Determination of Transport Network Equilibria – Using the Method of Intersection of Straight Line Approximations of Demand and Cost Curves

Mads Paulsen

Kongens Lyngby 2013
B.Sc.-2013-33

Under guidance of:

Bo Friis Nielsen, DTU Compute
Jeppe Rich, DTU Transport
Otto Anker Nielsen, DTU Transport

DTU Compute
Technical University of Denmark
Matematiktorvet, building 303B
DK-2800 Lyngby, Denmark
Tel: +45 4525 3031
Fax: +45 4588 1399
compute@compute.dtu.dk
www.compute.dtu.dk
B.Sc.-2013-33

DTU Transport
Technical University of Denmark
Bygningstorvet, building 116B
DK-2800 Lyngby, Denmark
Tel: +45 4525 6500
Fax: +45 4593 6533
transport@transport.dtu.dk
www.transport.dtu.dk
B.Sc.-2013-33

Abstract

The determination of the intersection between the demand curve, $D(t)$, and the travel time curve, $t(D)$, is a keystone in transportation systems analysis. The determination is a troublesome iterative process that includes an external mode choice loop as well as an internal route choice loop. Since the internal loop has to run until termination for each iteration of the external loop, alternative algorithms that seek to reduce the number of outer iterations are of great interest. As such this thesis presents the method of intersection of straight line approximations of demand and cost curves (the I method). Numerical experiments indicate that the method is superior to a straightforward conventional approach, but that it fails to compete with the most efficient methods currently available. The main drawback of the I method is its sensitivity to mutual correlation among the various parts of the network. This problem can not be addressed within the current structure of the I method. The thesis concludes that it is unlikely for any two-point based interpolation method to outperform the best currently available methods.

Preface

This thesis was prepared as a joint thesis at Technical University of Denmark between Department of Applied Mathematics and Computer Science and Department of Transport in fulfilment of the requirements for acquiring an BSc Eng in Mathematics and Technology.

The main goal of the thesis is to implement and test a new method for determining the mode specific transport demand, called *The Method of Intersection of Straight Line Approximations of Demand and Cost Curves* or simply *The I Method*. The method is a lot different from the usual way of determining the transport demand, since it makes use of the trends of demand and travel times between each iteration. The idea behind the method was presented in [RNC13] by Jeppe Rich and Otto Anker Nielsen from Department of Transport, Technical University of Denmark in corporation with Guilio Erberto Cantarella from Department of Transportation Engineering, University of Naples Federico II. This study can be seen as a continuation of this work and is meant to implement the method and test it on a small-scale test network. The method has been implemented and tested in MatLab[1].

The thesis consists of 6 chapters. The first chapter gives a brief introduction to the transport network equilibria. The second chapter introduces the fundamentals of transport demand analysis required to understand the determination methods presented later on. Chapter 3 presents the I method, the idea behind it, and the derivation of it. Before doing so, it explains the standard method and the concept of method of repeated approximations (MRA) and method of successive averages (MSA). Chapter 4 deals with the necessary preparations prior

[1] MATLAB, Release 2012a, The Mathworks Inc., Natick Massachusetts, United States.

to the tests of the I method. The small-scale transport network and the the concept of Monte Carlo generated OD matrices are introduced alongside with the calibration of some important constants of the models. In chapter 5 the I method is tested thoroughly, and the results of the tests are discussed. Finally, chapter 6 gives an explanation of why the I method does not perform as good as intended. It also suggests some changes to the method (leading to a method called *the C method*), and results of tests using these suggestions are discussed too.

Conclusively, I would like to thank my advisers Bo Friis Nielsen from Department of Applied Mathematics and Computer Science, Technical University of Denmark as well as Jeppe Rich and Otto Anker Nielsen from Department of Transport, University of Denmark for taking the time to come up with the idea of this interesting project and for guiding my through the project. Also, I would like to thank Finn Kuno Christensen, Department of Applied Mathematics and Computer Science, Technical University of Denmark for developing the LaTeX template used in the thesis.

Lyngby, 31-August-2013

Mads Paulsen

Contents

List of Tables

List of Figures

List of Symbols

$a_{D,\Omega,c}$	Constant term of the linear approximation of the demand for car travel of OD pair Ω.
a_k	MSA coefficient.
$a_{t,\Omega,c}$	Constant term of the linear approximation of the average travel time for car travel between OD pair Ω.
α_l	Constant used to calculate the travel time for car on link l.
$b_{D,\Omega,c}$	Slope of the linear approximation of the demand for car travel of OD pair Ω.
$b_{t,\Omega,c}$	Slope of the linear approximation of the average travel time for car travel between OD pair Ω.
β_l	Constant used to calculate the travel time for car on link l.
C_l	Car capacity of link l.
$C_\bullet$	Total car capacity of the network.
$\bar{D}_\Omega$	Expected demand between any two OD pairs.
$D_\bullet$	Total demand of the network.
$D_{l,c}$	Demand for car travel on link l.
$D_{l,p}$	Demand for public transport on link l.
D_Ω	Demand of the OD pair Ω.
$D_{\Omega,c}$	Demand for car transport of the OD pair Ω.
$D^*_{\Omega,c}$	Final demand for car transport of the OD pair Ω .
$D_{\Omega,p}$	Demand for public transport of the OD pair Ω.
$D_{r,c}$	Demand for car travel on route r.
$D_{r,p}$	Demand for public transport on route r.

$D^{\diamond}_{r,c}$	Demand for public transport on route r, when internal loop is in equilibrium.
$\tilde{D}_{\Omega,c}$	Demand at the intersection of the linear approximations of demand and average travel times for cars between OD pair Ω.
$\Delta_{\bullet}$	Expected total demand of the network.
$D_{c,\bullet}$	Total demand of car travel on the network when in equilibrium.
G_{Ω}	Set of routes connecting OD pair Ω.
G_r	Set of links contained in the route r.
γ	Measure of the time elasticity.
$k_{\max}$	Maximum number of iterations of the external loop.
$k_{i,\max}$	Maximum number of iterations of the internal loop.
κ_c	Car preference constant.
μ	Market share of car travel on the network.
l	Symbol denoting a link.
μ_0	Market share of cars under free flow conditions.
n_l	Number of links in the network.
n_{Ω}	Number of distinct OD pairs in the network.
n_r	Number of distinct routes in the network.
Ω	Symbol denoting an OD pair (two paired nodes).
Ω_r	OD pair sharing the same origin and destination as route r.
$P_{\Omega,c}$	The probability of choosing a car when travelling between the OD pair Ω.
r	Symbol denoting a route.
$t_{l,c}$	Travel time by car for travelling through link l.
$t_{l,p}$	Public transport time for travelling through link l.
$t_{r,c}$	Travel time by car for travelling through route r.
$t^{\diamond}_{r,c}$	Travel time by car for travelling through route r, when internal loop is in equilibrium.
$t_{r,p}$	Travel time by public transport for travelling through route r.
$t_{\Omega,c}$	Average travel time by car between the OD pair Ω.
$t^{*}_{\Omega,c}$	Final average travel time by car between the OD pair Ω.
$t_{\Omega,p}$	Average travel time by public transport between the OD pair Ω.
$\tilde{t}_{\Omega,c}$	Average travel time at the intersection of the linear approximations of demand and average travel times for cars between OD pair Ω.
$t_{0,l,c}$	Free flow travel time for cars travelling link l.
τ	Tolerance of the stopping criteria of the external loop.
τ_i	Tolerance of the stopping criteria of the internal loop.
U_c	Utility function for cars.
U_p	Utility function for public transport.
$y_{D,\Omega,c}$	Linear approximation of the demand for car travel of the OD pair Ω.
$y_{t,\Omega,c}$	Linear approximation of the average travel time for car travel of the OD pair Ω.

Chapter 1

Introduction

This introductory chapter gives a short presentation to the main problem area of the thesis. It seeks to provide the reader with the necessary theoretical overview to understand the basic theory of transport demand analysis presented in chapter 2. Thus the chapter can be seen as a prerequisite, but it also seeks to guide the reader through the content of the thesis, as well as stating the main limitations of the thesis.

The thesis deals with the determination of transport network equilibria. Between any two nodes in a transport network (a so-called *OD pair*, denoted by Ω) there is a function representing the demand for car transport, $D_{\Omega,c}(t_{\Omega,c})$, as well as a function representing the average travel time by car, $t_{\Omega,c}(D_{\Omega,c})$. The equilibrium demand and average travel time between the OD pair, Ω, is denoted by $(D^*_{\Omega,c}, t^*_{\Omega,c})$ and suffices $D^*_{\Omega,c} = D_{\Omega,c}(t^*_{\Omega,c})$ and $t^*_{\Omega,c} = t_{\Omega,c}(D^*_{\Omega,c})$. When the equilibrium demand and average travel time between all OD pairs of the transport network have been found, the transport network is said to be in equilibrium.

The demand and travel times of car transport affect each other, because an increasing number of cars travelling between the same two points will naturally lead to a decreasing average speed due to congestion. This causes the average travel times to increase, and people to select alternatives routes to avoid congestion.

If the primary link between an OD pair is heavily congested, a lot of people will search for alternative routes, whereas only a few will deviate from the usual route if the congestion is minor. This means that the route choice of the cars is strongly dependent on the demand for car travel. It is obvious that the route choice of the cars affect the average travel time by car since each route has its own travel time. It is also true that the amount of people choosing a car over public transport (the mode choice) is dependent on the travel times. As such, it can be seen that the network equilibrium can only be found by using a method that allows the mode choice and route choice to interact with each other.

This study includes only two modes of transportation, cars and public transportation, and uses a fixed overall demand between any OD pair, D_{Ω}. Since all other transportation modes (bicycles, planes and roller skates etc.) are omitted, a unique equilibrium demand for public transport, $D^*_{\Omega,p}$, can be found as $D^*_{\Omega,p} = D_{\Omega} - D^*_{\Omega,c}$. The corresponding equilibrium average travel time for public transport could be determined by, $t^*_{\Omega,p} = t_{\Omega,p}(D^*_{\Omega,p})$, but is actually already known right from the beginning. This is because the travel times of public transport is assumed to be unaffected by congestion.

Since $t^*_{\Omega,p}$ is known from the beginning, it is tempting to simply calculate $D^*_{\Omega,p}$ by $D_{\Omega,p}(t^*_{\Omega,p})$, but this is *not* possible, since the demand for public transport is also (and mainly) dependent on the travel time of the car transport. Instead a 2-stage model involving an external mode choice loop, and an internal route choice loop has to be used. For each iteration of the mode choice loop, the route choice loop has to run until termination. When the mode choice loop reaches convergence, the network equilibrium has been found.

Since the internal loop has to terminate for each iteration of the external loop, making each iteration of the external route rather “expensive”, it is of great interest to reduce the number of iterations of the external loop. The thesis seeks to do this by implementing and testing a new algorithm called *the I method.* The method uses straight line approximations of the demand and cost curves based on the previous two iterations. The idea was presented in [RNC13], and is quite groundbreaking in the sense that no prior studies have dealt with utilising the trend of the demand and cost curves to determine transport equilibria. At least not to the knowledge of the author.

The tests of the I method will be carried out on a small-scale transport network using Monte Carlo generated fixed OD matrices. If the I method turns out to be successful under these conditions, further investigations of the method should be conducted using a large transport network, and with the use of proper trip generation and trip distribution. These two aspects are the main limitations of the thesis.

The thesis is meant to be read chronologically, but depending on the interest and the background of the reader, it might not be necessary to read the entire thesis. Chapter 2 provides the background theory of the thesis, and can be skipped if the reader is already familiar with the basics of transport network analysis such as the BPR formula, discrete choice models and random utility theory. In chapter 3 the concepts of the method of repeated approximations (MRA) and the method of successive averages (MSA) are presented alongside with the I method. The chapter is very important for the understanding of the I method. Chapter 4 is concerned with the test network and the calibration of some constants used in the simulations. The chapter gives a broader understanding of the test results presented in chapter 5 but is not essential. As metioned, the test results are found in chapter 5, whereas the conclusive chapter 6 discusses why the I method fails to behave as intended. This chapter is important for understanding the downsides of the I method, although the final section, section 6.3 can be skipped if the reader is solely interested in the I method.

Chapter 2

Fundamentals of Transportation Demand Analysis

In transport systems analysis, it is of great importance to be able to find the intersection between the demand curve, $D(t)$, and the cost curve, $t(D)$. The reason is that this intersection point reveals the actual estimated amount of traffic flow (demand) and travel time (cost) there will be on each link (road) and route in the traffic network. It also reveals the amount of passengers travelling between two nodes (cities) travelling by car, and how large the average travel time between the two nodes are.

This chapter introduces the basics of transport demand analysis. In section 2.1 an example featuring a very simple transport network will be presented alongside with the BPR formula. In section 2.2 the general 4-stage model will be introduced. In the following section (2.3), a reduced model sufficing the analysis needed for this study is introduced. Section 2.4 presents the theory of random utility and contains some relevant examples too. Finally, section 2.5 discusses the problems occurring when a transport network is extended to consist of multiple OD pairs.

The chapter is primarily based on [Ric10], and where no other references are given it is implicit that [Ric10] is the source.

2.1 Introduction

Later on in this chapter, transport networks with multiple possible routes and modes will be presented. But in order to get a gentle introduction to the field, we will begin with a very simple network.

Example 2.1.1 (The Simplest Transport Network)
In this example we consider the simplest transport network possible consisting of two nodes (cities) connected by a single link (road) with a capacity of $C_l = 75$. The travel time for a car travelling on the link is given by

$$t_c(D_c) = t_{0,c}\left(1 + \alpha\left(\frac{D_c}{C_l}\right)^{\beta}\right) = 5\left(1 + 0.5\left(\frac{D_c}{75}\right)^2\right),$$

where D_c is the amount of cars travelling between the two cities. Furthermore, we assume that there is no public transport between the two cities, and no alternative vehicles (bicycles, roller skates, planes etc.) are available, see figure 2.1.

Figure 2.1: The transport network of example 2.1.1

Let the demand between the two cities be given by $D = 50$. Since there are no other transport options than choosing a car, we clearly have the demand for car travel to be $D_c = D = 50$. It is also obvious that the corresponding travel time by car, t_c, will be

$$t_c(50) = 5\left(1 + 0.5\left(\frac{50}{75}\right)^2\right) = 6.11$$

We can summarise this, by stating that given a demand of 50, the resulting demand and cost is given by $(D_c^*, t_c^*) = (50, 6.11)$.

□

2.1.1 The BPR Formula

The formula used for calculating travel times in example 2.1.1 is a very important formula, called the *BPR Formula.* The formula is a very common way of calculating travel times. It says that for a specific link l, the travel time by car can be calculated by

$$t_{l,c}(D_{l,c}) = t_{0,l}\left(1+\alpha_l\left(\frac{D_{l,c}}{C_l}\right)^{\beta_l}\right).\text{[Ric10]} \tag{2.1.2}$$

Here $t_{0,l}$ is the so called *free flow* travel time of the link, meaning the travel time required given that there are no other cars on the road. Furthermore, α_l and β_l are constants, and in this study $\alpha_l = 0.5$, $\forall l$, whereas $\beta_l \in \{2, 4, 6\}$ is link specific. Finally, C_l is the link specific capacity for cars [1], while $D_{l,c}$ is the demand for cars on link l.

The model tells us, that as traffic increases each vehicle have to drive slower in order to prevent crashing, leading to a higher travel time. Since $\frac{D_{l,c}}{C_l} < 1$, α_l serves as the maximum percentage of travel time added by congestion. In our case it is assumed that it takes 50% longer to travel on a fully congested link, corresponding to a speed reduction of 33%. β is a measure of the curvature of the travel time. For a high β, the first amount of traffic have almost no influence, whereas the travel times increases fast for high demands, see figure 2.2.

It should be stressed that the formula is only used to determine the travel time for cars. The travels time for public transportation are assumed to be equal to the free flow travel time, regardless of the demand,

$$t_{l,p} = t_{0,l}. \tag{2.1.3}$$

2.2 The General 4-Stage Model

The BPR formula tells how to calculate the corresponding travel time of a link to any demand of the link. Unfortunately, in order to use the formula, it is necessary to have a model that can determine the demand for the links.

If demand ($D(t)$) and travel time ($t(D)$) had been two simple functions, a solution – (D^*, t^*) satisfying $t(D^*) = t^*$ and $D(t^*) = D^*$– could have been found

[1] The capacity of a link can both be interpreted as the total capacity including both directions or only one of them. For a discussion about this, see the end of section 4.2.3.

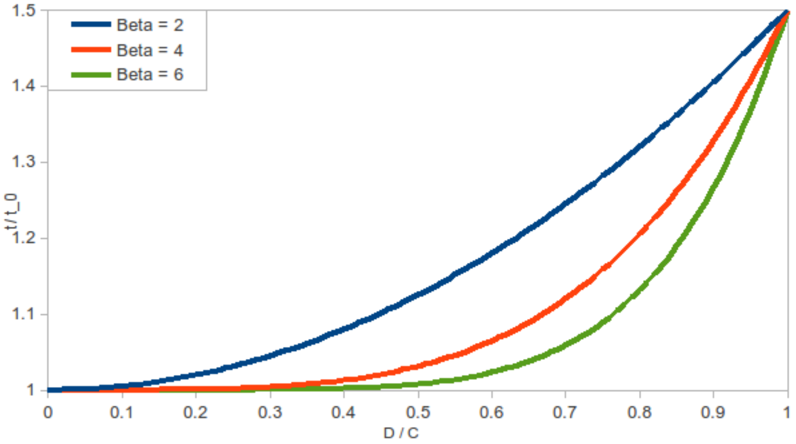

Figure 2.2: Travel time as a function of the ratio between demand and capacity for various values of β.

analytically. Unfortunately, they interact on each other in a very complicated way, and as such, the solution must be found by an iterative algorithm instead, see chapter 1.

The model needed is called *4-stage model*. It consists of the following 4 steps

- Trip generation,
- Trip distribution,
- Mode choice,
- Route assignment. [Ric10]

The above presentation of the model shows the 4 stages of the model, and it is tempting to believe that when all 4 stages have been completed, all demands and travel times have been determined. Unfortunately, this is *not* the case since the 4-stage model in fact is a 4-stage loop that keep repeating until a stable solution has been found.

Although this thesis will only use the last two stages of the model, it is important to know what has been let out. As such a brief explanation of the first two stages follows below.

2.2.1 Trip Generation and Distribution

The trip generation and the trip distribution are the first two steps of the 4-stage model. The trip generation step involves the determination of generated and attracted trips for each node of the network. For instance a node with a lot of work places will attract many trips, whereas a node representing an area on the country side will only have attract and generate few trips.

The trip distribution step describes how the generated and attracted trips are distributed between the different nodes of the network. All else being equal two nodes that are close to each other will be more likely to share the demand between them than two nodes that are far apart.

It is beyond the scope of this study to dig any further into these two steps, since this study only deals with a small fictive test network (see section 4.1). In this case, the trip generation and distribution are unimportant as long as the total demand has a realistic size relative to the network. How a realistic total demand has been determined in this study can be seen in section 4.2.3, whereas section 4.3 shows how a random trip distribution has been created by the use of Monte Carlo simulation.

2.3 The Reduced 2-Stage Model

It should be stressed clearly, that due to the approach of Monte Carlo simulation chosen in this study, the 4-stage model reduces to a 2-stage model, since the trip generation and distributions will be constant and as so are uninfluenced by the mode and route choice steps.

Thus ignoring the trip generation and trip distribution steps, the 4-stage model reduces to a 2-stage model consisting of two loops,

- Mode choice (the *external* loop).
- Route choice (the *internal* loop).

Throughout the study, the terms internal and external loop will be used frequently.

2.3.1 Mode Choice – The External Loop

The mode choice normally follows the trip distribution step, meaning that the current demand between each pair of nodes in the network (a so-called *OD pair*) has been determined. In our case using the 2-stage model, the total demand between any OD pair in the network, D_Ω, has been determined. The mode choice step now seeks to determine how many of these people will choose a specific transportation mode between each OD pair.

In practice there are extraordinarily many transportation modes available. For short distances people might choose to use roller skates, skateboards, bicycles or to walk. All these transportation modes will be unrealistic for long distances, and for certain trips - like crossing the Atlantic - planes will be the only realistic option.

In this study, we will only consider car travel and transportation by public transport. We could have included different type of public transport, but since this would not change the probability of choosing a car, it is unnecessary to include since this study focuses on car travel.

How people are assumed to choose between car and public transport are dependent on the average travel time of both modes (and will be explained in section 2.4).

If an iteration of the mode choice stage has come to the conclusion, that the majority of the demand between an OD pair is to be carried out by car, the corresponding average travel time by car will be very high due to congestion, see eq. (2.1.2), p. 7. In the next iteration of the mode choice stage, people will be more likely to use public transport because it is a relatively better option than in the previous iteration. This pattern continues throughout the iteration process. Therefore, it is clear that the mode choice stage has to be run several times before the external solution for each OD pair, $(D^*_{\Omega,c}, t^*_{\Omega,c})$, can be determined.

2.3.2 Route Choice – The Internal Loop

The route choice step follows the mode choice step. This means that it is known how many people will travel between each OD pair using each transportation mode.

People are assumed to be more likely to choose a route with a low travel time than a route with a high travel time (exactly how will be presented in section

2.4). The travel times for public transportation are almost independent of the amount of people choosing that route, and it is customary to assume that the travel times of public transportation is fully independent of the demand. This assumption has also been chosen in this study, see section 2.1.2.

As introduced in equation (2.1.2), section 2.1.1, the travel time on a link is dependent on the demand of the link. Thus, if initially most people are assumed to choose a route A because it has a low travel time, the links of route A become congested and the travel time increases. Now the situation has changed, and route A might be a rather slow route forcing people to choose a different route.

It is therefore clear, that for any specific mode demand between each OD pair, Ω, the route choice step has to run several iterations in order to give the internal solution for $(D^{\diamond}_{\Omega,c}, t^{\diamond}_{\Omega,c})$. In other words, for every iteration of the external loop, the internal loop has to run until convergence has been reached. This issue will be illustrated in example 2.4.4, section 2.4.3.

2.4 Discrete Choice Models

So far the mode and route choice steps have been introduced briefly, but no specific formula or method for determining the probability for each choice has been presented. A formula capable of calculating this is called a *discrete choice model*, and such a model will be presented in section 2.4.2.

However, in order for any discrete choice model to be able to calculate the probability of a certain choice, the concept of utility functions have to be introduced.

2.4.1 Utility Functions

In order to choose between two or more choices, it seems like a good idea to have a measure of "how good" each choice is. This is exactly the purpose of the so-called *utility functions*. For each option, O, the corresponding utility function, U_O, contains information about the utility achieved by choosing option O. A typical utility function for a transportation option is given by

$$U_O = \gamma t_O + \kappa_O, \tag{2.4.1}$$

where t_O is the travel time of option O. γ is a measure of the time elasticity, and will always be negative. It can be interpreted such that if $|\gamma|$ is big, then the demand of option O will be very sensitive to changes in t_O. κ_O is the option preference constant which emulates that some options have higher demand due to comfort, safety etc., even though the travel time is the same. For more information about the exact values of γ and κ_O used in this study, see section 4.2.1 and section 4.2.2.

2.4.2 The Logit Model

Having introduced the concept of utility functions, it is possible to introduce a discrete choice model. One of the most widely used methods is the *logit model*. Denoting an option by the letter O, the probability of choosing option O is then given by

$$P_O = \frac{e^{U_O}}{\sum\limits_{\forall i} e^{U_i}}. \tag{2.4.2}$$

It is easy to show that $P_O \in]0,1[, \forall O$ and $\sum\limits_{\forall O} P_O = 1$, making the logit model a valid discrete probability density function, see [JFM11].

Alternative models do exist, but dealing with more than one discrete choice model would be beyond the scope of this study. For more information on these models, such as the generalized extreme value models, the probit model and the mixed logit model, see [Tra09].

So far it has been assumed that each choice has a deterministic utility, U_O. This method is overly simplified, and in reality not all people will have the same utility for this choice. In order to correct for this, a stochastic error term can be added to the utility function. For person i the utility for option O can then be written as,

$$U_{O,i} = U_O + \varepsilon_{O,i},$$

where $\varepsilon_{O,i}$ is a randomly distributed term. If person i is faced with a number of options, it can then be assumed that person i chooses option O, if and only if $U_{O,i} > U_{U,i}, \forall U \neq O$. This is equivalent to

$$\begin{aligned} U_O + \varepsilon_{O,i} &> U_U + \varepsilon_{U,i} \qquad \Leftrightarrow \\ \varepsilon_{O,i} - \varepsilon_{U,i} &> U_U - U_O. \end{aligned}$$

The probability of the above happening is determined by the distribution of $\varepsilon_{O,i}$ (which is the same as for $\varepsilon_{U,i}$). It is commonly assumed that the error terms come from the *standard Gumbel distribution* given by

$$f(\varepsilon_{O,i}) = e^{-\varepsilon_{O,i}} e^{-e^{-\varepsilon_{O,i}}}, \qquad F(\varepsilon_{O,i}) = e^{-e^{-\varepsilon_{O,i}}}.$$

When this is the case, it turns out that $P(U_{O,i} > U_{U,i}), \forall U \neq O$ is exactly the logit model as given in (2.4.2). Thus, by using the logit model we actually emulate the fact that the choices are probabilistic. [Tra09]

A slightly different approach is to simulate every error term for each person stochastically, and then make the person choose the choice with the higher utility. Although this would increase realism and possibly decrease the number of external iterations required to converge, this study will stick to making probabilistic discrete choices based on the logit model. The reason is that this thesis is solely dealing with a comparative study of the performance of an alternative algorithm for the external loop. A stochastic approach is most likely not to strongly favour one of the methods over the other, and as such the improved performance would be unimportant. Also the added noise from the stochastic error terms could possibly make it harder to find any patterns revealing why the new method is performing as it is.

