way of working of this program, Figure 10.1 shows the diagram of the procedure dependence.

10.2.1 Main Program

```
clc
clear all
clear
close all

options = optimset('UseParallel','always');

global linksplotted;
global hroutesplotted;
global nodesplotted;
global nPath;
```

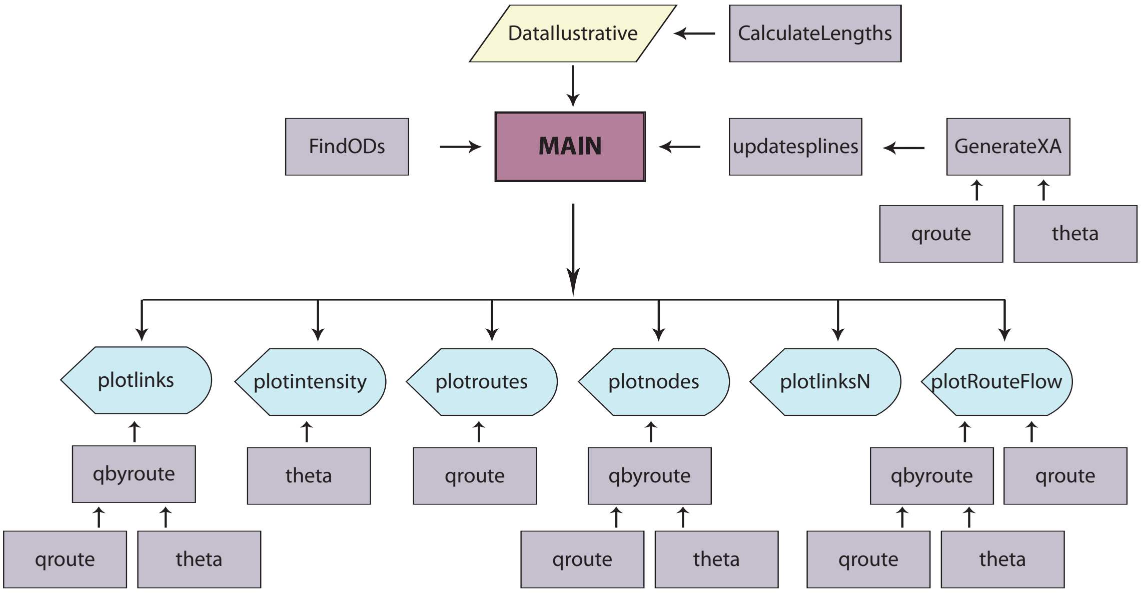

Figure 10.1: Diagram of the procedure dependence in the code for the FIFO rule consistent dynamic model

```
global colors;
global links;
global m;
global t;
global nnodes;
global nlinks;
global pathsLinks;
global nmaxlinks;
global alpha;
global fbeta;
global XAmax;
global plotoption;
global printoption;
global rho;
global linkroutes;
global nlinkroutes;
global nroutelinks;
global tsize1;
global tsize;
global nfigures;
global MaxIi;
global Ii;
global saturationmax;
global linksatmax;
global pp;
global routesplotted;
global ttt;
global TA1;
global error;
global maxtime;
global correctcontrol;
global S;
global satur;
global tmax;
global increment;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% ProjectNumber=1 : Ilustrative
% ProjectNumber=2 : Nguyen Dupuis
% ProjectNumber=3 : Cuenca
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
ProjectNumber=1;
nfigures=0;
tiempo0=clock;
routesplotted=[];
linksplotted=[];
hroutesplotted=[];
nodesplotted=[];

% Fixing the seed value to repeat the same random numbers
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
rand('seed',12345678);

% Data iterations and print
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
maxtimepermitted=5000;
nmaxnoderoutes=10;
nmaxlinkroutes=100;
nmaxnodelinks=5;
printoption=0;
plotoption=0;
itermax=100;
tol=0.1;
rho=1.0;    % relax coefficient
m=2;    % number of waves of the initial flow

% Data
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
switch ProjectNumber
    case 1
        DataIlustrative;
    case 2
        DataNguyenDupuis;
    case 3
        DataCuenca219;
    otherwise
end

t=0:increment:tmax;
tsize=length(t);
tsize1=tsize-1;

nlinks = length(links(:,1));
satur=zeros(nlinks,length(t));

% Generate random colors
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
colors=zeros(max(nPath,nlinks),3);
for i=1:max(nPath,nlinks)
    colors(i,:)=[rand rand rand];
end

[NI,nmaxlinks]=size(pathsLinks);
nroutelinks=zeros(1,nPath);     % number of links per route
for r=1:nPath
    for j=1:nmaxlinks
        if pathsLinks(r,j)>0
            nroutelinks(r)=nroutelinks(r)+1;
        end
    end
end

[nPathOD,OD,nOD]=FindODs(links,pathsLinks);
% OD: the set of OD-pair
% nOD: index of each OD-pair
% nPathOD: per each path indicates the OD index
nnodes=max(max(links));

% prints table of route links
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for r=1:nPath
    fprintf(’%3.0f ’,r);
    for j=1:nmaxlinks
        if pathsLinks(r,j)>0
            fprintf(’& %3.0f ’,pathsLinks(r,j));
        else
            fprintf(’&  ’);
        end
    end
    fprintf(’\\\\\n’);
end

% Calculates the output links of each node (outputnodelinks)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
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 initialization
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
ttt=zeros(nlinks,tsize);
TA1=zeros(nlinks,tsize);
for i=1:tsize
    TA1(:,i)=t(i);  % link enter time
end
for i=1:nlinks
    ttt(i,:)=t(:)+alpha(i);  % initial link exit time associated to TA1
end

ppp=pchip(t,t);
pp=struct(ppp);
for ll=1:nlinks
    % initial spline (exit time, enter time)
    pp(ll)=pchip(ttt(ll,:),TA1(ll,:));
end

% Starts iterative program
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
S=zeros(nlinks);
for ll=1:nlinks
    S(ll)=alpha(ll)*(1+fbeta(ll))/XAmax(ll);
end
error=100*tol;
iter=0;
maxtime=maxtimepermitted/2;
done=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));
        if plotoption==1
            plotintensity;
        end
    else
        fprintf('The proposed flow blocks the network with maxtime=%9.2f exceeding
        maxtimepermitted=%9.2f\n',maxtime,maxtimepermitted);
        done=1;
    end
end

if done==0
    tiempo1=clock;
    fprintf('\n Tiempo de cpu :  %0.5g segundos\n',time(tiempo1,tiempo0));
    plotroutes(routesplotted); % flow at the origin of each path
    plotintensity; % route start time with the link exit time of each path
    plotlinks(linksplotted); % traffic flow intensity at the exit each link
    plotnodes(nodesplotted); % flow intensity at each node with the time
    plotlinksN(linksplotted); % link travel time with the link enter time
    plotRouteFlow; % Link flow evolution
    plotNumberUsers;
    fprintf(' maximum travel time = %9.3f hours\n',maxtime);
end
```

10.2.2 Data reading procedure

We show the illustrative example data only, because the other examples follow the same scheme.

```
% DataIlustrative

ProjectName='Ilustrative';
links = int8([
    1 2
    1 5
    1 6
    1 3
    2 5
    6 5
    3 6
    5 4
    6 4
    ]);

nlinks = length(links(:,1));

delta=ones(1,nlinks);
pathsLinks = int8([
    1 5 8 0
    2 8 0 0
    3 9 0 0
    3 6 8 0
    4 7 9 0
    4 7 6 8
    5 8 0 0
    7 6 8 0
    7 9 0 0
    ]);

Points =100*[2.00 4.00; 0.00 3.00; 4.00 3.00;2.00 0.00; 1.00 2.00; 3.00 2.00];

tmax=32;
increment=0.2;
optionhcurves=1;
routesplotted=[1 2 3 4 5 6 7 8 9];
linksplotted=[1 2 3 4 5 6 7 8 9];
```

```
hroutesplotted=[1; 2; 3; 4; 5; 6; 7; 8; 9];
nodesplotted=[1 2 3 4 5 6];
hfunctionsplotted=[1 2 3 4 5 6 7 8 9];

selectedroute=6;
samax=ones(1,nlinks)*100;   % speed
[nPath NI]=size(pathsLinks);
m=2;
Nroutes=zeros(nPath,m);     % total daily flow rate
for i=1:nPath
    for j=1:m
        Nroutes(i,j)=1*(rand(1,1)*800+3200);
    end
end
CalculateLengths;
fbeta=ones(1,nlinks);
gamma=2;
for i=1:nlinks
    % number of vehicles on link i leading to a travel time alpha(1+fbeta)
    XAmax(i)=linklength(i)*100;
end
XAmax(6)=linklength(6)*25; % link 6 has 25% of XAmax for the other links

muroutes=ones(nPath,m);
for r=1:nPath
    for i=1:m
        % peak hour around 7:30 and 14:00 hours
        muroutes(r,i)=7.5+rand(1,1)+(i-1)*(6.5+rand(1,1));
        sigmaroutes(r,i)=rand(1,1)*1+1.5;
    end
end
```

10.2.3 Splines updating (updatesplines)

```
function updatesplines
% Update the splines

global pp;
global nlinks;
global TA1;
global ttt;
global error;
global maxtime;

GenerateXA; % link exit time of each link (ttt)
error=0;
for ll=1:nlinks
    coeff0=pp(ll).coefs;
    % spline (exit time, enter time)
    pp(ll)=pchip(ttt(ll,:),TA1(ll,:));
    error=error+max(max(abs(coeff0-pp(ll).coefs)));
end
maxtime=max(max(max(ttt)));
```

10.2.4 Calculation of the link exit time (GenerateXA)

```
function GenerateXA
% Calculates the link exit time of each link (ttt)

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;

saturationmax=0;
XA=zeros(nlinks,tsize);
E=zeros(nlinks,length(t));

correctcontrol=1;
Numbercorrections=0;
for kk=1:length(t)
    for ll=1:nlinks
        tkk=ttt(ll,kk);     % link exit time
        tkk1=TA1(ll,kk);    % link enter time associated to tkk
        E0=0;
        XA0=0;
        sat=0;
        for p=1:nPath
            found=0;
            j=1;
            while j<=nroutelinks(p) && found==0
                a=pathsLinks(p,j);
                if a==ll
                    % starting time of a user who reaches the end of link
                    % of a route p at time tkk
                    tt0=theta(ll,p,tkk);
                    % starting time of a user who reaches the end of link
                    % of a route p at time tkk1
                    tt1=theta(ll,p,tkk1);
                    %cumulative flow at origin node of the path p at tt0
                    [NI,c0]=qroute(p,tt0,1);
                    %cumulative flow at origin node of the path p at tt1
                    [NI,c1]=qroute(p,tt1,1);
                    % cumulative flow at the link exit by the time ttk
                    E0=E0+c0;
                    % number of vehicles inside the link
                    XA0=XA0+c0-c1;
                    found=1;
                    if j<nroutelinks(p)
                        % saturation of upstream links
                        sat=max(sat,satur(pathsLinks(p,j+1),kk));
                    end
                end
                j=j+1;
            end
        end
        E(ll,kk)=E0;
        XA(ll,kk)=XA0;
        aux2=XA0/XAmax(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(ll)*(1+fbeta(ll)*aux2^gamma+delta(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
         tkk=tkk1+tau; % updates the link exit time
         if kk>1 && (ttt(ll,kk-1)+Q>tkk)
             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(ll,kk-1),tkk,Q);
                 correctcontrol=0;
             end
             % updates the link exit time with the queue dissipation term
```

```
            tkk=ttt(ll,kk-1)+Q;
        end
        if kk>1 && (Q<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(ll,kk-1),tkk,Q);
        end
        ttt(ll,kk)=rho*tkk+(1-rho)*ttt(ll,kk);
    end
end
if Numbercorrections>0
    fprintf('Numbercorrections=%3.0f \n',Numbercorrections);
end


for kkk=2:length(t)
    for ll=1:nlinks
        if ttt(ll,kkk)<TA1(ll,kkk)
            fprintf('ttt(ll,kkk)=%8.3f TA1(ll,kkk)=%8.3f\n',ttt(ll,kkk),TA1(ll,kkk));
        end
    end
end
```

10.2.5 Calculation of the departure path time (theta)

```
function tt1=theta(a,p,tt)
% returns the starting time of a user who reaches the end of link of a route p at time 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(ll),tt1);
    end
end
```

10.2.6 Calculation of the inflow rate at the origin of path (qroute)

```
function [z,cumflow]=qroute(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
```

```
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(route,i)*normpdf(tt,muroutes(route,i),sigmaroutes(route,i));
    end
    if option==1 || option>2
        cumflow=cumflow+Nroutes(route,i)*(normcdf(tt,muroutes(route,i),
        sigmaroutes(route,i)));
    end
end
```

10.2.7 Calculation of the flow by route (qbyroute)

```
function [z,totalcumflow]=qbyroute(l,tt,r1,rr,option)
% Calculates the flows of link l due to routes r1 to rr

global pathsLinks;
global nroutelinks;

eps=0.001;
z=0;
totalcumflow=0;
for r=r1:rr
    for j=1:nroutelinks(r)
        if pathsLinks(r,j)==l
            % starting time of a user who reaches the end of link l
            % of the route r at time tt
            hp=theta(l,r,tt);
            [aux,cumflow]=qroute(r,hp,option);
            if option>1
                h1p=(theta(l,r,tt+eps)-hp)/eps;
                z=z+aux*h1p;
            end
            if option==1 || option>2
                totalcumflow=totalcumflow+cumflow;
            end
        end
    end
end
```

