July 14, 2018

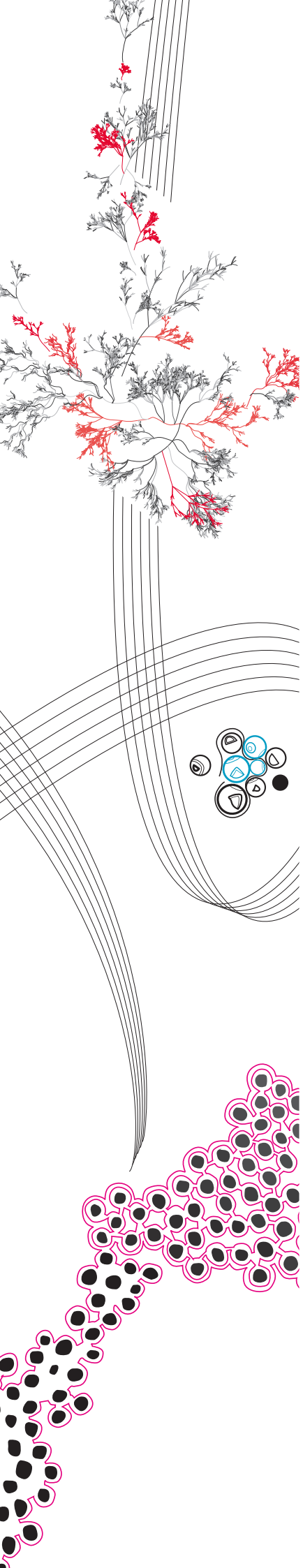
MASTER THESIS

MATRIX ESTIMATION WITH STAQ

Bernike Rijksen

Applied Mathematics
Discrete Mathematics and Mathematical Programming

Supervisors:
Georg Still, Marc Uetz (University of Twente)
Luuk Brederode, Luc Wismans (Dat.Mobility)



dat. mobility

UNIVERSITY OF TWENTE.

Preface

This master thesis finishes my master Applied Mathematics at the University of Twente. In September I started my internship at DAT.Mobility, which is part of the Goudappel Groep. Goudappel Groep gives advises on all kinds of mobility problems and within DAT.Mobility all expertise on the area of big data, model methodology, geographic information systems and software development is brought together. My assignment was to study the matrix estimation problem for the assignment model STAQ. And although this problem is only a small subproblem within a big system of models which work together to forecast traffic flows on a network, I like the idea that this research attributes to practice.

An important part of my internship was to develop an understanding for the given problem and the related models. Herein I have learned a lot from my supervisors at DAT.Mobility, Luuk Brederode and Luc Wismans. They provided me with the right literature and models and were always willing to answer my questions. By reading papers and testing different hypotheses on small test networks in Excel and Matlab, I obtained knowledge and feeling for the given problem, which resulted in many ideas and conclusions. But there was a lot more to investigate. Therefore I decided to continue with this subject as my final master project. I am happy with this decision because during the past months I have been able to analyse and answer many of the open questions there were at the end of my internship. The weekly discussions with Georg Still have attributed a lot to these results. He was always helpful and critical on my work. Although the matrix estimation method for STAQ is still in development, I am proud of what I reached. Luuk Brederode will continue working on the developed matrix estimation method and I hope he will keep me informed.

Past year I have learned a lot and therefore I would like to thank in the first place Georg Still, Luuk Brederode and Luc Wismans. They have shown me how to look at the matrix estimation problem for STAQ from both a mathematical and a traffic engineering point of view. Where I started reading the paper about matrix estimation for STAQ having questions at almost every sentence, I am now able to give feedback on the new version of this paper. This has been a valuable experience. Also I would like to thank Marc Uetz and Jan-Kees van Ommeren for reading my final work.

Abstract

This research focuses on the matrix estimation problem for the assignment model STAQ. In the matrix estimation process, additional road information is used to refine the OD-matrices which are estimated in the first three steps of the four step model. The chosen assignment model, in this case STAQ, is needed in this matrix estimation process. As such, the matrix estimation problem is typically expressed as a bi-level optimization problem. Because bi-level problems are known to be NP-hard, in practice heuristic methods are used to search for solutions. Conventional matrix estimation methods as developed for traditional static traffic assignment models are not directly suitable for STAQ. The approximations of the link flows as considered in these heuristic methods cannot cope with capacity constraints. Therefore Brederode et al. [5] propose a matrix estimation method for STAQ, in which the link flows are approximated using a first order Taylor approximation. In this study it has been shown that the current approach to determine the sensitivities of the assignment matrix to changes in the OD-demands, which are required in the first order Taylor approximation of the link flows, can lead to significant errors. It should be investigated on large scale networks how big those errors are. It might be needed to adapt the way of approximating the link flows or to choose a different heuristic approach. Furthermore from the characteristics of the simplified upper level optimization problem as considered within the proposed matrix estimation method it has been concluded that this optimization problem can be solved best using the `quadprog` solver in *Matlab*. However also the currently implemented `fmincon` interior point algorithm is suitable.

Notation List

N		set of nodes
R	$\subseteq N$	set of origins
S	$\subseteq N$	set of destinations
RS	$\lvert R \times S \rvert$	set of OD-pairs
P		set of paths
$\tilde{P}$	$\subseteq P$	set of paths with travel time measurements
A		set of directed links
$\tilde{A}$	$\subseteq A$	set of directed links with count measurements
I	$\subseteq A$	set of inlinks
J	$\subseteq A$	set of outlinks
IJ		set of turns
D		vector of OD-demands
Q	$\lvert P \rvert$	vector of path demands
T	$\lvert IJ \rvert$	vector of turn demands
Y	$\lvert A \rvert$	vector of link demands
y	$\lvert A \rvert$	vector of link flows
$\tilde{y}$	$\lvert \tilde{A} \rvert$	vector of observed link flows
t	$\lvert IJ \rvert$	vector of turn flows
τ	$\lvert P \rvert$	vector of path queueing delays
$\tilde{\tau}$	$\lvert \tilde{P} \rvert$	vector of observed path queueing delays
α	$\lvert IJ \rvert$	vector of turn based reduction factors
$\boldsymbol{D}$	$\lvert R \times S \rvert$	OD-matrix
$\boldsymbol{A}$	$\lvert \tilde{A} \rvert \times \lvert R \times S \rvert$	assignment matrix
$\boldsymbol{B}$	$\lvert \tilde{A} \rvert \times \lvert P \rvert$	crossing fraction matrix
$\boldsymbol{P}$	$\lvert P \rvert \times \lvert R \times S \rvert$	route fraction matrix
D_{rs}	$\in D$	demand from origin r to destination s
Q_p	$\in Q$	demand on path p
T_{ij}	$\in T$	turn demand from inlink i to outlink j
Y_a	$\in Y$	demand on link a
y_a	$\in y$	flow on link a
$\tilde{y}_a$	$\in \tilde{y}$	observed flow on link a
t_{ij}	$\in t$	flow from inlink i to outlink j
τ_p	$\in \tau$	average path queueing delay
$\tilde{\tau}_p$	$\in \tilde{\tau}$	observed average path queueing delay
α_{ij}	$\in \alpha$	reduction factor on turn ij
A_a^{rs}	$\in \boldsymbol{A}$	fraction of demand from OD-pair rs that flows over link a
$\hat{\alpha}_a^p$	$\in \boldsymbol{B}$	reduction factor on path p till link a
ψ_p^{rs}	$\in \boldsymbol{P}$	fraction of OD-demand rs on path p

Contents

Chapter 1

Background information

Mobility plays an important role in modern life. To improve and schedule traffic and transportation, models are used to forecast traffic flows on networks. The goal of this master thesis is to mathematically describe and solve a subproblem within a certain traffic model. More specific: I studied the matrix estimation problem for the four step traffic model with assignment model STAQ. This chapter provides the required background information about the four step model, the matrix estimation problem and the assignment model STAQ. In the next chapter the exact research question of this master thesis is introduced.

1.1 Four step model

The four step model is the most commonly used traffic and transportation model to forecast traffic flows on networks. In this model the network of the studied area is represented in the form of a directed graph (N, A), where N are the nodes which represent the junctions in the network and A are the links which represent the roads in the network. The studied area is divided in several zones for which characteristic socio-economic data is available. All these zones are connected to the network via centroids, which are one or more nodes in the centre of the zone. Trips in the network always start and end at these centroids. Commencing with this directed graph and the available socio-economic data, the four step model estimates trips per transport mode and the use of the network in terms of flows per link. The model consists of the following steps [9] [13]:

1. **Trip generation** In the first step, based on the available socio-economic data, the number of people arriving in and departing from each zone are determined. After this step it is clear how many trips originate and terminate at each centroid, but the origin and destination of specific trips are not yet connected.