For more information on stochastic simulation, see [LK00] for a general approach or [Tra09] for stochastic simulation for discrete choice models.

2.4.3 Examples of Simple Transport Networks Requiring Discrete Choices

Now that we know how to choose between two different options, we are ready to extend example 2.1.1 by adding public transport to the network.

Example 2.4.3 (Simple network requiring mode choice)
We consider the same transport network as in example 2.1.1 with the addition of a train running between the two cities, see figure 2.3.

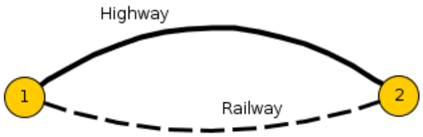

Figure 2.3: The transport network of example 2.4.3

Regardless of the amount of passengers on the train, it takes the train 5 time units to travel between the two cities (t_p=5). We still have D = 50 and the travel time on the road given by

$$t_c(D_c) = 5\left(1 + 0.5\left(\frac{D_c}{75}\right)^2\right).$$

Additionally, the following constants have been given

$$\gamma = -1, \qquad \kappa_c = 1.5, \qquad \kappa_p = 0.$$

Since there is no route choice involved in this example, our 2-stage models reduces to a 1-step model where only the mode choice is under consideration.

The way to get through the first iteration of the mode choice step is to assume there is no traffic on the network, and calculating the corresponding travel times. In this case we will have $t_c = t_p = 5$, which yields the following utility functions.

$$\begin{aligned} U_c &= \gamma t_c + \kappa_c = -1 \cdot 5 + 1.5 = -3.5, \\ U_p &= \gamma t_p + \kappa_p = -1 \cdot 5 + 0 = -5. \end{aligned}$$

Based on these utility functions, the initial probability of choosing a car over public transportation can be calculated by

$$P_c = \frac{e^{-3.5}}{e^{-3.5} + e^{-5}} = 0.8176.$$

And since the car demand is equal to the overall demand times the probability of choosing a car, the following demand is obtained by

$$D_c = 0.8176 \cdot 50 = 40.88.$$

This demand is then used to calculate the next travel etc. This process has to run for several iterations before convergence is reached. The calculations of these will be omitted, but the iteration pattern can be seen in figure 2.4. The solution to this system turns out to be

$$(D_c^*, t_c^*) = (35.84, 5.57).$$

□

So far we have seen how to handle a transport network with a single link and with public transport. In reality, it is very seldom that there is only one possible route to the destination. This next example, which will be the first example where both loops of the 2-stage model will be active, seeks to show how to handle such a case.

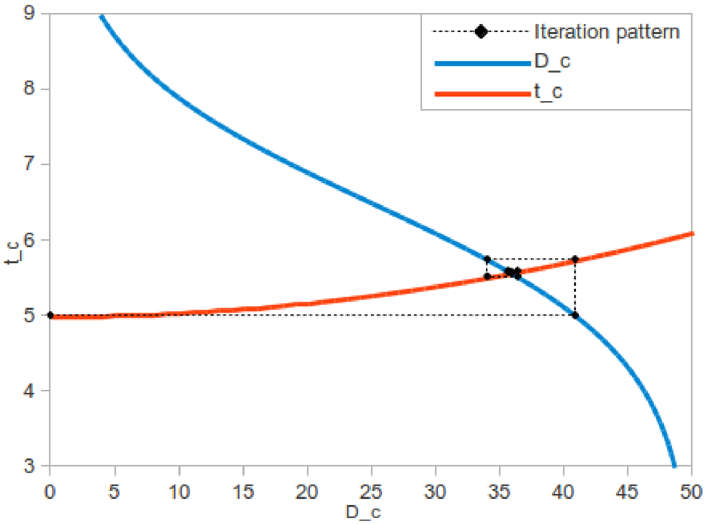

Figure 2.4: Iteration pattern of example 2.4.3

Example 2.4.4 (Simple network requiring mode and route choice)

In this example we continue with the setup from 2.4.3, but with a minor change. A new freeway between the two cities has been build, see figure 2.5. Under free flow conditions, it takes 4.5 time units to travel it, and it has a capacity of 100. It has a β parameter equal to 4, so that the travel time of the new link (link 2) becomes

$$t_{2,c} = t_{0,l}\left(1 + 0.5\left(\frac{D_{2,c}}{C_2}\right)^{\beta_2}\right) = 4.5\left(1 + 0.5\left(\frac{D_{2,c}}{100}\right)^4\right).$$

Due to so-called *induced demand* (see [Ger09]) caused by the newly built freeway between the two cities, the demand have increased to reach $D = 100$.

As in example 2.4.3, we first assume that there is no traffic on the network, and then determine the corresponding travel times, $t_{1,c} = t_p = 5$ and $t_{2,c} = 4.5$.

As known from before, the next step is to calculate the utility function for car travel and public transportation. Normally, the travel time (t_c) used in the utility function of cars would be the demand-weighted average of the travel times of the possible routes. Since, in this first iteration, both demands are assumed to be 0, we simply use the average given by $t_c = \frac{t_{1,c}+t_{2,c}}{2} = 4.75$. This

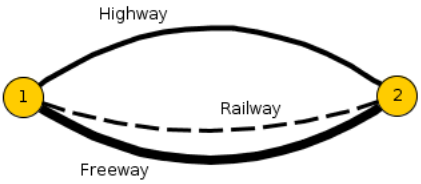

Figure 2.5: The transport network of example 2.4.4

gives us the following utility functions.

$$U_c = \gamma t_c + \kappa_c = -1 \cdot 4.75 + 1.5 = -3.25,$$
$$U_p = \gamma t_p + \kappa_p = -1 \cdot 5 + 0 = -5.$$

The corresponding demand is then

$$D_c = 100 \cdot \frac{e^{-3.25}}{e^{-3.25} + e^{-5}} = 85.20.$$

Now, since we have more than 1 possible route for the car demand, we have to initiate the route choice stage. In order to choose between the routes, the logit model is used, and it yields

$$D_{1,c} = 85.20 \cdot \frac{e^{-5}}{e^{-5} + e^{-4.5}} = 37.30$$
$$D_{2,c} = D - D_{1,c} = 47.90.$$

By using these demands, we can calculate the corresponding travel times of each link

$$t_{1,c} = 5 \cdot \left(1 + 0.5\left(\frac{37.30}{75}\right)^2\right) = 5.62$$
$$t_{2,c} = 4.5 \cdot \left(1 + 0.5\left(\frac{47.90}{100}\right)^4\right) = 4.55.$$

These travel times can then be used to calculate the demand of both routes.

Since the internal loop has to run for many iterations, the calculations will be skipped, and instead figure 2.6 shows the iteration pattern of the internal loop. It can be see that the method converges towards the internal solutions $(D^{\diamond}_{1,c}, t^{\diamond}_{1,c}) = (29.13, 5.38)$ and $(D^{\diamond}_{2,c}, t^{\diamond}_{2,c}) = (56.07, 4.72)$.

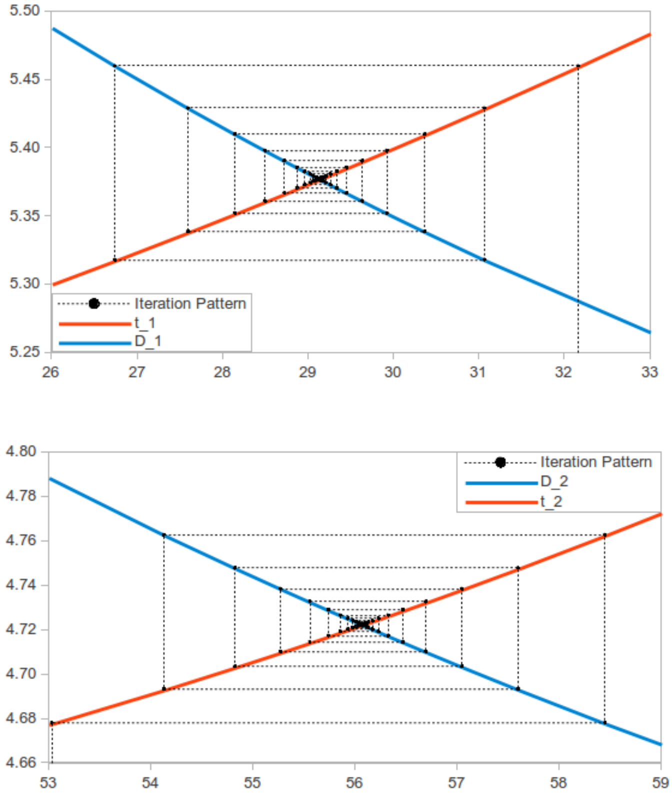

Figure 2.6: Iteration pattern of link 1 and 2 of example 2.4.4.

Using the internal solution, the average travel time by car can be calculated as

$$t_c = \frac{D_{1,c}t_{1,c} + D_{2,c}t_{2,c}}{D} = \frac{29.13 \cdot 5.38 + 56.07 \cdot 4.72}{85.20} = 4.95.$$

This can be used to determine the car demand of the next external iteration. The example can be completed by repeating the above process until the difference in the car demand is sufficiently small.

□

2.5 Networks with Multiple OD Pairs

After studying the last two examples, the required prerequisites to understand a complex transport system are almost fulfilled. One thing still needs to be introduced, though.

In example 2.4.4 we allowed multiple possible routes for the car demand. Both of these routes only consisted of 1 link, connecting the same two nodes. In a more advanced transport network, there are several nodes, and thus also a lot of OD pairs.

When going through the 2-stage model of a transport network with multiple OD pairs, each stage of the model has to be completed for each OD pair before continuing. This is due to the fact that the system can not be solved for 1 isolated OD pair, since the solution is dependent on the solution of the other OD pairs. For more information about this see section 6.2.

When there are multiple OD pairs, there will also be routes consisting of several links. In this case, the demand for each link is simply the sum of the demand of all routes containing this link. Formally, this can be written as

$$D_{l,c} = \sum_{\forall r | l \in G_r} D_{r,c}.$$

Using this information, the travel time for each link can be calculated by the BPR formula (see section 2.1.1),

$$t_{l,c} = t_{0,l,c} \left(1 + \alpha_l \left(\frac{D_{l,c}}{C_l} \right)^{\beta_l} \right).$$

This allows the calculation of the travel time for each route, by simply adding the travel times of the links contained in the route

$$t_{r,c} = \sum_{\forall l \in G_r} t_{l,c}.$$

And finally, by knowing the travel time of each route, the demand for each route can be calculated as

$$D_{r,c} = D_{\Omega_r,c} \frac{e^{\gamma t_{r,c}}}{\sum\limits_{\forall s \in G_{\Omega_r}} e^{\gamma t_{s,c}}},$$

where Ω_r is the OD pair corresponding to route r.

Chapter 3

Methods for Determination Transport Network Equilibria

As seen in the preceding chapter, when determining the demand and travel time of all OD pairs, routes and links of a transport network, an iterative approach has to be used. This approach consists of an external and an internal loop. The standard way of dealing with this is presented in section 3.1. In section 3.2, an alternative algorithm for the external loop (the I method) will be presented. Finally, some suggestions for alterations which have been considered during the development of the algorithm will be presented in section 3.3.

3.1 The Standard Method

After having introduced the fundamentals of transportation demand analysis in the preceding sections, we are now ready to introduce what will be referred to as the *standard method* in the rest of this study. Later on in this section, we will take a look at the standard method using the method of successive (weighted) averages (MS(W)A), but for now the method of repeated approximations will be presented.

3.1.1 MRA - The Method of Repeated Approximations

The Method of Repeated Approximations, generally known as (MRA), sounds much more complicated than it is. It is actually just the pure iterative approach used so far. The name comes from the fact, that for every stage of the model, the approximations are based on the former approximations. Since these stages repeat themselves gradually making better and better approximations, the method is called the method of repeated approximations. In order to really understand the I method, it is needed to get an understanding of the standard method, and how MSA is applied in this case.

It is the simplest case and does not use any trends what so ever. One can think of the method as if it was taking a snapshot of the transport network after each stage in the 2-stage model. Based on this snapshot, it makes the best possible approximation, and takes another snapshot.

A more structured review of the algorithm follows below.

Algorithm 3.1.1 (The standard method using MRA)

1. *For each OD-pair, calculate the average free flow travel times,* $t_{\Omega,c,1}$.
2. *Set* $k = 1$.
3. *For each OD-pair, calculate the k'th demand for car traffic,* $D_{\Omega,c,k}$*, based on the latest average travel time by cars,* $t_{\Omega,c,k}$*. That is* $D_{\Omega,c,k} = D_{\Omega,c}(t_{\Omega,k})$.
4. *Increment k by 1.*
5. *For each OD-pair, run the internal (route assignment) loop in order to calculate the new average travel times,* $t_{\Omega,c,k}$.
6. *Repeat step 3-5 until* $||D_{\Omega,c,k} - D_{\Omega,c,k-1}|| \leq \tau$ *for all OD-pairs or* $k \geq k_{max}$.

An example of the iteration process of the standard method of repeated approximations for a single OD pair can be seen in figure 3.1. The figure also shows the iteration process of the method of successive averages, which will be the topic of the following subsection.

3.1.2 MSA - The Method of Successive Averages

The method of repeated approximations is the most basic approach, and in some cases it turns out to be too simple, in the sense that under certain conditions the standard methods may fail to converge due to the steepness of the demand and cost curves. In these cases, convergence can be secured by using a very simple successive approach, called the method of successive averages (MSA).

Let I_k denote the k'th iteration using MSA. Furthermore, let $\hat{I}_{k+1}$ denote the values obtained by applying MRA on I_k. Then the MSA alters the next iteration in the following way

$$I_k = a_k \hat{I}_k + (1 - a_k) I_{k-1}, \quad a_k \in]0, 1]. \tag{3.1.2}$$

The a_k's are chosen such that a_k is a decreasing convex functions of k, and $k \in \mathbb{N}^+$.

Studies ([BC01, LHH09, Can97, CP01] among others) have shown that the number of external iterations required can be lowered dramatically by applying MSA averaging to the standard method. There are two obvious ways to apply the MSA averaging - either applying it to the demand function or to the cost function. Earlier studies have shown that applying it to the cost function is far more efficient and robust than applying it on the demand. [RNC13]

The iteration pattern of the MSA method can be seen in figure 3.1. It is seen that the convergence is much faster than that of the MRA method. It almost seems as if the MRA method is pulled towards the solution. The reason is that the it does not go all the way down or up to the cost function, but remains somewhere in between. The next iteration takes it back on level with the demand function, but in a place much closer to the solution than the MRA would have done. According to [RNC13] it can be considered as a type of contraction, although it differs from the mathematical definition of a contraction.

A common and very straightforward way to calculate a_k, is by letting $a_k = \frac{1}{k}$. When using this formula, it can be shown by a proof of induction, that for any k, all iterations are weighted exactly equally.

Lemma 3.1.3 *Let I_k denote the k'th iteration using MSA with $k \in \mathbb{N}^+$. Furthermore, let $\hat{I}_k$ be the values obtained by using MRA on I_{k-1} for $k \geq 2$ and $\hat{I}_1 = I_1$. If I_k is calculated by eq.* (3.1.2) *with $a_k = \frac{1}{k}$, then*

$$I_k = \frac{1}{k} \sum_{j=0}^{k-1} \hat{I}_{j+1}.$$

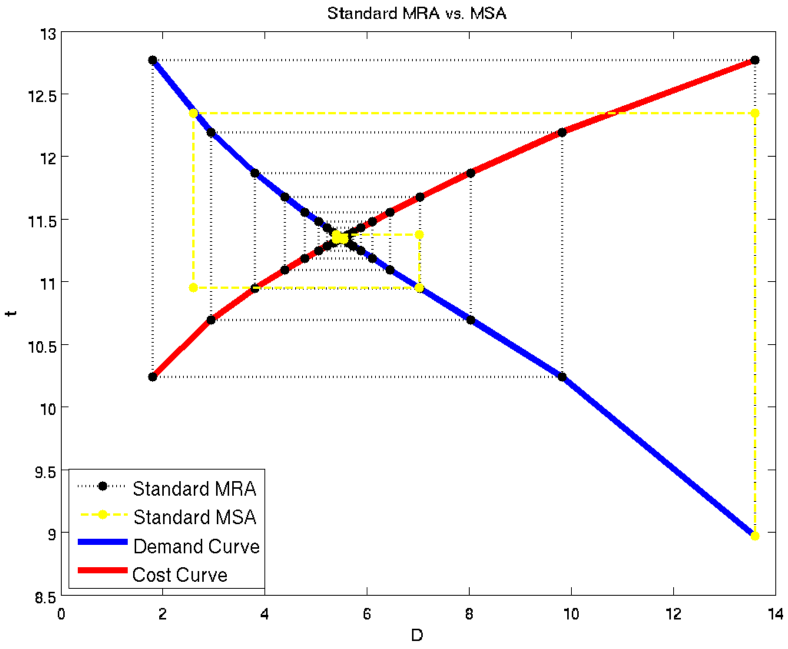

Figure 3.1: Comparison of the standard method using MRA and MSA.

Proof. It clearly holds for $k = 1$, since

$$I_1 = \frac{1}{1} \cdot \hat{I}_1 = \frac{1}{1} \sum_{j=1}^{1} \hat{I}_j.$$

Assuming that it holds for $k-1$, it can be shown that it also holds for k,

$$\begin{aligned} I_k &= \frac{1}{k} \cdot \hat{I}_k + \left(1 - \frac{1}{k}\right) \cdot I_{k-1} \\ &= \frac{1}{k} \cdot \hat{I}_k + \frac{k-1}{k} \cdot \frac{1}{k-1} \sum_{j=1}^{k-1} \hat{I}_j \\ &= \frac{1}{k} \sum_{j=1}^{k} \hat{I}_j. \end{aligned}$$

□

Another common way to calculate the MSA coefficient, is by using

$$a_k = \frac{1}{k^d}, \quad d \leq 1.$$

By selecting $d = \frac{2}{3}$, corresponding to $a_k = k^{-\frac{2}{3}}$, the method introduced in [Pol90] is obtained. Since $d \leq 1$ the Polyak-type MSA coefficient converges to 0 slower than than the conventional MSA coefficient. Numerical experiments of [BC01] show that the Polyak-type MSA coefficient benefits from this by being more efficient than the conventional MSA approach.

However, in [LHH09] it is suggested, that although $a_k = k^{-\frac{2}{3}}$ does perform well, it may converge to zero too slowly to be optimal (see figure 3.2).

3.1.3 MSWA - The Method of Successive Weighted Averages

In order to adjust for the slow decay of the Polyak-type MSA coefficient, [LHH09] introduces an alternative way of calculating the MSA coefficient, called *the method of successive weighted averages (MSWA)*. In this case, the MSA coefficient is given by

$$a_k = \frac{k^d}{\sum_{j=1}^{k} j^d}, \quad d \geq 0. \tag{3.1.4}$$

When $d = 0$, the method equals standard MSA averaging,

$$a_k = \frac{k^0}{\sum_{j=1}^{k} j^0} = \frac{1}{\sum_{j=1}^{k} 1} = \frac{1}{k}.$$

Figure 3.2 compares the MSA coefficient of [Pol90] (to the left) with the MSA coefficient of [LHH09] (to the right). It is seen that for the [LHH09]-type MSA coefficient, the higher the d, the slower the decrease of a_k. The opposite holds for the MSA coefficient of [Pol90]. Numerical experiments from [LHH09] gives ambiguous results, but indicate that MSA coefficients of this type may be preferable compared to the Polyak-type MSA coefficient. As such this thesis will use MSA coefficients based on the theory from [LHH09].

In general, when d is chosen as an integer, a closed form expression exists. Since $d = 1, 2, 3, 4, 5$ will be used in this study, the closed form expressions of these are calculated below. The formulas for $\sum_{j=1}^{k} j^d$ for various d has been used in

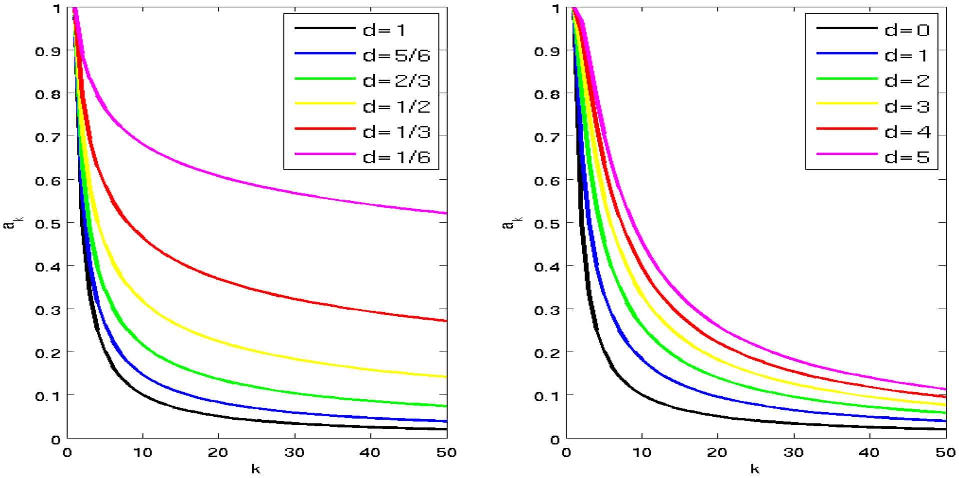

Figure 3.2: The Polyak type (left) and Liu type (right) MSA coefficient as a function of k for various values of d.

the calculations (see [Wei]).

$$\frac{k^1}{\sum_{j=1}^{k} j^1} = \frac{k}{\frac{1}{2}k(k+1)} = \frac{2}{k+1},$$

$$\frac{k^2}{\sum_{j=1}^{k} j^2} = \frac{k^2}{\frac{1}{6}k(k+1)(2k+1)} = \frac{6k}{(k+1)(2k+1)},$$

$$\frac{k^3}{\sum_{j=1}^{k} j^3} = \frac{k^3}{\frac{1}{4}k^2(k+1)^2} = \frac{4k}{(k+1)^2},$$

$$\frac{k^4}{\sum_{j=1}^{k} j^4} = \frac{k^4}{\frac{1}{30}k(k+1)(2k+1)(3k^2+3k-1)} = \frac{30k^3}{(k+1)(2k+1)(3k^2+3k-1)},$$

$$\frac{k^5}{\sum_{j=1}^{k} j^5} = \frac{k^5}{\frac{1}{12}k^2(k+1)^2(2k^2+2k-1)} = \frac{12k^3}{(k+1)^2(2k^2+2k-1)}.$$

The algorithm of the standard method using MSWA on the cost function is presented below.

Algorithm 3.1.5 (The standard method using MSA on $t_{\Omega,c}$)

1. *For each OD-pair, calculate the average free flow travel times, $t_{\Omega,c,1}$.*
2. *Set $k = 1$.*
3. *For each OD-pair, calculate the k'th demand for car traffic, $D_{\Omega,c,k}$, based on the latest travel times, $t_{\Omega,c,k}$. That is $D_{\Omega,c,k} = D_{\Omega,c}(t_{\Omega,c,k})$.*
4. *Increment k by 1.*
5. *For each OD-pair, perform the internal (route assignment) loop in order to calculate the new travel time, $t_{\Omega,c,k}$.*
6. *Calculate a_k.*
7. *For each OD-pair, set $t_{\Omega,c,k} = a_k \hat{t}_{\Omega,c,k} + (1 - a_k) t_{\Omega,c,k-1}$.*
8. *Repeat step 3-7 until $||D_{\Omega,c,k} - D_{\Omega,c,k-1}|| \leq \tau$ for all OD-pairs or $k \geq k_{max}$.*

Since MSWA is a also a method of successive averages, the "W" is only used to differentiate it from an equally weighted MSA. For the sake of simplicity, and since this study will not deal with the method of equally weighted successive averages (except for in the internal loop), MSA will be used as an acronym for the method of weighted successive averages as well.

3.1.4 The Route Choice Loop (The Internal Loop)

The preceding algorithms have made use of an internal loop – the route assignment loop. In this loop the demand of every OD pair is spread out on routes according to the theory of section 2.4 and 2.5.

The algorithm uses a standard MSA on the link travel times where the coefficient is calculated as $a_k = \frac{1}{k-1}$. The reason why $a_k = \frac{1}{k-1}$ is used as opposed to $a_k = \frac{1}{k}$, is due to the fact that the loop has to start with $k = 2$ in order for the code to work.

The algorithm assigning routes to the demand between each OD pair can be seen below.

Algorithm 3.1.6 (Internal loop / Route assignment loop)

1. *Let $t_{r,c,2} = t_{r,p}$.*

2. *Set* $k_i = 2$
3. *Calculate* $D_{r,c,k_i} = D_{\Omega_r,c,k} \frac{e^{\gamma t_{r,c,k_i}}}{\sum\limits_{r \in G_{\Omega_r}} e^{\gamma t_{r,c,k_i}}}$
4. *Calculate* $D_{l,c,k_i} = \sum_{\forall r | l \in G_r} D_{r,c,k_i}$.
5. *Calculate* $a_{k_i} = \frac{1}{k_i - 1}$.
6. *Calculate* $t_{l,c,k_i} = a_{k_i} t_{0,l} \cdot \left(1 + \alpha_l \left(\frac{D_{l,c}}{C_l}\right)^{\beta_l}\right) + (1 - a_{k_i}) t_{l,c,k_i}$.
7. *Increment* k_i *by 1.*
8. *Calculate* $t_{r,c,k_i} = \sum_{l \in G_r} t_{l,c,k_i - 1}$
9. *Repeat 3-8 until* $|D_{\Omega,c,k_i} - D_{\Omega,c,k_i - 1}| < \tau_i$ *for all* Ω *or* $k_i > k_{i,max}$.
10. *Set* $t_{r,c,2} = t_{r,c,k_i}$*, and begin next loop from step 2.*

It is worth noting that the above algorithm uses the route assignment from the last external iteration as the initial guess for the next iteration. This can decrease the number of internal iterations dramatically, but can also in some cases lead to instability that may make the internal loop fail to converge.