10.2.8 Graphic representation

Starting time with the link exit time (plotintensity)

```
function plotintensity
% Plot the route start time with the link exit time of each path

global t;
global TA1;
global ttt;
global nroutelinks;
global nfigures;
global ProjectName;
global pathsLinks;
global colors;
global hfunctionsplotted;

ymax=max(max(TA1));
xmax=max(max(max(ttt)));
nnn=3;
[mm,nn]=size(hfunctionsplotted);
ymax=ymax*1.1;

for iii=1:mm
    nfigures=nfigures+1;
    figure('Name',strcat('hfunction',num2str(nfigures)));
    for i=1:nn
        r=hfunctionsplotted(iii,i);
        if r>0
            x=min(t(1),min(min(min(ttt)))):0.2:max(max(max(ttt)));
            y=zeros(1,length(x));
            subplot(ceil(nn/nnn),nnn,i);
            plot(x,x,'LineWidth',2);
            grid on;
            hold on;
            axis('equal');
             axis([t(1) ymax t(1) xmax]);
            for j=nroutelinks(r):-1:1;
                ll=pathsLinks(r,j);
                for ii=1:length(x)
                    % starting time of a user who reaches the end of link
                    % ll of a route r at time x
                    y(ii)=theta(ll,r,x(ii));
                end
                % link exit time of each link (ttt)
                xx(1:length(t))=ttt(ll,:);
                    plot(y,x,'LineWidth',2,'Color',[.2 0.4 0]);
                    vvv=[x(1) y  x(length(x))];
                    uuu=[x(1)  x x(length(x))];
                    fill(vvv,uuu,colors(ll,:));
            end
            text(20,0.1*max(xx),['Path ' num2str(r)]);
            xlabel('\theta_a^p(t)');
            ylabel('t','Rotation',0);
        end
        hold off;
        print('-deps2c','-r600',sprintf('%s_%s.eps',ProjectName,get(gcf,'Name')))
    end
end
```

Flow at the origin of each path (plotroutes)

```
function plotroutes(routesplotted)
% Plot the flow at the origin of each path with the time

global nfigures;
global ProjectName;
global colors;

[mm,nn]=size(routesplotted);
x=0:0.2:24;
maxx=max(x);
y=x;
maxyy=0;

for ii=1:mm
    for i=1:nn
        r=routesplotted(ii,i);
        if r>0
            for j=1:length(x)
                % flow intensity at origin node
                y(j) = qroute(r,x(j),2);
            end
            maxy=max(y);
            maxyy=max(maxy,maxyy);
        end
    end
end
maxyy=1.1*maxyy;
for ii=1:mm
    nfigures=nfigures+1;
    figure('Name',strcat('paths',num2str(nfigures)));
    for i=1:nn
        r=routesplotted(ii,i);
        if r>0
            for j=1:length(x)
                % flow intensity at origin node
                y(j) = qroute(r,x(j),2);
            end
            subplot(ceil(nn/2),2,i);
            plot(x,y,'LineWidth',2.0);
            fill(x,y,colors(r,:));
            text(maxx-7,0.8*maxyy,['Path ' num2str(r)]);
            axis([0 maxx 0 maxyy]);
            xlabel('t');
            ylabel('h_p(t)','Rotation',0);
            hold on;
        end
    end
    hold off;
    print('-deps2c','-r600',sprintf('%s_%s.eps',ProjectName,get(gcf,'Name')))
end
```

Flow intensity at the exit each link (plotlinks)

```
function plotlinks(linksplotted)
% Plot traffic flow intensity at the exit each link

global nPath;
global pathsLinks;
global t;
global nroutelinks;
global nfigures;
global ProjectName;
global colors;
global routesplotted;

nnn=2;
[mm,nn]=size(linksplotted);
x=t;
maxx=max(x);
y=x;
ymax=0;

for ii=1:mm
    for i=1:nn
        l=linksplotted(ii,i);
        if l>0
            toplot=0;
            for rr=nPath:-1:1
                jj=1;
                while jj<=nroutelinks(rr) && toplot==0
                    if pathsLinks(rr,jj)==l
                        toplot=1;
                    end
                    jj=jj+1;
                end
                if toplot==1
```

```
                    for j=1:length(x)
                        % flows of link l due to routes 1 to rr at time x
                        [y(j),NI] = qbyroute(l,x(j),1,rr,2);
                    end
                    ymax=max(ymax,max(y));
                end
            end
        end
    end
end

for ii=1:mm
    nfigures=nfigures+1;
    figure('Name',strcat('Flowexitinglinks',num2str(nfigures)));
    for i=1:nn
        l=linksplotted(ii,i);
        if l>0
            subplot(ceil((nn+1)/nnn),nnn,i);
            grid on;
            hold on;
            toplot=0;
            for rr=nPath:-1:1
                jj=1;
                while jj<=nroutelinks(rr) && toplot==0
                    if pathsLinks(rr,jj)==l
                        toplot=1;
                    end
                    jj=jj+1;
                end
                if toplot==1
                    for j=1:length(x)
                         % flows of link l due to routes 1 to rr at time x
                        [y(j),NI] = qbyroute(l,x(j),1,rr,2);
                    end
                    plot(x,y);
                    uuu=[x(1)  x x(length(x))];
                    vvv=[0 y  0];
                    fill(uuu,vvv,colors(rr,:));
                    hold on;
                    axis([t(1) maxx 0 ymax]);
                    xlabel('t');
                    ylabel('g_a(t)','Rotation',0);
                end
            end
            text(maxx-3,0.8*ymax,['Link ' num2str(l)]);
            hold on;
        end
    end
    mom=length(routesplotted);
    beta=3.5;
    ll=1;
    ym=2.5*ll;
    xm=max(beta*ym,ll*(3*mom+1)/2);
    ss=(xm-mom*ll)/(mom+1);
    subplot(ceil((nn+1)/nnn),nnn,nn+1);
    xxx=ss;
    for jjj=1:mom
        rectangle('Position',[xxx ll/2 ll ll],'FaceColor',
        colors(routesplotted(jjj),:));
        if mom>7
            text(xxx+ll/2,2*ll,['R ' num2str(routesplotted(jjj))],
            'HorizontalAlignment','center','FontSize',6);
        else
            text(xxx+ll/2,2*ll,['Route ' num2str(routesplotted(jjj))],
            'HorizontalAlignment','center','FontSize',6);
        end
        xxx=xxx+ss+ll;
    end
    hold on;
    axis([t(1) xm 0 ym]);
    axis('equal');
    axis('off');
    hold off;
    print('-deps2c','-r600',sprintf('%s_%s.eps',ProjectName,get(gcf,'Name')))
end
```

Flow intensity at each node (plotnodes)

```
function plotnodes(nodesplotted)
% Plot flow intensity at each node with the time

global links;
global nPath;
global pathsLinks;
global t;
global nroutelinks;
global nfigures;
global MaxIi;
global Ii;
global ProjectName;
global colors;
global routesplotted;

nnn=2;
[mm,nn]=size(nodesplotted);
x=min(min(min(t))):0.2:max(max(max(t)));
maxx=max(x);
minx=min(x);
```

```
y=x;
u=zeros(1,length(x));
ymax=0;
for ii=1:mm
    for i=1:nn
        for j=1:length(x)
            y(j)=0;
        end
        node=nodesplotted(ii,i);
        if node>0
            for kk=1:MaxIi(node)
                l=Ii(node,kk);
                for j=1:length(x)
                    % flows of link l due to all routes
                    y(j)=y(j)+ qbyroute(l,x(j),1,nPath,2);
                end
                ymax=max(ymax,max(y));
            end
            for rr=nPath:-1:1
                jj=1;
                toplot=0;
                while jj<=nroutelinks(rr) && toplot==0
                    l=pathsLinks(rr,jj);
                    if links(l,2)==node
                        toplot=1;
                    end
                    jj=jj+1;
                end
                if toplot==1
                    for j=1:length(x)
                        % flows of link l due to each route
                        y(j)=qbyroute(l,x(j),rr,rr,2);
                    end
                end
            end
        end
    end
end

for ii=1:mm
    nfigures=nfigures+1;
    figure('Name',strcat('Flowenteringnodes',num2str(nfigures)));
    for i=1:nn
        for j=1:length(x)
            y(j)=0;
        end
        node=nodesplotted(ii,i);
        if node>0
            subplot(ceil((nn+1)/nnn),nnn,i);
            for kk=1:MaxIi(node)
                l=Ii(node,kk);
                for j=1:length(x)
                    % flows of link l due to all routes
                    y(j)=y(j)+ qbyroute(l,x(j),1,nPath,2);
                end
                u=y;
            end
            plot(x,u);
            grid on;
            hold on;
            axis([minx maxx 0 ymax]);
            done=0;
            for rr=nPath:-1:1
                 jj=1;
                toplot=0;
                while jj<=nroutelinks(rr) && toplot==0
                    l=pathsLinks(rr,jj);
                    if links(l,2)==node
                        toplot=1;
                    end
                    jj=jj+1;
                end
                if toplot==1
                    for j=1:length(x)
                        % flows of link l due to each route
                        y(j)=qbyroute(l,x(j),rr,rr,2);
                    end
                    uuu=[x(1)  x x(length(x))];
                    vvv=[0 u  0];
                    if done==0
                        fill(uuu,vvv,colors(rr,:));
                        hold on;
                        done=1;
                        u(:)=u(:)-y(:);
                    else
                        plot(x,u);
                        fill(uuu,vvv,colors(rr,:));
                        hold on;
                        u(:)=u(:)-y(:);
                    end
                end
            end
            text(maxx-3,0.8*ymax,[’Node ’ num2str(node)]);
            xlabel(’t’);
            ylabel(’r_n(t)’,’Rotation’,0);
            hold on;
        end
    end
```

```
    mom=length(routesplotted);
    beta=3.5;
    ll=1;
    ym=2.5*ll;
    xm=max(beta*ym,ll*(3*mom+1)/2);
    ss=(xm-mom*ll)/(mom+1);
    subplot(ceil((nn+1)/nnn),nnn,nn+1);
    xxx=ss;
    for jjj=1:mom
        rectangle('Position',[xxx ll/2 ll ll],'FaceColor',
        colors(routesplotted(jjj),:));
        if mom>7
            text(xxx+ll/2,2*ll,['R ' num2str(routesplotted(jjj))],
            'HorizontalAlignment','center','FontSize',6);
        else
            text(xxx+ll/2,2*ll,['Route ' num2str(routesplotted(jjj))],
            'HorizontalAlignment','center','FontSize',6);
        end
        xxx=xxx+ss+ll;
    end
    hold on;
    axis([0 xm 0 ym]);
    axis('equal');
    axis('off');
    hold off;
    print('-deps2c','-r600',sprintf('%s_%s.eps',ProjectName,get(gcf,'Name')))
end
```

Link travel time (plotlinksN)

```
function plotlinksN(linksplotted)
% For each link, plot the link travel time with the link enter time

global t;
global nfigures;
global ProjectName;
global colors;
global ttt;
global pp;
global maxtime;

nnn=2;
[mm,nn]=size(linksplotted);
x=min(t):0.2:max(t);
y=x;
maxx=max(max(x),maxtime);
ymax=0;
for ii=1:mm
```

```
    for i=1:nn
        l=linksplotted(ii,i);
        if l>0
            x=ttt(l,1):0.2:ttt(l,length(ttt(l,:)));
            y=x;
            for iii=1:length(x)
                y(iii)=x(iii)-ppval(pp(l),x(iii));
            end
            ymax=max(ymax,max(y));
        end
    end
end

ymax=1.1*ymax;
for ii=1:mm
    nfigures=nfigures+1;
    figure('Name',strcat('Flowexitinglinks',num2str(nfigures)));
    for i=1:nn
        l=linksplotted(ii,i);
        if l>0
            subplot(ceil((nn+1)/nnn),nnn,i);
            grid on;
            hold on;
            x=ttt(l,1):0.2:ttt(l,length(ttt(l,:)));
            y=x;
            for iii=1:length(x)
                y(iii)=x(iii)-ppval(pp(l),x(iii));
            end
            plot(x,y,'LineWidth',2.0);
            uuu=[x(1)  x x(length(x))];
            vvv=[0 y  0];
            fill(uuu,vvv,colors(l,:));
            text(maxx-3,0.95*ymax,['Link ' num2str(l)]);
            axis([t(1) maxx 0 ymax]);
            xlabel('t');
            ylabel('D_a(t)','Rotation',0);
            hold on;
        end
    end
    mom=length(linksplotted);
    beta=3.5;
    ll=1;
    ym=2.5*ll;
    xm=max(beta*ym,ll*(3*mom+1)/2);
    ss=(xm-mom*ll)/(mom+1);
    subplot(ceil((nn+1)/nnn),nnn,nn+1);
    xxx=ss;
    for jjj=1:mom
        rectangle('Position',[xxx ll/2 ll ll],'FaceColor',colors(linksplotted(jjj),:));
        if mom>7
            text(xxx+ll/2,2*ll,['L 'num2str(linksplotted(jjj))],'HorizontalAlignment',
            'center','FontSize',6);
        else
            text(xxx+ll/2,2*ll,['Link ' num2str(linksplotted(jjj))],
            'HorizontalAlignment','center','FontSize',6);
        end
        xxx=xxx+ss+ll;
    end
    hold on;
    axis([0 xm 0 ym]);
    axis('equal');
    axis('off');
    hold off;
    print('-deps2c','-r600',sprintf('%s_%s.eps',ProjectName,get(gcf,'Name')))
end
```