2. **Trip distribution** In the second step the number of trips between each origin and destination pair (OD-pair) is determined. This number is called the trip demand of the OD-pair. The gravity model[1] is a frequently used method to obtain these trip demands. The trip demands of all OD-pairs

[1] For more information about the gravity model see [20].

are represented in a matrix, this matrix is called the origin-destination matrix (OD-matrix). So the OD-matrix describes the number of trips from every centroid in the studied network to every other centroid in the network.

3. **Modal split** In the third step for each OD-pair it is decided how the trips are divided over different transport modes. Examples of transport modes are car, public transport and bicycle. After this step the number of trips between all OD-pairs are known per mode (there is determined an OD-matrix per mode), but it is not known which route on the network each trip will take. In practice step 2 and step 3 are often combined.

4. **Assignment** Finally in the fourth step the trip demands of all OD-pairs are distributed along the network. So for each trip it is determined which route in the network it will take. There is a wide variety of models developed to deal with this assignment problem, these models are called assignment models. When all traffic demand is assigned to the network, values for the flows and speeds on the links can be deduced.

So following these four steps, the four step model finds a forecast of the traffic flows on a network. In practice a feedback loop is performed within the four step model. This is needed because the gravity model used in step two requires the travel time between each OD-pair as an important aspect of the generalized costs[2] between OD-pairs, but there are no travel times other then free-flow travel times known before the fourth step.

1.2 Assignment models

As described, there is a wide variety of assignment models available. Within this study the assignment model STAQ is used (studied), which stands for Static Traffic Assignment with Queueing. The assignment model STAQ is explained in section 1.4. In this section first a general introduction to assignment models is given, such that the matrix estimation problem and the assignment model STAQ can be explained and placed within this context.

Traffic assignment models describe the interaction between road travel demand and road infrastructure supply. Given the travel demand on a network, assignment models describe how this travel demand is distributed along the network. All assignment models consist (implicitly or explicitly) of a route choice submodel and a network loading submodel. The route choice submodel determines path flows, based on travel demands and travel times. The network loading submodel propagates path flows through the network and yields travel times. This is shown in figure 1.1. Bliemer et al. have introduced a theoretical framework for the classification of traffic assignment models. Their classification is shortly discussed in this section, because it gives insight in the different assignment models available. In this theoretical framework assignment models are classified on their spatial, temporal and behavioural assumptions. In the remainder of this section the different model types within these three model classes are briefly

[2]For simplicity in the rest of this report the generalized costs are considered to consist of travel times only.

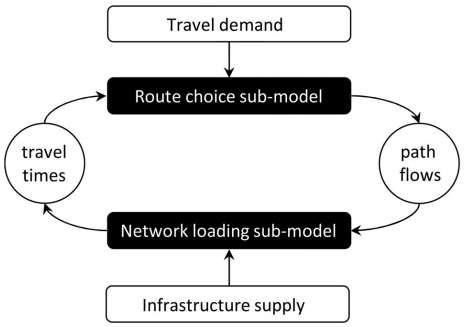


Figure 1.1: Interaction between travel demand and infrastructure supply [2].

explained. For more details the reader is referred to their article *"Genetics of traffic assignment for strategic transport planning"* [2].

1.2.1 Route choice submodel

Within the theoretical framework the behavioural assumptions describe the route choice submodel. The route choice submodel determines path flows based on given travel demands and considered travel times. To describe the route choice submodel, the behavioural assumptions define how the considered travel times are determined, and how the travellers are assumed to choose their routes. As a result of the behavioural assumptions, the theoretical framework distinguishes between three model types:

1. **Equilibrium models** In an equilibrium model an equilibrium flow is sought for in which no traveller can unilaterally change routes to improve his or her travel time. Congested travel times are considered by iterating between the route choice submodel and the network loading model. The routes can be chosen in a deterministic or stochastic way.

2. **One-shot models** In an one-shot model a single network loading is performed to determine the travel times on the routes. The routes are usually chosen using a logit-model (stochastic route choice).

3. **All-or-nothing models** In an all-or-nothing model typically free-flow travel times are used. The routes are chosen such that all travellers follow the fastest route (deterministic route choice). An all-or-nothing model is a special case of a one-shot model.

The three model types are given in a decreasing order of capability. Capability can be explained in terms of realism and complexity. The more capable a model type is, the more realistic this model is, but this makes the model also more complex. It can be seen that the equilibrium models are the most capable models. The assignment model used in this study (STAQ) can be applied as an equilibrium model. Therefore equilibrium models are explained in more detail in the remainder of this subsection. However STAQ can also be used as an one-shot or an all-or-nothing model [4].

In an equilibrium model congested travel times are considered. The difficulty of considering congested travel times is that these travel times are not fixed. The level of congestion and hence the travel time on a route is depending on the route choice, and the route choice is on its turn depending on the amount of congestion on each possible route. Because of this interaction between route choice and travel times, equilibrium models iterate between the route choice submodel and the network loading submodel to account for congested travel times. The route choice submodel calculates path flows which are used by the network loading model to update the travel times. Then these updated travel times are used by the route choice submodel to calculate new path flows (etc.). Note that in one-shot models and all-or-nothing models the travel times are considered to be fixed. Hence these models do not iterate between the route choice submodel and the network loading submodel, and thereby these models do not consider congested travel times.

The route choice in an equilibrium model can be determined in a stochastic way, like in an one-shot model, or in an deterministic way, like in an all-or-nothing model. Hence one-shot models and all-or-nothing models can be seen as a single iteration of an equilibrium model. In an equilibrium model the route choice tends to comply with Wardrop's first principle:

> *"The journey times in all routes actually used are equal and less than those which would be experienced by a single vehicle on any unused route"* [19].

So an equilibrium is sought for, in which no traveller can unilaterally change routes to improve his or her travel time. Such an equilibrium is called a User Equilibrium (UE). A feedback loop is performed between the route choice submodel and the network loading model till this equilibrium state is reached. The travellers are assumed to be non-cooperative, so they exhibit selfish behaviour. This is in contrast to system optimal models, which minimize the total (or average) travel time in the system and assume travellers to cooperate.

1.2.2 Network loading submodel

Within the theoretical framework the spatial assumptions describe the network loading submodel. The network loading submodel loads the path flows on the network and yields travel times. The spatial assumptions define if and how the network loading submodel accounts for congestion (delays and queues). As a result of the spatial assumptions there are four model types distinguished. They are described below in a decreasing order of capability. The less capable model types can be derived from the more capable model types by making simplifying assumptions:

1. **Capacity- and storage-constrained models** In capacity- and storage-constrained models, both the capacity of flow and the storage of queues on road segments are constrained. These models can be applied to all possible traffic conditions, including very heavy traffic when queues can grow longer than the road length and spill back onto upstream road segments occurs.

2. **Capacity-constrained models** In capacity-constrained models, there are no constraints on the storage of queues on road segments and as such spillback does not occur. These models are suitable for light to heavy traffic conditions in which short queues can form.

3. **Capacity-restrained models** In capacity-restrained models, flows can also exceed the physical road capacity and, therefore, queues are not described explicitly. These models are only suitable for light to medium traffic conditions in which the flow does not exceed the capacity, but some slight delays may occur due to increasing density.

4. **Unrestrained models** In unrestrained models, there are fixed (usually free-flow) travel conditions and travel times. These models are only suitable for light traffic conditions in which flow increases linearly with density, indicating that vehicles drive at maximum speed.

Actually the more capable the model type, the better these models reflect the theoretical relationship which exists between flow and density. This theoretical relationship between the flow and density can be empirically observed from traffic counts and can be described in a fundamental diagram (see figure 1.2). Each point in this fundamental diagram represents a specific steady traffic state. When the density is below the critical density, there is no congestion and no queues appear. The corresponding branch of the fundamental diagram is called the free-flow branch. Densities higher then the critical density are a result of congestion and queues on the road. This branch in the fundamental diagram is called the congested branch. All model types explicitly or implicitly[3] assume a fundamental diagram. In figure 1.3 for each model type an example of a fundamental diagram is shown. The fundamental diagram of the least capable model type only reflects part A of the theoretical fundamental diagram in figure 1.2 correctly. The more capable the model type, the better it reflects the theoretical fundamental diagram. The most capable model type is suitable for all given traffic conditions and hence reflects all parts of the theoretical fundamental diagram correctly (A, B, C and D).