Better methods to obtain convergence of the internal loop probably exist, but it has not been the focus of this study which focuses primarily on convergence of the external loop. It may lead to methods struggling to converge in the route choice loop, and during the performance tests towards the end of the study (see section 5), it will be necessary to take into account that the internal loop might not be constructed ideally.

3.1.5 Fake Convergence

When using iterative algorithms without having any prior knowledge about the actual solution, a stopping criteria based on the latest iterations of the algorithm is needed. Generally, a stopping criteria is a criteria, that indicates that not much will change if the algorithm kept running for a longer time. In other words, when the stopping criteria is fulfilled, the latest iterations were more or less identical, and the current guess is probably as good as it gets.

One known danger of using such a stopping criteria, is that an algorithm might fulfil the stopping criteria even though it is not near the actual solution – especially, if using heavy MSA. This situation will be denoted as *"fake convergence"* in this study. The cause behind this phenomenon becomes evident by rewriting the left-hand side of the stopping criteria,

$$\begin{aligned}||D_{\Omega,c,k} - D_{\Omega,c,k-1}|| &= ||D_{\Omega,c}(t_{\Omega,c,k}) - D_{\Omega,c}(t_{\Omega,c,k-1})|| \\ &= ||D_{\Omega,c}(a_k\hat{t}_{\Omega,c,k} + (1-a_k)t_{\Omega,c,k-1}) - D_{\Omega,c}(t_{\Omega,c,k-1})||.\end{aligned}$$

Since $||D_{\Omega,c}(a_k t_{\Omega,c,k} + (1-a_k)t_{\Omega,c,k-1}) - D_{\Omega,c}(t_{\Omega,c,k-1})||$ will be close to 0, when a_k is close to 0, a too fast decreasing a_k might suffice the stopping criteria solely because a_k is low, and not because $D_{\Omega,c,k} \approx D^*_{\Omega,c}$, $\forall\Omega$.

The case is even more extreme, if the MSA averaging is used directly on the demands. Then the stopping criteria becomes

$$\begin{aligned}\tau &\geq ||D_{\Omega,c,k} - D_{\Omega,c,k-1}|| \\ &\geq ||a_k\hat{D}_{\Omega,c,k} + (1-a_k)D_{\Omega,c,k-1} - D_{\Omega,c,k-1}|| \\ &\geq a_k||\hat{D}_{\Omega,c,k} - D_{\Omega,c,k-1}||,\end{aligned}$$

which can be rearranged to yield

$$||\hat{D}_{\Omega,c,k} - D_{\Omega,c,k-1}|| \leq \frac{\tau}{a_k}.$$

In this case it is pretty clear, that this can be fulfilled if either $||\hat{D}_{\Omega,c,k} - D_{\Omega,c,k-1}||$ or $||a_k||$ is sufficiently small. Remember that $D_{\Omega,c,k-1}$ is an averaged value including information of every demand occurred in the iteration process, and as such $||\hat{D}_{\Omega,c,k} - D_{\Omega,c,k-1}||$ is a measure of how close $\hat{D}_{\Omega,c,k}$ is to the current best guess of $D^*_{\Omega,c}$. Only if this difference is small, convergence has been reached righteously.

The best way to avoid fake convergence is to avoid extremely fast descending MSA functions. In the case of Liu-type MSA it will be to prevent a too low value of d, see figure 3.2, p. 24. $d = 0$ corresponds to the standard MSA averaging, and as a general rule of thumb one should be careful when using d-values lower than this, which is also suggested in that article since $d \geq 0$.

3.2 The I Method – The Method of Intersection of Straight Line Approximations of Demand and Cost Curves

In this section the method of intersection of straight line approximations of demand and cost curves (*the I method*) will be introduced. Later on in the section, a derivation and an algorithm of the method will follow. But firstly, the general idea of the I method will be presented.

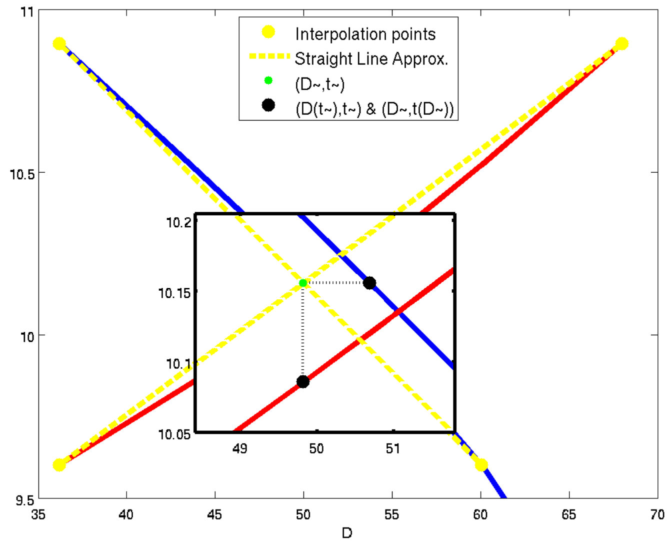

Figure 3.3: The first intersection of straight line approximations using the I method.

3.2.1 The Idea of the I Method

In [RNC13] an idea for a new iterative algorithm is presented. The idea is to let the 4 latest points of the iteration process approximate the system linearly, in the sense that one straight line approximates the demand curve, and another straight line approximates the cost function. The intersection of these two lines (denoted by $(\tilde{D}_{\Omega,c}, \tilde{t}_{\Omega,c})$) is expected to be a good approximation of $(D^*_{\Omega,c}, t^*_{\Omega,c})$, hopefully resulting in an algorithm requiring few iterations to converge.

It is obvious that neither the cost function nor demand function is a straight line itself, but on the majority of both curves, the assumption of linearity is expected to be decent. When the functions are not formed by straight lines, the intersection $(\tilde{D}_{\Omega,c}, \tilde{t}_{\Omega,c})$ might not lie on neither the demand nor cost function. As such, the method will not use $(\tilde{D}_{\Omega,c}, \tilde{t}_{\Omega,c})$ as the next point directly. Instead each of the coordinates will be used as input in the other function, such that the two resulting points from the iteration will be $(\tilde{D}_{\Omega,c}, t(\tilde{D}_{\Omega,c}))$ and $(D(\tilde{t}_{\Omega,c}), \tilde{t}_{\Omega,c})$.

The method might be quite hard to understand without a visual sketch of the method. Such a sketch can be seen in figure 3.3 and 3.4, where the first two iterations have been plotted for the first two iterations of the I method for a

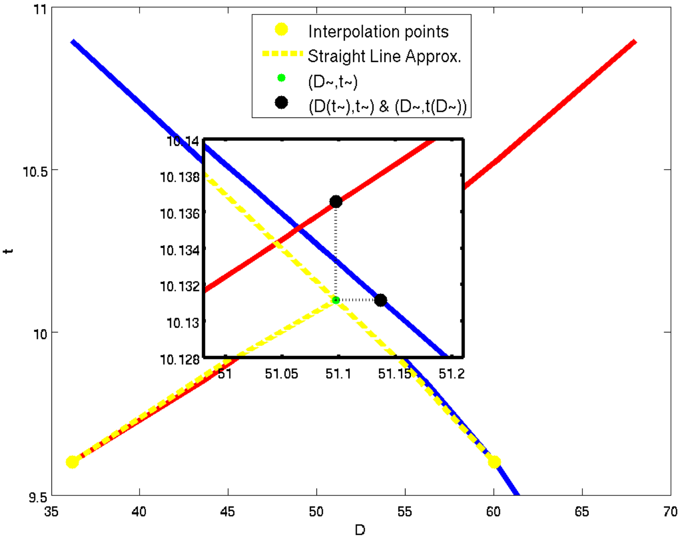

Figure 3.4: The second intersection of straight line approximations using the I method.

single OD pair. The yellow dots are the interpolation points, which are connected by the yellow interpolation lines. The green dot marks the intersection point $(\tilde{D}_{\Omega,c}, \tilde{t}_{\Omega,c})$, whereas the black dots are the evaluations of the intersection points, $((D(\tilde{t}_{\Omega,c}), \tilde{t}_{\Omega,c}))$ and $((\tilde{D}_{\Omega,c}, t(\tilde{D}_{\Omega,c})))$.

As seen in the figures, the I method can not be applied initially, since it requires 2 interpolation points per function, yielding a total of 4 points. The first 4 points using the standard MRA are $(D_{\Omega,c,1}, t_{\Omega,c,1})$, $(D_{\Omega,c,1}, t_{\Omega,c,2})$, $(D_{\Omega,c,2}, t_{\Omega,c,2})$, and $(D_{\Omega,c,2}, t_{\Omega,c,3})$. These 4 points are actually sufficient to initiate the I method, but since the fifth point $(D_{\Omega,c,3}, t_{\Omega,c,3})$ can be calculated easily without having to run the internal loop, the fifth point will also be calculated using the standard MRA method. This makes $(D_{\Omega,c,2}, t_{\Omega,c,2})$ and $(D_{\Omega,c,3}, t_{\Omega,c,3})$ the two interpolation points for the demand function, and $(D_{\Omega,c,1}, t_{\Omega,c,2})$ and $(D_{\Omega,c,2}, t_{\Omega,c,3})$ the two interpolation points for the cost function.

3.2.2 Derivation of the intersection point $(\tilde{D}_{\Omega,c}, \tilde{t}_{\Omega,c})$

Now that the general idea and visual introduction to the I method has been presented, a derivation of the straight line approximations and the intersection between them is needed.

Let the points used to approximate the demand curve be denoted by $(D_{\Omega,c,k-1}, t_{\Omega,c,k-2})$ and $(D_{\Omega,c,k}, t_{\Omega,c,k-1})$, and the points used to approximate the cost curve be denoted $(D_{\Omega,c,k-1}, t_{\Omega,c,k-1})$ and $(D_{\Omega,c,k}, t_{\Omega,c,k})$.

Furthermore, let the line approximating the demand and cost curve for OD pair Ω be denoted by y_{D_Ω} and y_{t_Ω}, respectively. Since both are straight lines, it will be possible to write both of them on the form

$$y_{D_\Omega}(D) = b_{D_\Omega} \cdot D + a_{D_\Omega} \tag{3.2.1}$$

$$y_{t_\Omega}(D) = b_{t_\Omega} \cdot D + a_{t_\Omega}, \tag{3.2.2}$$

where the a's and b's are real coefficients.

Using the general formulas for a and b for a line passing through $(x_1, y_1), (x_2, y_2)$,

$$b = \frac{y_2 - y_1}{x_2 - x_1},$$

$$a = y_2 - bx_2,$$

it is easy to compute the needed a and b's:

$$b_{D_\Omega} = \frac{t_{\Omega,c,k-1} - t_{\Omega,c,k-2}}{D_{\Omega,c,k} - D_{\Omega,c,k-1}}, \qquad a_{D_\Omega} = t_{\Omega,c,k-1} - b_{D_\Omega} D_{\Omega,c,k},$$

$$b_{t_\Omega} = \frac{t_{\Omega,c,k} - t_{\Omega,c,k-1}}{D_{\Omega,c,k} - D_{\Omega,c,k-1}}, \qquad a_{t_\Omega} = t_{\Omega,c,k} - b_{t_\Omega} D_{\Omega,c,k}.$$

It is worth noting that the above is not valid if $D_{\Omega,c,k} = D_{\Omega,c,k-1}$. This turns out to be a purely mathematical problem, that does not occur in practice. The stopping criteria of the iteration process is $||D_{\Omega,c,k} - D_{\Omega,c,k-1}|| < \tau$, and the process will only continue as long as there is significant changes in some of the demands. When this is the case, there will also be minor changes in the other OD pairs (see section 6.2).

We will now search for the formula for the intersection point between $y_{D_\Omega}(D)$ and $y_{t_\Omega}(D)$, denoted by $(\tilde{D}_{\Omega,c}, \tilde{t}_{\Omega,c})$. By denoting the D-coordinate of the intersection by $\tilde{D}_{\Omega,c}$ we get,

$$\begin{aligned} b_{t_\Omega}\tilde{D}_{\Omega,c} + a_{t_\Omega} &= b_{D_\Omega}\tilde{D}_{\Omega,c} + a_{D_\Omega} \qquad \Leftrightarrow \\ \tilde{D}_{\Omega,c} &= \frac{a_{D_\Omega} - a_{t_\Omega}}{b_{t_\Omega} - b_{D_\Omega}}. \end{aligned}$$

Yet again, this is not valid if $b_{t_\Omega} = b_{D_\Omega}$. But this is also very unlikely to happen, since b_{t_Ω} is the slope of an increasing function, whereas b_{D_Ω} is the slope of a decreasing function.

By inserting the above into $y_{D_\Omega}(D)$ (or $y_{t_\Omega}(D)$), we get

$$\tilde{t}_{\Omega,c} = b_{t_\Omega}\tilde{D}_{\Omega,c} + a_{t_\Omega}.$$

3.2.3 The I Method Using MRA

After having introduced the thought and the derivation of the I method, a proper algorithm is ready to be introduced. And since any further explanation of the method will provide less information than simply stating the actual algorithm, the algorithm is presented directly below.

Algorithm 3.2.3 (The I method using MRA)

1. *Follow the standard method for the first 3 iterations (that is until $D_{\Omega,c,1}$, $D_{\Omega,c,2}$, $D_{\Omega,c,3}$, $t_{\Omega,c,1}$, $t_{\Omega,c,2}$, $t_{\Omega,c,3}$ are known for all OD-pairs (Ω)).*

2. *For all Ω, set $\tilde{t}_{\Omega,c,3} = t_{\Omega,c,3}, \tilde{t}_{\Omega,c,2} = t_{\Omega,c,2}, \tilde{D}_{\Omega,c,3} = D_{\Omega,c,2}$, and $\tilde{D}_{\Omega,c,2} = D_{\Omega,c,1}$.*

3. *Set $k = 3$.*

4. *For all Ω, calculate $b_{D_\Omega} = \frac{\tilde{t}_{\Omega,c,k} - \tilde{t}_{\Omega,c,k-1}}{D_{\Omega,c,k} - D_{\Omega,c,k-1}}$, $a_{D_\Omega} = \tilde{t}_{\Omega,c,k} - b_{D_\Omega}D_{\Omega,c,k}$, $b_{t_\Omega} = \frac{t_{\Omega,c,k} - t_{\Omega,c,k-1}}{\tilde{D}_{\Omega,c,k} - \tilde{D}_{\Omega,c,k-1}}$, and $a_{t_\Omega} = t_{\Omega,c,k} - b_{t_\Omega}\tilde{D}_{\Omega,c,k}$.*

5. *Calculate $\tilde{D}_{\Omega,c,k+1} = \frac{a_{D,\Omega} - a_{t_\Omega}}{b_{t,\Omega} - b_{D,\Omega}}$ and $\tilde{t}_{\Omega,c,k+1} = b_{t_\Omega}\tilde{D}_{\Omega,c,k+1} + a_{t_\Omega}$.*

6. *Calculate $D_{\Omega,c,k+1} = D_{\Omega,c}(\tilde{t}_{\Omega,c,k+1})$ and $t_{\Omega,c,k+1} = t_{\Omega,c}(\tilde{D}_{\Omega,c,k+1})$.*

7. *Increment k by one.*

8. *Repeat step 4-7 until $||D_{\Omega,c,k} - D_{\Omega,c,k-1}|| \leq \tau$ for all OD-pairs or $k \geq k_{max}$.*

It is worth noting that the I method does *not* use MSA on the first initial iterations prior to the straight line approximations. Although it could take some of the interpolation points closer to the equilibrium point, it would also cause the interpolation points of the cost function to lie far from the cost function.

Also, when the I method performs the internal loop, it should be stressed clearly, that it does not use $D_{\Omega,c,k}$ to calculate the travel times, but $\tilde{D}_{\Omega,c,k}$.

3.2.4 The I Method Using MSA

Algorithm 3.2.3 is the simplest form of the I method, and it did not use MSA. In the following section, possible ways to apply MSA to the I method will be discussed.

Applying MSA to the average route cost of each OD pair (t_{Ω}) is an essential part of traditional MSA averaging with the standard method. Since the OD demand for car is calculated using $t_{\Omega,c}$, the MSA-averaging also stabilizes the resulting demand, $D_{\Omega,c}$. This does not only provide fast convergence, it also makes the method robust in the sense that convergence is possible even for situations where the standard method without MSA would have had diverged.

This can also be done with the I method. Unfortunately, the I method uses $D_{\Omega,c}(\tilde{t}_{\Omega,c})$ rather than $D_{\Omega,c}(t_{\Omega,c})$, why the MSA averaging will not influence the demands much. It will of course have an impact since the $t_{\Omega,c}$'s are used to calculate the next $\tilde{t}_{\Omega,c}$, but expect the impact is expected to be relatively small compared to the impact observed for the standard method.

Another problem that occurs when using MSA on $t_{\Omega,c}$, is the fact that the point estimates of the cost function $(a_k\hat{t}_{\Omega,c}(\tilde{D}_{\Omega,c,k}) + (1 - a_k)t_{\Omega,c}(\tilde{D}_{\Omega,c,k+1}))$ will no longer be an estimate of the actual cost function (see figure 3.1), but will lie somewhere in between due to the "contraction". This may not sound as a big problem at first glance, but since this point is used to make a straight line approximation of the cost function, it is clear that the method will end up making an approximation of a different curve. Luckily, this curve will also go through the equilibrium point $(D^*_{\Omega,c}, t^*_{\Omega,c})$ why the method might still converge.

The I method with MSA on $t_{\Omega,c}$ is described in the following algorithm.

Algorithm 3.2.4 (I method w/ MSA on $t_{\Omega,c}$)

1. *Follow the standard method using MSA on $t_{\Omega,c}$ for the first 3 iterations*

(that is until $D_{\Omega,c,1}$, $D_{\Omega,c,2}$, $D_{\Omega,c,3}$, $t_{\Omega,c,1}$, $t_{\Omega,c,2}$, $t_{\Omega,c,3}$ are known for all OD-pairs (Ω)).

2. *For all Ω, set $\tilde{t}_{\Omega,c,3} = t_{\Omega,c,3}, \tilde{t}_{\Omega,c,2} = t_{\Omega,c,2}, \tilde{D}_{\Omega,c,3} = D_{\Omega,c,2}$, and $\tilde{D}_{\Omega,c,2} = D_{\Omega,c,1}$.*

3. *Set $k = 3$.*

4. *For all Ω, calculate $b_{D_\Omega} = \frac{\tilde{t}_{\Omega,c,k}-\tilde{t}_{\Omega,c,k-1}}{D_{\Omega,c,k}-D_{\Omega,c,k-1}}$, $a_{D_\Omega} = \tilde{t}_{\Omega,c,k}-b_{D_\Omega}D_{\Omega,c,k}$, $b_{t_\Omega} = \frac{t_{\Omega,c,k}-t_{\Omega,c,k-1}}{\tilde{D}_{\Omega,c,k}-\tilde{D}_{\Omega,c,k-1}}$, and $a_{t_\Omega} = t_{\Omega,c,k} - b_{t_\Omega}\tilde{D}_{\Omega,c,k}$.*

5. *Calculate $\tilde{D}_{\Omega,c,k+1} = \frac{a_{D_\Omega}-a_{t_\Omega}}{b_{t_\Omega}-b_{D_\Omega}}$ and $\tilde{t}_{\Omega,c,k+1} = b_{t_\Omega}\tilde{D}_{\Omega,c,k+1} + a_{t_\Omega}$.*

6. *Calculate a_k*

7. *Calculate $D_{\Omega,c,k+1} = D_{\Omega,c}(\tilde{t}_{\Omega,c,k+1})$ and $t_{\Omega,c,k+1} = a_k\hat{t}_{\Omega,c}(\tilde{D}_{\Omega,c,k+1})+(1-a_k)t_{\Omega,c,k}$.*

8. *Increment k by one.*

9. *Repeat step 4-8 until $||D_{\Omega,c,k}-D_{\Omega,c,k-1}|| \leq \tau$ for all OD-pairs or $k \geq k_{max}$.*

MSA on $t_{\Omega,c}$ might make the I method a little more stable. But since the corresponding OD demand is not calculated by using this $t_{\Omega,c}$, but by using $\tilde{t}_\Omega$, the demands may still fluctuate unrestrained.

One way to overcome this problem is to use MSA directly on $D_{\Omega,c}$ too. In this case the straight line approximations will be of two curves that differ from the cost and demand functions which might be a problem (see the above discussion of MSA on $t_{\Omega,c}$). A strict algorithm will not be presented since the differences between this and algorithm 3.2.4 are minuscule.

Another approach that might stabilize the I method is to apply the MSA average on $\tilde{t}_{\Omega,c}$. The reason behind this, is that it is the cost used to calculate the demand. But since $(\tilde{D}_{\Omega,c}, \tilde{t}_{\Omega,c})$ is the intersection of the straight line approximations of the demand and cost function, it would seem weird to only use MSA on one of the coordinates, and it could possibly cause the method to be very unstable. Instead it is more obvious to apply MSA on both $\tilde{t}_{\Omega,c}$ and $\tilde{D}_{\Omega,c}$.

As it was the case for the previous method, a strictly formulated algorithm will not be presented since the changes between this and algorithm 3.2.4 are very small. It should be mentioned though, that when using this method the 3 initial iterations (that is run with the standard method) should be run using MRA.

Thus, when applying MSA to $\tilde{t}_{\Omega,c}$ and $\tilde{D}_{\Omega,c}$ the a_k should be calibrated such that $a_4 = 1$, meaning that a_k should be calculated as

$$a_k = \frac{(k-3)^d}{\sum\limits_{j=1}^{k-3} j^d}.$$

3.3 Alternative Interpolation Methods

In this section a bunch of alternative interpolation methods, that could have been used to approximate the demand and cost curves will be presented. An actual code where these methods are implemented is beyond the scope of this study, why this section is mostly meant as an inspiration for further research.

3.3.1 Progressively Weighted Least Squares Method

When approximating a line to explain the behaviour of some data points, the least squares method is normally used. Since it is assumed that after each iteration, we are getting closer to the real equilibrium. This knowledge can be implemented to the least squares method by weighting the latest iterations higher than the first iterations. Thus, instead of minimizing the square error, the line we are searching for is then minimizing the weighted square errors [JFM11]. As opposed to the I method, this will also give some weight to the early iterations.

The method might not be very useful, since the previous iterations that the I method has “left behind” is in fact of less importance than the following iterations that are closer to the equilibrium point. Thus, it might be better to disregard the early iterations fully (as done in the I method) than to give them a small weight. Especially since the goal is to make a good approximation between the last two iterations where the equilibrium is expected to be, rather than making an approximation that gives a good overall approximation.

Because of this, the method of progressively weighted least squares will not be discussed any further in this study.

3.3.2 Higher Order Polynomial Interpolation

In 3.3.1 a method of improving the accuracy of the approximations by including extra interpolation points was introduced. When dealing with more than two interpolation points, another way to increase accuracy is to include a 2nd order polynomial term to the approximation, allowing curvature for the approximation.

Unfortunately, this method would not be fruitful. The difference between any two 2nd order polynomials is also a 2nd order polynomial. For any 2nd order polynomial P_2, it holds that $P_2(x) = 0$ is true for either 2, 1 or 0 values of x, why the method could result in more than one intersection between the two approximations. How to deal with this problem is not obvious. And since the severity of the problem increases when there is no intersection, approximations using polynomials of order 2 or higher can be ruled out.

3.3.3 Exponential Interpolation

A way to overcome the problems occurring from higher order polynomials, but without giving up on adding a bit of curvature to the approximation, is to interpolate using exponential functions of the form

$$f(x) = ae^{bx}.$$

Two exponential functions will always either intersect nowhere (same b and different a), 1 place or everywhere (same b and a) – the exact same situation as for two straight lines.

Although it can be a good thing to add a bit of curvature to the approximations, the exponential interpolation has a major disadvantage in the cases where the demand or cost almost form a straight line, or where the curves are concave. In this case an exponential curvature would be assumed, although the assumption may turn out to anything but true. And since there are very few cases, where an exponential interpolation will be a good approximation, the exponential interpolation will not be examined any further in this study.

Chapter 4

Test Preparations

Before tests of the I method can be run, a series of important questions regarding the test network, important constants, and the OD matrix generation principle has to be answered. In section 4.1, the small-scale test network used in this study will be presented. Section 4.2 will handle the calibration of some of the important constants used in various formulas. Finally, section 4.3 explains how Monte Carlo generation of random OD matrices can emulate the trip generation and trip distribution steps from the 4-Stage model (see section 2.2, p. 7).

4.1 The Test Network

In order to test the I method, a network on which a series of tests can be run has to be constructed. The chosen test network is a rather simple network consisting of 9 nodes and 16 links. The network can be seen in figure 4.1.

The network can be interpret as a city with a city center in node 9, having an inner and an outer ring road. The nodes 1-4 lie on the outer ring, and the nodes 5-8 lie on the inner ring. The nodes are numbered clockwise from the outside to the inside.