Link flow evolution (plotRouteFlow)

```
function plotRouteFlow
% Route flow at origin and corresponding link flow evolution

global ProjectName;
global selectedroute;
global colors;
global pathsLinks;
global nroutelinks;
global nfigures;
global maxtime;

xmax=maxtime;
ymax=0;
rr=selectedroute;
x=0:0.2:xmax;
y=zeros(1,length(x));
for i=1:length(x)
    % flow intensity at origin node
    y(i)=qroute(rr,x(i),2);
end
ymax=max(ymax,max(y));
for j=1:nroutelinks(rr)
    l=pathsLinks(rr,j);
    if l>0
        for i=1:length(x)
            % flows of link l due to the last route at time x
            y(i)=qbyroute(l,x(i),rr,rr,0);
        end
    end
end
```

```
nfigures=nfigures+1;
figure('Name',strcat('RouteFlowEvolution',num2str(nfigures)));
subplot(nroutelinks(rr)+1,1,1);
for i=1:length(x)
    % flow intensity at origin node
    y(i)=qroute(rr,x(i),2);
end
plot(x,y,'LineWidth',2.0);
fill(x,y,colors(rr,:));
xmax=max(xmax,max(x));
text(1,0.8*ymax,['Path ' num2str(rr)]);
xlabel('t');
ylabel('h_p(t)','Rotation',0);
grid on;
hold on;

for j=1:nroutelinks(rr)
    l=pathsLinks(rr,j);
    if l>0
        for i=1:length(x)
            % flows of link l due to the last route at time x
            y(i)=qbyroute(l,x(i),rr,rr,2);
        end
        ymax=max(ymax,max(y));
    end
end

for j=1:nroutelinks(rr)
    l=pathsLinks(rr,j);
    if l>0
        for i=1:length(x)
            % flows of link l due to the last route at time x
            y(i)=qbyroute(l,x(i),rr,rr,2);
        end
        subplot(nroutelinks(rr)+1,1,j+1);
        plot(x,y,'LineWidth',2.0);
        fill(x,y,colors(rr,:));
        ymax=max(ymax,max(y));
        text(1,0.8*ymax,['Link ' num2str(l)]);
        xlabel('t');
        ylabel('g_a(t)','Rotation',0);
        grid on;
        hold on;
    end
end
ymax=ceil(ymax/100)*110;
axis([0 xmax 0 ymax]);
print('-deps2c','-r600',sprintf('%s_%s.eps',ProjectName,get(gcf,'Name')))
```

10.2.9 Other procedures

Finding the OD-pair (FindODs)

```
function [nPathOD,OD,nOD]=FindODs(links,pathsLinks)
% OD: the set of OD-pair
% nOD: index of each OD-pair
% nPathOD: per each path indicates the OD index

[nPath nmaxj]=size(pathsLinks);
PathOD=zeros(nPath,2);
nPathOD=zeros(nPath,1);
nOD=0;
OD=[];
for i=1:nPath
    if pathsLinks(i,1)>0
        PathOD(i,1)=links(pathsLinks(i,1),1);
        PathOD(i,2)=links(pathsLinks(i,1),2);
        for j=2:nmaxj
            if pathsLinks(i,j)>0
                PathOD(i,2)=links(pathsLinks(i,j),2);
            end
        end
    end
    done=0;
    j=1;
    while j<=i-1 && done==0
        if PathOD(i,1)==PathOD(j,1) && PathOD(i,2)==PathOD(j,2)
            done=1;
            nPathOD(i)=nPathOD(j);
        end
      j=j+1;
    end
    if j==i && done==0
       nOD=nOD+1;
       nPathOD(i)=nOD;
       OD=[OD; PathOD(i,:)];
    end
end
```

Calculation of the link length y free-flow speed (CalculateLengths)

```
function CalculateLengths
% Calculates the link length and the free flow link travel time

global linklength;
global links;
global nlinks;
global Points;
global samax;
global alpha;

linklength=zeros(1,nlinks);
for i=1:nlinks
    linklength(i)=sqrt((Points(links(i,1),1)-Points(links(i,2),1))^2+
    (Points(links(i,1),2)-Points(links(i,2),2))^2);
    if linklength(i)==0
        fprintf('Link %4.0f has length %8.3f  %5.0f -- %5.0f\n',i,linklength(i),
        links(i,1),links(i,2));
    end
    alpha(i)=linklength(i)/samax(i);   % free flow link time
end
```

10.3 Code for the G. Beta-Gaussian Bayesian network model

This section deals with the Matlab implementation of the Generalized Beta-Gaussian Bayesian network model which has been exposed in chapter 8.

10.3.1 Main Program

```
clc
clear all
clear
close all

global ProjectName;
global ProjectNumber;
global StationNames;
global todasfechas;
global DATA;
global fecha;
global nstations;
global ndays;
global ntimes;
global simulationoption;
global nfigures;
global increment;
global TimesCDF;
global selectedlinks;
global dataplot;
global t;
global Removethreshold;
global Datapreprocessed;
global selectedsimulation1;
global nstationLatexTable;
global MarkovOrder;

%General data
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
ProjectName='Vermont';
ProjectNumber='1';
dataplot=0;
nfigures=0;
Removethreshold=40;
t=0:23;
selectedlinks=[1,2,3,4];
TimesCDF=[10:1:16];
increment=1;
Datapreprocessed=0;
MarkovOrder=3;
simulationoption=0;
selectedsimulation1=1;
nstationLatexTable=3;

% Reads data from file
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if Datapreprocessed==1
    fid1=fopen('DATOSVERMONT1.dat','r');
    fprintf('Preprocessing data\n');
    nstations=84;
    ndays=365;
    ntimes=24;

    % Allocates memory for variables
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    StationNames=cell(1,nstations);
    todasfechas=cell(1,ndays);
    DATA=zeros(nstations,ndays,ntimes);
    fecha=cell(nstations,ndays);

    % Initializes dates to 'missing'
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    for i=1:nstations
        for j=1:ndays
            fecha{i,j}='missing';
        end
    end
    nstations=0;
    ntodasfechas=0;

    % Reads line by line
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    tline = fgetl(fid1);
    while ischar(tline)
```

```
% Reads one line of data
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
CC = textscan(tline, '%s %s %d %d %d %d %d %d %d %d %d %d %d %d %d %d %d %d %d %d
                     %d %d %d %d %d %d ');

% identifies a change in station
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if strcmp(CC{1},'Station')
    stat=CC{2};

    % Looks for station name in StationNames list
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    nstation=findStationNumber(StationNames,stat);

    % If station name is not in list adds it to the list
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    if nstation==0
        nstations=nstations+1;
        StationNames(nstations)=stat;
        nstation=nstations;
    end
else

    % Concatenates date and day of week
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    date=strcat(CC{1},CC{2});

    % Identifies day of week
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    if strcmp(CC{2},'Mon')
        col=1;
    elseif strcmp(CC{2},'Tue')
        col=2;
    elseif strcmp(CC{2},'Wed')
        col=3;
    elseif strcmp(CC{2},'Thu')
        col=4;
    elseif strcmp(CC{2},'Fri')
        col=5;
    elseif strcmp(CC{2},'Sat')
        col=6;
    elseif strcmp(CC{2},'Sun')
        col=7;
    end

    % If Mon throughout Thu stores data
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    if col<5
```

```
                % Locates date in list of dates
                %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
                nfecha=findStationNumber(todasfechas,date);

                % If not in list adds it to the list
                %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
                if nfecha==0
                    ntodasfechas=ntodasfechas+1;
                    todasfechas(ntodasfechas)=date;
                    nfecha=ntodasfechas;
                end

                % Reads day data
                %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
                for k=1:ntimes
                    DATA(nstation,nfecha,k)=CC{k+2};
                end
                fecha(nstation,nfecha)=date;
            end
        end

        % Reads a new line of data
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        tline = fgetl(fid1);
    end

    % Closes file
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    fclose(fid1);

    % Stores actual values of nstations and ndays
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    nstations=nstation;
    ndays=ntodasfechas;
    fprintf('After initial filtering (Mon-Thu), we have registered %5.0f stations and
            %5.0f days\n',nstations,ndays);

    % Stores Matlab data variables
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    save([ProjectName 'StationNames.mat'],'StationNames');
    save([ProjectName 'todasfechas.mat'],'todasfechas');
    save([ProjectName 'DATA.mat'],'DATA');
    save([ProjectName 'fecha.mat'],'fecha');
    save([ProjectName 'mixture.mat'],'nstations','ndays','ntimes');
else

    % reads Matlab data variables
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    load([ProjectName 'StationNames.mat'],'StationNames');
    load([ProjectName 'todasfechas.mat'],'todasfechas');
    load([ProjectName 'DATA.mat'],'DATA');
    load([ProjectName 'fecha.mat'],'fecha');
    load([ProjectName 'mixture.mat'],'nstations','ndays','ntimes');
end
tt=1:24;
aux=zeros(1,24);
for nstation=1:nstations
    nfigures=nfigures+1;
    figure('Name',strcat('predictions',num2str(nfigures)));
    for simul=1:ntimes
        aux(:)=DATA(nstation,simul,:);
        plot(tt,aux,'-b','LineWidth',0.001);
        hold on;
    end
    grid on;
    hold off;
end
GenerateRandomColors(nstations);

% Detects and removes outliers
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Removeoutliers;

% Transforms initial data to generalized beta and standard multivariate normal variables
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
TransformstoGBetaandNormal

% Calculates mean matrix
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
CalculatesMeans

% Make predictions
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
MakePredictions;
```

10.3.2 Removing outliers procedure (Removeoutliers)

```
function Removeoutliers
% Detects and removes outliers

global ProjectName;
global StationNames;
global DATA;
global fecha;
global nstations;
global ndays;
global ntimes;
global dataplot;
global nfigures;
global t;
global Removethreshold;

% Calculates means and standard deviations of each station-hour pair
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
mean=zeros(nstations,ntimes);
sigma=zeros(nstations,ntimes);
for i=1:nstations
    for k=1:ntimes
        ndata=0;
        for j=1:ndays
            if ~strcmp(fecha{i,j},'missing')
                ndata=ndata+1;
                mean(i,k)=mean(i,k)+DATA(i,j,k);
            end
        end
        mean(i,k)=mean(i,k)/ndata;
        for j=1:ndays
            if ~strcmp(fecha{i,j},'missing')
                sigma(i,k)=sigma(i,k)+power(DATA(i,j,k)-mean(i,k),2);
            end
        end
        sigma(i,k)=sqrt(sigma(i,k)/ndata);
    end
end

% Marks outlier data with 'miss' date
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
error=zeros(1,nstations);
u=zeros(1,ntimes);
v=zeros(1,ntimes);
for i=1:nstations
    for j=1:ndays
        if ~strcmp(fecha{i,j},'missing')
            error(i)=0;
            u(:)= DATA(i,j,:);
            for k=1:24
                v(k)=abs((u(k)-mean(i,k))/sigma(i,k));
                error(i)=error(i)+v(k);
            end
            error1=max(v);
            if error(i)>Removethreshold || error1>4
                fecha{i,j}='miss';
                fprintf('eliminates station %5.0f and day %5.0f: %s\n',i,j,fecha{i,j});
            end
        end
    end
```

```
end

% Calculates new means and standard deviations of each station-hour pair
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
mean1=zeros(nstations,ntimes);
for i=1:nstations
    for k=1:ntimes
        ndata=0;
        for j=1:ndays
            if ~strcmp(fecha{i,j},’missing’) && ~strcmp(fecha{i,j},’miss’)
                ndata=ndata+1;
                mean1(i,k)=mean1(i,k)+DATA(i,j,k);
            end
        end
        mean1(i,k)=mean1(i,k)/ndata;
    end
end

% Removes outliers
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for i=1:nstations
    for j=1:ndays
        if strcmp(fecha{i,j},’miss’)
            fecha{i,j}=’missing’;
        end
    end
end

% Plots data and removes outliers
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if dataplot==1
    for i=1:nstations
        nfigures=nfigures+1;
        figure(’Name’,strcat(’Data’,num2str(nfigures)));
        minu=999999;
        maxu=0;
        for j=1:ndays
            if ~strcmp(fecha{i,j},’missing’)
                u(:)= DATA(i,j,:);
                if ~strcmp(fecha{i,j},’miss’)
                    plot(t,u,’LineWidth’,4,’color’,[105 200 92]/255);
                else
                    plot(t,u,’--’,’LineWidth’,4,’color’,’red’);
                end
                minu=min(minu,min(u));
                maxu=max(maxu,max(u));
                hold on;
            end
        end
        plot(t,mean(i,:),'LineWidth',4,'color','blue');
        plot(t,mean1(i,:),'LineWidth',2,'color','yellow');
        text(11,0.95*minu+0.05*maxu,strcat(StationNames(i),'-','Station'),'FontSize',18);
        print('-deps2c','-r600',sprintf('%s_%s.eps',ProjectName,get(gcf,'Name')));
        hold off;
    end
end
save([ProjectName 'fecha.mat'],'fecha');
```