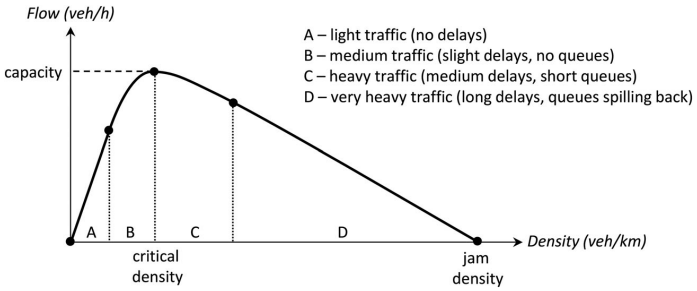


Figure 1.2: Theoretical relationship between flow and density [2].

[3]For the unrestrained and capacity-restrained model types, in practice only a travel time function is defined. This function describes the relation between flow and travel time.

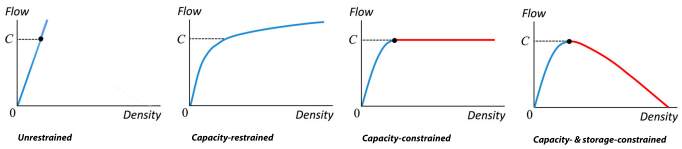


Figure 1.3: Fundamental diagrams [2].

While the fundamental diagram only shows flows (veh/h) and densities (veh/km), the speed of a vehicle (km/h) can be determined using the fundamental relationship that speed equals flow divided by density [2], or equivalently:

$$\text{flow} = \text{speed} \cdot \text{density} \tag{1.1}$$

So, given the flow and density, it is possible to determine the corresponding speed and hence the corresponding travel time[4] on a link. Note that for unrestrained and capacity-restrained model types, given a link flow, the corresponding density can be uniquely determined from the fundamental diagram. The travel times are separable; They are only depending on the flow on the link itself. Whereas for capacity- (and storage)-constrained model types, given a link flow, also the corresponding density is needed to determine the speed and hence the travel time on the link from the fundamental diagram[5]. The travel times are non-separable; The travel time on a link depends on the level of congestion on the link, which on its turn depends on the flows on all other links.

1.2.3 Temporal assumptions

Finally within the theoretical framework the temporal assumptions define if and how a time dimension is considered within the assignment model. As a result of the temporal assumptions the theoretical framework distinguishes between three model types. They are described below in decreasing order of capability:

1. **Dynamic models** Dynamic models consider a time-varying travel demand and generally (but not necessarily) multiple time periods for route choice. Within each time period there exist smaller time steps for network loading. In these models variations over time in path flows, link flows and travel times are explicitly taken into account.

2. **Semi-dynamic models** Semi-dynamic models often consider only a single step for network loading within each route choice period, but traffic flows can be propagated between route choice periods. These models can be seen as a sequence of static models, in which the result from a previous period is taken into account in the next period.

3. **Static models** Static models consider a stationary travel demand and only a single time period for the route choice and the network loading. It is assumed that traffic outside this time period does not influence flows or

[4]The travel time on a link (h) equals the length of the link (km) divided by the speed (km/h) on the link.

[5]See [11] for the relation between the flow-density, speed-density and speed-flow curves.

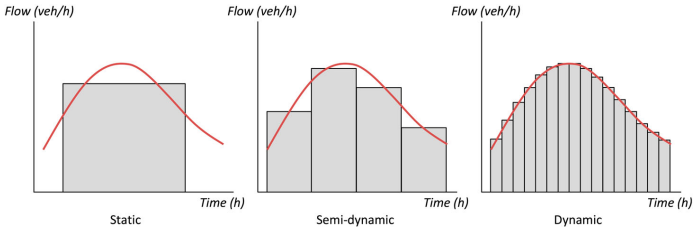


Figure 1.4: Relation time periods and travel demand [2].

travel times in the considered period. Within the network loading step all traffic reaches the destination. Static models result in an average image of the traffic in the considered time interval.

In figure 1.4 it is shown how static, semi-dynamic and dynamic models represent the travel demand. The more capable the model, the better it reflects the time-varying travel demand (the red line). Note that this figure only visualizes the differences between the model types in travel demand, other relevant differences have been described above.

1.2.4 Classification

In the previous subsections the different model types within the three model classes of the theoretical framework for the classification of traffic assignment models are described. The behavioural and temporal assumptions distinguish between three model types and the spatial assumptions distinguish between four model types. Combining the three different model classes, 36 different assignment model types can be described. This is shown in the framework in figure 1.5. Note that not all model types distinguished in the framework exist or can exist in practice. The goal of this framework is to make it possible to classify and compare existing traffic assignment models.

The least capable model type according to this framework is a static unrestrained all-or-nothing traffic assignment model. The most capable model type is a dynamic capacity and storage-constrained equilibrium traffic assignment model. Of course it seems best to build and always use the most capable model. But it turns out that this model is not ideal in practice. The dynamic capacity and storage-constrained equilibrium traffic assignment model is poorly scalable, data intensive and it has even been proven that there does not always exist an equilibrium for this model type [6]. Therefore, to support policy development and planning in strategic applications on large scale congested networks, the assignment model STAQ is developed [4]. STAQ circumvents problems with scalability and data-intensiveness, because this model does not consider a time dimension. The assignment model STAQ is a static capacity-constrained and[6] capacity- and storage-constrained assignment model which can be applied as an equilibrium model, and will be further introduced in section 1.4. First, in the next section the matrix estimation problem introduced.

[6]The assignment model STAQ consists of two phases, see section 1.4.

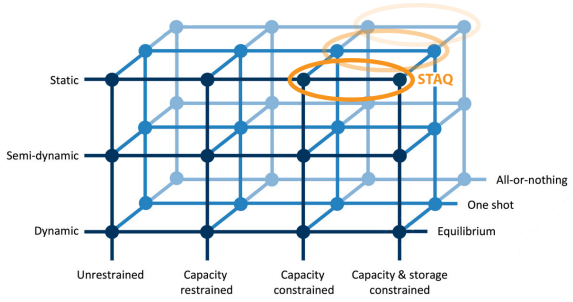


Figure 1.5: Framework classification traffic assignment models, edited from [2].

1.3 Matrix estimation problem

The matrix estimation problem for the assignment model STAQ is the central subject of this master thesis. This section provides a general introduction to the matrix estimation problem.

Matrix estimation is a process which aims to improve the results of the four step traffic model by adding exogenous information. In the four step traffic model, after step three, OD-matrices have been determined per mode. These OD-matrices are determined by the behavioural models in the first three steps of the four step model (see section 1.1). The behavioural models do not always guarantee good results, because the socio-economic data (survey-data) used, only describes the average mobility behaviour over an entire study area. To refine the results of the four step traffic model, additional traffic measurements can be used to improve the modelled OD-matrices. This process is called matrix estimation. In figure 1.6 the position of the matrix estimation process within the four step traffic model is visualized. The traffic measurements which are used for this process can be of many types and sources, most frequently used are traffic counts. Traffic counts register the traffic flow on a certain location in the network.

To improve a modelled OD-matrix using traffic counts, the modelled OD-matrix should be adjusted such that it better fits to measured traffic flows. The difficulty is that the adjusted OD-matrix cannot be directly compared with the measured traffic flows. OD-matrices describe the traffic demand for each OD-pair and the traffic flows caused by these OD-demands are only known after an assignment is performed. In other words, there is no one-to-one correspondence between OD-matrices and traffic counts. Therefore the assignment problem, in which the demand of a given OD-matrix is distributed along the network, is embedded within the matrix estimation problem[7] [13]. Because of this embedded assignment problem, the matrix estimation problem is typically expressed as a bi-level optimization problem. In the upper level the modelled OD-matrix is improved based on available traffic measurements, while in the lower level the

[7]So note that the chosen assignment model within the four step traffic model is not only used in the fourth step, but also within the matrix estimation process.