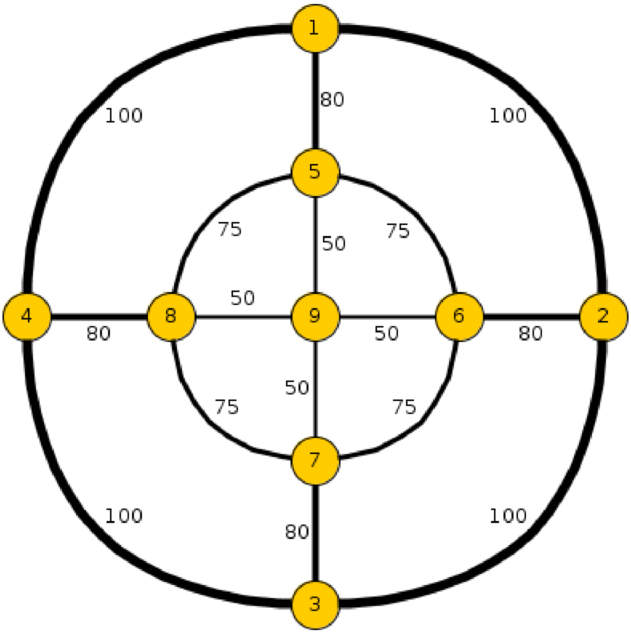

Figure 4.1: The test network and the capacity of each link.

Alongside each link of the network, the capacity of the link has been written. The outer ring has a capacity of 100, the roads connecting the outer and inner ring have a capacity of 80, the inner ring has a capacity of 75, and the roads connecting the city center with the inner ring have a capacity of 50.

Also, on each link there is some sort of public transport, that is assumed to have no maximum capacity. The type of public transport is irrelevant, but could be bus rapid transport (BRT), light rail rapid transit (LRRT), metro etc. There is no public transport outside of the links.

4.1.1 Travel Time of Each Link

It is assumed that the free flow speeds of each road are identical. This means that the free flow travel time of each route is only based on the length of the road. The free flow travel time from any of the outer points to the center is set to be 4. The free flow travel time on all other links can then be calculated

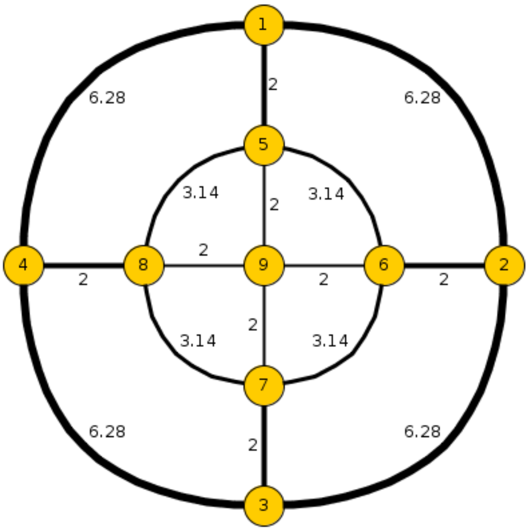

Figure 4.2: The test network and the free flow travel time of each link.

geometrically, so that e.g. the free flow travel time between node 1 and 2 is equal to 2π and so forth. The network with all free flow travel times shown can be seen in figure 4.2.

The travel times shown are the free flow travel times. As introduced in the BPR formula (eq. (2.1.2)), section 2.1.1, the travel time increases as congestion occurs,

$$t_{l,c}(D_{l,c}) = t_{0,l}\left(1+\alpha_l\left(\frac{D_{l,c}}{C_l}\right)^{\beta_l}\right).$$

The effect of congestion is determined by the link-specific β-constant, which can be seen for all links in figure 4.3.

Recall that public transportation is assumed to be unaffected by congestion, and as such the travel times for public transportation is equal to the free flow travel times seen in figure 4.2.

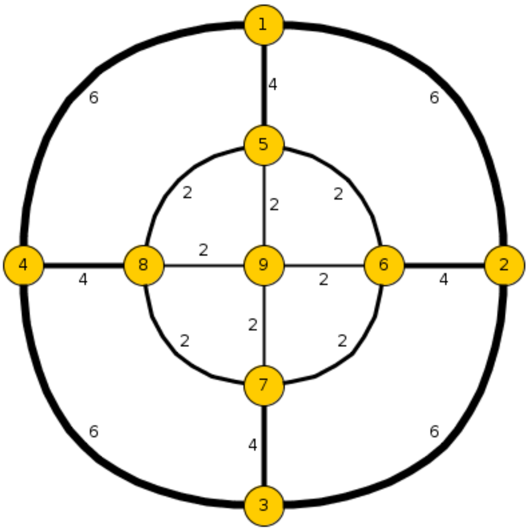

Figure 4.3: The test network and the β_l-values of each link.

4.1.2 The Routes

Besides the nodes and links, the network has a tremendous amount of routes too – 2007 to be exact. These routes represents every possible connection between any two nodes of the network, without passing through the same link twice. Since the routes from A to B, are the same as the routes from B to A (just in reverse order), there is a total of

$$\binom{9}{2} = \frac{9!}{2! \cdot 7!} = \frac{8 \cdot 9}{2} = 36$$

different OD pairs[1]. This means that on average there are more than 55 possible route between any two OD pairs. Clearly, some routes are more likely than others, and an approach where each route had the same possibility would be foolish.

It is obvious that a route with a shorter travel time, should be more likely than a route with a high travel time, and as introduced in section 2.4.2 this choice is emulated by a logit model, where the constants γ and κ_c determines the initial

[1] Normal studies would have used twice as many (72) OD pairs for this network. See the end of section 4.2.3 why half of the OD pairs can be ignored.

probability of choosing a given route. The values of these will be calibrated in the following section.

4.2 Calibration of Various Constants

This section seeks to determine realistic values of some constants used in the transportation models. The constants that are to be calibrated includes a measure of the time elasticity γ, the car preference constant κ_c and the expected total amount of demand on the network $\Delta_\bullet$.

4.2.1 The Measure of Time Elasticity (γ)

The time elasticity is a very important parameter for route (and mode) choice. If the time elasticity is rather small (in absolute terms), the cars will spread out on the network, whereas a rather high time elasticity will result in almost everyone choosing the fastest route. Note that γ is not equal to the time elasticity. Still, γ and time elasticity will be used interchangeably throughout the report, since it is the only measure of the time elasticity that occurs in the study.

In order to determine a realistic value γ, three examples will be provided. In each case, a couple of routes will be selected, and for each route an approximate estimate of the amount of cars choosing this route will be made. Afterwards a series of tests will be run with various values of γ. These tests will determine which values of γ result in a realistic route choice, and which values of γ that do not.

It should be stressed that in these tests, the amount of passengers choosing the specific route is under the assumption that each route can be driven at free flow speed. Of course, as the network gets congested these numbers might not be realistic any more. This is not a problem, since the routes in this case will have updated travel times that correspond to the current traffic load. These tests only serve to calibrate the value of γ.

We will begin with an example between two adjacent "corner points" on the outer ring.

Example 4.2.1 (From 1 to 2)
In the following example we are looking at three different routes (a red, a blue,

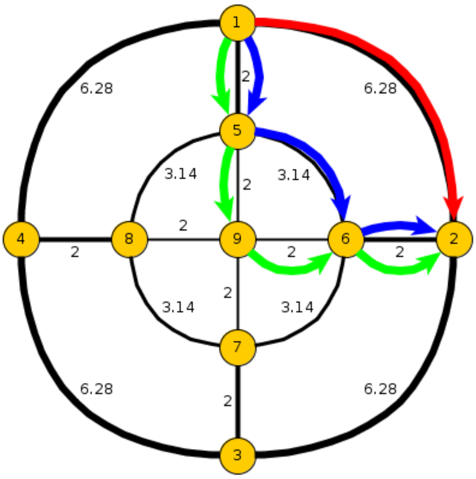

Figure 4.4: Three highlighted routes from node 1 to 2.

and a green) leading from node 1 to node 2. The routes can be seen in figure 4.4.

From the figure, it seems as is if we would expect at least 40% of the travellers to use the red route. If less than 40% choose this route, the time elasticity should be increased.

The percentage of travellers using each route is presented in the table below (table 4.1) for various values of γ.

The table shows that in order to get at least 40% of the passengers to choose the red route, $|\gamma|$ must be more than 0.4. It can also be seen, that when $|\gamma| \leq 0.6$, more than 8% of the passengers choose a fourth route. Since there are only three obvious routes in this example, it seems as if $|\gamma|$ should be chosen higher

γ	-0.2	-0.4	-0.6	-0.8	-1.0	-1.2	-1.4	-1.6	-1.8	-2.0
Red (%)	14.0	33.6	46.8	55.3	61.7	67.1	71.7	75.9	79.5	82.6
Blue (%)	11.8	23.8	27.9	27.8	26.1	23.9	21.6	19.2	16.9	14.8
Green (%)	9.9	16.9	16.7	14.0	11.1	8.5	6.5	4.8	3.6	2.7
Others (%)	64.4	25.8	8.7	3.0	1.1	0.4	0.1	0.1	0.0	0.0

Table 4.1: Route probability for three highlighted routes from node 1 to 2 for various values of γ

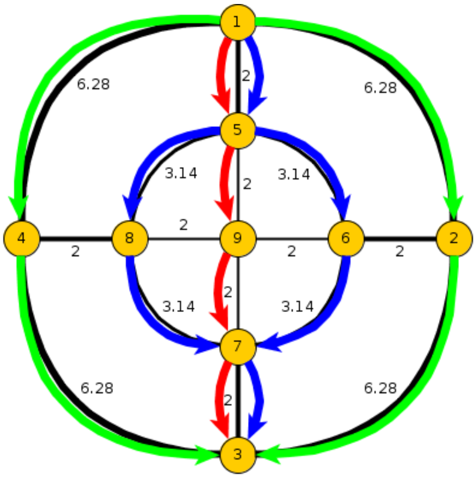

Figure 4.5: Three highlighted routes from node 1 to 3

than 0.6.

□

Example 4.2.2 (From 1 to 3)

This is an example of route choice between the two nodes that are furthest apart (1 and 3). Since the nodes are so far apart, there is a lot of parallel realistic routes in this case. The 3 chosen routes can be seen in figure 4.5. Note that the green route is *not* the third fastest option, but is solely chosen because it is a quite intuitive route.

Once again the percentage of travellers choosing a specific route for various time elasticities have been computed and can be seen in table 4.2. Note that the numbers for the green and blue routes are the combined numbers, where the left and right alternative of the two have been added.

γ	-0.2	-0.4	-0.6	-0.8	-1.0	-1.2	-1.4	-1.6	-1.8	-2.0
Red (%)	8.6	22.0	38.8	55.7	70.0	80.7	87.9	92.5	95.4	97.2
Blue (%)	10.9	17.7	19.7	18.0	14.3	10.5	7.2	4.8	2.5	2.0
Green (%)	6.9	7.1	5.0	3.2	1.5	0.7	0.3	0.1	0.1	0.0
Others (%)	73.6	53.3	36.5	23.5	14.2	8.2	4.6	2.5	1.4	0.7

Table 4.2: Route probability for three highlighted routes from node 1 to 3 for various values of γ

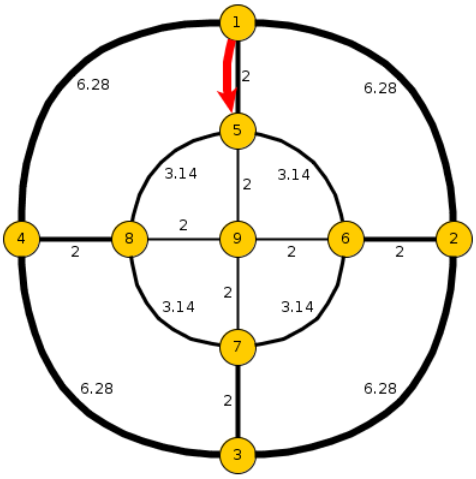

Figure 4.6: The only obvious route between node 1 and 5.

The table reveals that when $|\gamma|$ exceeds 1.4, only 7.2% of the passengers or less would choose the blue route. Since the blue route is a rather obvious choice, it would seem as if $|\gamma|$ should be kept smaller than 1.4.

□

Example 4.2.3 (From 1 to 5)

This example is in contrast to example 4.2.2, where there were many realistic routes. In this case there is only 1 realistic route, and choosing another route than this would be highly unlikely. Thus it is expected that at most 0.5% will choose an alternative route. It is illustrated in figure 4.6.

Once again the percentages have been calculated, and they can be seen in table 4.3. It is seen that we $|\gamma|$ should be bigger than 0.6, since any $|\gamma| \leq 0.6$ makes too many people choose an unrealistically long route.

□

γ	-0.2	-0.4	-0.6	-0.8	-1.0	-1.2	-1.4	-1.6	-1.8	-2.0
Red (%)	37.8	89.4	98.7	99.8	100.0	100.0	100.0	100.0	100.0	100.0
Others (%)	62.2	10.6	1.3	0.2	0.0	0.0	0.0	0.0	0.0	0.0

Table 4.3: Route probability for the highlighted route from node 1 to 5 for various values of γ

.

Summarizing the above examples, we get that $\gamma \in]-1.4, -0.6[$ in order to obtain a realistic value. Therefore, the forthcoming performance tests could use any of these values, but to be on the safe side, the middle-most value ($\gamma = -1$) has been chosen for the forthcoming tests.

4.2.2 The Mode Preference Constant (κ_c)

Another important constant is the car preference constant κ_c. This constant is the only one of the constants that can be determined without any tests. In fact, when one important constant is known it can be determined purely mathematically. The constant (μ_0) that has to be known is the market share of cars on the network, under the assumption that all transportation modes have the same travel time for all possible routes. Since this transport network is a fictive one, the number is unknown. A value of $\mu_0 = 0.85$ has been selected in this study.

Recall that the probability of choosing a car over public transportation is calculated by the logit model

$$P_{\Omega,c} = \frac{e^{U_{\Omega,c}}}{e^{U_{\Omega,c}} + e^{U_{\Omega,p}}} = \frac{1}{1 + e^{U_{\Omega,p} - U_{\Omega,c}}},$$

where the utility functions for car and public transport are given by

$$\begin{aligned} U_{\Omega,c} &= \gamma t_{\Omega,c} + \kappa_c, \\ U_{\Omega,p} &= \gamma t_{\Omega,p}. \end{aligned}$$

Thus, when the two travel times are equal the probability of choosing a car reduces to

$$P_{\Omega,c} = \frac{1}{1 + e^{-\kappa_c}}.$$

Since we want the probability to be equal to μ_0 we get the following,

$$\begin{aligned} \mu_0 &= \frac{1}{1 + e^{-\kappa_c}} && \Leftrightarrow \\ e^{-\kappa_c} &= \frac{1}{\mu_0} - 1 && \Leftrightarrow \\ \kappa_c &= -\ln\left(\frac{1}{\mu_0} - 1\right). \end{aligned}$$

This can be rewritten so that the car preference constant is given as

$$\kappa_c = \ln\left(\frac{\mu_0}{1 - \mu_0}\right). \tag{4.2.4}$$

Since we have chosen $\mu_0 = 0.85$ in this study, the resulting car preference constant will be

$$\kappa_c = \ln\left(\frac{0.85}{1-0.85}\right) = \ln\left(\frac{0.85}{0.15}\right) \approx 1.735.$$

4.2.3 The Demand between each OD-Pair (D_Ω)

Normally the demand between an OD pair is determined by a gravity formula, which is dependent on socio-economic measures of the two, and the distance between them (see [Voo56] for the original development or [Ric10] for an introduction).

More importantly, the road network is constantly being altered (new roads are built, public transport is increased, roads are extended, tolls are applied etc.) so that the ratio between the flow and the capacity on the network is not too low, and not close to 1.

In our case, we have the opposite task. We have a road network with a given capacity, and we seek to find a realistic demand, so that the flow will not be too low, also not too close to the network capacity.

In order to solve this problem, some shorthand notation will come in handy. Let $C_\bullet$ denote total car capacity of the traffic network. It holds that

$$C_\bullet := \sum_{l=1}^{n_l} C_l = 1220.$$

Likewise, the total demand is given by

$$D_\bullet := \sum_{O=1}^{n_\Omega} D_\Omega.$$

The total demand is going to be shared by cars and public transportation. Thus the above only gives a maximum total car demand ($D_{\bullet,c}$),

$$D_{\bullet,c} < D_\bullet.$$

Because of this, intuitively, it could be a good idea to simply set $D_\bullet = C_\bullet$, since the demand for public transport would be subtracted, leading to a $D_{\bullet,c}$ a realistic amount below $C_\bullet$. Unfortunately, the situation is more complicated as such, which the following example seeks to show.

Example 4.2.5
A traveller travelling from node 1 to node 3 will have to use at least 2 links to get to his destination (see figure 4.5, p. 43). But most of the time, more than two links will be used. In fact, the fastest route passes through the center of the graph, and in this case 5 links will be used. By doing this, even though the demand only counts for 1, the remaining total capacity is reduced by 5.

□

Even though example 4.2.5 is the most extreme case of the network, it still shows an important point. If all travellers had the same effect on the network as the one from the example, the total capacity would be exceeded if one has simply chosen to put $D_\bullet = C_\bullet$, unless the public transport would carry an unrealistically high amount of the demand.

Thus we need to come up with a different approach. One way to do it, is to check if $D_\bullet$ was realistic, retrospectively, by calculating the market share by cars (μ) after the test has been run. This can be done by the following formula,

$$\mu := \frac{1}{n_\Omega} \sum_{O=1}^{n_\Omega} \frac{D_{\Omega,c}}{D_\Omega}.$$

If $D_\bullet$ was too high, the demand would be so big, that the flow on the link would approach their capacity, while "there was still travellers left to place in the system". Thus these will be placed in the public transport so to say, making the amount of people using public transport relatively great, corresponding to a low μ ($\mu < 0.65$).

On the other hand, if $D_\bullet$ was too low, there would be no congestion on the roads, and the benefits from riding a car (included in the mode preference constant κ_c (see section 4.2.2)) would exceed the relatively small time loss due to congestion. Thus almost everyone would travel by car resulting in a high μ. $\mu > 0.75$ might indicate that $D_\bullet$ was too low.

4.3 Generation of Random OD-Matrices Using Monte Carlo Simulation

There are plenty of ways to generate random OD-matrices. In this study an extremely simple approach is used. For each OD matrix, the expected total demand, $\Delta_\bullet$, is given as a parameter, and then the demand for each OD-pair

D_Ω is draw randomly from the uniform distribution $U\left(1, 2\frac{D_\bullet}{n_\Omega}\right)$. By denoting $\frac{D_\bullet}{n_\Omega}$ by $\bar{D}_\Omega$, we say that

$$D_\Omega \sim U(1, 2\bar{D}_\Omega).$$

By doing this, the distribution of $D_\bullet$ will suffice the following,

$$\begin{aligned}
\mathrm{E}[D_\bullet] &= n_\Omega \mathrm{E}(D_\Omega) = n_\Omega \frac{1 + 2\tilde{D}_\Omega}{2} = \Delta_\bullet + \frac{n_\Omega}{2}, \\
\mathrm{Var}(D_\bullet) &= n_\Omega \mathrm{Var}(D_\Omega) = n_\Omega \frac{1}{12}(2\bar{D}_\Omega - 1)^2 = \frac{n_\Omega}{3}\left(\bar{D}_\Omega^2 - \bar{D}_\Omega + \frac{1}{4}\right) \lessapprox \frac{\Delta_\bullet^2}{3n_\Omega}, \\
\mathrm{SD}(D_\bullet) &= \sqrt{\frac{n_\Omega}{3}\left(\bar{D}_\Omega^2 - \bar{D}_\Omega + \frac{1}{4}\right)} \lessapprox \frac{\Delta_\bullet}{\sqrt{3n_\Omega}}.
\end{aligned}$$

In the above we have used that the variance of a uniform distribution $U(a, b)$ is equal to $\frac{1}{12}(b - a)^2$ [JFM11].

The following example seeks to show the size of the variability of the approach.

Example 4.3.1
In our study we have 36 OD-pairs, such that $n_\Omega = 36$. Assume that we want to do a Monte Carlo generation of an OD-matrix with the parameter $\Delta_\bullet = 450$. Then, with 95% confidence, we can say that the resulting $D_\bullet$ will approximately lie in the interval

$$\begin{aligned}
\Delta_\bullet + \frac{n_\Omega}{2} \pm 1.96\sqrt{\frac{n_\Omega}{3}\left(\bar{D}_\Omega^2 - \bar{D}_\Omega + \frac{1}{4}\right)} &= 468 \pm 1.96\sqrt{\frac{36}{3}\left(\left(\frac{450}{36}\right)^2 - \frac{450}{36} + \frac{1}{4}\right)} \\
&\approx [387, 550].
\end{aligned}$$

□

It can be seen, that when we want the total to be equal to 450 the width of the 95% confidence interval is about 163 wide. Luckily, by looking at the formula, it seen that the width of the confidence interval is approximately proportional to $\Delta_\bullet$, such that for smaller values of $\Delta_\bullet$ (and hence also $D_\bullet$), the width will decrease proportionally.

By using this approach it could seem as if the market share is only dependent on $D_\bullet$. This is definitely not the case. If all the demand is concentrated on OD-pairs that are fairly close to each other, the total flow would be much less than if the demand was concentrated on OD-pairs far apart. And since a big flow will result in large car costs, the market share of the cars would be smaller in this case and vice versa.

Nevertheless, $D_\bullet$ is still of great importance, and as such it serves as an easy way to "handle" the size of μ.

It should be noted, that this study uses symmetric OD matrices, which means that if there is a demand of x passengers between A and B, there is an equal demand between B and A. This will (almost) always be the case for a full day OD matrix. Sometimes, this does not emulate reality sufficiently, and the OD matrices are split into different sections of the day, causing the OD matrix to be non-symmetric.

Since symmetric OD matrices are used (and there are no one-way streets in the network) it is fair to let the OD pair between 1 and 2 be equal to the OD pair between 2 and 1. This allows the number of OD pairs to be reduced from 72 to 36 OD pairs.

When a demand between an OD pair is stated, it is not obvious if this is solely the demand between A and B or the demand between both A and B as well as B and A. It does not matter which of the two interpretation is chosen, as long as the interpretation of the corresponding capacity for each link C_l is reflecting this as well.

CHAPTER 5

Comparative Monte Carlo Testing of the I Method

In this chapter the results of a series of tests will be presented. In order to allow comparison, tests have been run for both the I method and the standard method. Section 5.1 gives a general introduction to the tests. It is followed by section 5.2 that deals with the comparison of the I method and standard method when both methods are using MRA. Finally, section 5.3 compares the two methods when applying MSA.

5.1 Introductory Remarks Regarding the Tests

This section serves to provide a small introduction to the series of tests regarding the number of tests, the used parameters, and how the test results will be evaluated.

For each method 30 randomly generated OD matrices have been constructed[1], and the number of external iterations required to reach convergence has been noted. The matrices were constructed using $\Delta_\bullet = 450$, and the internal loop

[1] Each method has faced the same 30 OD matrices due to seeding.

was allowed to run a maximum of 1000 iterations per external iteration. The market share of cars ranged from 67.5% to 73.5% which seems to be fairly realistic according to the discussion about a realistic total demand in section 4.2.3.

Whenever MSA has been used, the standard method has used MSA on $t_{\Omega,c}$, whereas MSA on three different variable combinations have been tried for the I method. In each case, the d-values have been set to 1, 2, 3, 4, and 5

5.1.1 Two Different Tolerances

The tolerance of the stopping criteria has been run under both strict ($\tau = 0.001$) and more relaxed conditions ($\tau = 0.01$). Recall that the expected demand between each OD pair is $\frac{\Delta_\bullet}{n_\Omega} = \frac{450}{36} = 12.5$. And since less than 80% of this is expected to be travelled by car, the car demand between each OD pair, will on average be less than 10. This makes $\tau = 0.001$ corresponds to a relative change of less than 0.01% on average. And since the internal loop uses a tolerance that is 100 times smaller, the restrictions for convergence of the internal loop are indeed very strict when using $\tau = 0.001$.

By re-running all tests again with the tolerance to $\tau = 0.01$ we might see a change in the amount of external iterations needed for the I method and its different MSA averaging methods. They might all improve at the same rate, but the only way to find out if the previous tolerance was favouring or disfavouring one of the methods, is to run the test with another tolerance. Thus, all methods have been run with two different tolerances.

5.1.2 How to Evaluate the Test Results

Each method will be evaluated on two parameters. The number of required iterations to reach convergence is the key parameter, but the failure rate – that is the number of matrices that failed to converge divided by the total number of matrices – will be evaluated too. The reason why the failure rate is not weighted that much, is that a non-converging OD matrix is caused by an internal loop failing to converge in less than 1000 iterations. A faster internal convergence might have been possible with a better algorithm for the internal loop, but since this is beyond the scope of this thesis (see section 3.1.4), the failure rate will only be looked at secondarily.

Due to the lack of convergence for some OD matrices for certain methods, it has not been possible to make a fully fair average. Some methods can at most use 22 OD matrices to determine the average number of required iterations, whereas some methods include all tests, including some under-performing tests. In order to correct for this, the best 20 tests have been selected for each method, and an average has been made solely on the basis these 20 tests. The 95% confidence interval of this measure has been calculated by

$$\bar{x}_{20} \pm t_{0.025}(19)\frac{S_{\bar{x}_{20}}}{\sqrt{20}} \approx \bar{x}_{20} \pm 0.468 \cdot S_{\bar{x}_{20}},$$

with $\bar{x}_{20}$ being the mean of the top 20 performances, and $S_{\bar{x}_{20}}$ being the sample standard deviation of the top 20 performances.

Also, when comparing two specific methods, a paired t-test has been used. In these cases the relevant p-value describing the probability of obtaining a mean value lower than the observed mean from the method under consideration, given that the true mean of the method under consideration is equal to the true mean of the other method used in the comparison. [JFM11]

5.2 Using MRA

It is now time for the presentation of the first series of results. The first tests have been using the I method and standard method, respectively, using MRA. The full results of these tests can be found in table A.1 and A.2, appendix A, p. 83ff [2]. The tests have been run for two different tolerances, and the tests with the strict tolerance will be evaluated initially.