10.3.3 Transformation to GBeta & $N(\mu, \sigma)$ (TransformstoGBetaandNormal)

```
function TransformstoGBetaandNormal
% Transforms initial data to generalized beta and standard multivariate normal variables

global ProjectName;
global DATA;
global fecha;
global nstations;
global ndays;
global ntimes;
global nfigures;
global increment;
global Nsimulations;
global TimesCDF;
global selectedlinks;
global DATANormal;
global dmin;
global dmax;
global alpha;
global beta;
global indices;
global Datapreprocessed;

if Datapreprocessed==1
    TimesCDF1=TimesCDF;

    % Alocates memory for variables
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    alpha=zeros(nstations,ntimes);
    beta=zeros(nstations,ntimes);
    dmin=zeros(nstations,ntimes);
    dmax=zeros(nstations,ntimes);
    indices=ones(nstations,ntimes);
    val1=zeros(1,ndays);
    DATANormal=zeros(nstations,ndays,ntimes);

    for i=1:ndays
        proba1(i)=i/(ndays+1);
```

```
end

val2=zeros(1,ndays);
for nstation=1:nstations
    for time=1:ntimes
        val2=zeros(1,ndays);
        ndays1=0;
        for day=1:ndays
            if ~strcmp(fecha(nstation,day),'missing')
                ndays1=ndays1+1;
                val2(ndays1)=DATA(nstation,day,time);
            end
        end
        val=zeros(1,ndays1);
        val(:)=val2(1:ndays1);

        % Calculates minimum and maximum values
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        valmin=min(val);
        valmax=max(val);

        % Corrects constant day data
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        if valmax-valmin<0.05
            indices(nstation,time)=0;
        end

        % If small or null range small range values are simulated
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        if abs((valmin-valmax)/valmax)<0.00001
            val=val+valmax/100000*rand(Nsimulations,1);

            % Updates minimum and maximum values
            %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
            valmin=min(val);
            valmax=max(val);
        end
        val1=val;

        % Extends ranges in both directions
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        dmin(nstation,time)=1.001*valmin-0.001*valmax;
        dmax(nstation,time)=1.05*valmax-0.05*valmin;

        % Reduces variable to [0,1] interval
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        val=(val-dmin(nstation,time))/(dmax(nstation,time)-dmin(nstation,time));

        % Estimates beta paramaters based on val data
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
param=betafit(val);

% Stores beta parameters
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
alpha(nstation,time)=param(1);
beta(nstation,time)=param(2);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% Plots observations and generalized beta model cdf
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
doplot=0;
for i=1:length(TimesCDF1)
    if time==TimesCDF1(i)
        doplot=1;
    end
end
doplot1=0;
for i=1:length(selectedlinks)
    if nstation==selectedlinks(i)
        doplot1=1;
    end
end
doplot=doplot*doplot1;
if doplot==1
    nfigures=nfigures+1;
    figure('Name',strcat('CDF',num2str(nfigures)));
    val1=sort(val1);
    proba1=zeros(ndays1,1);
    for i=1:ndays1
        proba1(i)=i/(ndays1+1);
    end
    plot(val1, proba1,'o');
    proba=zeros(ndays1,1);
    for ii=1:ndays1
        val1(ii)=dmin(nstation,time)+(dmax(nstation,time)-dmin(nstation,time))*
                 proba1(ii);
        proba(ii)=betacdf(proba1(ii),param(1),param(2));
    end
    hold on;
    plot(val1, proba,'-','LineWidth',4);
    text(0.02*dmax(nstation,time)+0.98*dmin(nstation,time),0.9,['Link '
         num2str(nstation) '  Time ' num2str(time*increment)]);
    hold off
    print('-deps2c','-r600',sprintf('%s_%s.eps',ProjectName,get(gcf,'Name')));
end

% Transform beta values to standard normal values
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
            if max(val)>1 || min(val)<0
                PrintMatrix('val1',val,'%8.3f')
            end
            for iii=1:length(val)
                aaaa=betacdf(val(iii),param(1),param(2));
                val(iii)=Norminv1(aaaa,0,1);
            end
            DATANormal(nstation,1:length(val),time)=val(:);
        end
    end

    % Stores matlab variables
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    save([ProjectName 'DATANormal.mat'],'DATANormal');
    save([ProjectName 'dmin.mat'],'dmin');
    save([ProjectName 'dmax.mat'],'dmax');
    save([ProjectName 'alpha.mat'],'alpha');
    save([ProjectName 'beta.mat'],'beta');
    save([ProjectName 'indices.mat'],'indices');
else

    % Stores matlab variables
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    load([ProjectName 'DATANormal.mat'],'DATANormal');
    load([ProjectName 'dmin.mat'],'dmin');
    load([ProjectName 'dmax.mat'],'dmax');
    load([ProjectName 'alpha.mat'],'alpha');
    load([ProjectName 'beta.mat'],'beta');
    load([ProjectName 'indices.mat'],'indices');
end
```

Calculation of the normal cdf (Norminv1)

```
function u=Norminv1(x,mu,sigma)
% Calculates normal inverse cumulative distribution function, but correcting the extremes

if x>0.999940941
    u=mu+3.85*sigma;
elseif x<5.9059e-005
    u=mu-3.85*sigma;
else
    u=norminv(x,mu,sigma);
end
```

10.3.4 Calculation of mean (CalculatesMeans)

```
function CalculatesMeans
% Calculates the mean matrix and allocates in memory
```

```
global ProjectName;
global DATANormal;
global fecha;
global nstations;
global ndays;
global ntimes;

% Allocates memory for variables
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Means=zeros(nstations,ntimes);

% Calculates mean matrix
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for nstation=1:nstations
    daysavailable=0;
    for nday=1:ndays
        if ~strcmp(fecha(nstation,nday),’missing’)
            daysavailable=daysavailable+1;
            for time=1:ntimes
                Means(nstation,time)=Means(nstation,time)+DATANormal(nstation,nday,time);
            end
        end
    end
    Means(nstation,:)=Means(nstation,:)/daysavailable;
end

save([ProjectName ’Means.mat’],’Means’);
```

10.3.5 Making predictions procedure (MakePredictions)

```
function MakePredictions

global ProjectName;
global increment;
global plotoption;
global simulationoption;
global timevaluesLatexTable;
global varLatexTableData;
global nstationLatexTable;
global t1;
global MarkovOrder;
global selectedsimulation1;
global selectedsimulation;
global n1;
global n2;
global nnn;
global maxlink;
global minlink;
global covar1;
global actualvaluenormal;
global corre;
global predictedvalue;
global std;
global SampleSize;
global PredictionsErrors;
global std1;
global indexZZ;
global mean;
global Means;
global nstations;
global ndays;
global ntimes;

% Memory allocation for data in Latex Table
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
varLatexTableData=zeros(100,3,length(timevaluesLatexTable));

% Reads indices to avoid singular matrix
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
load([ProjectName,'indices.mat'],'indices');

% Reads Project Data
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
ReadData;

% Initializes variables and allocates memory for them
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
PredictionsErrors=zeros(ndays,nstations,ntimes);
SampleSize=zeros(ntimes,1);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% n1=Number of evidential variables
% n2=number of predicted variables
% nnn=n1+n2 total number of variables involved
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
n1=nstations*MarkovOrder;
n2=ntimes-t1;
nnn=n1+n2;

% Clock initialization
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
tiempo00=clock;

% Defines range of the simulation for cycle
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if simulationoption==1
    daymin=1;
    daymax=ndays;
else
```

```
    daymin=selectedsimulation1;
    daymax=selectedsimulation1;
end

% Allocates memory for variables
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
std1=zeros(1,n2);

% Performs simulation for cycle
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for selectedsimulation=daymin:daymax

    % Evaluates cpu time
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    tiempo1=clock;
    fprintf('Simulation =%5.0f cputime=%12.5g segundos\n',selectedsimulation,etime
           (tiempo1,tiempo00));

    % Performs nstation cycle
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    for nstation=1:nstations
        tiempo0=clock;

        % Calculates mean and covar matrices
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        CalculateMeanVar(nstation);
        if nstationLatexTable==nstation && selectedsimulation==selectedsimulation1

            % Plots means for nstation and all times
            %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
            PlotCurve(mean,strcat('mean',num2str(nstation)),0,0);

            % Select optimal variables to predict nstation value at given time
            % predictedtime
            %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
            OptimalVarSelection;
            tiempo1=clock;
            fprintf('\n Tiempo de cpu:%0.5g segundos nstation=%6.0f\n',etime(tiempo1,
                    tiempo0),nstation);
        end
        predictedtime=n1+1;

        % Calculates row of correlation matrix associated with column predictedtime
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        CalculatesRowCorrelMatrix(predictedtime);

        % Plots correlation coefficient for variable predictedtime
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        if nstation==nstationLatexTable && selectedsimulation==selectedsimulation1
```

```
            PlotCurve(corre,strcat('corre',num2str(nstation)),1,0.2);
        end

        % Performs initial selection of variables
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        InitialVarSelection;

        % Calculates mean and covariance matrix given the evidence Z
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        OptimalVarSelection;
        indexY=n1+1:nnn;
        indexZ=indexZZ;
        UpdateMeanVar(indexY,indexZ);

        % Calculates semiamplitude confidence interval
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        for iii=1:n2
            std1(iii)=2*sqrt(covar1(iii,iii));
        end
        std=[0 std1];

        % Plots generalized Beta densities
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        PlotGBeta(nstation);

        % Accumulates prediction errors
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        aux=Transformtobeta([actualvaluenormal(1,1) predictedvalue'],nstation,t1);
        actualvalue=Transformtobeta(actualvaluenormal,nstation,t1);
        for sss=1:n2+1
            time=t1+sss-1;
            PredictionsErrors(selectedsimulation,nstation,time)=(aux(sss)-actualvalue(sss))
                                                                /Means(nstation,time);
            SampleSize(time)=SampleSize(time)+1;
        end

        % Plots real and predicted flows with 0.95-confidence bands
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        if plotoption==1
            PlotRealPredicted(nstation,actualvalue,aux);
        end
    end

        % Calculates and Prints RMSE tables
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    CalculateandPrintRMSE;

end
close all;
```

```
% Plots flow data used in the statistical analysis
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if plotoption==1
    minlink=ones(nstations,1)*999999;
    maxlink=zeros(nstations,1);

    for nstation=1:nstations
        if nstation==1
            for jj=1:ntimes
                y(jj)=(jj-1)*increment;
            end
        end
        % Plot the observed Generalized Beta variables
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        PlotData(nstation,y);
    end
end
```

Data reading procedure (ReadData)

```
function ReadData
% Read the initial values and write on screen a summary

global ProjectNumber;
global ProjectName;
global TimesCDF;
global selectedlinks;
global t1;
global MarkovOrder;
global selectedsimulation1;
global timevaluesLatexTable;
global linkvarLatexTable;
global Missingdata;
global Nsimulations;
global nlinks;
global nPath;
global tmax;
global increment;
global optionhcurves;
global routesplotted;
global linksplotted;
global hroutesplotted;
global nodesplotted;
global hfunctionsplotted;

switch ProjectNumber
    case 1
        DataVermont;
        ProjectName='Vermont';
    case 2
        DataNguyenDupuis;
        ProjectName='NguyenDupuis';
    case 3
        DataCuenca219;
        ProjectName='Cuenca219';
    otherwise
end

fprintf('******************************************************\n');
fprintf('ProjectName= %s \n',ProjectName);
fprintf('******************************************************\n');
fprintf('MarkovOrder= %4.0f \n',MarkovOrder);
fprintf('Nsimulations= %4.0f \n',Nsimulations);
fprintf('linkvarLatexTable= %4.0f \n',linkvarLatexTable);
fprintf('selectedsimulation1= %4.0f \n',selectedsimulation1);
fprintf('t1= %4.0f \n',t1);
fprintf('nlinks= %4.0f \n',nlinks);
fprintf('nPath= %4.0f \n',nPath);
PrintMatrix('selectedlinks',selectedlinks,'%4.0f');
PrintMatrix('TimesCDF',TimesCDF,'%4.0f');
PrintMatrix('timevaluesLatexTable',timevaluesLatexTable,'%4.0f');
fprintf('tmax= %6.0f \n',tmax);
fprintf('increment= %8.2f \n',increment);
fprintf('optionhcurves= %4.0f \n',optionhcurves);
PrintMatrix('routesplotted',routesplotted,'%4.0f');
PrintMatrix('linksplotted',linksplotted,'%4.0f');
PrintMatrix('hroutesplotted',hroutesplotted','%4.0f');
PrintMatrix('nodesplotted',nodesplotted,'%4.0f');
PrintMatrix('hfunctionsplotted',hfunctionsplotted,'%4.0f');
PrintMatrix('Missingdata',Missingdata,'%4.0f');
fprintf('******************************************************\n');
```

Calculation of mean and covariance (CalculateMeanVar)

```
function CalculateMeanVar(link)
% Calculates mean and covar matrices

global ProjectName;
global nlinks;
global hh;
global t1;
global Nsimulations;
global MarkovOrder;
global n1;
global nnn;
global maxlink;
global minlink;
global covar;
global mean;
global Z0;
global Z;
global selectedsimulation;
global meanindices;

% Opens file
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
filename=[ProjectName,'NORMAL.dat'];
fid=fopen(filename,'r');

% Reads heading data
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
nlinks=fread(fid,1,'integer*4');
hh=fread(fid,1,'integer*4');
Nsimulations=fread(fid,1,'integer*4');
A=zeros(nlinks,hh);

% Stores the link numbers of flow time data used
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
meanindices=[];
for ii=MarkovOrder:-1:1
    meanindices=[meanindices 1:nlinks];
end
for ii=1:hh-t1+1
    meanindices=[meanindices link];
end

% Calculates means
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
mean=zeros(1,nnn);

for j=1:Nsimulations
    for ii=1:nlinks
        A(ii,:)=fread(fid,hh,'real*4');
    end
    for ii=MarkovOrder:-1:1
        low=(ii-1)*nlinks+1;
        up=ii*nlinks;
        mean(low:up)=mean(low:up)+A(1:nlinks,t1-ii+1)';
    end
    low=n1+1;
    up=n1+hh-t1;
    mean(low:up)=mean(low:up)+A(link,t1+1:hh);

    % Updates minima and maxima
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    maxlink(link)=max([maxlink(link) max(A(link,:))]);
    minlink(link)=min([minlink(link) min(A(link,:))]);
end
mean=mean/Nsimulations;

% Rewinds file
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
frewind(fid);

% Reads heading data
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
nlinks=fread(fid,1,'integer*4');
hh=fread(fid,1,'integer*4');
Nsimulations=fread(fid,1,'integer*4');

% Calculates covariance matrix
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
 B=zeros(1,nnn);
covar=zeros(nnn,nnn);
for j=1:Nsimulations
    for ii=1:nlinks
        A(ii,:)=fread(fid,hh,'real*4');
    end

    % Real flow data and evidences must be transformed to normal N(0,1)
    % data
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    for ii=MarkovOrder:-1:1
        low=(ii-1)*nlinks+1;
        up=ii*nlinks;
        B(low:up)=A(1:nlinks,t1-ii+1)';
    end
    low=n1+1;
    up=n1+hh-t1;
    B(low:up)=A(link,t1+1:hh);

    if j==selectedsimulation
        Z0= A(link,t1:hh)';
        Z=B(1:n1)';
    end
    aaaux=B-mean;
    covar=covar+aaaux'*aaaux;
end
covar=covar/Nsimulations;

fclose(fid);
```