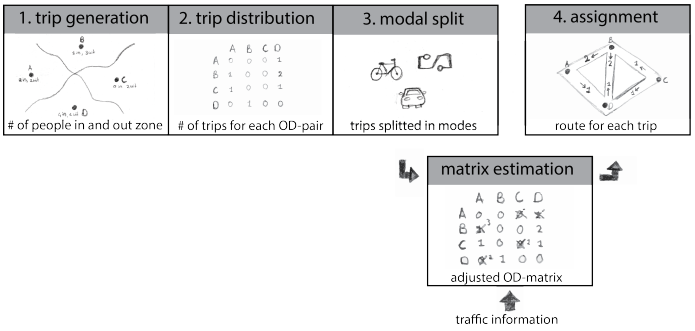


Figure 1.6: Matrix estimation within the four step model.

traffic assignment problem is solved. When traffic counts are the only traffic measurements used, the matrix estimation problem is formulated as follows:

$$\begin{aligned} \min_{D} F(D) &= \alpha f_1(D, D^0) + (1-\alpha) f_2(y, \tilde{y}) \\ \text{s.t. } y &= \text{Assign}(D) \\ D &\geq 0 \end{aligned} \tag{1.2}$$

D	vector/matrix of estimated OD-demands
D^0	vector/matrix of prior (modelled) OD-demands
y	vector of estimated link flows
$\tilde{y}$	vector of observed link flows
f_1	distance function
f_2	distance function
α	$\in [0, 1]$ weighting factor

Note that in the upper level of this bi-level optimization problem not only the distances between the observed and estimated link flows, but also the distances between the modelled (prior) and estimated OD-demands are minimized. This term is included, to not deviate too much from the OD-matrix as estimated in the first three steps. The distance functions f_1 and f_2 should be defined to make the problem concrete. These distance functions can be of many forms, for an overview the reader is referred to the master thesis of Smits [13]. Furthermore the weighting factor α gives the decision maker the opportunity to express his or her confidence in the prior matrix against the traffic counts. A smaller value of α puts more weight on the counts-part (f_2) of the objective function, while a larger value of α puts more weight on the prior-part (f_1) of the objective function. There does not exist a standard way of setting the α-value. In practice the weighting factor is calibrated through trial and error. Note that both objective function parts have a different scale. The prior-part concerns all OD-pairs while the counts-part concerns all observed links. Therefore a normalization should be applied to allow the weighting factor α to give a meaningful interpretation, expressing the relative importance of both terms on a scale of zero to one [5].

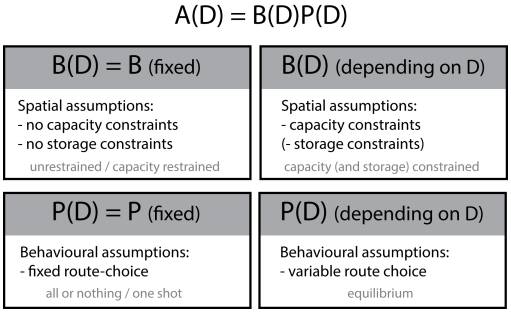


Figure 1.7: The assignment matrix.

In the lower level of the bi-level optimization problem, an assignment model Assign(D) is embedded, which can be any of the available assignment models. The assignment model distributes a given demand D along the network, which results in flows $y(D)$ on all links. According to Frederix et al. [8], for each possible assignment model the relation between the given OD-demands and the resulting link flows can be described by an assignment matrix $\boldsymbol{A}(D)$. This assignment matrix can be further subdivided into a crossing fraction matrix and a route fraction matrix:

$$\begin{aligned} y(D) &= \boldsymbol{A}(D)D, \\ &= \boldsymbol{B}(D)\boldsymbol{P}(D)D. \end{aligned} \tag{1.3}$$

D	vector of OD-demands, $\lvert R \times S \rvert$
$y(D)$	vector of estimated link flows, $\lvert \tilde{A} \rvert$
$\boldsymbol{A}(D)$	assignment matrix, $\lvert \tilde{A} \rvert \times \lvert R \times S \rvert$
$\boldsymbol{B}(D)$	crossing fraction matrix, $\lvert \tilde{A} \rvert \times \lvert P \rvert$
$\boldsymbol{P}(D)$	route fraction matrix, $\lvert P \rvert \times \lvert R \times S \rvert$
$\tilde{A}$	set of links with count measurements
R	set of origins
S	set of destinations
P	set of paths

The elements of the crossing fraction matrix $\boldsymbol{B}(D)$ express the proportion of a route flow that passes a link, thereby these elements describe the spatial propagation of the route flows through the network by the network loading submodel[8]. The elements of the route fraction matrix $\boldsymbol{P}(D)$ express the proportion of an OD-flow choosing a certain route, thereby these elements reflect the behavioural assumptions of route choice submodel. Finally the temporal assumptions of the assignment model are reflected in the dimensions of the assignment, crossing fraction and route fraction matrices. If more time periods are considered, these dimensions are correspondingly enlarged. It can be concluded that the assignment matrix $\boldsymbol{A}(D)$ is depending on the chosen assignment model.

[8]When there are no capacity and storage constraints considered, the crossing fraction matrix is fixed to the link-path incidence matrix.

In figure 1.7 it is shown that for some assignment models the crossing fraction matrix and the route fraction matrix are fixed matrices; They are not depending on D. It is interesting to note that if both the route fraction and the crossing fraction matrix are assumed to be fixed matrices, then the matrix estimation problem can be reformulated to a single level problem. In this case the assignment matrix is the same (and hence explicitly known) for all given D, so there exists a linear relationship between the link flows and the OD-flows and as such the vector of estimated link flows y can be formulated explicitly as a function of the OD-demand D. Hence the given bi-level matrix estimation problem can be reformulated as a single level problem. Such a reformulation to a single level problem is possible for all assignment models for which the assignment matrix $\boldsymbol{A}(D)$ is known explicitly. The assignment matrix of the assignment model STAQ cannot be expressed explicitly as a function of the OD demands. Therefore the matrix estimation problem for STAQ is (remains) a bi-level optimization problem.

1.4 STAQ

In this section the traffic assignment model STAQ is introduced. STAQ stands for Static Traffic Assignment with Queueing. As the name already indicates, STAQ can be seen as a static traffic assignment model. So the results of STAQ give an average image of the traffic in the considered time interval. In the next two subsections the route choice submodel and the network loading submodel of the assignment model STAQ are described, following the paper of Brederode et al [4]. Subsequently STAQ is placed within the theoretical framework for the classification of traffic assignment models as described in subsection 1.2 and its capability is discussed.

1.4.1 Route choice submodel

In figure 1.8 the modelling framework for the assignment model STAQ is shown. When used as an equilibrium model, STAQ iterates between its route choice submodel and its network loading submodel, such that congested travel times are considered. The main characteristics of STAQ are derived from its network loading model and as such its route choice submodel is interchangeable [4]. The route choice submodel within STAQ as considered in this study, consists of three components which interact to determine path flows from the given travel times:

- **Route set generator** The route set generator creates a set of routes based upon a given transport network.
- **Route choice model** The route choice model uses generalized costs, which are mainly based on the travel times calculated by the network loading model, to compute route fractions for all route alternatives between an OD-pair.
- **Route demand calculator** The route demand calculator computes path demands (in the figure called route demands), based on the given OD-demand and the calculated route fractions.

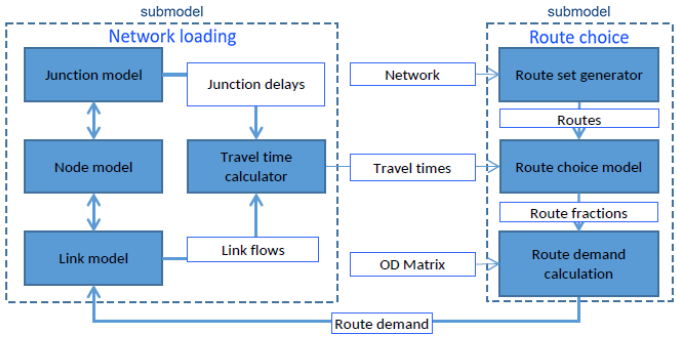


Figure 1.8: STAQ modelling framework, edited from [4].

1.4.2 Network loading submodel

The network loading submodel within STAQ consists of four different components which interact to propagate the given route demands through the network and calculate the resulting travel times:

- **Link model** The link model describes for each link the relation between flow and density in the form of a fundamental diagram. It determines from the path demands as given from the route model in interaction with the node model the corresponding link flows.
- **Node model** The node model seeks for a consistent solution in terms of flows transferred over an intersection. It accounts for flow restrictions due to merge and diverge interactions between flows. This model can transfer the effect of capacity restrictions on downstream to upstream links and the effect of demand changes on upstream to downstream links.
- **Junction model** The junction model accounts for the effect of limited supply due to conflict points on a junction itself and the way the junction is regulated. Furthermore it calculates travel time delays due to passing a junction.
- **Travel time calculator** The travel time calculator derives travel times from the output as calculated by the link model, node model and junction model.