5.2.1 Using a Strict Tolerance ($\tau = 0.001$)

Figure 5.1 summarises the results of the test, by having a diagram showing the failure rate of the tests to the list, and a boxplot of the top 20 tests of each method to the right.

In the diagram of the failure rates, it is seen that the I method is rather unstable, since it fails to converge in 26.7% of the cases. For comparison the standard method only fails to converge within 200 external iterations in 6.7% of the cases.

[2] In these table a third method called the *C method* is also listen. It will be introduced in section 6.3.

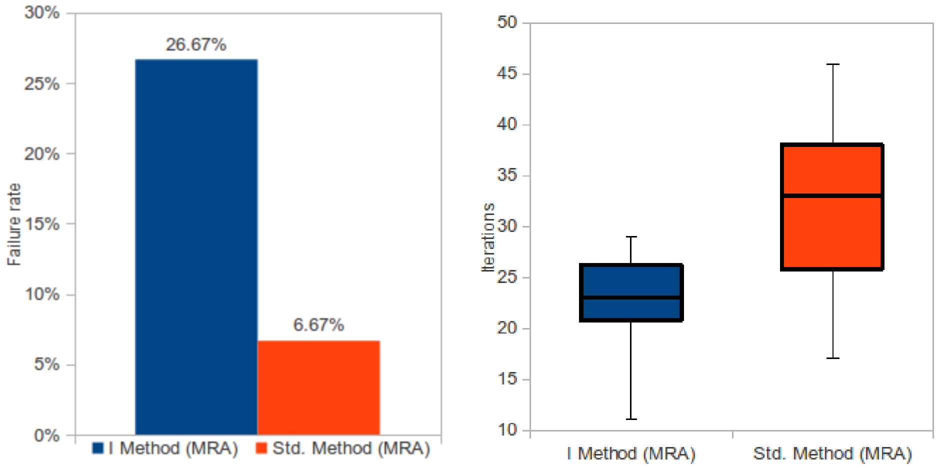

Figure 5.1: Summary of the tests of the I method and standard method using MRA and $\tau = 0.001$

The instability put aside, the I method performs much better than the standard method, beating it in 20 out of the 22 cases where convergence was achieved. Figure 5.1 also shows the boxplot of the top 20 tests for both the I method and the standard method. It is seen that both the minimum, first quartile, median, third quartile and maximum is lower than those of the standard method. Also, it is seen that the fifth best test of the standard method required about the same number of iterations as the fifteenth best test of the I method.

To sum up, when using a tolerance of $\tau = 0.001$, the I method performs much better on average on the standard method, when both methods are using MRA. Nevertheless, it should be kept in mind, that the failure rate of the I method was very high compared to that of the standard method.

5.2.2 Using a More Relaxed Tolerance ($\tau = 0.01$)

The tests for the I method and the standard method using MRA have also been run under the tolerance of $\tau = 0.01$ too. Figure 5.2 contains the information of the failure rate and the general performance. It is seen that the less strict tolerance has decreased the failure rate from 26.7% to a mere 3.3%. In fact, using the more relaxed tolerance, the failure rate of the I method is lower than that of the standard method which remains unchanged at 6.7%.

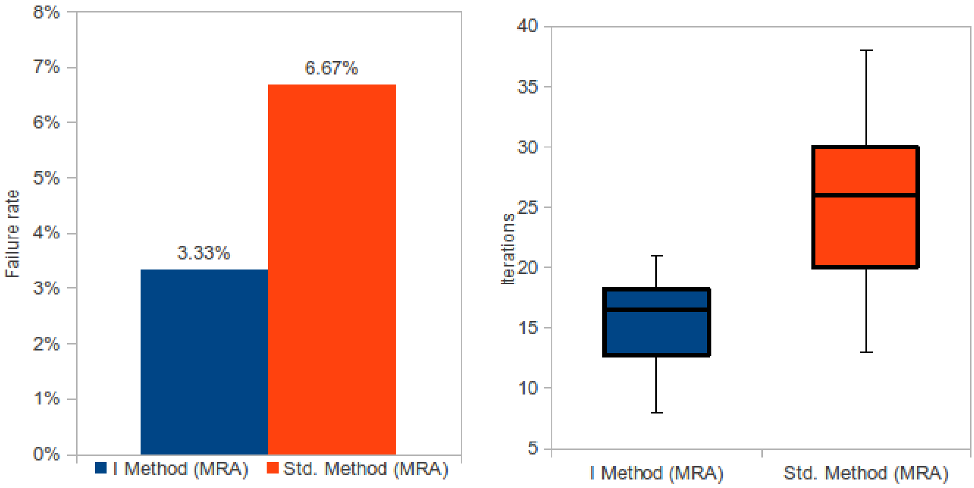

Figure 5.2: Summary of the tests of the I method and standard method using MRA and $\tau = 0.01$

Figure 5.2 also reveals that the I method in general requires much less iterations to converge than the standard method. It has even improved compared to the standard method, since the third quartile of the I method now is less than the first quartile of the standard method.

The relative change of the median of iterations needed for the I method and the standard method was almost equal, with a relative drop of around 27% for both methods.

To sum up, the tolerance did almost have the same impact on performance for both methods. But the I method managed to improve the stability a lot, making it undoubtedly superior to the standard method using MRA and a relaxed tolerance.

5.3 Using MSA

In this section the results of the tests using MSA will be presented and discussed. MSA averaging on $t_{\Omega,c}$ is the most commonly used method to secure fast convergence for the standard method. These tests will show if a similar dramatical drop in required iterations can be seen when using the I method. MSA for the I method will be discussed first, and will later be compared to the standard method using MSA. The full results can be found in appendix A, p. 81.

5.3.1 Using a Strict Tolerance ($\tau = 0.001$)

Once again the tests under the strict tolerance will be discussed first. Table 5.1 sums up the failure rates of all of the different approaches for the I method. The I method is quite stable when using MSA averaging on $t_{\Omega,c}$, with a failure rate between 0% and 3.3% depending on the value of d. Likewise when MSA is used on both $t_{\Omega,c}$ and $D_{\Omega,c}$, where the average failure rate is 3.3%. When using MSA on $\tilde{t}_{\Omega,c}$ and $\tilde{D}_{\Omega,c}$ on the other hand, the I method is quite unstable. Depending on the d-value the method only converges in 80%-93% of the cases. For comparison, it can be said that the standard method using MSA on $t_{\Omega,c}$ converged in all of the cases.

Figure 5.3 gives an idea of the number of required iterations to reach converge for each method. As explained in the introduction to this chapter, an average has been calculated on the basis of the 20 best performances of each method. This is indicated by a marker in figure 5.3. The 95%-confidence intervals have been shown as straight lines in the plot.

It is seen that the standard method in fact did perform a lot better when applying MSA. When using $d = 5$, the average number of external iterations required to reach convergence for the top 20 tests has been reduced by more than 73%, achieving an average of only 8.45 iterations. $d = 2, 3, 4$ did a little worse, but still impressive, job with between 8.65 and 12.05 iterations required for convergence on average. $d = 1$ uses an average of 18.15 external iterations.

As expected, the I method fails to benefit much from MSA. It is evident from figure 5.3, that MSA on $t_{\Omega,c}$ and $D_{\Omega,c}$ results in a much slower convergence, requiring between 30 and 40 iterations to converge on average. MSA on $t_{\Omega,c}$ generally requires around 25 iterations to converge, which is also worse than the

MRA	26.7%				
	$d = 1$	$d = 2$	$d = 3$	$d = 4$	$d = 5$
MSA on t_Ω	3.3%	0%	0%	3.3%	3.3%
MSA on t_Ω & D_Ω	3.3%	0%	3.3%	3.3%	6.7%
MSA on $\tilde{t}_\Omega$ & $\tilde{D}_\Omega$	20%	20%	16.7%	6.7%	6.7%

Table 5.1: Failure rate of the tests for the I method using MSA and $\tau = 0.001$

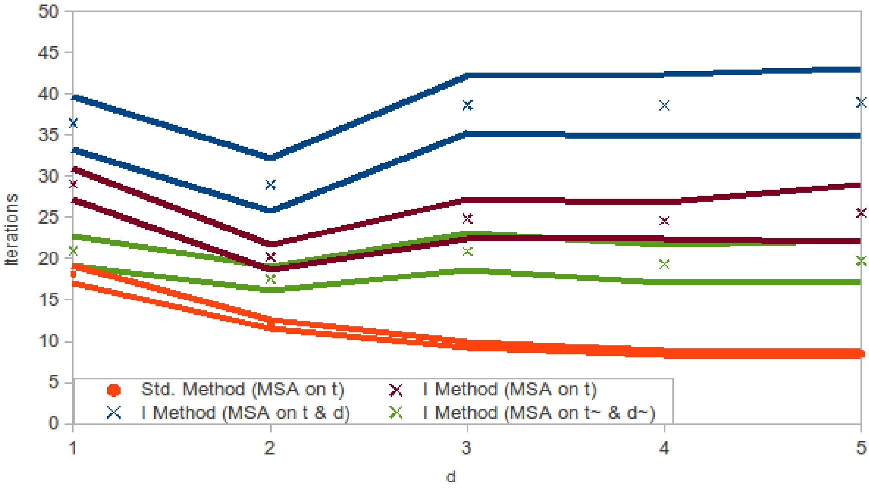

Figure 5.3: Number of required iterations of the tests for the I method and standard method using MSA and $\tau = 0.001$

average for the I method using MRA. When MSA is used on $\tilde{t}_{\Omega,c}$ and $\tilde{D}_{\Omega,c}$ the required number of iterations is around 20 on average, and it is the method that performs the best when the stability issues are ignored.

In addition to figure 5.3, table 5.2 also seeks to explain the performance of the I method using various MSA. It shows the p-value from the one-sided hypothesis test based on a paired t-test compared to the MRA, where the alternative hypothesis has been that MSA is better. All valid measurements (that is all the OD matrices where both methods converged) have been used in the tests.

It can be seen that none of the methods perform significantly better than the MRA on a 95%-confidence level. The p-values of both methods when using

	$d = 1$	$d = 2$	$d = 3$	$d = 4$	$d = 5$
MSA on t_Ω	> 0.5	0.1061	> 0.5	> 0.5	> 0.5
MSA on t_Ω & D_Ω	> 0.5	> 0.5	> 0.5	> 0.5	> 0.5
MSA on $\tilde{t}_\Omega$ & $\tilde{D}_\Omega$	0.2876	0.0762	0.1846	0.1935	0.3882

Table 5.2: p-values of the I method of various MSA approaches in a one-sided paired t-test compared to the MRA approach, with the alternative hypothesis being that the MSA is better, and using $\tau = 0.001$

$d = 2$ are rather low, though, and it can not be ruled out that these methods actually perform a little better than the MRA when $d = 2$. But it is crystal clear that no MSA method gives the I method a noticeable boost – unlike the standard method that changes from being a bad method using MRA, to being a very efficient method using MSA on $t_{\Omega,c}$.

To sum up, the standard method benefits immensely from using MSA on $t_{\Omega,c}$. The I method fails to get the same improvement, and none of the observed improvements were significant on a 95% confidence level. It should be noted, though, that using MSA on $t_{\Omega,c}$ managed to increase stability a lot without making it perform worse.

5.3.2 Using a More Relaxed Tolerance ($\tau = 0.01$)

So far we have witnessed that the standard method using MSA on $t_{\Omega,c}$ was superior to any other method under a strict tolerance. The difference is so huge that obtaining competitive results just by relaxing the tolerance seems unrealistic. Nevertheless, the series of tests has been run with the tolerance $\tau = 0.01$ too, especially to see if MSA on various variables can lead to significantly improved results for the I method. Once again the full results can be found in appendix A, p. 81ff.

Table 5.3 shows the failure rate of the various methods. It is seen that the I method manages to converge for every OD matrix for any value of d when using MSA on $t_{\Omega,c}$. The I method using MSA on both $t_{\Omega,c}$ and $D_{\Omega,c}$ also manages to converge every time when using $d = 1, 2, 3, 5$. When using $d = 4$ it fails to converge a single time. The I method using MSA on $\tilde{t}_{\Omega,c}$ and $\tilde{D}_{\Omega,c}$ shows great improvements regarding stability using this more relaxed tolerance. It converges in 100% of the OD matrices when $d \geq 3$. For both $d = 1$ and $d = 2$ it only fails to converge a single time. The standard method using MSA on $t_{\Omega,c}$ once again manages to converge in all 150 cases.

MRA	3.3%				
	$d = 1$	$d = 2$	$d = 3$	$d = 4$	$d = 5$
MSA on t_Ω	0%	0%	0%	0%	0%
MSA on t_Ω & D_Ω	0%	0%	0%	3.3%	0%
MSA on $\tilde{t}_\Omega$ & $\tilde{D}_\Omega$	3.3%	3.3%	0%	0%	0%

Table 5.3: Failure rate of the tests for the I method using MSA and $\tau = 0.01$

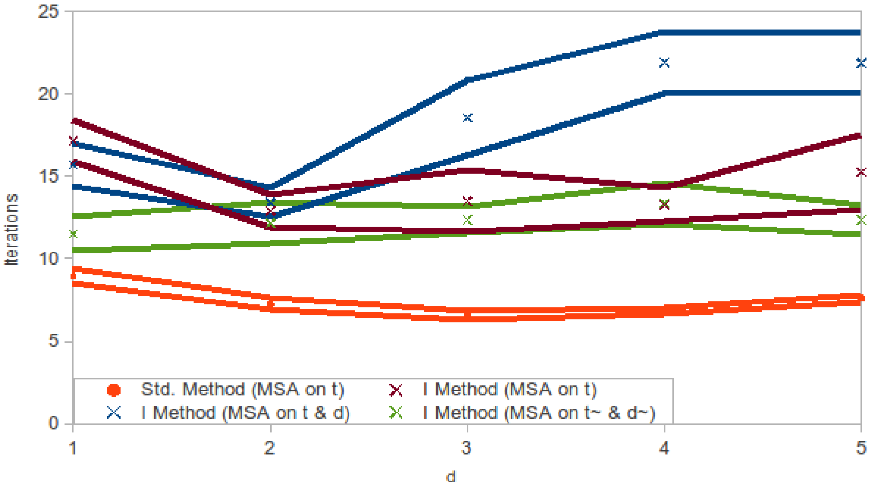

Figure 5.4: Number of required iterations of the tests for the I method and standard method using MSA and $\tau = 0.01$

Figure 5.4 shows the average and the confidence interval of the average of the top 20 performances for each method. Yet again, it is seen that the standard method clearly outperforms the I method when using MSA on $t_{\Omega,c}$. With a properly selected d, the standard method requires as little as 6.55 iterations on average for $d = 3$. Using MRA the standard method required an average of 25.05 iterations, which makes the drop correspond to almost 75%.

It is harder to make a conclusion about the various MSA approaches of the I method under this tolerance than under the previous. In figure 5.4 it is readily seen that the lines of the different methods cross each other. What they all have in common, is that their performances remain far from that of the standard method using MSA on $t_{\Omega,c}$. It is seen that all three methods manage to get an

	$d = 1$	$d = 2$	$d = 3$	$d = 4$	$d = 5$
MSA on t_Ω	> 0.5	0.0151	0.3928	0.5000	> 0.5
MSA on t_Ω & D_Ω	> 0.5	> 0.5	> 0.5	> 0.5	> 0.5
MSA on $\tilde{t}_\Omega$ & $\tilde{D}_\Omega$	0.0001	0.0026	0.0987	0.0004	0.0001

Table 5.4: p-values of the I method of various MSA approaches in a one-sided paired t-test compared to the MRA approach, with the alternative hypothesis being that the MSA is better, and using $\tau = 0.01$

average lower than 15, when a good d is chosen. It is also seen that MSA on $t_{\Omega,c}$ and $D_{\Omega,c}$ is very sensitive to the value of d, whereas MSA on $\tilde{t}_{\Omega,c}$ and $\tilde{D}_{\Omega,c}$ performs decently regardless of the d-value.

In order to determine whether or not any of these MSA approaches performed better than the I method using MRA, once again paired t-tests have been used. The p-values of these tests can be found in table 5.4.

Once again, it is easily seen that MSA on both $t_{\Omega,c}$ and $D_{\Omega,c}$ undoubtedly fails to improve the performance of the I method. Also, notice that MSA on $t_{\Omega,c}$ is quite sensitive to the chosen d, and once again performs much better when using $d = 2$. In this case the p-value is less than 2%, and the difference is in fact significant on most commonly used significance levels. As mentioned earlier, MSA on $\tilde{t}_{\Omega,c}$ and $\tilde{D}_{\Omega,c}$ is not that sensitive to the chosen d-value. Still, it is seen that $d = 3$ is a far worse choice than the others, making the p-value almost 10%. All other d-values result in p-values below 1%, why there is clear significance that this MSA is in fact improving the performance.

It should be mentioned though, that the improvements are rather small. The standard method managed to reduce its top 20 average of required iterations for convergence by 70%, whereas the I method is far from getting a similar performance boost. The relative improvement is only 25%.

To summarise the results from the more relaxed tolerance, it is still very clear that the I method fails to achieve the same benefits from MSA compared to the standard method. Nevertheless, using this relaxed tolerance, the I method did improve significantly compared to the MRA approach when using MSA on $\tilde{t}_{\Omega,c}$ and $\tilde{D}_{\Omega,c}$. Improvements were also seen for MSA on $t_{\Omega,c}$, but they were small compared to those seen by MSA on $\tilde{t}_{\Omega,c}$ and $\tilde{D}_{\Omega,c}$. Finally, the stability of the MSA approached improved tremendously with the change of tolerance.

Chapter 6

Diagnosis of the I Method

In this chapter, a diagnosis of the I method will be made. The basis of the diagnosis is a $(D_{\Omega,c}, t_{\Omega,c})$-plot which will be presented in section 6.1. Information from this section implies that the mutual correlation among the OD pairs has a great impact on the I method. The correlation is investigated thoroughly in section 6.2. Finally, section 6.3 uses the knowledge from the previous sections to present an alternative version of the I method – the C method – which will be tested, discussed, and evaluated.

6.1 Tracking Down the I Method Step by Step

In order to get a better understanding of the I method, it might be useful to look at a $(D_{\Omega,c}, t_{\Omega,c})$-plot for a single OD pair. The plot will, for a given OD pair Ω, show the points $(\tilde{D}_{\Omega,k}, t_{\Omega,c,k})$ and $(D_{\Omega,c,k}, \tilde{t}_{\Omega,c,k})$ after each iteration.

In figure 6.1 and 6.2 the first two iterations of the I method have been plotted for the OD-pair 2-4 ($\Omega = 10$) using OD matrix 14 from the Monte Carlo simulations. Just as in figure 3.3 and 3.4 in section 3.2.1, the demand and cost function are represented as a blue and a red line, respectively. The yellow dots indicate the interpolation points, which are connected by the yellow linear approximations.

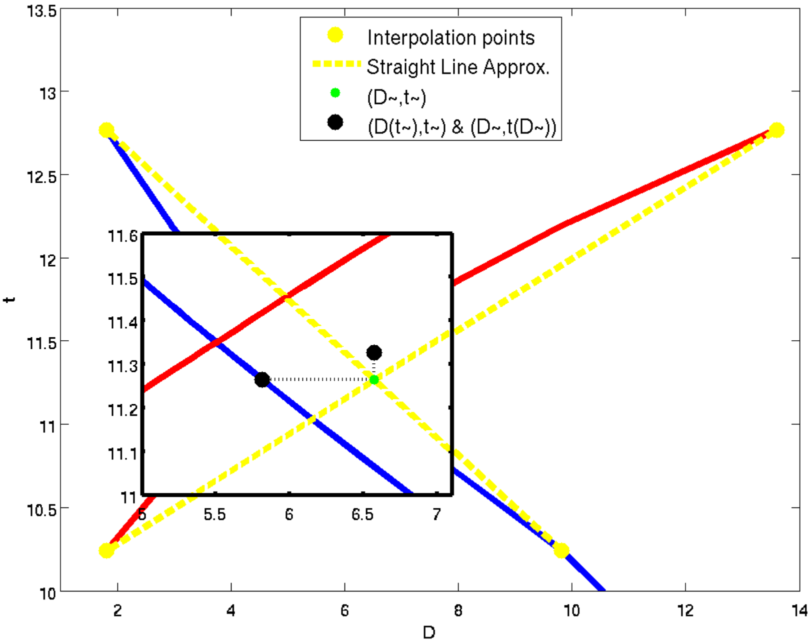

Figure 6.1: The first intersection of straight line approximations using the I method on the test network using OD matrix-14 and observing $\Omega = 10$.

The intersection point, $(\tilde{D}_{\Omega,c}, \tilde{t}_{\Omega,c})$ is marked with a green dot. Finally, the resulting points, $(\tilde{D}_{\Omega,c}, t_{\Omega,c}(\tilde{D}_{\Omega,c}))$ and $(D_{\Omega,c}(\tilde{t}_{\Omega,c}), \tilde{t}_{\Omega,c})$ are denoted by black dots.

It is seen that $(D_{\Omega,c}(\tilde{t}_{\Omega,c}), \tilde{t}_{\Omega,c})$ is quite close to the equilibrium point already after the first iteration. Unfortunately, the other point, $(\tilde{D}_{\Omega,c}, t_{\Omega,c}(\tilde{D}_{\Omega,c}))$, is far from the equilibrium, and also very far from the cost function it is supposed to approximate. When looking at the second iteration, the trend continues. The intersection point was very close to the approximation of the demand function of the previous iteration, why there is not much change in this point. But once again the approximation of the cost function is far from the cost function.

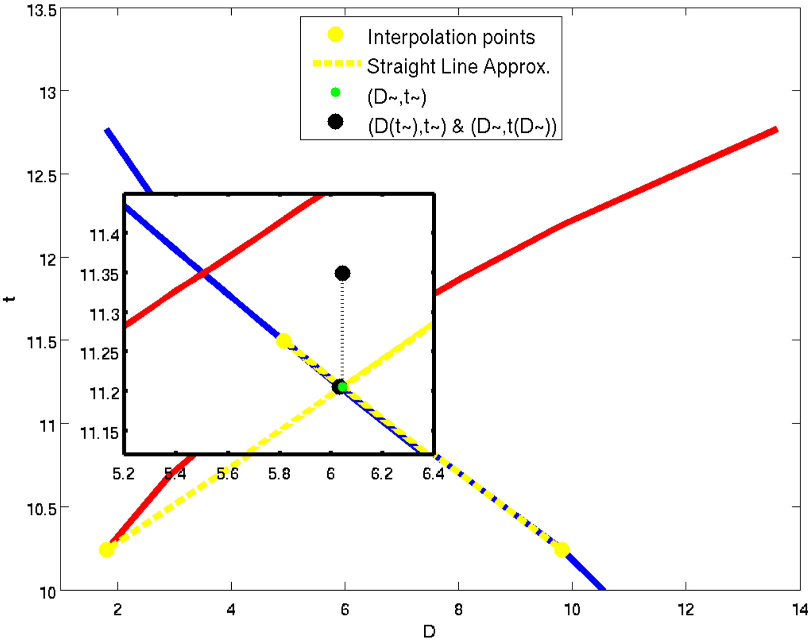

Figure 6.2: The second intersection of straight line approximations using the I method on the test network using OD-matrix 14 and observing $\Omega = 10$.

6.1.1 Mathematical Explanation of the Poor Approximation of the Cost Function

In order to determine why the I method fails to make a good approximation of the cost function, but approximates the demand function perfectly, it is necessary to dig into the mathematics of the two functions.

Recall that the normal 4-stage model has been reduced to a 2 step model (see section 2.3). In this model the demand between each OD pair is fixed for each test, why the demand can be calculated accurately solely by using $t_{\Omega,c}$,

$$D_{\Omega,c} = \frac{1}{1 + e^{\gamma(t_{\Omega,p} - t_{\Omega,c}) - \kappa_c}}.$$

Thus, no matter which method is used, it will always be possible to find a unique car demand for any average car travel time between any OD pair.

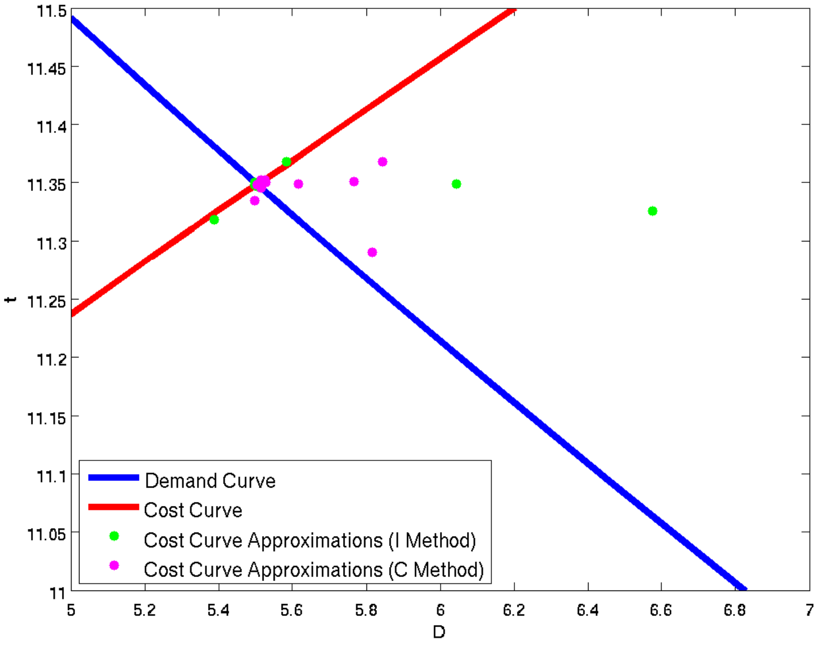

Figure 6.3: Point estimates of the cost function for the I method and C method, respectively.