Selection of optimal variables (OptimalVarSelection)

```
function OptimalVarSelection
```

```
% Select optimal variables to predict link value at given time tcenter

global stdcenterthreshold;
global timevaluesLatexTable;
global t1;
global n1;
global covar;
global cofcorr;
global index;
global stdcenter;
global covar1;
global tcenter;
global indexZZ;

for tttt=1:length(timevaluesLatexTable)
    tcenter=n1+timevaluesLatexTable(tttt)-t1;

    % Calculates row of correlation matrix
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    CalculatesRowCorrelMatrix(tcenter);

    % Calculates mean and covariance matrix given the evidence Z
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    InitialVarSelection;

    indexZ=[];
    varZ=[covar(tcenter,tcenter)];
    done=0;
    stdcenter=sqrt(covar(tcenter,tcenter));
    while done==0 && length(cofcorr)>2 && stdcenter>stdcenterthreshold
        [cofcorr ibb]=sort(cofcorr,'descend');
        index=index(ibb);
        if cofcorr(1)<0.05
            done=1;
        else
            indexZ=[indexZ index(1)];
            index=index(2:end);
            indexY=[tcenter index];
            UpdateMeanVar(indexY,indexZ);
            varZ=[varZ covar1(1,1)];
            stdcenter=sqrt(covar1(1,1));
            aauu=diag(covar1);
            if min(aauu)<0.000001
                done=1;
                indexZ=indexZ(1:length(indexZ)-1);
                indexY=[indexY indexZ(end)];
            else
                cofcorr=abs(covar1(1,:)/sqrt(aauu(1)));
                cofcorr=cofcorr./sqrt(aauu');
                cofcorr=cofcorr(2:end);
            end
        end
    end
    % Stores (link, time, variance) to be printed
    interpret1(indexZ,tttt,varZ);
end
% Writes a latex table: Order in which variables are selected for different prediction
%times
dovarLatexTable;
indexZZ=indexZ;
```

Calculation correlation matrix (CalculatesRowCorrelMatrix)

```
function CalculatesRowCorrelMatrix(tcenter)
% Calculates row of correlation matrix associated with column tcenter

global covar;
global corre;

aux=1./diag(covar);
corre=abs(covar(tcenter,:))*sqrt(aux(tcenter));
corre=corre.*sqrt(aux');
```

Selection of the initial variables (InitialVarSelection)

```
function InitialVarSelection
%Among the no-missing data, only selects variables with a correlation bigger than 0.10

global nlinks;
global cofcorr;
global indices;
global Missingdata;
global MarkovOrder;
global corre;
global index;
global indexgood;
global t1;

nmissing=length(Missingdata);
index=[];
cofcorr=[];
for mm=1:nlinks
    if indices(mm,t1)==1
        insert=1;
        for rrr=1:nmissing
            if mm==Missingdata(rrr)
                insert=0;
            end
        end
        if insert==1
            for ii=MarkovOrder:-1:1
                jj=(ii-1)*nlinks+mm;
                if corre(jj)>0.10
                    index=[index jj];
                    cofcorr=[cofcorr corre(jj)];
                end
            end
        end
    end
end
indexgood=index;
```

Updating mean and covariance procedure (UpdateMeanVar)

```
function UpdateMeanVar(indexY,indexZ)
%Updates the mean and covariance matrix

global covar;
global Z;
global EYZ;
global covar1;
global mean;

EY=mean(indexY);
EZ=mean(indexZ);
SYY=covar(indexY,indexY);
SYZ=covar(indexY,indexZ);
SZZ=covar(indexZ,indexZ);
SZZ1=inv(SZZ);
EYZ=EY'+SYZ*SZZ1*(Z(indexZ)-EZ');
covar1=SYY-SYZ*SZZ1*SYZ';
```

Transformation to G. Beta (Transformtobeta)

```
function x=Transformtobeta(y,nstation,torigin)
% Transforms the normal variables to Generalized Beta

global dmin;
global dmax;
global alpha;
global beta;
global muT;
global sigmaT;
global criticalnormal;

n=length(y);
for i=1:n
```

```
    dmin1=dmin(nstation,torigin+i-1);
    dmax1=dmax(nstation,torigin+i-1);
    alpha1=alpha(nstation,torigin+i-1);
    beta1=beta(nstation,torigin+i-1);
    muT1=muT(nstation,torigin+i-1);
    sigmaT1=sigmaT(nstation,torigin+i-1);
    y1=sigmaT1*y(i)+muT1;
    if y1<-criticalnormal
        aux=normcdf(-criticalnormal,0,1);
    elseif y1>criticalnormal
        aux=normcdf(criticalnormal,0,1);
    else
        aux=normcdf(y1,0,1);
    end
    aux1=betainv(aux,alpha1,beta1);
    x(i)=dmin1+(dmax1-dmin1)*aux1;
end
```

Calculation of RMSE (CalculateandPrintRMSE)

```
function CalculateandPrintRMSE
% Calculates and Prints RMSE tables

global nlinks;
global increment;
global hh;
global t1;
global Nsimulations;
global selectedsimulation;
global SampleSize;
global PredictionsErrors;

ppp=t1:1:24;
PredictionRMSE=zeros(nlinks,hh);

for i=1:nlinks
    for j=1:hh
        for selectedsimulation=1:Nsimulations
            PredictionRMSE(i,j)=PredictionRMSE(i,j)+PredictionsErrors(selectedsimulation,i,
                                j)^2;
        end
    end
end
for j=1:hh
    if SampleSize(j)>0
        PredictionRMSE(:,j)=sqrt(PredictionRMSE(:,j)/SampleSize(j));
    end
end
qqq=PredictionRMSE(:,ppp);
```

```
LatexPrint(ppp*increment,qqq,t1*increment);
```

10.3.6 Example of data file (DataVermont)

We show the Vermont-State example data.

```
function DataVermont

global ProjectName;
global optionhcurves;
global linksplotted;
global hfunctionsplotted;
global nlinks;
global fbeta;
global delta;
global tmax;
global increment;
global TimesCDF;
global selectedlinks;
global t1;
global MarkovOrder;
global selectedsimulation1;
global timevaluesLatexTable;
global linkvarLatexTable;
global Missingdata;
global Nsimulations;
global nfigures;
global dataplot;
global readsDataVermont;

% readsDataVermont=1;
ProjectName='Vermont';
nstations=84;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% FACTOR used to divide data values
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
FACTOR=1;
selectedlinks=[1,2,3,4];
TimesCDF=[10:1:16];
Nsimulations=16;
tmax=23;
optionhcurves=1;
linksplotted=[1 2 3 4 5 6 7 8 9];
hfunctionsplotted=[1 2 3 4 5 6 7 8 9];
increment=1;
timeincrement=increment;
t=0:23;
x=min(t):timeincrement:max(t);
u=x;
```

```
if readsDataVermont==1
    DATA=zeros(nstations,24,10);
    StationData=zeros(1,nstations);
    StationNames=cell(size(StationData));

    % Opens file to write data
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    filename=[ProjectName,'.dat'];
    fid=fopen(filename,'w+');

    % Opens file to read data

    fid1=fopen('DATOSVERMONT1.dat','r');

    % Reads heading data
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    tline = fgetl(fid1);
    nstation=1;
    store=0;
    j=0;
    nstations=0;
    while ischar(tline)
        disp(tline);
        CC = textscan(tline, '%s %s %d %d %d %d %d %d %d %d %d %d %d %d %d %d %d %d %d %d
                              %d %d %d %d %d %d ');
        if strcmp(CC{1},'Station')
            stat=CC{2};
            if store==1
                StationData(nstation)=j;
            end
            store=1;
            nstation=findStationNumber(StationNames,stat);
            if nstation==0
                nstations=nstations+1;
                StationNames(nstations)=stat;
                nstation=nstations;
            end
            j=StationData(nstation);
        else
            if strcmp(CC{2},'Mon')
                col=1;
            elseif strcmp(CC{2},'Tue')
                col=2;
            elseif strcmp(CC{2},'Wed')
                col=3;
            elseif strcmp(CC{2},'Thu')
                col=4;
            elseif strcmp(CC{2},'Fri')
                col=5;
            elseif strcmp(CC{2},'Sat')
                col=6;
            elseif strcmp(CC{2},'Sun')
                col=7;
            end

            % Elliminates Fri, Sat and Sun
            %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
            if col<5
                j=j+1;
                for k=1:24
                    DATA(nstation,k,j)=CC{k+2}/FACTOR;
                    u(k)= DATA(nstation,k,j);
                end
            end
        end
        tline = fgetl(fid1);
    end
    StationData(nstation)=j;

    % Filtering data
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    mean=zeros(nstations,24);
    sigma=zeros(nstations,24);
    for i=1:nstations
        for k=1:24
            for j=1:StationData(i)
                mean(i,k)=mean(i,k)+DATA(i,k,j);
            end
            mean(i,k)=mean(i,k)/StationData(i);
            for j=1:StationData(i)
                sigma(i,k)=sigma(i,k)+power(DATA(i,k,j)-mean(i,k),2);
            end
            sigma(i,k)=sqrt(sigma(i,k)/StationData(i));
        end
    end

    % Plots data and calculates ndata, the number of valid data in each station
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    valido=zeros(nstations,max(StationData));

    ndata=zeros(1,nstations);
    error=zeros(1,nstations);
    incompletestations=[];
    incompletedays=[];
    for i=1:nstations
        if dataplot==1
            nfigures=nfigures+1;
            figure('Name',strcat('Data',num2str(nfigures)));
        end
        minu=min(min(DATA(i,:,1:StationData(i))));
        maxu=max(max(DATA(i,:,1:StationData(i))));
        for j=1:StationData(i)
            error(i)=0;
            for k=1:24
                u(k)= DATA(i,k,j);
                error(i)=error(i)+abs((u(k)-mean(i,k))/sigma(i,k));
            end
            if error(i)<40
                valido(i,j)=1;
                ndata(i)=ndata(i)+1;
                if dataplot==1
                    plot(t,u,'LineWidth',4,'color',[105 200 92]/255);
                end
            else
                incompletestations=[incompletestations i];
                incompletedays=[incompletedays j];
                fprintf('eliminates data %5.0f of station %5.0f\n',j,i);
                if dataplot==1
                    plot(t,u,'--','LineWidth',4,'color','red');
                end
            end
            hold on;
        end
        if dataplot==1
            plot(t,mean(i,:),'LineWidth',4,'color','blue');
            text(11,0.95*minu+0.05*maxu,strcat(StationNames(i),'-','Station'),'FontSize',
                 18);
            print('-deps2c','-r600',sprintf('%s_%s.eps',ProjectName,get(gcf,'Name')));
            hold off;
        end
    end
    sort(ndata)
    incompletestations=sort(union(incompletestations,incompletestations));
    incompletedays=sort(union(incompletedays,incompletedays));
    PrintMatrix('incompletestations',incompletestations,'%4.0f');
    PrintMatrix('incompletedays',incompletedays,'%4.0f');

    close all;

    [nstations,ntimes,Nsimulations]=size(DATA);
    tt=1:24;
    for nstation=1:floor(nstations/10):nstations
        nfigures=nfigures+1;
        figure('Name',strcat('predictions',num2str(nfigures)));
        for simul=1:Nsimulations
            if valido(nstation,simul)
                plot(tt,DATA(nstation,:,simul),'-b','LineWidth',4);
            else
                plot(tt,DATA(nstation,:,simul),'-r','LineWidth',4);
            end
            hold on;
        end
        grid on;
        hold off;
    end

    Nsimulationsmin=100;
    nlinks=0;
    for i=1:nstation
        if ndata(i)>=Nsimulationsmin
            nlinks=nlinks+1;
        end
    end

    % Generate random colors
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    GenerateRandomColors(nlinks)
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

    delta=ones(1,nlinks);
    fbeta=ones(1,nlinks);

    for i=1:nstation
        fprintf('i=%6.0f sum=%8.0f\n',i,sum(valido(i,:)));
    end
    save([ProjectName,'valido.mat'],'valido');

    % Calculates Means
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    MeanAll=zeros(1,nstations*24);
    k=0;
    for nstation=1:nstations
        for time=1:24
            k=k+1;
            ndata1=0;
            for simul=1:Nsimulations
                if valido(nstation,simul)
                    ndata1=ndata1+1;
                    MeanAll(k)=MeanAll(k)+DATA(nstation,time,simul);
                end
            end
            MeanAll(k)=MeanAll(k)/ndata1;
        end
    end
    save([ProjectName,'MeanAll.mat'],'MeanAll');
    h=1;
```

```
% Calculates covar matrix
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
CovarAll=zeros(nstations*24,nstations*24);
mdata=zeros(nstations*24,nstations*24);
k1=0;
for nstation1=1:nstations
    fprintf('nstation1=%5.0f of %5.0f\n',nstation1,nstations);
    for time1=1:24
        k1=k1+1;
        k2=0;
        for nstation2=1:nstations
            for time2=1:24
                k2=k2+1;
                if k1>=k2
                    for simul=1:Nsimulations
                        if valido(nstation1,simul) && valido(nstation2,simul)
                            mdata(k1,k2)=mdata(k1,k2)+1;
                            CovarAll(k1,k2)=CovarAll(k1,k2)+(DATA(nstation1,time1,
                                     simul)-MeanAll(k1))*(DATA(nstation2,time2,simul)-
                                     MeanAll(k2));
                        end
                    end
                end
            end
        end
    end
end
CovarAll=CovarAll./mdata;
for k1=1:nstations*24-1
    for k2=k1+1:nstations*24
        CovarAll(k1,k2)=CovarAll(k2,k1);
    end
end
eig(CovarAll);
save([ProjectName,'CovarAll.mat'],'CovarAll');
save([ProjectName,'DATACovarAll.mat'],'DATA');

% Initializes Means
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
meanvalues=zeros(nstations,24);

% Writes heading data
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
fwrite(fid,nlinks,'integer*4');
fwrite(fid,length(x),'integer*4');
fwrite(fid,Nsimulations,'integer*4');
fprintf('Stored format\n');
countnlinks=ones(1,nstation);
```

```
    % Calculates mean values
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    for j=1:Nsimulations
        ii=0;
        for i=1:nstations
            if ndata(i)>=Nsimulationsmin
                while valido(i,countnlinks(i))==0
                    countnlinks(i)=countnlinks(i)+1;
                end
                ii=ii+1;
                for k=1:24
                    fwrite(fid,DATA(i,k,countnlinks(i)),'real*4');
                end
                meanvalues(ii,:)=meanvalues(ii,:)+DATA(i,:,countnlinks(i));
                countnlinks(i)=countnlinks(i)+1;
            end
        end
    end
    meanvalues=meanvalues/Nsimulations;

    % Saves mean data
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    filename8=[ProjectName,'Means.mat'];
    save(filename8,'meanvalues');
    PrintMatrix('valido',valido,'%2.0f');

    % Prints station names
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    fprintf('Station Names:\n');
    for i=1:nstation
        if ndata(i)>=Nsimulations
            fprintf('%7.0f %s counts=%8.0f\n',i,StationNames{i},countnlinks(i));
        end
    end

    fclose(fid);
    fclose(fid1);
end

% Point where prediction is to be done (multiplied by increment provides
% the corresponding time)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
t1=3;

% Number of previous time increments from which prediction is to be based
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
MarkovOrder=3;
```

```
% Number of simulation used to predict flows (reference case)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
selectedsimulation1=1;

% Prediction time points for which the optimal selection of variables is done
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
timevaluesLatexTable=[16,18,20,22,24];

% Link for which the optimal selection of variables is done
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
linkvarLatexTable=1;

% Links that are missing
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Missingdata=[];
```