STAQ can be applied to congested networks in which queues can grow longer than the road length and spillback to upstream road segments occurs. To deal with congestion the network loading model considers two phases: a squeezing phase and a queueing phase. The squeezing phase deals with the flow metering effect of congestion, assuming capacity but no storage constraints and the queueing phase deals with the spillback effect of congestion, assuming both capacity and storage constraints. STAQ-squeezing can also be used as a stand-alone network loading model, but STAQ-queueing needs the squeezing phase outcomes for initialisation[9].

[9]The queueing phase uses initial sending and receiving flows for every link as calculated by the squeezing phase.

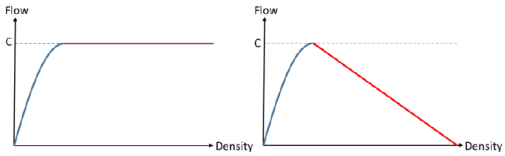


Figure 1.9: Fundamental diagrams STAQ -squeezing (left) and -queueing (right) [4].

Both phases of STAQ use the same node model and the same junction model, but different link models. The link models differ by the form of their fundamental diagram. This is shown in figure 1.9. The free-flow branch (blue) of both fundamental diagrams is identical. The fundamental diagrams differ in the congested branch (red). The congested branch of the fundamental diagram for STAQ-squeezing satisfies a maximum flow constraint. In this way the squeezing phase takes into account flow metering: the flow on a link can never become higher than the maximum flow of capacity of that link. However the maximum density is not constrained. As a consequence in the squeezing phase vertical queues are implied. All demand on a link that exceeds the maximum capacity forms a vertical point queue on the upstream node of this link[10]. Hence the squeezing phase detects the locations and the severity of active bottlenecks in the network. Then the spillback effect of congestion is considered in the next phase, the queueing phase. The fundamental diagram of STAQ-queueing constrains both the flow and the density. Therefore the queueing phase accounts for both the flow metering and the spillback effects. The fundamental diagrams in figure 1.9 are meant to show the difference between the squeezing and queueing phase, they are not explicitly implemented in the models.

In practice the network loading model first considers the squeezing phase and then the queueing phase. During the squeezing phase the STAQ algorithm iterates[11] between the network loading submodel (squeezing) and the route choice submodel till an User Equilibrium is approximated. After this phase the bottlenecks in the network are determined. Then, to determine the spillback and secondary effects of these bottlenecks, in the queueing phase the STAQ algorithm performs one iteration with the network loading submodel (queueing). Only one iteration is performed, because in real world applications on heavily congested networks it is not always possible to approximate an equilibrium to a level that is sufficient for strategic transport model applications [6]. Although no time dimension exists with respect to the in- and output of STAQ, the queueing phase uses internally a time dimension to allow for the spatial interaction between all different spillback and flow metering effects. Brederode et al. [4] describe that the main reason to split the algorithm in two phases is to maintain scalability when calculating spillback and secondary effects of bottlenecks. Additional reasons are that the flow metering and spillback effects can be analysed separately and that the squeezing phase compensates for the lack of a pre-study-period warm-up.

[10] Such an assignment model is called a residual queueing model.

[11] Note that this only holds when STAQ is used as an equilibrium assignment model, otherwise no iterations are preformed and no User Equilibrium is approximated.

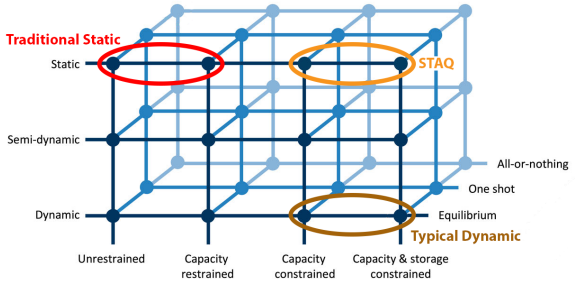


Figure 1.10: Placement of STAQ within the framework, edited from [2].

1.4.3 Capability of STAQ

It can be concluded that STAQ is a static traffic assignment model and from the fundamental diagrams in figure 1.9 it can be seen that STAQ is capacity-constrained in the squeezing phase and capacity- and storage-constrained in the queueing phase. Note that STAQ technically spoken is not static in the queueing phase, but its input and output are both static of nature. The theoretical framework in figure 1.10 shows that STAQ is a more capable assignment model than the traditional static traffic equilibrium assignment (STA) models, which assume separable monotonously increasing travel time functions and hence are unrestrained or capacity-restrained models. Traditional STA models cannot cope with capacity constraints, nor represent flow metering and queue formation, while STAQ takes into account these physical effects of congestion. Therefore, according to Brederode et al, STAQ is a viable alternative for traditional STA models to support policy development and planning in strategic applications on large scale congested networks [5]. STAQ combines the advantages of classical static and typical[12] dynamic assignment models; It is a computationally fast and scalable model which takes into account the physical effects of congestion. Note that in figure 1.10 the traditional STA model, STAQ and the typical DTA model are indicated as equilibrium assignment models. While comparing these three model types they are considered to be equilibrium models. However, they can also be used as a one-shot or AON assignment models. The algorithms as used within STAQ are explained in detail in chapter 3.

[12]Nowadays dynamic models are typically capacity- (and storage-) constrained model types. But note that unconstrained and capacity-restrained dynamic models also exist.

Chapter 2

Introduction

In the previous chapter the matrix estimation problem and the assignment model STAQ have been introduced. Recall that the assignment problem, and hence the chosen assignment model, is embedded within the matrix estimation problem. This research contributes to the development of a solution method for the matrix estimation problem with the assignment model STAQ. Within this chapter the goal of this research and the research question for this master thesis are introduced and motivated. Besides, the corresponding solution approach and the structure of this report are explained.

2.1 Research motivation

Matrix estimation methods for traditional static traffic assignment (STA) models are studied extensively and are readily available [5]. Generally these assignment models assume separable monotonously increasing travel time functions (see subsection 1.2.2), and as such, using these models it is relatively easy to determine how the prior OD-matrix should be adjusted to achieve similar modelled and observed link flows. Contrary to traditional STA models, STAQ takes into account capacity constraints and represents flow metering and queueing effects. This results in a more realistic, but also more complex assignment model. The implicitly considered travel time functions within STAQ are not separable any more; The travel times on the links are not only depending on the flow on the link itself, but also on the flows on all other links. This makes it more complex to determine how the prior OD-matrix should be adjusted to achieve similar modelled and observed link flows. Therefore, available matrix estimation methods, which are developed for traditional STA models, are not directly suitable for the assignment model STAQ. This research aims at developing a suitable matrix estimation method for STAQ. Within this master thesis a specific method for matrix estimation with STAQ is investigated. This proposed method is developed by Brederode et al. [5][1] (it is still under development) and is based on the conventional matrix estimation method for traditional STA models. The conventional matrix estimation method for traditional STA models and the proposed matrix estimation method for STAQ are discussed in the next two subsections.

[1] Currently there is worked on a new version of this paper.

2.1.1 Conventional matrix estimation method

Generally the matrix estimation problem is a bi-level optimization problem (see section 1.3); In the upper level problem the distances between the prior and estimated OD-matrix and between the observed and estimated link flows are minimized. However, the estimated link flows corresponding to the estimated OD-matrix, are (for most assignment models) only known explicitly after solving the assignment problem for this OD-matrix, hence after solving the lower level assignment problem. So the evaluation of the upper level objective function requires solving the lower level optimization problem, whose functional form is generally unknown. It is important to realize that bi-level problems are intrinsically hard to solve. They are neither differentiable anywhere nor convex. This holds even when the objective functions of the upper level and lower level and the constraints are all linear, because the objective function of the upper level is decided by the solution function of the lower level problem and therefore is neither linear nor differentiable [18]. Bi-level problems are even shown to be NP-hard [1], which implies that they cannot be solved to optimality within polynomial time. Therefore in practice, heuristic methods are used to search for solutions which are not guaranteed to be optimal or perfect, but which are sufficient for practical applications.