For $t_{\Omega,c}$, on the other hand, this is *not* the case. Recall that $t_{\Omega,c}$ is the average travel time for a person travelling by car between the OD pair Ω, and that this is determined by the travel time of each route between Ω, which actually quite dependent on overlapping routes (see section 6.2.1). Thus, $t_{\Omega,c}$ is calculated by using $D_{\Upsilon,c}, \forall\Upsilon$, and not only for Ω, and as a consequence, the corresponding average travel time can not be uniquely determined solely on the basis of $D_{\Omega,c}$.

The above is the most severe problem of the I method. The problem is illustrated in figure 6.3 which shows how the I method fails to approximate the cost function accurately. The figure also shows the approximations made by the C method, which will be introduced in section 6.3.

On the other hand, since the standard method does not use any predictions using trends from an isolated OD pair, the problem does not occur for the standard method. Of course, when using MSA, the estimation of the cost function is in fact terrible. The difference is that when using the standard method, each

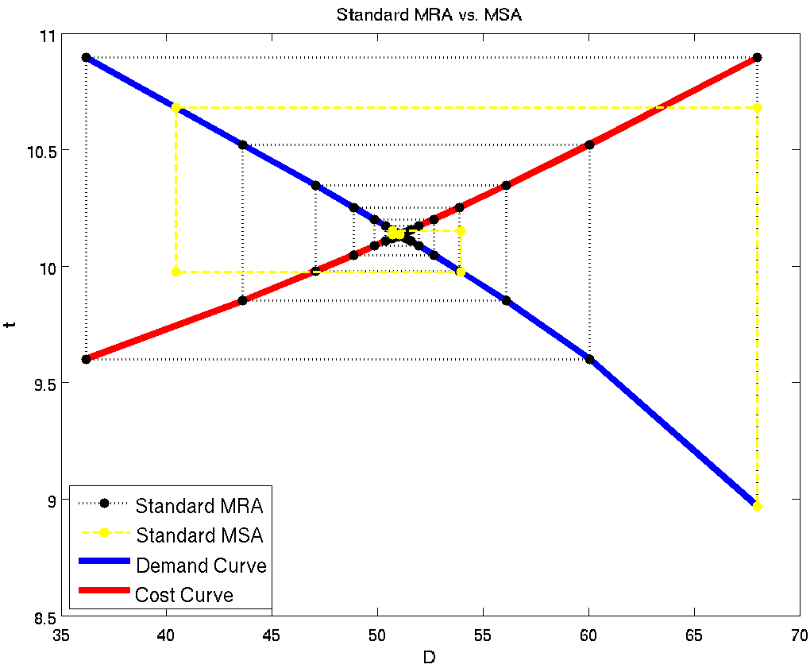

Figure 6.4: The iteration pattern of the standard method using MSA and MRA, respectively, on the test network using OD-matrix 14 and observing $\Omega = 10$.

estimate is a point estimate, and it is not used as an estimator of the trend of the cost function. Figure 6.4 shows how the standard method iterates robustly towards the solution when using MSA and MRA.

6.2 Mutual Correlation of the OD Pairs

Section 6.1.1 showed that the correlation will have a severe impact on the I method. The goal of this section is to determine, how this impact is seen, and especially to check if there are certain OD pairs that have greater impact on the correlation than others. The section will also deal with a discussion of applicable ways to use this knowledge to improve the I method.

In certain situations, the I method behaves in a very unexpected way that is far from the more or less ideal case presented in section 3.2.1, p. 28. Because the demand and cost of Ω is not entirely dependent on the latest cost and demand for Ω, but also on other OD pairs ($\Upsilon \neq \Omega$), sometimes $t_{\Omega,c}$ is increasing even though the demand is decreasing and vice versa.

In order to investigate this, we consider a single OD pair, Ω. If $D_\Upsilon = 0$, $\forall \Upsilon \neq \Omega$, then $t_{\Omega,c}$ would be relatively small since the only traffic in the network is caused by D_Ω. Now, let $D_\Upsilon = k$, $\forall \Upsilon \neq \Omega$, and let k increase. As k increases, more traffic have to be absorbed by the network, and more congestion will appear, making $t_{\Omega,c}$ higher.

The following example seeks to show this effect numerically.

Example 6.2.1 ($t_{\Omega,c}$ is also dependent on D_Υ, $\Upsilon \neq \Omega$.)
Consider the following OD matrix

$$\begin{bmatrix} 0 & k & 50 & k & k & k & k & k & k \\ k & 0 & k & k & k & k & k & k & k \\ 50 & k & 0 & k & k & k & k & k & k \\ k & k & k & 0 & k & k & k & k & k \\ k & k & k & k & 0 & k & k & k & k \\ k & k & k & k & k & 0 & k & k & k \\ k & k & k & k & k & k & 0 & k & k \\ k & k & k & k & k & k & k & 0 & k \\ k & k & k & k & k & k & k & k & 0 \end{bmatrix}.$$

We will now investigate what happens to the final demand and cost for cars between node 1 and 3 as k is increased. Since the OD pairs have been given ascending numbers based on the contained nodes (so that node 1 to node 3 corresponds to $\Omega = 2$), the values can be found in the following two variables, $D^*_{2,c}$ and $t^*_{2,c}$. The results are summarised in figure 6.5.

As expected, it is seen that $t^*_{2,c}$ is an increasing function of k, and as a consequence $D^*_{2,c}$ is a decreasing function of k.

□

6.2.1 Overlapping and Disjunct OD Pairs

We have now seen that the demand and cost of a specific OD pair is also dependent on the other OD pairs. Intuitively, for a single OD pair under consideration, all other OD pairs will not have the same impact. It is assumed that overlapping routes will have a greater impact than perpendicular routes.

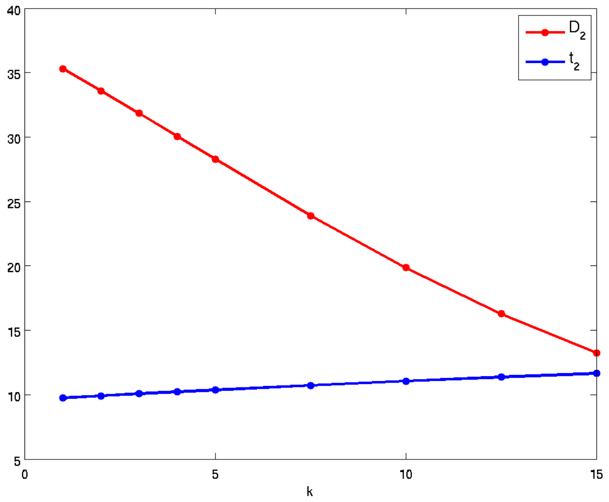

Figure 6.5: $D^*_{2,c}$ and $t^*_{2,c}$ as functions of the demand of all other OD pairs

The following example seeks to test this hypothesis.

Example 6.2.2 (Dependency of overlapping routes)
In order to test for the effect of overlapping routes, we will yet again see what happens to the final demand and cost of OD pair number 2. The OD matrix for this test case is given below,

$$\begin{bmatrix} 0 & 0 & 50 & 0 & 0 & 0 & k & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 50 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ k & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$[1]

OD pair number 6, that connects node 1 and 7,is expected to use many of the same links as OD pair number 2 (see figure 4.1, section 4.1, p. 38). By letting the demand of all other OD pairs be negligible small, and increasing D_6 gradually, it is possible to see how two overlapping routes affect each other.

[1] Due to the inability of the MatLab code to deal with no demands, all the zeros have been set to 10^{-4}

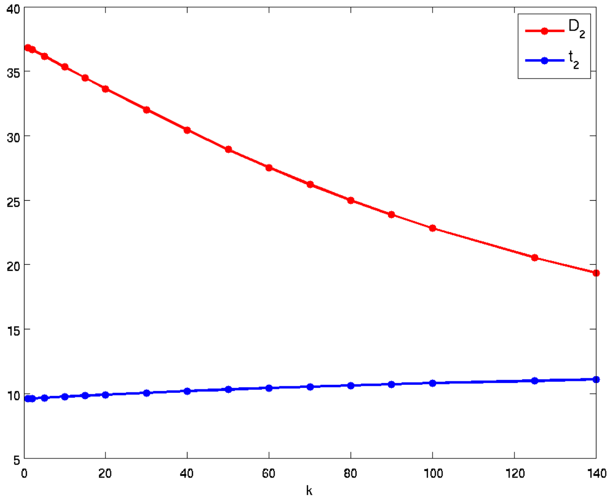

Figure 6.6: $D^*_{2,c}$ and $t^*_{2,c}$ as functions of D_6

The results are summarised figure 6.6. As expected, the demand and cost of OD pair number 2 is heavily influenced by the increasing demand of OD pair 6. □

It is clearly seen that the demand of an OD pair with many overlapping route had a huge impact on the results. Example 6.2.3 seeks to show whether or not a "perpendicular" OD pair will also have an effect.

Example 6.2.3 (Dependency of perpendicular routes)
Once again the demand of OD pair number 2 has been fixed to 50. In this test, the demand of OD Pair number 10 between note 2 and 4 will be varied, corresponding to the following OD matrix,

$$\begin{bmatrix} 0 & 0 & 50 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & k & 0 & 0 & 0 & 0 & 0 \\ 50 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & k & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

The results from these tests are summarised in figure 6.7. It is seen that the demand of OD pair 10 has almost no effect on the cost and demand of OD pair

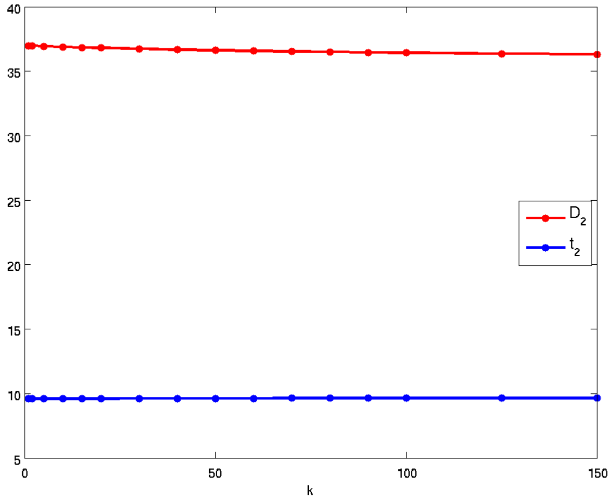

Figure 6.7: $D^*_{2,c}$ and $t^*_{2,c}$ as functions of D_{10}

2. From this it seems fair to conclude that perpendicular routes only have a minuscule impact.

□

6.2.2 The Inability of the I Method to Include Correlation

Previously, when using the I method, the method has worked isolated on one OD pair at the time. By doing this the dependency of all the other OD pairs has been ignored. But as seen in section 6.2, the other OD pairs do in fact have quite some influence on the demand and cost of each OD pair.

In order to correct this "error" in the previously introduced I method, it could intuitively be a very good idea to look at more than one OD pair at the time. By doing this it could be possible to include the impact of the other OD pairs in the model.

Unfortunately, the I method is based on straight line approximations of the demand and cost curves using only two interpolation points per curve. This only allows the determination of 2 coefficients (the constant term a and the slope coefficient b). Therefore, it is not possible to include any term correcting for the correlation of the other OD pairs.

If more information is wanted in each approximation, the number of included interpolation points have to be increased. Doing this would be beyond the scope of this study, but it could be a way to improve the performance of the I method. However, it should be kept in mind that adding interpolation points also delays the first "real" iteration of the I method.

6.3 The C Method - A Combination of the Standard and I Method

In this section the method called the *C method* will be explained and tested. It will begin with explaining the thought behind the C method.

6.3.1 The Idea of the C method

Due to the inability of the I method to approximate the cost function properly, the C method only seeks to approximate the demand function (but not the cost function) on the basis of the intersection point. Thus, instead of using $t_{\Omega,c}(\tilde{D}_{\Omega,c})$ and $D_{\Omega,c}(\tilde{t}_{\Omega,c})$ the C method uses $D_{\Omega,c}(\tilde{t}_{\Omega,c})$ and $t(D_{\Omega,c}(\tilde{t}_{\Omega,c}))$.

The C method only uses half of the information obtained by solving the linear system, but could possibly benefit from a better approximation of the cost curve. At least the thought behind the method is that by evaluating the travel time using the demand function, and not by a geometrical approach, the corresponding approximation of the cost function might be better. Figure 6.3 on p. 64, section 6.1.1 showed that the C method did in fact approximate the cost function in a different manner. If it was better or worse was hard to tell though.

An iteration pattern of the C method can be seen in figure 6.8 and 6.9. It is similar to the I method, but there is one major difference. When the I method has run for two iterations using intersections, the quadrilateral formed by the four intersection points can form any quadrilateral. When using the C method, the intersection points will always be vertically aligned in pairs of two, because the point estimate of the cost function is calculated by the demand function. As such the four points form a trapezoid, because two of the sides are parallel with each other. This will have an effect of the position of the next intersection, but it is hard to determine if it is beneficial or not.

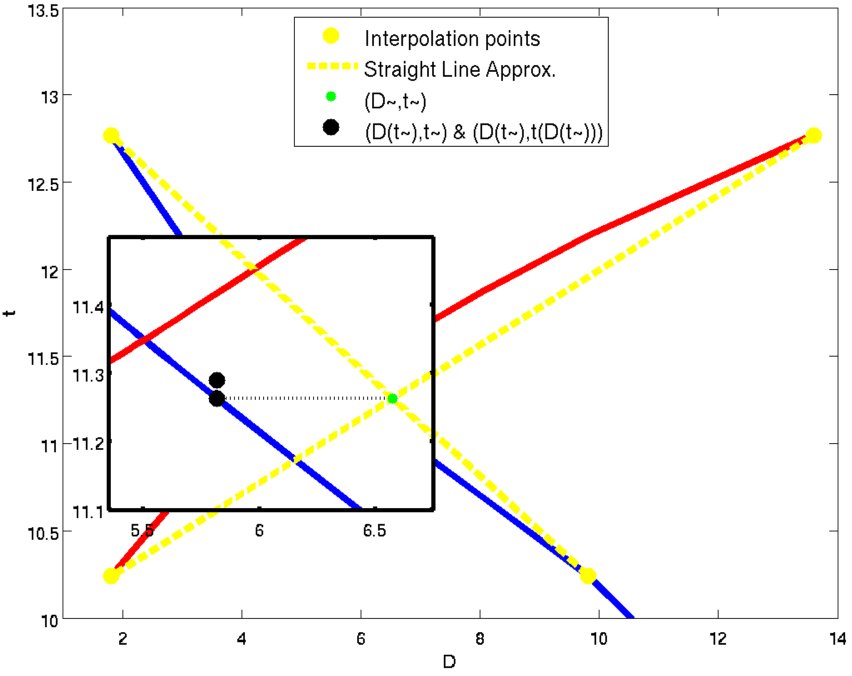

Figure 6.8: The first intersection of straight line approximations using the C method on the test network using OD-matrix 14 and observing $\Omega = 10$.

The algorithm of the method is presented below.

Algorithm 6.3.1 (The C method)

1. *Follow the standard method for the first 3 iterations (that is until* $D_{\Omega,c,1}$, $D_{\Omega,c,2}$, $D_{\Omega,c,3}$, $t_{\Omega,c,1}$, $t_{\Omega,c,2}$, $t_{\Omega,c,3}$ *are known for all OD-pairs* (Ω)*).*

2. *For all* Ω*, set* $\tilde{t}_{\Omega,c,3} = t_{\Omega,c,3}$, $\tilde{t}_{\Omega,c,2} = t_{\Omega,c,2}$, $\breve{D}_{\Omega,c,3} = D_{\Omega,c,2}$*, and* $\breve{D}_{\Omega,c,2} = D_{\Omega,c,1}$.

3. *Set* $k = 3$.

4. *For all* Ω*, calculate* $b_{D_\Omega} = \frac{\tilde{t}_{\Omega,c,k}-\tilde{t}_{\Omega,c,k-1}}{D_{\Omega,c,k}-D_{\Omega,c,k-1}}$, $a_{D_\Omega} = \tilde{t}_{\Omega,c,k} - b_{D_\Omega} D_{\Omega,c,k}$, $b_{t_\Omega} = \frac{t_{\Omega,c,k}-t_{\Omega,c,k-1}}{\breve{D}_{\Omega,c,k}-\breve{D}_{\Omega,c,k-1}}$*, and* $a_{t_\Omega} = t_{\Omega,c,k} - b_{t_\Omega} \breve{D}_{\Omega,c,k}$.

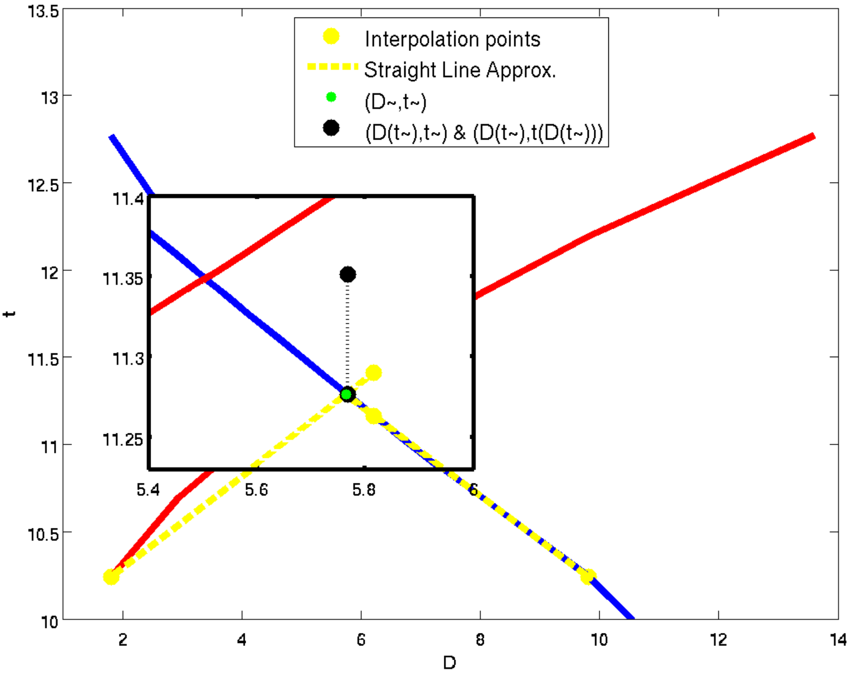

Figure 6.9: The second intersection of straight line approximations using the C method on the test network using OD-matrix 14 and observing $\Omega = 10$.

5. *Calculate* $\tilde{t}_{\Omega,c,k+1} = b_{t_\Omega} \frac{a_{D_\Omega} - a_{t_\Omega}}{b_{t_\Omega} - b_{D_\Omega}} + a_{t_\Omega}$.

6. *Calculate* $D_{\Omega,c,k+1} = D_{\Omega,c}(\tilde{t}_{\Omega,c,k+1})$ *and* $t_{\Omega,c,k+1} = t_{\Omega,c}(D_{\Omega,c,k+1})$.

7. *Set* $\breve{D}_{\Omega,c,k+1} = D_{\Omega,c,k+1}$.

8. *Increment* k *by one.*

9. *Repeat step 4-8 until* $||D_{\Omega,c,k} - D_{\Omega,c,k-1}|| \leq \tau$ *for all OD-pairs or* $k \geq k_{max}$.

It might seem weird to use $\breve{D}_{\Omega,c}$ since it is almost always equal to $D_{\Omega,c}$. For $k \geq 5$, it could have been omitted, but for the first two "real" iterations of the C method it is required in order to adjust for the fact that the coordinate progression changes from $(D_{\Omega,c,k}, t_{\Omega,c,k})$ and $(D_{\Omega,c,k}, t_{\Omega,c,k+1})$ to $(D_{\Omega,c,k}, \tilde{t}_{\Omega,c,k})$ and $(D_{\Omega,c,k}, t_{\Omega,c,k})$.

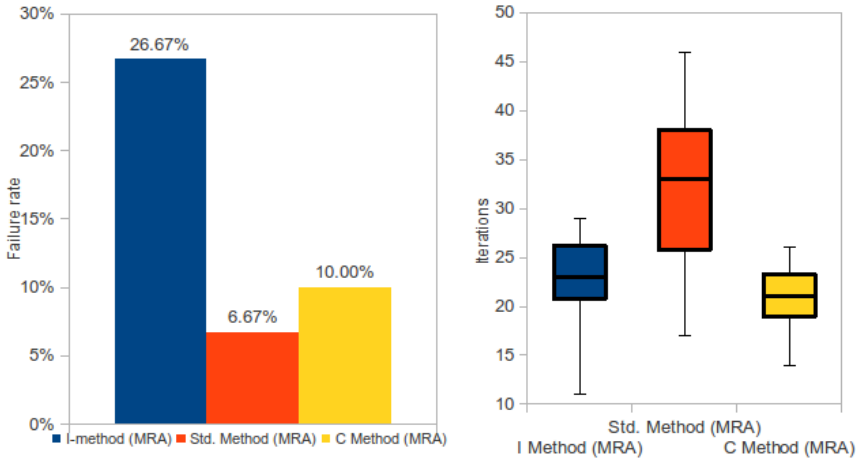

Figure 6.10: Summary of the test for the I method, standard method, and C method using MRA and $\tau = 0.001$

Whereas the I method used MSA on three different combinations of variables, the C method will only try to use MSA on $t_{\Omega,c}$ or on $\tilde{t}_{\Omega,c}$. It is hard too determine theoretically if any of the two MSA methods will perform better than the MRA approach. The performance tests in the following two subsections will reveal the answer for the test case.

6.3.2 Performance Tests of the C Method Using MRA

The tests of the C method have been run under the same conditions as the others (see section 5.1). This includes running the tests under both a strict and a relaxed tolerance.

Once again the evaluation of the method will begin by looking at the failure rate of the method using MRA under a strict stopping criteria. A figure of this can be seen on the left side of figure 6.10. It is seen that the C method is much more stable than the I method, managing to converge in 90% of the cases. Still it is a little more unstable than the standard MRA method, which only has a failure rate of 6.7%.

The right side of the same figure illustrates the number of required iterations when convergence was reached. It is seen that the C method generally performs better than the two other methods, and has smaller range. The best performance

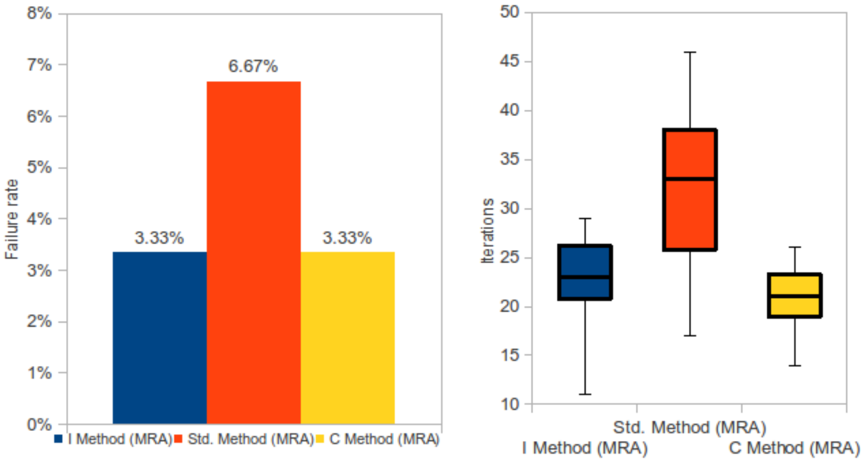

Figure 6.11: Summary of the test for the I method, standard method, and C method using MRA and $\tau = 0.01$

of the I method is better than the best performance of the C method, though. This is also why the C method is not significantly better than the I method on the basis of a paired t-test. The relevant p-value is in this case 0.1978.

The tests have also been run under a relaxed tolerance ($\tau = 0.01$). An illustrative summary of these tests can be seen in figure 6.11. Just as for the I method, the stability increases a lot by the relaxed tolerance. It makes the C method converge in 96.7% of the cases, making it just as stable as the I method, and more stable than the standard method.

The increase of stability is also reflected in the boxplot on the right side of figure 6.11. The range for the C method is far smaller than for any of the other methods. This also results in a smaller p-value, and this time the p-value for the C method being better than the I method on the basis of a paired t-test is 0.0781, which makes the C method questionably better than the I method.

To summarise, the C method is a lot more stable than the I method. It converges more often, and its range is much smaller. The performance on the other hand, is quite similar, and the C method only performed significantly better than the I method under a relaxed stopping criteria and a confidence level below 90%.

MRA	10.0%				
	$d = 1$	$d = 2$	$d = 3$	$d = 4$	$d = 5$
MSA on t_Ω	0%	0%	0%	0%	0%
MSA on $\tilde{t}_\Omega$	3.3%	0%	0%	0%	0%

Table 6.1: Failure rate of the tests for the C method using MSA and $\tau = 0.001$

6.3.3 Performance Tests of the C Method Using MSA

It was seen that the C method did not differ much from the I method when using MRA. In order to find out if a different conclusion can be obtained when using MSA, a series of tests has been run.

It turns out that MSA applied on the C method is a very good way to secure convergence. This is can be derived from table 6.1, that shows the failure rate of each method. When using a strict tolerance, MSA on $t_{\Omega,c}$ converges every time, and MSA on $\tilde{t}_{\Omega,c}$ also has a success rate of 100% for 4 of the 5 d-values. When $d = 1$ it fails to converge in 3.3% of the cases.

Although the stability has been increased, the performance did not improve by the MSA. This can be seen in table 6.2 and 6.3, which shows the relevant p-values from the paired t-tests for the C method using MSA on $t_{\Omega,c}$ and $\tilde{t}_{\Omega,c}$, respectively, compared to both the I method and C method using MRA, where the MSA is assumed better in the alternative hypothesis. It also shows the average for all tests, and for the top 20 tests. It can be seen that none of the MSA approaches manages to get a better average than any of the MRA methods, regardless of the d-value.