10.3.7 Graphic representation

General data (PlotCurve)

```
function PlotCurve(data,Name1,option,threshold)
% Plots the given data when they are bigger than a threshold

global ProjectName;
global nfigures;
global meanindices;

nfigures=nfigures+1;
figure('Name',Name1);
plot(1:length(data),data,'-o');
hold on
if option==1
    for i=1:length(data)
        if data(i)>threshold
            text(i,1.03*data(i),num2str(meanindices(i)));
        end
    end
end
hold off;
print('-deps2c','-r600',sprintf('%s_%s.eps',ProjectName,get(gcf,'Name')));
```

Generalized Beta densities (PlotGBeta)

```
function PlotGBeta(link)
% Plots generalized Beta densities

global ProjectName;
global increment;
global dmin;
global dmax;
global alpha;
global beta;
global nfigures;
global TimesCDF;
global t1;
global Z0;
global colors;
global EYZ;
global std;
global selectedlinks;
global plotoption;
global Missingdata;

doplot=0;
for i=1:length(selectedlinks)
    if link==selectedlinks(i)
        doplot=1;
    end
end
minx=99999999;
maxx=0;
maxy=0;
if doplot==1 && plotoption==1
    nfigures=nfigures+1;
    if isempty(Missingdata)
        figure('Name',strcat('GBeta',num2str(link)));
    else
        figure('Name',strcat('GBetaM',num2str(link)));
    end
    mus=[Z0(1,1) EYZ'];
    stds=std/2;

    % Allocates memory
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    xborderline=zeros(length(TimesCDF),1);
    yborderline=zeros(length(TimesCDF),1);
    aborderline=zeros(length(TimesCDF),1);
    bborderline=zeros(length(TimesCDF),1);

    for i=1:length(TimesCDF)
        time=TimesCDF(i);
        tt=time-t1+1;
        a=dmin(link,t1+tt-1);
        b=dmax(link,t1+tt-1);
        alpha1=alpha(link,t1+tt-1);
        beta1=beta(link,t1+tt-1);
        mu=mus(tt);
        sigma=stds(tt);
        xx=a+(b-a)/10000:(b-a)/5000:b;
        yy=zeros(1,length(xx));
        for jjj=1:length(xx)
            % Calculates the Generalized Beta densities given the normal distributions
            yy(jjj)=varchange(a,b,alpha1,beta1,mu,sigma,xx(jjj));
        end
        yy1=yy>0.01;
        yy=yy(yy1);
        xx=xx(yy1);
        minx=min([minx min(xx)]);
        maxx=max([maxx max(xx)]);
        maxy=max([maxy max(yy)]);
        zz=ones(length(xx),1)*TimesCDF(i)*increment;
        plot3(zz,xx,yy,'-','LineWidth',3,'Color',colors(i,:))
        grid on
        hold on;
        [aaa,bbb]=max(yy);
        yborderline(i)=aaa;
        xborderline(i)=xx(bbb);
        aborderline(i)=xx(1);
        bborderline(i)=xx(end);
    end
    hold on;

    view([-45 15]);
    hold on;
    xyz=[TimesCDF'*increment xborderline yborderline]';
    fnplt(cscvn(xyz),'black',4)
    xyz=[TimesCDF'*increment xborderline yborderline*0]';
    fnplt(cscvn(xyz),'black',2,'--')
    xyz=[TimesCDF'*increment aborderline yborderline*0]';
    fnplt(cscvn(xyz),'black',2)
    xyz=[TimesCDF'*increment bborderline yborderline*0]';
    fnplt(cscvn(xyz),'black',2)
    axis([TimesCDF(1)*increment TimesCDF(end)*increment minx maxx 0 1.06*maxy]);
    xlabel('Time');
    ylabel(strcat('Link ',' ',num2str(link),' travel time'));
    zlabel('Density f(x)');
    hold on;
    print('-deps2c','-r600',sprintf('%s_%s.eps',ProjectName,get(gcf,'Name')));
    hold off
end
```

Real and predicted flows (PlotRealPredicted)

```
function PlotRealPredicted(link,actualvalue,aux)
% Plots real and predicted flows with 0.95-confidence bands

global ProjectName;
global nfigures;
global t1;
global predictedvalue;
global std;
global std1;
global increment;
global n2;
global Missingdata;

y=zeros(n2+1,1);
for jj=1:n2+1
    y(jj)=(t1-1+jj)*increment;
end

nfigures=nfigures+1;
if isempty(Missingdata)
    figure('Name',strcat('Real&Predicted',num2str(nfigures)));
else
    figure('Name',strcat('Real&PredictedM',num2str(nfigures)))
end
plot(y, aux,'-b','LineWidth',6);
hold on;
auxlo=Transformtobeta([actualvalue(1,1) predictedvalue']-std,link,t1);
plot(y,auxlo,'--b','LineWidth',2);
hold on;
auxup= Transformtobeta([actualvalue(1,1) predictedvalue']+std,link,t1);
plot(y,auxup,'--b','LineWidth',2);
hold on;
grid on;
plot(y,actualvalue ,'-r','LineWidth',4);
aux=Transformtobeta(predictedvalue'-std1,link,t1);
axis([min(y) max(y) min(0,min(aux)) 1.1*max([max(auxup) max(actualvalue)])]);
text(0.9*min(y)+0.1*max(y),0.1*max([max(auxup) max(actualvalue)]),['Link ' num2str(link)]);
hold off;
print('-deps2c','-r600',sprintf('%s_%s.eps',ProjectName,get(gcf,'Name')));
```

Observed Generalized Beta variables (PlotData)

```
function PlotData(link,y)
% Plot the observed Generalized Beta variables

global ProjectName;
global nlinks;
global nfigures;
global hh;
global t1;
global Nsimulations;
global selectedsimulation;
global maxlink;
global minlink;

% Opens the file
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
filename=[ProjectName,'NORMAL.dat'];
fid=fopen(filename,'r');

nfigures=nfigures+1;
figure('Name',strcat('LinkTravelTimeF',num2str(nfigures)));
nlinks=fread(fid,1,'integer*4');
hh=fread(fid,1,'integer*4');
Nsimulations=fread(fid,1,'integer*4');
A=zeros(nlinks,hh);
meanlink=zeros(nlinks,hh);

for j=1:Nsimulations
    for ii=1:nlinks
        A(ii,:)=fread(fid,hh,'real*4');
    end
    if j==selectedsimulation
        AA=A(link,:);
        minlink(link)=min([minlink(link) Transformtobeta(A(link,:),link,1)]);
        maxlink(link)=max([maxlink(link) Transformtobeta(A(link,:),link,1)]);
    else

        % Plots observed data of link
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        plot(y,Transformtobeta(A(link,:),link,1),'-b','LineWidth',2);
        hold on;
    end
    minlink(link)=min([minlink(link) Transformtobeta(A(link,:),link,1)]);
    maxlink(link)=max([maxlink(link) Transformtobeta(A(link,:),link,1)]);

    % Plots mark where observations are taken
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    plot(y(t1),minlink(link),'o','MarkerEdgeColor','k','MarkerFaceColor','r','MarkerSize',
         16);
    hold on;
    meanlink=meanlink+A;
end

% Plots real data
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
plot(y, Transformtobeta(AA,link,1),'-r','LineWidth',6);
meanlink=meanlink/Nsimulations;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
plot(y, Transformtobeta(meanlink(link,:),link,1),'--g','LineWidth',6);
text(5,0.9*maxlink(link)+0.1*minlink(link),['Link ' num2str(link)]);
```

```
xlabel('t');
ylabel('D_a(t)','Rotation',0);

% Adjusts axis
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
axis([min(y) max(y) minlink(link) maxlink(link)]);
hold off;
print('-deps2c','-r600',sprintf('%s_%s.eps',ProjectName,get(gcf,'Name')));
fclose(fid);
```

10.3.8 Other procedures

Selected variables for prediction in LaTeX(dovarLatexTable)

```
function dovarLatexTable
% Writes a latex table: Order in which variables are selected for different prediction
% times

global ProjectName;
global timevaluesLatexTable;
global varLatexTableData;
global increment;
global Missingdata;

[m n p]=size(varLatexTableData);

if isempty(Missingdata)
    filename=['Latex',ProjectName,'Var.dat'];
else
    filename=['Latex',ProjectName,'VarM.dat'];
end
fid=fopen(filename,'w');

fprintf(fid,'\\begin{table}\n');
fprintf(fid,'\\centering\n');
fprintf(fid,'{\\tiny\n');
fprintf(fid,'\\renewcommand{\\tabcolsep}{1mm}\n');
fprintf(fid,'\\renewcommand{\\arraystretch}{1.0}\n');
fprintf(fid,'\\begin{tabular}{|c');
for k=1:p
    fprintf(fid,'|c|c|c');
end
fprintf(fid,'|}\n');
fprintf(fid,'\\hline\n');
for k=1:p
    fprintf(fid,'& \\multicolumn{3}{|c|}{');
    aux=timevaluesLatexTable(k)*increment;
    aa=floor(aux);
    bb=(aux-aa)*60;
    if bb==0
        fprintf(fid,'%2.0f:00}',aa);
    else
        fprintf(fid,'%2.0f:%2.0f}',aa,bb);
    end
end
fprintf(fid,' \\\\\n');
fprintf(fid,'\\hline\n');
fprintf(fid,'i ');
for k=1:p
    fprintf(fid,'&link&time&var');
end
fprintf(fid,' \\\\\n');
fprintf(fid,'\\hline\n');
for i=1:m
    if varLatexTableData(i,1,k)>0
        fprintf(fid,'%3.0f',i);
        for k=1:p
            aux=varLatexTableData(i,2,k);
            aa=floor(aux);
            bb=(aux-aa)*60;
            if bb==0
                fprintf(fid,' & %3.0f & %2.0f:00 & %6.3f ',varLatexTableData(i,1,k),aa,
                        varLatexTableData(i,3,k));
            else
                fprintf(fid,' & %3.0f & %2.0f:%2.0f & %6.3f ',varLatexTableData(i,1,k),aa,
                        bb,varLatexTableData(i,3,k));
            end
        end
        fprintf(fid,' \\\\\n');
    end
end
fprintf(fid,'\\hline\n');
fprintf(fid,'\\end{tabular}\n');
fprintf(fid,'\\caption{\\label{t683}Order in which variables are selected for different
        prediction times and time of prediction $12$:$00$.}}\n');
fprintf(fid,'\\end{table}\n\n');

fclose(fid);
```

Storage of variables (interpret1)

```
function interpret1(indexZ,tt,varZ)
%Stores (link, time, variance) to be printed

global varLatexTableData;
global increment;
global t1;
global nlinks;
```

```
for i=1:length(indexZ)
   n=indexZ(i);
   m=floor((n-1)/nlinks);
   aux=(t1-m)*increment;

   varLatexTableData(i,1,tt)=n-m*nlinks;
   varLatexTableData(i,2,tt)=aux;
   varLatexTableData(i,3,tt)=varZ(i);
end
```

Calculation of Generalized Beta densities (varchange)

```
function [z]=varchange(a,b,alpha,beta,mu,sigma,x)
% Calculates the Generalized Beta densities given the normal distributions

aux=max(normpdf(norminv(betacdf((x-a)/(b-a),alpha,beta),0,1),0,1),0.000000001);
z=normpdf(norminv(betacdf((x-a)/(b-a),alpha,beta),0,1),mu,sigma)*betapdf((x-a)/
  (b-a),alpha,beta)/((b-a)*aux);
```

Printing in matricial format (PrintMatrix)

```
function PrintMatrix(s0,A,s)
[nn mm]=size(A);
fprintf(s0);
fprintf('\n');
for i=1:nn
    for j=1:mm
        fprintf(s,A(i,j));
    end
    fprintf('\n');
end
```

Generate a set of random colors (GenerateRandomColors)

```
function GenerateRandomColors(n)
% Generate random colors

global colors;

colors=zeros(n,3);
for i=1:n
    colors(i,:)=[rand rand rand];
end
```

Searching the station name (findStationNumber)

```
function nstation=findStationNumber(StationNames,stat)
```

```
% Looks for station name in StationNames list

n=length(StationNames);
i=1;
nstation=0;
while i<=n && nstation==0
    if strcmp(stat,StationNames(i))
        nstation=i;
    else
        i=i+1;
    end
end
```

Notation

α	parameter of the statistical function.
α_a	free-flow travel time of link a.
$\bar{\alpha}_{pi}$	vector of parameters associated with path p.
$\beta(\alpha, \theta)$	Beta function with parameters α and θ.
β_a	parameter of the travel cost function associated with link a.
β_{ij}	regression coefficient of X_j in the regression of X_i on their parents.
δ_a	parameter of the travel time function associated with link a.
δ_a^r	terms of the path-link incidence matrix, where r denotes the path and a, the link.
δ_{pqr}	parameter of the Gumbel distribution associated with route r with origin-destination p-q.
ϵ_{rs}	active hyperparameter of the shifted-Gamma function corresponding to route r and the s element in the sample.
$\hat{\epsilon}_{rs}$	posterior active hyperparameter of the shifted-Gamma function corresponding to route r.
$\eta_a(t)$	link a travel time associated with a vehicle that leaves link a at time t.
η_r	active hyperparameter of the shifted-Gamma function corresponding to route r.
$\hat{\eta}_r$	posterior active hyperparameter of the shifted-Gamma function corresponding to route r.
η_{pq}	dual variable associated with the origin-destination p-q.
γ	parameter of the travel cost function.
$\Gamma(\alpha)$	Gamma function.
γ_e	Euler constant.
γ_{ijpq}	parameters of calibration of the maximum entropy problem.
Λ	total number of queued vehicles at the link exit node.
λ	parameter of the shifted-Gamma model.
λ_a	dual variable. Travel cost associated with link a.
μ	mean.
μ_{pi}	mean parameter of the ith component of path p.