To solve the bi-level matrix estimation problem for traditional STA models, conventionally a heuristic algorithm is used that iteratively assigns the OD-vector from the upper level into the lower level and then solves the upper level problem using the assignment matrix from the lower level to approximate the relationship between the OD-flows and the link flows [8]. In figure 2.1 this solution approach is shown. Recall from equation (1.3) in section 1.3 that the relationship between the OD-flows and the link flows is determined by the assignment model and is given by the assignment matrix $\boldsymbol{A}(D)$. Within the conventional solution approach for traditional STA models, the relationship between the OD-flows and the link flows in the upper level is approximated, using the assignment matrix from the previous lower level (STA model) assignment:

$$\begin{aligned} y(D) &\approx \boldsymbol{A}(D^{k-1})D, \\ &= \boldsymbol{B}\boldsymbol{P}(D^{k-1})D. \end{aligned} \tag{2.1}$$

y	vector of estimated link flows
$\boldsymbol{A}(D^{k-1})$	assignment matrix from previous lower level assignment
D	vector of OD-demands
$\boldsymbol{B}$	crossing fraction matrix
$\boldsymbol{P}(D^{k-1})$	route fraction matrix from previous lower level assignment

The assignment matrix of a traditional STA model consists of a fixed crossing fraction matrix and a route fraction matrix which is depending on the OD-matrix. Traditional STA models do not take into account capacity and storage constraints and as such their crossing fraction matrix is equal to the (fixed) link path incidence matrix. However variations in route choice are taken into account, such that their route fraction matrix is depending on the OD-matrix[2]. So note

[2]See also figure 1.7 in section 1.3.

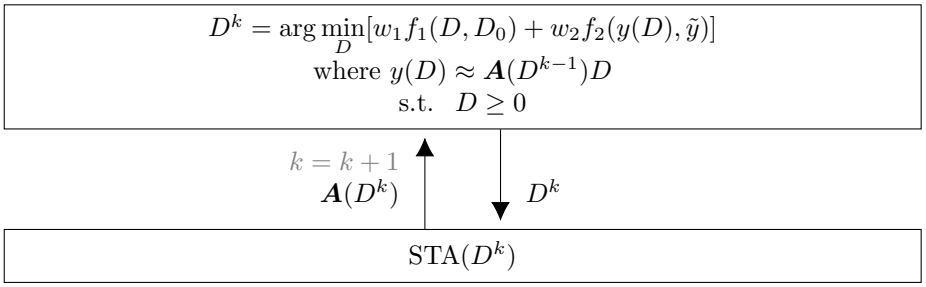


Figure 2.1: Conventional solution approach for traditional STA model.

that a linear relationship between the link flows and the OD-flows is assumed in the upper level, whereas in reality this relationship is non-linear, because actually the route fraction matrix of a traditional STA model is depending on the OD-matrix D. But by performing iterations between the upper level problem and the lower level problem, the heuristic algorithm finds solutions which are sufficient enough for practice.

2.1.2 Proposed matrix estimation method

The conventional matrix estimation method for traditional STA models as introduced in the previous subsection, is not directly suitable for the assignment model STAQ. Contrary to traditional STA models, STAQ considers capacity and storage constraints, hence it takes into account the physical effects of congestion. As such, its crossing fraction matrix is not fixed, but is depending on the OD-matrix. If the conventional matrix estimation method for traditional STA models would be directly used for STAQ, the same linear relationship would be assumed:

$$\begin{aligned} y(D) &\approx \boldsymbol{A}(D^{k-1})D, \\ &= \boldsymbol{B}(D^{k-1})\boldsymbol{P}(D^{k-1})D. \end{aligned} \tag{2.2}$$

$y(D)$	vector of estimated link flows
$\boldsymbol{A}(D^{k-1})$	assignment matrix from previous lower level assignment
D	vector of OD-demands
$\boldsymbol{B}(D^{k-1})$	crossing fraction matrix from previous lower level assignment
$\boldsymbol{P}(D^{k-1})$	route fraction matrix from previous lower level assignment

However, in reality there are two sources of non-linearity for STAQ. Considering STAQ, not only the route fraction matrix, but also the crossing fraction matrix is depending on the OD-flows. So the assignment matrix is even more depending on the corresponding OD-matrix. Therefore, the approximation of the link flows as given in equations (2.2), turns out to be not suitable for STAQ[3]; Considering STAQ, performing iterations between the upper level and lower level using (2.2), does not always lead to satisfying solutions.

[3]See example in subsection 4.2.3.

This is why Brederode et al. [5] propose to approximate the relationship between the OD-demands and the link flows for STAQ in a different way; They propose a first order Taylor approximation[4] around the previous OD-vector D^{k-1}:

$$\begin{aligned} y(D^{k-1}) &= \boldsymbol{A}(D^{k-1})D^{k-1}, \\ y(D) &\approx \boldsymbol{A}(D^{k-1})D^{k-1} + \frac{\delta\big(\boldsymbol{A}(D)D\big)}{\delta D}\Big|_{D=D^{k-1}}(D - D^{k-1}). \end{aligned} \tag{2.3}$$

The main difference between the conventional approximation[5] of the link flows and the Taylor approximation as given in (2.3) is the second term in equation (2.3) [8]. This term incorporates the sensitivity of the assignment matrix to changes in the OD-demands; Hence using this approximation, the (first order) response of the lower level problem is taken into account in the upper level problem. This idea is adopted from Frederix et al. [8]. Frederix et al. describe that (even for traditional STA models) using the approximation as in (2.3) is theoretically more sound. Herein they refer to Yang [22] and Tavana [17], which both discuss the importance of including the response of the lower level problem when solving the upper level problem.

According to Yang [22] the conventional matrix estimation method for traditional STA models solves the matrix estimation problem as if it were a Cournot Nash game, while in reality it is a Stackelberg game. In a Cournot Nash game, the upper level and lower level problem are treated in a parallel (symmetric) way. Indeed this symmetry can be recognized in the conventional solution approach for traditional STA models. Both in the upper level and in the lower level optimization problem only the latest solution of the other sub problem is known. In an iterative way there is sought for a mutually consistent solution. However actually the matrix estimation problem is a bi-level optimization problem, and as such in reality it has an asymmetric hierarchical structure. In the upper level optimization problem not only the latest solution of the lower level, but also the reaction of the lower level to a given upper level decision is known. Such an asymmetric game is also known as a Stackelberg game [10]. So the matrix estimation problem is a bi-level problem and hence a Stackelberg game, and therefore the response of the lower level should be taken into account in the upper level optimization problem.

Although Frederix et al. [8] describe that it is theoretically more sound to use a first order Taylor approximation to approximate the link flows, they also describe why this approximation is not preferred in practice; For most assignment models the relationship between the OD-demands and the link flows is not explicitly known (see section 1.3), so the sensitivity of the assignment matrix to changes in the OD-demands cannot be exactly determined. It is possible to numerically approximate this sensitivity using finite differences, but therefore it is needed to perform a lower level assignment for each OD-pair in the network. In most practical situations this leads to computation times which are not feasible. Therefore in practice many researchers use the conventional approximation of the link flows [8].

[4] This first order Taylor approximation is discussed in more detail in section 4.2.
[5] As given in (2.1) and (2.2) for respectively traditional STA models and STAQ.

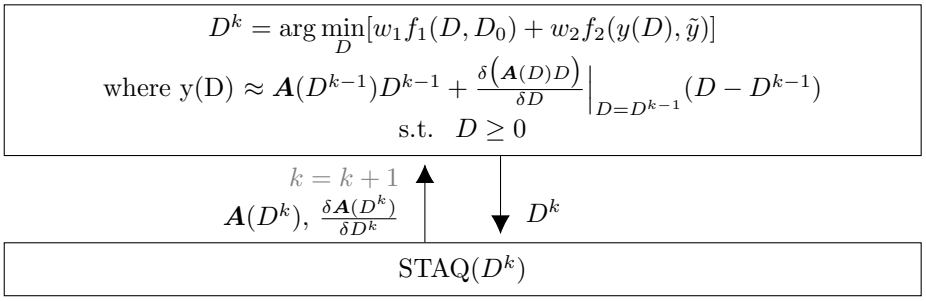


Figure 2.2: Proposed solution approach for STAQ.

Brederode et al. [5] explain in their paper that the sensitivity of the assignment matrix for STAQ can be easily determined without performing $|RS|$ times a lower level assignment, by using the node model within STAQ. As such they conclude that the approximation of the link flows as given in (2.3) can be used efficiently for STAQ. Hence for STAQ it is possible to take into account the response of the lower level in the upper level optimization problem, while computation times remain feasible. In figure 2.2 the solution approach for the matrix estimation problem with STAQ as proposed by Brederode et al. [5] is shown. The exact method they propose to determine the sensitivity of the assignment matrix to changes in the OD-demands is explained in chapter 4.