When running the tests under a relaxed tolerance, the stability of the C method

	C Method (MRA)	C Method (MSA on $t_{\Omega,c}$)				
		$d = 1$	$d = 2$	$d = 3$	$d = 4$	$d = 5$
$\bar{x}_{\text{All}}$	23.11	34.33	25.50	27.77	27.00	26.47
$\bar{x}_{20}$	20.70	31.05	21.10	23.85	21.95	20.65
p_I	0.1978	> 0.5	> 0.5	> 0.5	> 0.5	> 0.5
p_C	-	> 0.5	> 0.5	> 0.5	> 0.5	> 0.5

Table 6.2: p-values of the C method using MSA on $t_{\Omega,c}$ in a one-sided paired t-test compared to the MRA approach, with the alternative hypothesis being that the MSA is better, and using $\tau = 0.001$.

	C Method (MRA)	C Method (MSA on $\tilde{t}_{\Omega,c}$)				
		$d = 1$	$d = 2$	$d = 3$	$d = 4$	$d = 5$
$\bar{x}_{\text{All}}$	23.11	31.62	29.47	32.20	27.10	27.53
$\bar{x}_{20}$	20.70	23.80	20.20	24.55	20.05	22.15
p_I	0.1978	> 0.5	> 0.5	> 0.5	> 0.5	> 0.5
p_C	-	> 0.5	> 0.5	> 0.5	> 0.5	> 0.5

Table 6.3: p-values of the C method using MSA on $\tilde{t}_{\Omega,c}$ in a one-sided paired t-test compared to the MRA approach, with the alternative hypothesis being that the MSA is better, and using $\tau = 0.001$.

using MSA is total, in the sense that all methods converge for all OD matrices for all values of d (see table 6.4). This is in contrast to the MRA, which fails to converge in 3.3% of the cases.

To study the performance of the C method using MSA under a strict tolerance, we have tables 6.5 and 6.6, which are the counterparts of table 6.2 and 6.3. It is seen that using $d \geq 2$ for MSA applied on $t_{\Omega,c}$ results in fewer average number of required iterations than both the I and C method using MRA. Whereas it manages to get significantly better than the I method, by having a p-value of 0.0314, the difference is insignificant compared to the C-method using MRA. The relevant p-value is 0.1145, which tells that it can not be ruled out that the differences are caused by pure chance.

MSA on $\tilde{t}_{\Omega,c}$ improves the performance of the C method when $d = 1, 4, 5$. Whereas it is almost significantly better than the I method using MRA when $d = 4$ on a 95%-confidence interval, it is far from being significantly better than the C method using MRA.

MRA	3.3%				
	$d = 1$	$d = 2$	$d = 3$	$d = 4$	$d = 5$
MSA on t_Ω	0%	0%	0%	0%	0%
MSA on $\tilde{t}_\Omega$	0%	0%	0%	0%	0%

Table 6.4: Failure rate of the tests for the C method using $\tau = 0.01$

	C Method (MRA)	C Method (MSA on $t_{\Omega,c}$)				
		$d=1$	$d=2$	$d=3$	$d=4$	$d=5$
$\bar{x}_{\text{All}}$	16.48	19.30	16.27	15.17	15.80	16.17
$\bar{x}_{20}$	14.55	17.10	12.70	11.90	12.35	12.25
p_I	0.0781	> 0.5	0.1693	0.0314	0.1289	0.2514
p_C	-	> 0.5	0.4465	0.1145	0.3046	0.4492

Table 6.5: p-values of the C method using MSA on $t_{\Omega,c}$ in a one-sided paired t-test compared to the MRA approach, with the alternative hypothesis being that the MSA is better, and using $\tau = 0.01$.

To summarise the evaluation of the C method using MSA, it was seen that MSA increases the stability of the C method, although the C method using MRA was quite stable in itself. The required number of iterations to reach convergence was higher than both the I and C method using MRA, when facing a strict convergence criteria. This is in contrast to the tests using a relaxed tolerance, where the MSA did improve the performance a little in some cases. Although MSA on $t_{\Omega,c}$ and $d = 2$ was significantly better than the I method using MRA, a similar conclusion could not be made when compared to the C method.

	C Method (MRA)	C Method (MSA on $\tilde{t}_{\Omega,c}$)				
		$d=1$	$d=2$	$d=3$	$d=4$	$d=5$
$\bar{x}_{\text{All}}$	16.48	16.10	21.87	18.33	15.90	16.07
$\bar{x}_{20}$	14.55	12.55	14.85	13.50	12.95	12.35
p_I	0.0781	0.2003	> 0.5	> 0.5	0.0770	0.1626
p_C	-	0.3902	> 0.5	> 0.5	0.1970	0.2135

Table 6.6: p-values of the C method using MSA on $\tilde{t}_{\Omega,c}$ in a one-sided paired t-test compared to the MRA approach, with the alternative hypothesis being that the MSA is better, and using $\tau = 0.01$.

CHAPTER 7

Conclusion

The determination of the intersection of the demand curve, $D(t)$, and travel time curve, $t(D)$, between any two nodes in a transport network is of great importance in transportation systems analysis. This equilibrium is found iteratively using a model containing an external mode choice loop and an internal route choice loop. The two loops affect each other due to congestion, but as the iteration process approaches the equilibrium, the changes become smaller and smaller eventually revealing the equilibrium. This thesis sought to examine the use of the intersection between straight line approximations of the demand and cost curves (the I method) in the determination process. A corresponding MatLab implementation was created, and the method was tested on a small-scale network using various Monte Carlo generated fixed OD matrices. The results showed that the I method was superior to the standard method when using the method of repeated approximations (MRA), but that it did not benefit much from the method of successive averages (MSA). As a consequence, it was outperformed by the standard method using MSA. The main problem of the I method was the mutual correlation among similar OD pairs, which made the I method fail to approximate the cost function sufficiently well. It turned out to be impossible to correct for the mutual correlation, since the approximations were already fully determined. The thesis did not rule out the existence of an approach using the intersection of approximations of the demand and cost functions with the ability to outperform a properly chosen standard MSA, but suggested that such an approach in all likelihood will be using more than two interpolation points.

Appendix A

Results Tables of the Comparative Monte Carlo Tests of the I Method

In this appendix, the tables showing the results of the performance tests of the I method, standard method, and C method can be found. For more information about the tests and an analysis of the results, see section 5, p. 51 for information on the tests of the standard and I method, and section 6.3, p. 70 for more information on the C method.

The d's in the tables are the d-values used to calculate the MSA-coefficient,

$$a_k = \frac{k^d}{\sum_{j=1}^{k} j^d}.$$

The entry “NC” denotes that there was no convergence due to a diverging internal loop. The entry > 200 indicates that the method failed to converge within 200 external iterations, but that all internal loops converged up to this point.

The $\bar{x}$ has been calculated as the mean of all the tests that converged. The corresponding standard deviation belonging to the mean, is given by

$$S_{\bar{x}} = \frac{S_x}{\sqrt{n_c}},$$

where S_x is the standard deviation found among the converged tests, and n_c is the number of tests managing to converge.

The p-values in the bottom of the table are the p-values from the one-sided hypothesis tests. In the paired t-tests used to calculate the p-values all valid measurements (converged tests) have been used. p_I indicates the probability of obtaining a mean value lower than the observed mean from the method under consideration, given that the true mean of the method under consideration is equal to the true mean of the I method using MRA. Likewise, p_C is the p-value with respect to the C method using MRA.

OD-matrix no.	I Method	Standard Method	C Method	$D_\bullet$	μ
1	15	25	21	408	0.735
2	13	30	19	430	0.725
3	21	64	28	498	0.701
4	41	> 200	30	450	0.714
5	NC	46	24	488	0.705
6	21	41	24	466	0.709
7	NC	53	NC	495	0.701
8	NC	37	NC	483	0.716
9	21	41	26	481	0.709
10	23	27	22	464	0.715
11	NC	28	23	518	0.700
12	26	52	33	530	0.696
13	29	88	28	524	0.685
14	29	97	32	538	0.686
15	NC	38	19	447	0.716
16	19	21	19	451	0.724
17	26	28	20	486	0.709
18	21	55	NC	573	0.675
19	29	38	21	441	0.712
20	NC	37	24	443	0.722
21	11	17	17	435	0.733
22	NC	> 200	30	512	0.696
23	28	26	14	448	0.716
24	NC	93	29	486	0.703
25	23	36	15	452	0.715
26	26	37	22	440	0.725
27	27	66	26	480	0.715
28	20	23	21	416	0.723
29	30	42	20	476	0.706
30	23	22	17	440	0.726
$\bar{x}$	23.7	43.1	23.1	473.3	0.710
$S_{\bar{x}}$	1.4	4.1	1.0	7.1	0.003
p_I	-	> 0.5	0.1977	-	-
p_C	> 0.5	> 0.5	-	-	-

Table A.1: Results of the I, standard and C method using MRA and $\tau = 0.001$

OD-matrix no.	I Method	Standard Method	C Method	$D_\bullet$	μ
1	8	20	12	408	0.735
2	9	24	18	430	0.725
3	19	50	16	498	0.701
4	25	> 200	25	450	0.714
5	17	36	16	488	0.705
6	16	32	18	466	0.709
7	28	43	27	495	0.701
8	20	29	17	483	0.716
9	18	32	12	481	0.709
10	20	21	15	464	0.715
11	NC	22	17	518	0.700
12	18	41	19	530	0.696
13	22	69	16	524	0.685
14	15	76	NC	538	0.686
15	20	30	14	447	0.716
16	13	16	16	451	0.724
17	20	22	16	486	0.709
18	19	43	20	573	0.675
19	19	30	14	441	0.712
20	19	29	13	443	0.722
21	11	13	11	435	0.733
22	20	> 200	24	512	0.696
23	17	20	14	448	0.716
24	25	74	19	486	0.703
25	12	28	15	452	0.715
26	17	29	16	440	0.725
27	16	52	15	480	0.715
28	13	18	16	416	0.723
29	19	33	15	476	0.706
30	11	17	12	440	0.726
$\bar{x}$	17.4	33.9	16.5	473.3	0.710
$S_{\bar{x}}$	0.9	3.2	0.7	7.1	0.003
p_I	-	> 0.5	0.0781	-	-
p_C	> 0.5	> 0.5	-	-	-

Table A.2: Results of the I, standard and C method using MRA and $\tau = 0.01$

OD Matrix	Standard method (MSA on $t_{\Omega,c}$)					$D_\bullet$	μ
	$d=1$	$d=2$	$d=3$	$d=4$	$d=5$		
1	21	13	10	9	8	408	0.735
2	21	13	10	9	8	430	0.725
3	21	15	12	11	10	498	0.701
4	20	14	12	11	10	450	0.714
5	16	12	10	9	9	488	0.705
6	17	12	10	9	9	466	0.709
7	17	13	10	9	9	495	0.701
8	19	12	9	8	9	483	0.716
9	16	12	10	9	9	481	0.709
10	21	13	10	9	9	464	0.715
11	22	13	10	9	8	518	0.700
12	23	16	12	11	10	530	0.696
13	25	17	13	11	11	524	0.685
14	22	15	12	10	10	538	0.686
15	19	12	10	9	9	447	0.716
16	20	13	9	8	8	451	0.724
17	22	13	10	9	9	486	0.709
18	25	17	13	11	11	573	0.675
19	17	12	10	9	9	441	0.712
20	21	13	10	9	9	443	0.722
21	13	8	7	7	7	435	0.733
22	17	18	14	12	12	512	0.696
23	22	13	10	9	8	448	0.716
24	16	11	10	9	9	486	0.703
25	16	11	9	8	8	452	0.715
26	19	12	10	9	8	440	0.725
27	22	15	12	11	10	480	0.715
28	18	11	9	8	8	416	0.723
29	21	13	11	10	9	476	0.706
30	19	12	9	8	8	440	0.726
$\bar{x}$	19.6	13.1	10.4	9.3	9	473.3	0.710
$S_{\bar{x}}$	0.5	0.4	0.3	0.2	0.2	7.1	0.003
p_I	0.0054	~ 0	~ 0	~ 0	~ 0	-	-
p_C	0.0007	~ 0	~ 0	~ 0	~ 0	-	-

Table A.3: Results of the standard method using MSA on $t_{\Omega,c}$ and $\tau = 0.001$.

OD Matrix	Standard method (MSA on $t_{\Omega,c}$)					$D_\bullet$	μ
	$d=1$	$d=2$	$d=3$	$d=4$	$d=5$		
1	10	8	7	7	7	408	0.735
2	10	8	7	7	8	430	0.725
3	10	9	8	8	9	498	0.701
4	10	8	7	8	9	450	0.714
5	8	7	7	7	8	488	0.705
6	8	7	7	7	8	466	0.709
7	8	8	7	7	8	495	0.701
8	9	7	6	7	8	483	0.716
9	8	7	7	7	8	481	0.709
10	10	8	7	7	7	464	0.715
11	11	8	7	7	8	518	0.700
12	11	9	8	7	8	530	0.696
13	12	10	8	8	9	524	0.685
14	10	9	8	8	9	538	0.686
15	9	7	6	7	8	447	0.716
16	10	7	6	6	7	451	0.724
17	11	8	7	7	7	486	0.709
18	12	10	8	8	9	573	0.675
19	9	7	7	7	8	441	0.712
20	10	8	7	7	8	443	0.722
21	7	5	5	6	7	435	0.733
22	13	10	9	9	9	512	0.696
23	10	8	7	7	7	448	0.716
24	8	7	7	8	9	486	0.703
25	8	7	6	7	8	452	0.715
26	9	7	6	7	8	440	0.725
27	10	9	8	8	9	480	0.715
28	9	7	6	6	7	416	0.723
29	10	8	7	7	8	476	0.706
30	9	7	6	6	7	440	0.726
$\bar{x}$	9.6	7.8	7	7.2	8	473.3	0.710
$S_{\bar{x}}$	0.3	0.2	0.2	0.1	0.1	7.1	0.003
p_I	~ 0	~ 0	~ 0	~ 0	~ 0	-	-
p_C	~ 0	~ 0	~ 0	~ 0	~ 0	-	-

Table A.4: Results of the standard method using MSA on $t_{\Omega,c}$ and $\tau = 0.01$.

OD Matrix	I method (MSA on $t_{\Omega,c}$)					$D_\bullet$	μ
	$d = 1$	$d = 2$	$d = 3$	$d = 4$	$d = 5$		
1	30	21	23	29	29	408	0.735
2	28	20	39	18	35	430	0.725
3	28	20	38	27	37	498	0.701
4	34	28	40	42	31	450	0.714
5	44	31	25	30	40	488	0.705
6	29	21	31	29	31	466	0.709
7	30	23	40	30	39	495	0.701
8	36	29	21	40	29	483	0.716
9	34	22	30	33	51	481	0.709
10	33	23	27	26	18	464	0.715
11	36	23	25	34	28	518	0.700
12	35	23	36	23	24	530	0.696
13	41	23	24	25	30	524	0.685
14	0	26	36	47	0	538	0.686
15	23	24	33	32	34	447	0.716
16	29	12	21	18	13	451	0.724
17	31	24	23	0	33	486	0.709
18	33	32	30	47	38	573	0.675
19	38	21	30	26	26	441	0.712
20	25	19	19	22	22	443	0.722
21	17	12	14	26	13	435	0.733
22	33	31	30	32	46	512	0.696
23	42	21	29	21	30	448	0.716
24	30	27	32	56	37	486	0.703
25	32	19	56	21	37	452	0.715
26	30	35	27	21	35	440	0.725
27	27	26	45	34	28	480	0.715
28	36	18	19	18	14	416	0.723
29	28	21	36	57	30	476	0.706
30	31	18	17	18	13	440	0.726
$\bar{x}$	31.8	23.1	29.9	30.4	30	473.3	0.710
$S_{\bar{x}}$	1	1	1.7	2	1.8	7.1	0.003
p_I	> 0.5	0.1061	> 0.5	> 0.5	> 0.5	-	-
p_C	> 0.5	0.2896	> 0.5	> 0.5	> 0.5	-	-

Table A.5: Results of the I method using MSA on $t_{\Omega,c}$ and $\tau = 0.001$.

OD Matrix	I method (MSA on $t_{\Omega,c}$)					$D_\bullet$	μ
	$d = 1$	$d = 2$	$d = 3$	$d = 4$	$d = 5$		
1	19	12	19	14	30	408	0.735
2	18	12	10	13	13	430	0.725
3	31	15	14	17	29	498	0.701
4	32	23	21	27	29	450	0.714
5	20	16	14	18	34	488	0.705
6	18	14	21	14	14	466	0.709
7	13	17	18	24	32	495	0.701
8	14	23	17	14	14	483	0.716
9	14	14	21	27	34	481	0.709
10	17	14	10	13	13	464	0.715
11	22	12	10	14	13	518	0.700
12	17	18	22	21	19	530	0.696
13	25	18	13	14	19	524	0.685
14	22	24	28	35	26	538	0.686
15	21	14	21	14	14	447	0.716
16	21	12	9	10	10	451	0.724
17	18	12	10	25	24	486	0.709
18	26	17	15	25	21	573	0.675
19	16	14	22	14	14	441	0.712
20	16	13	13	13	14	443	0.722
21	13	8	9	9	9	435	0.733
22	22	26	24	22	26	512	0.696
23	18	12	22	13	13	448	0.716
24	21	22	14	26	29	486	0.703
25	21	9	18	14	29	452	0.715
26	28	19	18	13	13	440	0.725
27	21	14	25	14	22	480	0.715
28	14	12	9	10	10	416	0.723
29	20	18	30	23	27	476	0.706
30	15	12	9	10	10	440	0.726
$\bar{x}$	19.8	15.5	16.9	17.3	20.1	473.3	0.710
$S_{\bar{x}}$	0.9	0.8	1.1	1.2	1.5	7.1	0.003
p_I	> 0.5	0.0151	0.3928	0.5000	> 0.5	-	-
p_C	> 0.5	0.0292	> 0.5	> 0.5	> 0.5	-	-

Table A.6: Results of the I method using MSA on $t_{\Omega,c}$ and $\tau = 0.01$.

OD Matrix	I method (MSA on $t_{\Omega,c}$ & $D_{\Omega,c}$)					$D_\bullet$	μ
	$d=1$	$d=2$	$d=3$	$d=4$	$d=5$		
1	45	25	31	48	90	408	0.735
2	50	31	28	41	33	430	0.725
3	39	45	35	89	44	498	0.701
4	40	58	88	51	66	450	0.714
5	32	36	36	NC	NC	488	0.705
6	34	28	42	52	53	466	0.709
7	36	33	40	46	49	495	0.701
8	31	39	35	51	NC	483	0.716
9	38	69	33	36	75	481	0.709
10	41	29	NC	105	47	464	0.715
11	59	23	34	26	39	518	0.700
12	49	46	67	33	43	530	0.696
13	NC	58	40	33	49	524	0.685
14	63	43	49	63	48	538	0.686
15	48	21	51	66	64	447	0.716
16	35	25	60	39	27	451	0.724
17	51	37	51	48	38	486	0.709
18	49	47	49	117	58	573	0.675
19	56	38	41	45	44	441	0.712
20	30	30	74	45	26	443	0.722
21	23	23	27	34	30	435	0.733
22	71	48	71	54	84	512	0.696
23	36	31	34	62	26	448	0.716
24	48	33	57	35	39	486	0.703
25	27	49	120	30	44	452	0.715
26	31	20	61	37	70	440	0.725
27	49	70	40	51	47	480	0.715
28	30	20	61	41	50	416	0.723
29	45	40	33	28	32	476	0.706
30	39	18	44	25	24	440	0.726
$\bar{x}$	42.2	37.1	49.4	49.3	47.8	473.3	0.710
$S_{\bar{x}}$	2.1	2.6	3.8	4.1	3.3	7.1	0.003
p_I	> 0.5	> 0.5	> 0.5	> 0.5	> 0.5	-	-
p_C	> 0.5	> 0.5	> 0.5	> 0.5	> 0.5	-	-

Table A.7: Results of the I method using MSA on $t_{\Omega,c}$ & $D_{\Omega,c}$ and $\tau = 0.001$.

OD Matrix	I method (MSA on $t_{\Omega,c}$ & $D_{\Omega,c}$)					$D_\bullet$	μ
	$d=1$	$d=2$	$d=3$	$d=4$	$d=5$		
1	17	12	16	39	23	408	0.735
2	12	11	18	21	25	430	0.725
3	24	42	31	29	21	498	0.701
4	18	21	34	25	24	450	0.714
5	22	24	27	41	34	488	0.705
6	15	16	41	57	21	466	0.709
7	15	16	21	33	31	495	0.701
8	12	12	40	30	34	483	0.716
9	13	14	26	40	53	481	0.709
10	39	12	19	21	45	464	0.715
11	17	13	17	19	24	518	0.700
12	20	26	21	38	25	530	0.696
13	33	16	27	32	28	524	0.685
14	19	24	44	26	47	538	0.686
15	20	13	19	17	34	447	0.716
16	18	11	32	21	25	451	0.724
17	23	20	22	20	18	486	0.709
18	21	15	35	25	29	573	0.675
19	39	15	25	19	32	441	0.712
20	14	26	16	19	21	443	0.722
21	11	11	12	24	13	435	0.733
22	14	22	21	34	21	512	0.696
23	21	15	12	18	16	448	0.716
24	31	15	25	19	22	486	0.703
25	27	14	17	19	22	452	0.715
26	14	12	26	17	18	440	0.725
27	18	23	16	28	20	480	0.715
28	17	15	11	32	21	416	0.723
29	17	37	43	21	41	476	0.706
30	13	10	11	NC	30	440	0.726
$\bar{x}$	19.8	17.8	24.2	27	27.3	473.3	0.710
$S_{\bar{x}}$	1.4	1.4	1.8	1.8	1.7	7.1	0.003
p_I	> 0.5	> 0.5	> 0.5	> 0.5	> 0.5	-	-
p_C	> 0.5	> 0.5	> 0.5	> 0.5	> 0.5	-	-

Table A.8: Results of the I method using MSA on $t_{\Omega,c}$ & $D_{\Omega,c}$ and $\tau = 0.01$.

OD Matrix	I method (MSA on $\tilde{t}_{\Omega,c}$ & $\tilde{D}_{\Omega,c}$)					$D_\bullet$	μ
	$d=1$	$d=2$	$d=3$	$d=4$	$d=5$		
1	16	13	11	13	11	408	0.735
2	17	14	17	13	11	430	0.725
3	20	NC	25	27	20	498	0.701
4	NC	NC	NC	36	28	450	0.714
5	26	23	29	34	22	488	0.705
6	18	15	20	30	36	466	0.709
7	23	19	33	22	23	495	0.701
8	24	21	25	NC	24	483	0.716
9	29	19	20	18	29	481	0.709
10	20	20	27	30	24	464	0.715
11	22	28	NC	23	27	518	0.700
12	26	NC	25	15	21	530	0.696
13	NC	52	39	24	31	524	0.685
14	NC	22	20	18	36	538	0.686
15	22	15	25	18	22	447	0.716
16	NC	18	24	34	36	451	0.724
17	31	18	22	18	14	486	0.709
18	25	17	21	28	26	573	0.675
19	19	26	20	27	15	441	0.712
20	34	21	31	19	16	443	0.722
21	13	14	14	13	11	435	0.733
22	44	NC	NC	24	NC	512	0.696
23	NC	17	27	28	21	448	0.716
24	24	30	26	28	31	486	0.703
25	23	18	17	19	18	452	0.715
26	19	19	16	19	NC	440	0.725
27	NC	NC	NC	NC	25	480	0.715
28	19	15	NC	15	28	416	0.723
29	27	NC	31	28	26	476	0.706
30	16	13	16	14	18	440	0.726
$\bar{x}$	23.2	20.3	23.2	22.7	23.2	473.3	0.710
$S_{\bar{x}}$	1.4	1.7	1.3	1.3	1.4	7.1	0.003
p_I	0.2876	0.0762	0.1846	0.1935	0.3882	-	-
p_C	> 0.5	0.2111	> 0.5	0.3765	> 0.5	-	-

Table A.9: Results of the I method using MSA on $\tilde{t}_{\Omega,c}$ & $\tilde{D}_{\Omega,c}$ and $\tau = 0.001$.

OD Matrix	I method (MSA on $\tilde{t}_{\Omega,c}$) & $\tilde{D}_{\Omega,c}$)					$D_\bullet$	μ
	$d=1$	$d=2$	$d=3$	$d=4$	$d=5$		
1	9	9	10	8	8	408	0.735
2	13	9	12	9	11	430	0.725
3	11	13	16	14	18	498	0.701
4	13	NC	17	NC	17	450	0.714
5	14	16	15	16	15	488	0.705
6	10	10	10	14	11	466	0.709
7	15	15	11	16	16	495	0.701
8	11	18	15	16	16	483	0.716
9	12	12	11	20	12	481	0.709
10	15	18	13	17	20	464	0.715
11	11	17	14	16	23	518	0.700
12	15	16	12	19	12	530	0.696
13	NC	18	22	17	16	524	0.685
14	14	17	12	15	12	538	0.686
15	22	16	11	15	14	447	0.716
16	19	22	14	17	15	451	0.724
17	15	11	12	13	12	486	0.709
18	17	11	13	16	15	573	0.675
19	13	15	32	16	12	441	0.712
20	18	12	14	12	14	443	0.722
21	7	7	9	9	11	435	0.733
22	20	31	26	17	18	512	0.696
23	11	13	20	16	12	448	0.716
24	13	17	35	17	22	486	0.703
25	10	10	16	11	11	452	0.715
26	11	12	12	19	11	440	0.725
27	15	17	15	17	21	480	0.715
28	8	13	12	11	12	416	0.723
29	20	14	16	13	21	476	0.706
30	9	9	25	11	11	440	0.726
$\bar{x}$	13.5	14.4	15.7	14.7	14.6	473.3	0.710
$S_{\bar{x}}$	0.7	0.9	1.2	0.6	0.7	7.1	0.003
p_I	0.0001	0.0026	0.0987	0.0004	0.0001	-	-
p_C	0.0009	0.0164	0.3214	0.0272	0.0209	-	-

Table A.10: Results of the I method using MSA on $\tilde{t}_{\Omega,c}$ & $\tilde{D}_{\Omega,c}$ and $\tau = 0.01$.