μ_{pqr}	dual variable. Over-cost associated with route r with origin-destination p-q.
ν_i	parameter of the shifted-Gamma function associated to the number of users traveling through the OD-pair i.
Φ	cumulative distribution function of the standard normal random variable.
$\hat{\phi}_{pqr}(h, \omega)$	worst-case travel cost between the OD-pair p-q through the route r associated with the worst case demand $\hat{d}_{pq}$.
ϕ^*_{pqr}	stochastic minimum travel cost associated with route r with origin-destination p-q.
$\phi_{pqr}(h, \omega)$	travel cost between the OD-pair p-q through the route r associated with stochastic demand $d_{pq}(\omega)$ or link capacity $q_a(\omega)$.
Π_i	set of parents of node X_i in a directed acyclic graph.
π_i	parent of node i.
π_{pq}	equilibrium travel cost to travel from origin p to destination q.
ψ_i^2	conditional variance of X_i given $\Pi_i = \pi_i$.
ρ	traffic density.
ρ^*	dimensionless density.
ρ_c	critical density.
ρ_c^*	dimensionless critical density.
ρ_j	jam traffic density.
ρ_r	active hyperparameter of the shifted-Gamma function corresponding to route r.
$\hat{\rho}_r$	posterior active hyperparameter of the shifted-Gamma function corresponding to route r.
ρ_{iq}	minimum costs to go from node i to node q.
ρ_i	location parameter of the shifted-Gamma function associated to the number of users traveling through the OD-pair i.
ρ_i^a	location parameter of the shifted-Gamma function associated to the number of users using link a and traveling the OD-pair i.
Σ	variance-covariance matrix.
σ^2	variance.
σ_{pi}	standard deviation parameter of the ith component of path p.
Σ_{YY}	variance-covariance matrix of Y.
Σ_{YZ}	covariance matrix of Y and Z.
σ_{zz}	variance of Z.
$\tau(t)$	delay function.
$\tau_a^{-1}(t)$	entry time of a user that exits link a at time t.
$\tau_a(t)$	link traversal time for a vehicle entering the link a at time t.
Θ	domain of definition of the θ parameter.

θ	parameter of the statistical function.
$\theta^{p}_{a_i}(t)$	departure time from the origin of path p of a user who exits the ith link a_i of path p at time t.
θ^{a}_{i}	parameter of the shifted-Gamma function associated to the number of users using link a and traveling the OD-pair i.
θ_{pq}	parameter of the Gumbel distribution associated with origin-destination p-q.
ξ_{pqr}	random error term associated with the route r with origin-destination p-q.
a	link.
$A(x,t)$	cumulative flow past any point x by time t.
a_{ij}	link between nodes i and j.
b	lower end of the generalized Beta variable.
$\hat{b}_{at}$	estimated Beta parameter of the random variable X_{at} of link a at time t.
c	upper end of the generalized Beta variable.
$\hat{c}_{at}$	estimated Beta parameter of the random variable X_{at} of link a at time t.
C^{*}_{pqr}	least travel cost corresponding to route r with origin-destination p-q.
c_a	travel cost function associated with link a.
c_{a0}	free-flow conditions cost associated with link a.
C_a	integral of the travel cost function associated with link a.
C_{ij}	integral of travel cost function associated with link a_{ij}.
c_{ij}	travel cost function associated with link a_{ij}.
$c_{ij}(t)$	travel cost to go from the initial node i to the final node j associated with the departure time t.
C_{ir}	set of links in route r of OD-pair i.
C_{pqr}	perceived travel cost associated with route r with origin-destination p-q.
c_{pqr}	travel cost function associated with route r with origin-destination p-q.
$cost_{ij}$	minimum trip cost between any pair of nodes (i,j).
D	abscissa coordinate of a given point.
D^{*}	dimensionless abscissa coordinate of a given point.
$D_a(t)$	lower bound of link travel time for vehicles entering link a at time t.
d_{pq}	demand for the flows from origin p to destination q.
$\hat{d}_{pq}$	worst case demand associated with the origin p to destination q at time t.
E	set of evidential nodes.
$E_a(t)$	cumulative inflow up to time t for link a.
$e_a(t)$	inflow rate for link a at time t.

F_r	random amount of users choosing route r.
f_r	path flows.
$\hat{f}_r$	reconstructed path flows.
$f_{N(\mu,\sigma)}(t)$	probability density function (pdf) of the normal distribution with mean μ and standard deviation σ.
f_r^0	prior path flows.
$G(\cdot)$	survival function of a cumulative distribution function.
$G_a(t)$	cumulative flow associated with the exit of link a by time t.
$g_a(t)$	outflow rate for link a at time t.
$g_a^p(t)$	outflow rate of link a at time t due to path p.
$GBeta(\alpha, \theta; b, c)$	generalized Beta function with parameters α, θ, b and c.
$H_p(t)$	cumulative inflow rate at the origin of path p at time t.
$h_p(t)$	inflow rate at the origin of path p at time t.
h_{pi}	coefficients of the linear combination to generate $h_p(t)$.
h_{pqr}	flow on route r with origin-destination p-q.
$h^*_{pqr}(t)$	route inflows in a dynamic user equilibrium state.
$I_\beta(x; \alpha, \theta)$	incomplete Beta function.
k	observation.
L	link length.
n	number of nodes in a Bayesian network.
$n(r; \mathcal{SC})$	number of routes that can be distinguished from route r using the set $\mathcal{SC}$ of scanned links.
n_0	size of the virtual prior sample.
$N_i(t)$	maximum number of vehicles that can be inside cell i at time t (cell capacity).
$n_i(t)$	number of vehicles in cell i at time t.
n_p	number of function components.
n_r	sample size corresponding to route r.
n_{links}	number of network links.
nc	number of scanned links.
p	path.
P_{pqr}	share of drivers choosing a route r with origin-destination p-q.
p_{pqr}	probability of a user to select path r of OD-pair p-q.
q	traffic flow.
$q(t; \bar{\alpha}_{pi})$	parametric family of probability density functions with parameter $\bar{\alpha}_{pi}$.
q^*	dimensionless flow.
q_a	constant measuring the flow producing a given congestion level in the link a.
$q_a(\omega)$	stochastic link capacity.
$Q_i(t)$	maximum flow in cell i at time t.

q_m	maximum flow.
Q_{ak}	queue dissipation time at link a at time t_k.
r	route.
$r_n(t)$	flow intensity at node n and time t.
R_{pq}	set of routes with origin-destination p-q.
S	subset of variables.
$s_a(t)$	link a congestion ratio.
t	time.
t^*	dimensionless time.
t_0	time of prediction.
$T_a(t)$	link a entry time associated with a vehicle that leaves link a at time t.
T_i	number of users traveling through the OD-pair i.
t_i	OD-pair i flows.
t_k	set of discrete times to be considered for $k = 1, \dots, n$.
t_m	discrete time value.
t_s	predicted time.
t_a^{out}	link exit time of a user who enters link a at time t.
u	traffic velocity.
$U(0, 1)$	standard uniform random variable.
u^*	dimensionless velocity.
u_0	free-flow speed.
u_c	critical velocity.
u_w	shock wave speed.
u_{at}	random value from a standard $Beta(\alpha, \theta)$ variable.
$\hat{v}$	observed flow.
v^*	link flows resulting from an assignment problem.
v_a	flow on link a.
$\hat{v}_a$	link count information.
V_{ai}	number of users using link a and traveling the OD-pair i.
V_a	number of users using link a.
v_{ijpq}	flow through link a_{ij} going from origin node p to destination node q.
v_{ijq}	flow on link a_{ij} with destination q.
v_{ij}	total flow through link a_{ij}.
w_{ijp}	flow coming from a given origin node p and using link a_{ij}.
$\hat{w}_r$	flow associated to route r observed by plate scanning technique.
W_s	number of users passing through and only through a given subset $\mathcal{A}_s$ of links.
$\hat{X}$	estimate of variable X.
x	abscissa coordinate.
x^*	dimensionless abscissa coordinate.

x^0_{rs}	s element in a virtual sample.
x_0	abscissa associated with the initial time.
x^*_0	dimensionless abscissa associated with the initial time.
$x_a(t)$	number of vehicles on link a at time t (link traffic volume).
x^{max}_a	number of vehicles on link a leading to a travel time $\alpha_a(1+\beta_a)$.
x_i	value of the random variable associated with node i in a Bayesian network.
X_{at}	generalized Beta random variable associated with link a and time t.
x_{kat}	observation k of the random variable x of link a at time t.
x_{max}	maximum number of vehicles in a link. Capacity.
$y_i(t)$	maximum flow in cell i and the number of vehicles that coming from cell $i-1$ can enter cell i during the time interval $(t, t+1)$.
Y_{at}	standard normal random variable associated with link a and time t.
y_{kat}	standard normal variable value corresponding to observation k in link a at time t.
$Y_{pqr}(t)$	cumulative number of vehicles entering the network at origin p and exiting at destination q over route r at any time t.
$y_{pqr}(t)$	instantaneous flows that arrives at destination q from origin p at time t.
z_{ipq}	flow going from origin node p to destination node q and passing through node i.
$\mathcal{A}$	set of links of a traffic network.
$\mathcal{A}(n)$	set of links entering node n.
$\mathcal{A}_s$	subset of observed links.
$\mathcal{B}_0$	set of link-time pairs.
$\mathcal{D}$	set of origin-destination pairs of nodes.
$\mathcal{F}$	feasible set of scanned links observations.
$\mathcal{H}(\alpha, \theta, \lambda)$	shifted-Gamma family of densities.
$\mathcal{N}$	set of nodes of a traffic network.
$\mathcal{P}(a)$	set of paths containing link a.
$\mathcal{R}$	set of routes.
$\mathcal{R}_i$	set of paths of each OD pair i.
$\mathcal{S}(a)$	set of all links downstream link a in all its paths.
$\mathcal{SC}$	set of links to be scanned.

Bibliography

Anderson, T. W. (1984). *An introduction to multivariate statistical analysis*. John Wiley and Sons, New York.

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., Hato, E., and Kashiwadani, M. (2000). Origin-destination matrices estimation model using automatic vehicle identification data and its application to the han-shin exopressway network. *Transportation*, 27:419–438.

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.

Astarita, V. (1996). A continuous time link model for dynamic network loading based on travel time function. In *13th International Symposium on Theory of Traffic Flow*, pp. 79–103.

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 matrix 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 B*, 25:1:13–23.

Ben-Akiva, M. and Lerman, S. R. (1985). *Discrete choice Analysis: Theory and application to travel demand*. MIT Press, Cambridge, Massachussetts and London, England.

Bianco, L., Confessore, G., and Gentili, M. (2006). Combinatorial aspects of the sensor location problem. *Annals of Operations Research*, 144:201–234.

Burrel, J. E. (1968). Multiple route assignment and its application to capacity restraint. *4th International Symposium of the theory of traffic flows.*

Cantarella, G. E. (1997). A general fixed-point approach to multimode multi-user equilibrium assignment with elastic demand. *Transportation Science*, 31:107–128.

Carey, M. and Revelli, R. (1986). Constrained estimation of direct demand functions and trip matrices. *Transportation Science*, 20:3:143–152.

Cascetta, E. (1984). Estimation of trip matrices from traffic counts and survey data: a generalized least squares estimatorm. *Transportation Research*, 18B:289–299.

Cascetta, E. and Nguyen, S. (1988). A unified framework for estimating or updating origin/destination matrices from traffic counts. *Transportation Research*, 22B:437–455.

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., Castillo, C., Mínguez, R., and Ortigosa, D. (2006). A perturbation approach to sensitivity analysis in mathematical programming. *Journal of Optimization Theory and Applications.*

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., Mínguez, R., and Castillo, C. (2005). A closed formula for local sensitivity analysis in mathematical programming. *Engineering Optimization.*

Castillo, E., Conejo, A., Pedregal, P., García, R., and Alguacil, N. (2001). *Building and Solving Mathematical Programming Models in Engineering and Science.* Pure and Applied Mathematics: A Wiley-Interscience Series of Texts, onographs and Tracts. John Wiley & Sons Inc., New York.

Castillo, E., Gutiérrez, M., J., Hadi, A. S., and Solares, C. (1997a). Symbolic propagation and sensitivity analysis in gaussian Bayesian networks with application to damage assessment. *Artificial Intelligence in Engineering*, 11:173–181.

Castillo, E., Gutiérrez, J. M., and Hadi, A. S. (1997b). *Expert systems and probabilistic network models.* Springer Verlag, New York.

Castillo, E., Gutiérrez, J. M., and Hadi, A. S. (1997c). Sensitivity analysis in discrete Bayesian networks. *IEEE Transactions on Systems, Man and Cybernetics*, 26(7):412–423.