2.2 Research question

The proposed matrix estimation method for STAQ as given in figure 2.2 is still under development. Currently it is possible to find realistic solutions on small test networks, if within STAQ only the squeezing phase is considered, fixed route choice is assumed and no junction modelling is applied. This simplified variant of STAQ is shown in figure 2.3. Of course finally it should be possible to solve the matrix estimation problem with STAQ for both the squeezing and queueing phase, using STAQ as an equilibrium model and applying junction modelling. To reach this, the idea is to first add route choice, so use STAQ as an equilibrium model, and then add junction modelling to the currently developed model. Finally not only the squeezing but also the queueing phase should be considered in the future.

The developed matrix estimation model for STAQ squeezing with fixed route choice and no junction modelling is implemented in *Matlab*. In the current model the `fmincon` interior point algorithm is used to solve the upper level optimization problem. However, it is not known if this solver is (most) suitable for the given optimization problem. Therefore I was asked to find out whether `fmincon` of some other solver should be used, to solve the given upper level minimization problem. The corresponding research question is:

> *Which solver is (most) suitable for the upper level optimization problem within the matrix estimation method for the assignment model STAQ-squeezing as developed by Brederode et al. [5], assuming fixed route choice and no junction modelling?*

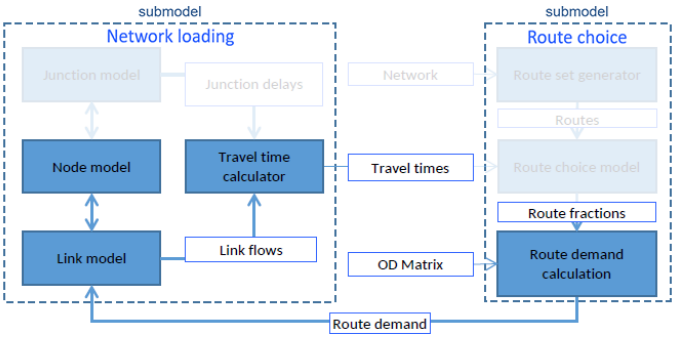


Figure 2.3: STAQ with no junction modelling, edited from [4].

2.3 Solution approach

To answer the research question, the given (upper level) optimization problem and the proposed solution method are mathematically formulated and analysed. Based on the mathematical characteristics of both the problem and the solution method a motivated choice for a suitable solver can be made. Analysing the given problem and solution method also resulted in recommendations for the further development of the matrix estimation method for STAQ.

The structure of this report is as follows; In chapter 3 the matrix estimation problem for the assignment model STAQ-squeezing with fixed route choice and no junction modelling is formulated. Subsequently in chapter 4 the developed solution method for this problem is described. Chapter 5 further investigates the characteristics of the given simplified upper level optimization problem. Finally in chapter 6 conclusions are drawn and recommendations for further research are given. Furthermore in this chapter the matrix estimation problem is discussed from a mathematical and practical point of view.

Chapter 3

Problem formulation

In section 1.3 the matrix estimation problem has been generally introduced and formulated. In this chapter the studied matrix estimation problem for STAQ-uesqueezing with fixed route choice and no junction modelling is described in detail. Assumptions within the formulation of this problem are adopted from the developed matrix estimation method for STAQ [5].

3.1 Upper level problem

The formulation of the matrix estimation problem as given in section 1.3, is the conventional form of the matrix estimation problem. As mentioned before, this formulation only accounts for count information, hence for observed link flows. However, nowadays ever more types of (big) data are available to modellers. As such, in the developed matrix estimation method for STAQ, it has been chosen to also take the average travel times along paths into account. Furthermore, the developed method assumes that there is information available to determine for each link whether the link is in a free-flow traffic regime or in a congested traffic regime. The matrix estimation problem considered in this study has the following form:

$$\begin{aligned} \min_{D} F(D) = \min_{D}\ & w_1 f_1(D, D^0) + w_2 f_2(y, \tilde{y}) + w_3 f_3(\tau, \tilde{\tau}) \\ \text{s.t. } & y, \tau = \text{Assign}(D) \\ & D \geq 0 \\ & +\text{links in right traffic regime} \end{aligned} \tag{3.1}$$

D	vector of estimated OD-flows
D^0	vector of prior OD-flows
y	vector of estimated link flows
$\tilde{y}$	vector of observed link flows
τ	vector of estimated path queueing delays
$\tilde{\tau}$	vector of observed path queueing delays
f_i	$i \in \{1, 2, 3\}$, distance functions
w_i	$i \in \{1, 2, 3\}$, weighting factors; $\sum_{i=1}^{3} w_i = 1$, $w_1 \geq 0$

In the upper level of this bi-level optimization problem not only the distances between the modelled (prior) and estimated OD-demands and the observed and estimated link flows are minimized, but also the differences between the observed and estimated path queueing delays. The average travel time on a path consists of the free-flow travel time and the average queueing delay on that path. And because the free-flow travel time on a path is fixed, only the path queueing delays are considered in the objective function. Note that the average travel time information is included in the objective function, while the traffic regime information is included in the constraints. Section 3.3 explains why the traffic regime information is needed to constrain the problem and formulates the traffic regime constraints explicitly.

The distance function used in the developed matrix estimation method is the squared L_2-norm. According to Smits [13], the advantage of this norm is that it does not rely on statistical assumptions. The distance functions in (3.1) get the following form:

$$\begin{aligned} f_1(D, D^0) &= \sum_{rs \in RS} (D_{rs} - D^0_{rs})^2, \\ f_2(y, \tilde{y}) &= \sum_{a \in \tilde{A}} (y_a - \tilde{y}_a)^2, \\ f_3(\tau, \tilde{\tau}) &= \sum_{p \in \tilde{P}} (\tau_p - \tilde{\tau}_p)^2. \end{aligned} \tag{3.2}$$

D_{rs}	demand from origin r to destination s
D^0_{rs}	apriori demand from origin r to destination s
y_a	estimated flow on link a
$\tilde{y}_a$	observed flow on link a
τ_p	estimated queueing delay on path p
$\tilde{\tau}_p$	observed queueing delay on path p
RS	set of OD-pairs
A	set of directed links
$\tilde{A} \subseteq A$	set of links with count measurements
P	set of all paths
$\tilde{P} \subseteq P$	set of all paths with travel time measurements

3.2 Lower level problem

In the lower level of the matrix estimation problem (3.1), the assignment model, Assign(D), distributes a given demand D over the network which results in flows on the links and implicitly in queueing delays (travel times) on the paths. In this study the assignment model STAQ-squeezing with fixed route choice and no junction modelling is considered. STAQ has been introduced in section 1.4 and is described in more detail in this section. The modelling framework of the STAQ variant as considered within this study is shown in figure 3.1. The junction model is drawn slightly blurred, to indicate this model is not considered. Furthermore keep in mind that only the squeezing-phase is considered.

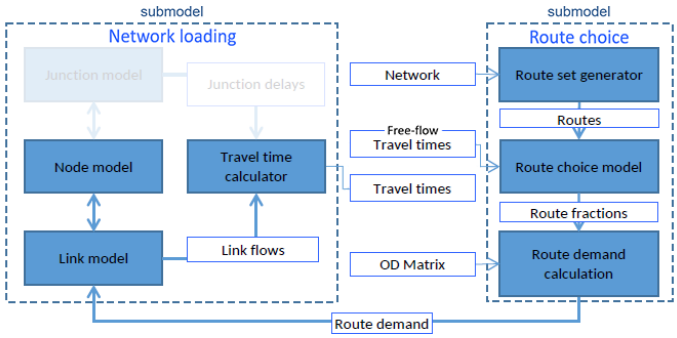


Figure 3.1: STAQ modelling framework, edited from [4].

It can be seen that the route choice submodel determines the path demand (route demand) given the network, the free-flow travel times and the OD-matrix D. The network loading submodel then determines the link flows and travel times, given the path demand from the route choice submodel. Note that normally STAQ iterates between its network loading submodel and its route choice submodel till convergence (see figure 1.8). Then a stochastic user equilibrium flow assignment has been approximated [4]. But within this study fixed route fractions are assumed, hence no iterations between the network loading model and the route choice model are performed. So the simplified variant of STAQ as considered within this study is not an equilibrium but a one shot assignment model.