OD Matrix	C method (MSA on $t_{\Omega,c}$)					$D_\bullet$	μ
	$d=1$	$d=2$	$d=3$	$d=4$	$d=5$		
1	36	17	33	21	21	408	0.735
2	29	22	44	31	18	430	0.725
3	27	25	42	44	33	498	0.701
4	35	41	25	39	36	450	0.714
5	39	25	21	23	22	488	0.705
6	35	22	35	27	46	466	0.709
7	42	44	33	29	34	495	0.701
8	31	24	31	21	30	483	0.716
9	37	23	22	32	39	481	0.709
10	32	20	25	18	14	464	0.715
11	29	27	31	18	18	518	0.700
12	30	23	31	23	31	530	0.696
13	48	41	27	25	38	524	0.685
14	52	18	33	33	40	538	0.686
15	33	22	25	21	22	447	0.716
16	34	21	18	18	13	451	0.724
17	28	22	29	40	18	486	0.709
18	44	28	27	49	46	573	0.675
19	34	22	24	31	28	441	0.712
20	32	27	13	23	18	443	0.722
21	31	12	19	18	13	435	0.733
22	33	44	38	29	34	512	0.696
23	39	14	24	18	27	448	0.716
24	37	31	21	26	35	486	0.703
25	25	33	21	21	26	452	0.715
26	30	27	22	21	21	440	0.725
27	35	24	36	41	26	480	0.715
28	32	23	27	25	13	416	0.723
29	31	23	30	31	21	476	0.706
30	30	20	26	14	13	440	0.726
$\bar{x}$	34.3	25.5	27.8	27	26.5	473.3	0.710
$S_{\bar{x}}$	1.1	1.5	1.3	1.6	1.8	7.1	0.003
p_I	> 0.5	> 0.5	> 0.5	> 0.5	> 0.5	-	-
p_C	> 0.5	> 0.5	> 0.5	> 0.5	> 0.5	-	-

Table A.11: Results of the C method using MSA on $t_{\Omega,c}$ and $\tau = 0.001$.

OD Matrix	C method (MSA on $t_{\Omega,c}$)					$D_\bullet$	μ
	$d=1$	$d=2$	$d=3$	$d=4$	$d=5$		
1	21	19	10	14	13	408	0.735
2	17	13	10	13	13	430	0.725
3	17	17	22	32	14	498	0.701
4	23	19	14	14	14	450	0.714
5	16	23	18	13	13	488	0.705
6	18	15	18	14	13	466	0.709
7	23	21	14	13	13	495	0.701
8	16	17	14	13	13	483	0.716
9	13	14	14	14	13	481	0.709
10	16	20	10	13	10	464	0.715
11	17	14	10	13	13	518	0.700
12	26	10	21	25	43	530	0.696
13	19	29	23	22	37	524	0.685
14	19	14	14	14	13	538	0.686
15	13	14	14	13	13	447	0.716
16	17	16	9	10	10	451	0.724
17	19	9	10	13	15	486	0.709
18	25	28	20	29	41	573	0.675
19	22	14	14	13	13	441	0.712
20	22	13	13	13	13	443	0.722
21	15	8	9	9	10	435	0.733
22	24	35	29	34	32	512	0.696
23	19	12	14	10	13	448	0.716
24	29	23	13	13	13	486	0.703
25	18	9	14	14	13	452	0.715
26	18	14	18	13	13	440	0.725
27	20	14	26	25	17	480	0.715
28	16	12	9	10	10	416	0.723
29	22	10	22	18	14	476	0.706
30	19	12	9	10	10	440	0.726
$\bar{x}$	19.3	16.3	15.2	15.8	16.2	473.3	0.710
$S_{\bar{x}}$	0.7	1.2	1.0	1.2	1.7	7.1	0.003
p_I	> 0.5	0.1693	0.0314	0.1289	0.2514	-	-
p_C	> 0.5	0.4465	0.1145	0.3046	0.4492	-	-

Table A.12: Results of the C method using MSA on $t_{\Omega,c}$ and $\tau = 0.01$.

OD Matrix	C method (MSA on $\tilde{t}_{\Omega,c}$)					$D_\bullet$	μ
	$d=1$	$d=2$	$d=3$	$d=4$	$d=5$		
1	20	17	15	17	24	408	0.735
2	46	16	39	14	15	430	0.725
3	28	23	53	29	33	498	0.701
4	43	72	35	51	47	450	0.714
5	21	20	67	28	21	488	0.705
6	28	32	28	23	22	466	0.709
7	67	31	30	33	23	495	0.701
8	20	20	27	31	28	483	0.716
9	50	26	20	36	25	481	0.709
10	19	18	30	15	28	464	0.715
11	32	31	27	18	19	518	0.700
12	55	76	61	44	34	530	0.696
13	68	66	38	44	35	524	0.685
14	30	51	63	34	34	538	0.686
15	20	26	14	16	27	447	0.716
16	24	36	27	17	23	451	0.724
17	27	16	22	18	38	486	0.709
18	30	32	30	39	20	573	0.675
19	21	18	28	21	23	441	0.712
20	34	32	31	43	42	443	0.722
21	23	23	30	21	15	435	0.733
22	35	52	55	26	46	512	0.696
23	20	14	22	16	30	448	0.716
24	43	24	29	27	29	486	0.703
25	>200	22	21	17	16	452	0.715
26	23	23	32	45	44	440	0.725
27	29	16	32	43	18	480	0.715
28	18	13	16	15	21	416	0.723
29	27	20	16	15	22	476	0.706
30	16	18	28	17	24	440	0.726
$\bar{x}$	31.6	29.5	32.2	27.1	27.5	473.3	0.710
$S_{\bar{x}}$	2.6	3.1	2.6	2.1	1.7	7.1	0.003
p_I	>0.5	>0.5	>0.5	>0.5	>0.5	-	-
p_C	>0.5	>0.5	>0.5	>0.5	>0.5	-	-

Table A.13: Results of the C method using MSA on $\tilde{t}_{\Omega,c}$ and $\tau = 0.001$.

OD Matrix	C method (MSA on $\tilde{t}_{\Omega,c}$)					$D_\bullet$	μ
	$d=1$	$d=2$	$d=3$	$d=4$	$d=5$		
1	9	10	13	10	10	408	0.735
2	17	9	14	11	10	430	0.725
3	16	23	18	13	26	498	0.701
4	23	19	13	15	24	450	0.714
5	12	15	27	19	20	488	0.705
6	13	24	18	20	24	466	0.709
7	16	23	17	21	13	495	0.701
8	11	14	15	14	13	483	0.716
9	15	16	21	15	13	481	0.709
10	11	17	12	18	15	464	0.715
11	11	13	12	13	12	518	0.700
12	31	31	20	19	27	530	0.696
13	23	46	19	28	18	524	0.685
14	15	54	17	21	28	538	0.686
15	16	21	13	11	11	447	0.716
16	26	18	14	16	12	451	0.724
17	15	11	13	11	12	486	0.709
18	16	19	20	14	14	573	0.675
19	12	34	14	11	12	441	0.712
20	18	13	14	12	15	443	0.722
21	9	9	9	9	9	435	0.733
22	25	70	31	31	20	512	0.696
23	12	10	67	19	13	448	0.716
24	12	14	14	19	13	486	0.703
25	14	20	11	16	10	452	0.715
26	14	15	14	13	28	440	0.725
27	37	27	18	16	13	480	0.715
28	9	27	9	10	19	416	0.723
29	16	14	39	21	19	476	0.706
30	9	20	14	11	9	440	0.726
$\bar{x}$	16.1	21.9	18.3	15.9	16.1	473.3	0.710
$S_{\bar{x}}$	1.2	2.5	2.0	1.0	1.1	7.1	0.003
p_I	0.2003	>0.5	>0.5	0.077	0.1626	-	-
p_C	0.3902	>0.5	>0.5	0.197	0.2135	-	-

Table A.14: Results of the C method using MSA on $\tilde{t}_{\Omega,c}$ and $\tau = 0.01$.

Appendix B

MatLab Implementations

This chapter shows the MatLab codes used in the performance tests of the standard, I and C method.

There might be redundancy in the codes, and possibly there are parts of the codes that could have been made more efficient. The primary goal of the codes have not been to make a very fast code, since it was never the intention of the code to use it on real-life transport networks. As such, it has been made sufficiently efficient to handle tests on a small test network (see section 4), but has also remained a manageable structure in cases where a more efficient coding could have been used.

Most of the variable names are directly taken from the theory presented in the study. Whenever this is not the case, a brief introduction to the variable has been stated before the code.

B.1 The Internal Loop

Since all other methods uses the internal loop, it is natural to start with the code of the internal loop. When other methods have called this code, the name internalLoop has been used. The code can be seen below:

MatLab code B.1.1 (The Internal Loop)

```
%Initialising various constants
k_i=2;
tau_i=tau/100;
t_lc=t_0l;
t_rc=t_lc*link';
D_rc=zeros(1,n_r);
D_lc=zeros(1,n_l);
converged_i=false;

%Setting the travel times of each route equal to
%the final travel times of the last external iteration.
t_rc(2,:)=t_rc_init;

while(k_i<k_imax);

% Calculating the route demands.
    for ODp=1:n_Omega
        D_rc(k_i,StartEnd(ODp,1):StartEnd(ODp,2)) =...
            D_Omegac(k,ODp).*exp(gamma*...
            t_rc(k_i,StartEnd(ODp,1):StartEnd(ODp,2)))./...
            sum(exp(gamma*t_rc(k_i,StartEnd(ODp,1):StartEnd(ODp,2))));
    end

    %Checking for convergence.
    if(norm(D_rc(k_i,:)-D_rc(k_i-1,:),inf)< tau_i)
        converged_i=true;
    end

    %Calculating link demands.
    D_lc(k_i,:)=D_rc(k_i,:)*link;

    %Calculating the standard MSA coefficient.
    a_k=1/(k_i-1);

    %Calculating the link travel times using the BPR formula and MSA.
    t_lc(k_i,:)=a_k*t_0l.*(1+alpha_l*(D_lc(k_i,:)./C_l).^beta_l) + ...
        (1-a_k)*t_lc(k_i-1,:);

    %Incrementing k_1.
    k_i=k_i+1;

    %Calculating the route travel times.
    t_rc(k_i,:)=t_lc(k_i-1,:)*link';
end

% Saving the final route cost for use in next external iteration.
t_rc_init=t_rc(k_i,:);
```

□

B.2 The Standard Method

In this section, the MatLab implementation of the standard method will be presented. The code consists of some network specific variables, that has been initialized prior to the code. A brief explanation of these variables follows below.

- `beta_l` is a vector containing the $\beta-$coefficient for the BPR formula (see section 2.1.1) for all the links of the network.
- `t_0l` is a vector containing the free flow travel time of all the links of the network.
- `link` is a binary matrix of size $n_r \times n_l$. A 1 in the (r, l)'th entry denotes that l is a part of route r. A 0 denotes that it is not.
- `StartEnd` is a $n_\Omega \times 2$ matrix. All the routes have a unique ID in such a way, that all the routes corresponding to the OD pair Ω have numbers in the interval between the $(\Omega, 1)$'th and the $(\Omega, 2)$'th entry, with the latter being the highest route ID for any route corresponding to Ω.

Having introduced these variables, the code is now ready to be presented.

MatLab code B.2.1 (The Standard Method)

```
%Initialisation of the constants.
tau=0.01;
k_max=200;
k_imax=1000;
Delta_dot=450;
mu_0=0.85;
kappa_c=-log(1/mu_0-1);
gamma=-1;
alpha_l=0.5;
beta_l=[6,6,4,6,4,6,4,4,2,2,2,2,2,2,2,2];
n_r=2007;
n_l=16;
n_Omega=36;

%% Determining the public transport OD-pair costs.
t_rp=t_0l*link';
D_rp=zeros(1,n_Omega);
t_Omegap=D_rp;

% For each OD pair, the probability of choosing each route is
% calculated using a logit model. Secondly, the average travel
% time of each route is calculated.
for ODp=1:n_Omega
    denominator = sum(exp(gamma*t_rp(StartEnd(ODp,1)...
        :StartEnd(ODp,2))));
    D_rp(StartEnd(ODp,1):StartEnd(ODp,2))=...
        exp(gamma*t_rp(StartEnd(ODp,1):StartEnd(ODp,2)))./denominator;
    t_Omegap(ODp)=D_rp(StartEnd(ODp,1):StartEnd(ODp,2))...
        *t_rp(StartEnd(ODp,1):StartEnd(ODp,2))';
end

% Equal in 1st iteration, because the flow is 0.
t_Omegac_init=t_Omegap;


%% Testing various OD-matrices
for od=1:30
    rng(436+od);
    %% Loading OD-matrix and transforming it to a vector.
    Dbar = Delta_dot/36;
    D_Omega=randi(round(2*Dbar),1,36);


    %% Testing for various values of d
    for d=1:5

        % The initial route travel times for cars are initiated.
        t_rc_init=t_rp;

        % Average travel times for each OD pair are initiated.
        t_Omegac = t_Omegac_init;

        % The demand for car travel on each OD pair is calculated,
        % using a logit model.
        for ODp=1:n_Omega
            D_Omegac(ODp)=1/(1+exp(-kappa_c))*D_Omega(ODp);
        end


        %% External loop
        k=1;
        converged=false;

        while(~converged && k <k_max)
            % Internal Loop
            internalLoop;

             %Checking for non-convergence of internal loop
            if(k_i>=k_imax)
                break;
            end

            % Updating demand and cost.
            k=k+1;

           %Calculating the MSA coefficient. For MRA is wanted, set a_k=1;
            a_k=k^d/sum((1:k).^d);
            for ODp=1:n_Omega
                t_Omegac(k,ODp)=a_k*(t_rc(k_i,StartEnd(ODp,1):...
                    StartEnd(ODp,2))*D_rc(k_i,StartEnd(ODp,1):StartEnd(...
                    ODp,2))'/D_Omegac(k-1,ODp))...
                    +(1-a_k)*t_Omegac(k-1,ODp);
                D_Omegac(k,ODp)=1/(1+exp(-(gamma*(t_Omegac(k,ODp)-t_Omegap(ODp))...
                    +kappa_c)))*D_Omega(ODp);
            end


            %Checking for convergence
            if(norm(D_Omegac(k,:)-D_Omegac(k-1,:),inf)<tau)
                converged=true;
            end

        end

        % The market share of car travel is calculated.
        mu=sum(D_Omegac(k,:))/sum(D_Omega);

    end
end
```

□

B.3 The I Method

The I method contains no further variables that are not already presented in the study, why the code will just be presented straight away.

MatLab code B.3.1 (The I Method)

```
%Initialisation of the constants.
tau=0.01;
k_max=200;
k_imax=1000;
Delta_dot=450;
mu_0=0.85;
kappa_c=-log(1/mu_0-1);
gamma=-1;
alpha_l=0.5;
beta_l=[6,6,4,6,4,6,4,4,2,2,2,2,2,2,2,2];
n_r=2007;
n_l=16;
n_Omega=36;

%% Determining the public transport OD-pair costs.
t_rp=t_0l*link';
D_rp=zeros(1,n_Omega);
t_Omegap=D_rp;

% For each OD pair, the probability of choosing each route is
% calculated using a logit model. Secondly, the average travel
% time of each route is calculated.
for ODp=1:n_Omega
    denominator = sum(exp(gamma*t_rp(StartEnd(ODp,1)...
        :StartEnd(ODp,2))));
    D_rp(StartEnd(ODp,1):StartEnd(ODp,2))=...
        exp(gamma*t_rp(StartEnd(ODp,1):StartEnd(ODp,2)))./denominator;
    t_Omegap(ODp)=D_rp(StartEnd(ODp,1):StartEnd(ODp,2))...
        *t_rp(StartEnd(ODp,1):StartEnd(ODp,2))';
end

% Equal in 1st iteration, because the flow is 0.
t_Omegac_init=t_Omegap;


%% Testing various OD-matrices
for od=1:30
    rng(436+od);

    %% Loading OD-matrix and transforming it to a vector.
    Dbar = Delta_dot/36;
    D_Omega=randi(round(2*Dbar),1,36);


    %% Testing for various values of d
    for d=1:5

        % The initial route travel times for cars are initiated.
        t_rc_init=t_rp;

        % Average travel times for each OD pair are initiated.
        t_Omegac = t_Omegac_init;

        % The demand for car travel on each OD pair is calculated,
        % using a logit model.
        for ODp=1:n_Omega
            D_Omegac(ODp)=1/(1+exp(-kappa_c))*D_Omega(ODp);
        end


        %% First internal loop
        %k needs to be updated for use in the internal loop.
        k=1;
        internalLoop;

        %Calculating the MSA coefficient. For MRA, set a_k=1;
         a_k=2^d/sum((1:2).^d);
         for ODp=1:n_Omega
             t_Omegac(2,ODp)=a_k*(t_rc(k_i,StartEnd(ODp,1):...
                 StartEnd(ODp,2))*D_rc(k_i,StartEnd(ODp,1):StartEnd(...
                 ODp,2))'/D_Omegac(k-1,ODp))...
                 +(1-a_k)*t_Omegac(k-1,ODp);
             D_Omegac(2,ODp)=1/(1+exp(-(gamma*(t_Omegac(2,ODp)-t_Omegap(ODp))...
                 +kappa_c)))*D_Omega(ODp);
         end




%% Second internal loop
%k needs to be updated for use in the internal loop.
        k=2;
        internalLoop;

        %Calculating the MSA coefficient. For MRA, set a_k=1;
         a_k=3^d/sum((1:3).^d);
         for ODp=1:n_Omega
             t_Omegac(3,ODp)=a_k*(t_rc(k_i,StartEnd(ODp,1):...
                 StartEnd(ODp,2))*D_rc(k_i,StartEnd(ODp,1):StartEnd(...
                 ODp,2))'/D_Omegac(k-1,ODp))...
                 +(1-a_k)*t_Omegac(k-1,ODp);
             D_Omegac(3,ODp)=1/(1+exp(-(gamma*(t_Omegac(3,ODp)-t_Omegap(ODp))...
                 +kappa_c)))*D_Omega(ODp);
         end

        %% External loop

        %Initialising Dtilde_Omegac and ttilde_Omegac
        ttilde_Omegac=t_Omegac;
        Dtilde_Omegac=zeros(3,36);
        Dtilde_Omegac(2:3,:)=D_Omegac(1:2,:);
        converged=false;
        k=3;

        while(~converged && k <k_max)
            % Calculating dtilde and ttilde;
            b_D = (ttilde_Omegac(k,:)-ttilde_Omegac(k-1,:))./...
                (D_Omegac(k,:)-D_Omegac(k-1,:));
            a_D = ttilde_Omegac(k,:)-b_D.*D_Omegac(k,:);
            b_t = (t_Omegac(k,:)-t_Omegac(k-1,:))./...
                (Dtilde_Omegac(k,:)-Dtilde_Omegac(k-1,:));
            a_t = t_Omegac(k,:)-b_t.*Dtilde_Omegac(k,:);
            Dtilde_Omegac(k+1,:) =(a_D-a_t)./(b_t-b_D)
            ttilde_Omegac(k+1,:) = (b_t.*Dtilde_Omegac(k+1,:)+a_t)+...

            %The internal loop is run
            internalLoop_I;

            %Checking for non-convergence of internal loop
            if(k_i>=k_imax)
                break;
            end

           %Calculating MSA coefficient. For MRA, set a_k=1.
            a_k=(k+1)^d/sum((1:(k+1)).^d)

           %Updating cost and demand.
            for Omega=1:n_Omega
                t_Omegac(k+1,Omega)=a_k*(t_rc(k_i,StartEnd(Omega,1):...
                    StartEnd(Omega,2))*D_rc(k_i,StartEnd(Omega,1):StartEnd(...
                    Omega,2))'/Dtilde_Omegac(k+1,Omega))+(1-a_k)*t_Omegac(k,Omega);

                D_Omegac(k+1,Omega)=1/(1+exp(-(gamma*(ttilde_Omegac(k+1,Omega)...
                     -t_Omegap(Omega))+kappa_c)))*D_Omega(Omega);
            end

            %Incrementing k.
            k=k+1;

            % Checking for convergence.
            if(norm(D_Omegac(k,:)-D_Omegac(k-1,:),inf)<tau)
                converged=true;
            end

        end

%Calculating the market share of cars.
        mu=sum(D_Omegac(k,:))/sum(D_Omega);
end
```

□

The procedure `internalLoop_I` is identical to `internalLoop`, except that `internalLoop_I` uses $\tilde{D}_{\Omega,c}$ instead of $D_{\Omega,c}$.

B.4 The C method

In order to avoid a lot of redundancy, the upper part of the code for the C method (denoted by 6 vertical dots) has been cropped. That part of the code is identical the the code of the I method. The rest of the code is as follows.

MatLab code B.4.1 (The C Method)

```
  :
  :
  :
  %% External loop
   %Initializing the external loop
        k=3;
        converged=false;
        ttilde_Omegac=t_Omegac;
        Dbreve_Omegac=zeros(3,36);
        Dbreve_Omegac(2:3,:)=D_Omegac(1:2,:);
        while(~converged && k <k_max)
```

```
    % Calculating Dbreve and ttilde;
    b_D = (ttilde_Omegac(k,:)-ttilde_Omegac(k-1,:))./...
        (D_Omegac(k,:)-D_Omegac(k-1,:));
    a_D = ttilde_Omegac(k,:)-b_D.*D_Omegac(k,:);
    b_t = (t_Omegac(k,:)-t_Omegac(k-1,:))./...
        (Dbreve_Omegac(k,:)-Dbreve_Omegac(k-1,:));
    a_t = t_Omegac(k,:)-b_t.*Dbreve_Omegac(k,:);
    ttilde_Omegac(k+1,:) =b_t.*(a_D-a_t)...
        ./(b_t-b_D)+a_t);

    %Incrementing k
    k=k+1;

    %Calculating the demands.
    for Omega=1:n_Omega
        D_Omegac(k,Omega)=1/(1+exp(-(gamma*(ttilde_Omegac(k,Omega)...
            -t_Omegap(Omega))+kappa_c)))*D_Omega(Omega);
    end

    %Running the internal loop.
    internalLoop;

    %Checking if internal loop converged.
    if(k_i>=k_imax)
        break;
    end

    %Calculating the cost function.
    for Omega=1:n_Omega
        t_Omegac(k,Omega)=t_rc(k_i,StartEnd(Omega,1):...
            StartEnd(Omega,2))*D_rc(k_i,StartEnd(Omega,1):StartEnd(...
            Omega,2))'/D_Omegac(k,Omega);
    end

    %Setting Dbreve_k = D_k.
    Dbreve_Omegac(k,:)=D_Omegac(k,:);

    % Checking for convergence.
    if(norm(D_Omegac(k,:)-D_Omegac(k-1,:),inf)<tau)
        converged=true;
    end
end
```

□

Bibliography

[BC01] J Bottom and I Chabini. Accelerated averaging methods for fixed point problems in transportation analysis and planning. *Preprints of Tristan IV, Azores, Portugal*, pages 69–79, 2001.

[Can97] Giulio Erberto Cantarella. A general fixed-point approach to multi-mode multi-user equilibrium assignment with elastic demand. *Transportation Science*, 31(2):107–128, May 1997.

[CP01] Ennio Cascetta and Maria Nadia Postorino. Fixed point approaches to the estimation of o/d matrices using traffic counts on congested networks. *Transportation Science*, 35(2):134–147, May 2001.

[Ger09] Haluk Gerçek. Is there a road ahead? In *Istanbul – City of Intersections*, Urban Age, page 47. Alfred Herrhausen Society, London School of Economics and Political Science, November 2009.

[JFM11] Richard Johnson, John Freund, and Irwin Miller. *Miller and Freund's Probability and Statistics for Engineers – International Edition.* Pearson Education, Inc., 8th edition, 2011.

[LHH09] Henry X. Liu, Xiaozheng He, and Bingsheng He. Method of successive weighted averages (mswa) and self-regulated averaging schemes for solving stochastic user equilibrium problem. *Networks and Spatial Economics*, 9(4):485–503, December 2009.

[LK00] Averill M. Law and W. David Kelton. *Simulation Modeling and Analysis.* McGraw-Hill, 3rd edition, 2000.

[Pol90] B.T. Polyak. New method of stochastic approximation type. *Automation and Remote Control*, 51(7):937–946, Jul 1990.

[Ric10] Jeppe Rich. *Introduction to Transport Models: Theory, Application and Implementation.* Department of Transport, Technical University of Denmark, 2800 Lyngby, Denmark, 5.06th edition, August 2010.

[RNC13] Jeppe Rich, Otto Anker Nielsen, and Giulio Erberto Cantarella. System convergence in transport models. Working Paper, 2013.

[Tra09] Kenneth E. Train. *Discrete Choice Models with Simulation.* Cambridge University Press, New York, 2nd edition, 2009.

[Voo56] Alan M. Voorhees. *A General Theory of Traffic Movement.* Institute of Traffic Engineers, New Haven Connecticut, 1956.

[Wei] Eric W. Weisstein. Power sum. From MathWorld – A Wolfram Web Ressource. http://mathworld.wolfram.com/PowerSum.html.