Castillo, E., Hadi, A. S., and Solares, C. (1996). Learning and updating of uncertainty in dirichlet models. *Machine Learning*, 26:3–63.

Castillo, E., Jiménez, P., Menéndez, J. M., and Conejo, A. (2008b). The observability problem in traffic models: algebraic and topological methods. *IEEE Transactions on Intelligent Transportation Systems*, 9(2):275–287.

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. (2008c). Trip matrix and path flow reconstruction and estimation based on plate scanning and link observations. *Transportation Research B*, 42:5:455–481.

Castillo, E., Menéndez, J. M., Nogal, M., Sánchez-Cambronero, S., and Jiménez, P. (2011). FIFO rule consistent model for the continuous dynamic network loading problem. *IEEE Transactions on Intelligent Transportation Systems*, DOI.: 10.1109/TITS.2011.2169668.

Castillo, E., Menéndez, J. M., and Sánchez-Cambronero, S. (2008d). Predicting traffic flow using Bayesian networks. *Transportation Research B*, 42:5:482–509.

Castillo, E., Menéndez, J. M., and Sánchez-Cambronero, S. (2008e). Traffic estimation and optimal counting location without path enumeration using Bayesian networks. *Computer Aided Civil and Infraestructure Engineering*, 23(2).

Castillo, E., Rivas, A., Jiménez, P., and Menéndez, J. M. (2009). The observability problem in traffic networks based on combining plate scanning and link observations. *Submitted*.

Castillo, E., Sarabia, J. M., Solares, C., and Gómez, P. (1999). Uncertainty analyses in fault trees and Bayesian networks using form/sorm methods. *Reliability Engineering and System Safety*, 65:29–40.

Celikoglu, H. B. (2007). A dynamic network loading model for traffic dynamics modeling. *IEEE Transactions on Intelligent Transportation Systems*, 8(4):575—-583.

Chandler, R. E., Herman, R., and Montroll, E. W. (1958). Traffic dynamics: studies in car following. *Operations Research*, 16(2):165–184.

Chang, K. C. and Fung, R. (1991). Symbolic probabilistic inference with continuous variables. *Proceedings of the Seventh Conference on Uncertainty in Artificial Intelligence*, pp. 77–85.

Childress, S. (2005). *Notes on Traffic Flow.*

Daganzo, C. (1995a). Properties of link travel time functions under dynamic loads. *Transportation research Part B*, 29B(2):95–98.

Daganzo, C. and Sheffi, Y. (1977). On stochastic models of traffic assignment. *Transportation Science*, 11:253–274.

Daganzo, C. F. (1992). The cell transmission model. Part I: A simple dynamic representation of highway traffic. Tech. Rep. UCB-ITS-PRR-93-7, Institute of Transportation Studies, Univesrity of California, Berkeley.

Daganzo, C. F. (1995b). The cell transmission model, part II: Network traffic. *Transportation Research Part B*, 29(2):79–93.

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, 39(1):1—-38.

Doblas, J. and Benitez, 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*, 39B:565–591.

E. Castillo, J. M. G., A. Cobo and Pruneda., E. (1998). *Functional Networks with Applications.* Kluwer Academic Publishers, Boston, Dordrecht, London.

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 trip estimation with user optimal assignmen. *Transportation Research B*, 22:69–79.

Friesz, T., Bernstein, D., Smith, T., Tobin, R., and Wie, B. (1993a). A variational inequality formulation of the dynamic network equilibrium problem. *Operations Research*, 41:179—-191.

Friesz, T., Bernstein, D., Smith, T., Tobin, R. L., and Wie, B. (1993b). 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.

Fritsch, F. N. and Carlson, R. E. (1980). Monotone piecewise cubic interpolation. *SIAM Journal on Numerical Analysis*, 17(2):238—-246.

Gazis, D. C., Herman, R., and Rothery, R. W. (1961). Nonlinear follow-the-leader models of traffic flow. *Operations Research*, 19(4):545–567.

Gentili, M. and Mirchandiani, P. (2005). Locating active sensors on traffic networks. *Annals of Operations Research*, 136(1):229–257.

Gilks, W., Richardson, S., and Spiegelhalter, D. J. (1996). *Markov Chain Monte Carlo in Practice.*. Chapman and Hall/CRC.

Hall, J. and Mars, P. (1998). The limitatuions of artificial neural networks for traffic prediction. In *Third IEEE Symposium on Computers and Communications*, pp. 8–12. IEEE, Athens, Greece.

Han, H. B., S. (2006). Consistent objetives and solution of dynamic user equilibrium models. *Transportation Research Part B*, 40:16–34.

Han, s. (2002). Dynamic traffic modelling and dynamic stochastic user equilibrium assignment for general road networks. *Transportation Research Part B*, pp. 225–249.

Hazelton, M. L. (2000). Estimation of origin-destination matrices from link flows on congested networks. *Transportation Research B*, 34B:549–566.

Hazelton, M. L. (2001). Inference for origin-destination matrices: estimation, reconstruction and prediction. *Transportation Research B*, 35:7:667–676.

Hazelton, M. L. (2003). Some comments on origin-destination matrix estimation. *Transportation Research*, 37A:811–822.

Hellinga, B. and Van Aerde, M. (1994). A statistical analysis of the reliability of using rgs vehicle probes as estimators of dynamic o-d departure rates. *IVHS Journalh*, 2(1):21–44.

Heydecker, B. and Addison, J. (1996). An exact expression of dynamic traffic equilibrium. In J. Lesort, ed., *Transportation and traffic theory*, pp. 359–383. Pergamon, Oxford.

Hopf, E. (1950). The partial differential equation $u_t + uu_x = \mu_{xx}$. *Communications in Pure and Applied Mathematics*, 3:201–230.

Janson, B. N. (1991). Dynamic traffic assignment for urban road networks. *Transportation Research*, 25B:43–61.

Janson, B. N. and Robles, J. (1995). A quasi-continuous dynamic traffic assignment model. *Transportation Research Record*, 1493:199–206.

Jensen, F. V. (2001). Bayesian networks and decision graphs. *Springer-Verlag*.

Kenley, C. R. (1986). *Influence diagram models with continuous variables*. Ph.d thesis, Stanford University, Stanford.

Klieter, G. (1992). Bayesian diagnosis in expert systems. *Proceedings of the AIJ92*.

Lax, P. D. (1954). Week solutions of nonlinear hyperbolic equations and their numerical computation. *Communications in Pure and Applied Mathematics*, 7:159–193.

Lee, S. and Fambro, D. B. (1999). Application of subsets autoregressive integrated moving average model for short-term freeway traffic volume forecasting. *Transportation Research Record*, (1678):179–188.

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. and Tung, Y. (2003). Network with degradable links: capacity analysis and design. *Transportation Research Part B*, 37(4):345—-363.

Lo, H. P. and Chan, C. P. (2003). Simultaneous estimation of an origi-destination matrix and link choice proportions using traffic counts. *Transportation Research A*, 37:771–788.

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 B*, 30:4:309–324.

Maher, M., Zang, X., and Van Vliet, D. (2001). A bi-level programming approach for trip matrix estimation and traffic control problems with stochastic user equilibrium link flows. *Transportation Research B*, 35:1:23–40.

Maher, M. J. (1983). Inferences on trip matrices from observations on link volumes: a Bayesian statistical approach. *Transportation Research*, 17B(6):435–447.

Maher, M. J. and Zhang, X. (1999). Algorithms for the solution of the congested trip matrix estimation problem. In *Transportation and traffic theory, Proceedings of the 14th International Symposium on Transportation and Traffic Theory*, pp. 445–469. Elsevier, Amsterdam.

Mannering, F. and Kilareski, W. (1998). *Principles of highway engineering and traffic analysis*. John Wiley and Sons, New York.

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.

Miller-Hooks, E. and Mahmassani, H. (2000). Least expected time paths in stochastic, time-varying transportation networks. *Transportation Science*, 34(2):198–215.

Nagurney, A. (1993). *Network Economics. A variational inequality approach..* Kluwer Academic Publisher.

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.

Nie, Y. and Zhang, H. M. (2005a). A comparative study of some macroscopic link models used in dynamic traffic assigment. *Networks and Spatial Economics*, 5:89–115.

Nie, Y. and Zhang, H. M. (2005b). Delay-function-based link models: their properties and computational issues. *Transportation Research Part B*, 39:729—751.

Paveri Fontana, S. L. (1975). On Boltzmann like treatments for traffic flow. *Transportation Research Part B*, 9:225–235.

Prigogine, I. and Herman, R. (1971). *Kinetic theory of vehicular traffic.* American Elsevier, New York.

Queen, C. and Albers, C. (2009). Intervention and causality: Forecasting traffic flows using a dynamic bayesian network. *Journal of the American Statistical Association*, 104(486):669–681.

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.

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. 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.

Sánchez-Cambronero, S., Castillo, E., Menéndez, J. M., and Jiménez, P. (2011). Dealing with error recovery in traffic flow prediction using bayesian networks based on license plate scanning data. *Journal of Transportation Engineering (ASCE)*, pp. doi:10.1061/(ASCE)TE.1943–5436.0000249.

Shachter, R. and Kenley, C. (1989). Gaussian influence diagrams. *Management Science*, 35(5):527–550.

Sheffi, Y. (1985). *Urban transportation networks: equilibrium analysis with mathematical programming methods*. N. Prentice Hall, Englewood Cliffs.

Sheffi, Y. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, San Mateo, CA.

Sheffi, Y. and Daganzo, F. (1978). Hypernetworks and supply-demand equilibrium obtained with disaggregate demand models. *Transportation Research Record*, 673:113–121.

Smith, M. J. (1979). The existence, uniqueness and stability of traffic equilibria. *Transportation Research*, 13B:295–304.

Smith, M. J. (1993). A new dynamic traffic model and the existence and calculation of dynamic user equilibria on congested capacity-constrained road networks. *Transportation Research*, 27B:49–63.

Spiess, H. (1987). A maximum likelihood model for estimating origin-destinations matrices. *Transportation Research B*, 21:5:395–412.

Stokes, G. G. (1845). On the theories of internal friction of fluids in motion. *Transactions of the Cambridge Philosofical Society*, 8.

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.

Sun, Z. C., S. and Zhang, Y. (2005). Traffic flow forecasting using a spatio-temporal Bayesian network predictor. *Lecture notes in computer science (LNCS)*, 3697:273–278.

Szeto, W. Y. and Lo, H. (2005). Properties of dinamyc traffic assigment with physical queues. *Journal of the Easten Asia Society for Transportation Studies*, 6:2108—-2123.

Tebaldi, C. and West, M. (1998). Inference on network traffic using link count data. *Journal of the American Statistical Association*, 93:557–576.

Tobin, R. L. and Friesz, T. L. (1988). Sensitivity analysis for equilibrium network flows. *Transportation Science*, 22:242–250.

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 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. (1994). Maximum likelihood estimation of an origin-destination matrix from a partial registration plate survey. *Transportation Research B*, 28:4:289–314.

Watling, D. P. (2002). A second order stochastic network equilibrium model, I:teorical foundation. *Transportation Science*, 36(2):149–166.

Wellman, M. P., Larson, K., M., F., and R., W. P. (1995). Path planning under time-dependent uncertainty. In *11th Conference on Uncertainty in Artificial Intelligence*, pp. 532—-539.

Whitham, G. B. (1974). *Linear and nonlinear wavess*. Wiley., New York.

William, B. M. (1999). *Modeling and forecasting vehicular traffic flow as a seasonal stochastic time series process*. Ph.D. thesis, University of Virginia, Charlottesvilla, VA.

Willumsen, L. G. (1978). Estimation of od matrix from traffic counts. a review.working paper 99. *Institute of Transportation Studies, University of Leeds*.

Wu, C. F. J. (1983). On the convergence properties of the em algorithm. *Annals of Statistics*, 11(1):95—-103.

Wu, J., Chen, Y., and Florian, M. (1998). The continuous dynamic network loading problem: A mathematical formulation and solution method. *Transportation Research Series B: Methodological*, 32(3):173–187. Cited by (since 1996) 50.

Xu, L. and Pearl, J. (1989). Structuring causal tree models with continuous variables. In *Uncertainty in Artificial Intelligence 3*, pp. 209–219. In Kanal, L. N., Levitt, T. S., and Lemmer, J. F., editors, North Holland, Amsterdam.

Xu, Y., Wu, J., Florian, M., Marcotte, P., and Zhu, D. (1999). Advances in the continuous dynamic network loading problem. *Transportation Science*, 33:341—-353.

Xu, Y. W., Wu, J. H., and Florian, M. (1998). An efficient algorithm for the continuous network loading problem: a dynaload implementation. In *Transportation Networks: Recent Methodological Advances*, pp. 51—-66. Elsevier Science Ltd., Amsterdam.

Yang, H. (1995). Heuristic algorithm for the bi-level origin-destination matrix estimation problem. *Transportation Research*, 29B:1–12.

Yang, H., Sasaki, T., Iida, Y., and Asakura, Y. (1992). Estimation of origin-destination matrices from link traffic counts on congested networks. *Transportation Research*, 26B:417–434.

Yu, Y. J. and Cho, M. G. (2008). A short-term prediction model for forecasting traffic information using Bayesian network. In *Third 2008 International Conference on Convergence and Hybrid Information Technology*, pp. 242–247. IEEE Computer Society.

Zhang, C., Chen, X., and Sumalee, A. (2011). Robust wardrop's user equilibrium assignment under stochastic demand and supply: Expected residual minimization approach. *Transportation Research Part B*, 45(3):534–552.

Zhou, X. and Chen, A. (2008). Comparative analysis of three user equilibrium models under stochastic demand. *Journal of Advanced Transportation*, 42(3):239–263.

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.

Zhu, D. and Marcotte, P. (2000). On the existence of solutions to the dynamic user equilibrium problem. *Transportation Science*, 34(4):402—-414.

Index