3.2.1 Network loading submodel

In figure 3.1 it can be seen that within the network loading submodel of STAQ, the node model, link model and junction model interact to determine junction delays and link flows, given the path demand from the route choice submodel. Because junction modelling is omitted in this study, in this study only the node model and link model interact to determine the link flows. The travel time calculator then determines the travel times corresponding to these link flows.

It is important to realize that traditional static traffic assignment (STA) models do not need a complex network loading submodel to determine link flows and travel times. In unrestrained and capacity restrained static assignment model types it is assumed that link flows can exceed link capacities (see section 1.2.2). Therefore, for these model types, the flow on a link simply equals the travel demand on that link and the travel demand on a link can be easily calculated by summing the path demands of the paths that use this link. From these link flows, travel times can then be calculated using a travel time function. Travel time functions can be easily defined, because for unrestrained and capacity restrained model types the travel time on a link is only depending on the flow on the link itself (see section 1.2.2). So it can be concluded that the network loading submodel for traditional STA models is only formed by a link flow function and a travel time function. In practice these functions are integrated within the route choice submodel, such that one optimization problem is obtained. This is shown in appendix A.1.

It can be seen that the network loading submodel for the more capable assignment model STAQ is much more complicated. Recall that STAQ is capacity restrained in the squeezing phase and capacity and storage constrained in the queueing phase. So in contrast to the classical static traffic assignment models, in both these phases the flows on the links are constrained (see section 1.2.2); Link flows cannot exceed the capacity of a link. Hence, within the network loading submodel of STAQ, a consistent set of link flows has to be determined that satisfies the given link capacities. This requirement on the link flows can be expressed by a set of supply constraints [16] on the network loading submodel:

$$y_j = \sum_i t_{ij} \leq R_j, \quad \forall j \in J. \tag{3.3}$$

y_j	flow on outlink j
t_{ij}	turn flow from inlink i to outlink j
R_j	available supply outlink j
$I \subseteq A$	set of inlinks
$J \subseteq A$	set of outlinks

These constraints describe that the inflow[1] on an outlink can never exceed the available supply of that link. This available supply on a link is defined by link geometry and spillback from downstream supply constraints [4]. Because in this study only the squeezing-phase is considered, spillback is not taken into account. Therefore, in this study, the available supply on an outlink is only defined by link geometry, hence by the capacity C_j of outlink j; So for STAQ-squeezing R_j in (3.3) can be replaced by C_j.

When more inlinks are competing for the limited supply of an outlink, the node model within the network loading submodel determines how this available supply is divided over the competing turns. It calculates a set of reduction factors which express on turning movement level the fraction of turn demand that can fit trough a turn:

$$t_{ij} = \alpha_{ij} T_{ij}, \quad \forall ij \in IJ. \tag{3.4}$$

t_{ij}	turn flow from inlink i to outlink j
α_{ij}	reduction factor on the turn from inlink i to outlink j
T_{ij}	turn demand from inlink i to outlink j
IJ	set of turns

These reduction factors expresses the proportion between the amount of flow that wants to flow trough a turn and the amount of flow that actually can flow trough the turn. If not all flow demand can flow trough the turn, the reduction factor on this turn is less then one and the blocked flow forms a point queue on the upstream node of the corresponding outlink. So during the squeezing phase it is assumed that more vehicles can flow into a link than may exit, because there are no constraints on the storage of queues (see section 1.2.2).

[1] Flows can be defined on on link level but also on turn level, path level or origin destination level. A turn is defined by an inlink, a node and an outlink.

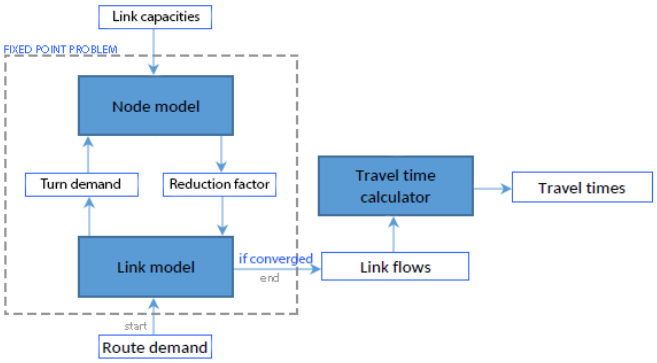


Figure 3.2: Flowchart network loading submodel of STAQ-squeezing without junction modelling [4].

An important assumption within the node model of STAQ, is that vehicles exit a link in a first in first out (FIFO) sequence; It is assumed that vehicles flow from an inlink into different outlinks in the same order they reached the end of the inlink. This means that if there is a vehicle which is unable to exit the inlink into its preferred outlink, all vehicles behind are prevented to continue regardless of their destination. So the outflow of an inlink is always restricted according to its most restrictive outlink. Therefore the node model determines the reduction factors in such a way, that the reduction factors for all turns which start from the same inlink are the same:

$$\alpha_{ij} = \alpha_i, \quad \forall ij \in IJ. \tag{3.5}$$

α_{ij} reduction factor on the turn from inlink i to outlink j
α_i reduction factor for inlink i

In figure 3.2, a flowchart of the network loading submodel of STAQ-squeezing is shown[2]. It is visualized how the node model and the link model interact to calculate for a given path demand the corresponding link flows and travel times. It can be seen that based on the given path demand (route demand) from the route choice submodel and the reduction factors from the node model, the link model calculates turn demands. From these turn demands and the given link capacities, the node model determines on its turn the reduction factors. Note that the turn demands are depending on the reduction factors, but the reduction factors are on their turn depending on the turn demands. Therefore, iterations are performed between the link model and the node model till a fixed point is reached[3]. After convergence the link model calculates the corresponding link flows. Finally from these link flows the travel time calculator determines the corresponding travel times.

[2]This flowchart visualizes a part of the STAQ modelling framework as shown in figure 3.1 for the squeezing phase.
[3]That this problem is indeed a fixed point problem is shown later on.

To determine the turn demands and the link flows, the link model uses equations (3.6) to (3.9) as given below:

$$T_{ij} = \sum_{p \in P_{ij}} \hat{\alpha}_i^p(D) Q_p, \quad \forall \in ij \in IJ. \tag{3.6}$$

$$t_{ij} = \sum_{p \in P_{ij}} \hat{\alpha}_j^p(D) Q_p, \quad \forall ij \in IJ. \tag{3.7}$$

$$Y_a = \sum_{i \in I_a} T_{ia}, \quad \forall a \in A. \tag{3.8}$$

$$y_a = \sum_{i \in I_a} t_{ia}, \quad \forall a \in A. \tag{3.9}$$

T_{ij}	turn demand from inlink i to outlink j
t_{ij}	turn flow from inlink i to outlink j
Y_a	demand on link a
y_a	flow on link a
Q_p	demand on path p
$\hat{\alpha}_i^p(D)$	reduction factor on path p till inlink i
$\hat{\alpha}_j^p(D)$	reduction factor on path p till outlink j
$P_{ij} \subseteq P$	set of paths that use turn ij
$I_a \subseteq I$	set of inlinks to outlink a

Recall that within the link model, the path demands (Q_p) are given from the route choice submodel. Furthermore within the link model the reduction factors as determined by the node model are given. From these turn based reduction factors, path based reduction factors ($\hat{\alpha}_a^p$) can be calculated. These path based reduction factors describe the fraction of traffic on path p that is not held up by supply constraints upstream from inlink a [5]:

$$\hat{\alpha}_a^p(D) = \prod_{ij \in IJ_{ap}} \alpha_{ij}(D), \quad \forall a \in A,\ p \in P. \tag{3.10}$$

$\hat{\alpha}_a^p(D)$	reduction factor on path p till link a, for OD-vector D
$\alpha_{ij}(D)$	reduction factor from inlink i to outlink j, for OD-vector D
IJ_{ap}	set of turns used by path p travelling from origin to link a

Note that the calculation of the turn demands (3.6) and turn flows (3.7) is similar. They only differ in whether the reduction factor on the turn itself is taken into account or not. To determine the turn demand on a turn, which is the amount of flow that wants to flow into the turn, the reduction factor on the turn itself should not be taken into account. But to determine the the actual flow on the turn, the reduction factor on the turn itself should be taken into account. The link demands (3.8) and link flows (3.9) can be easily determined from the turn demands and the turn flows.

To determine the reduction factors the node model uses the following node model function for node $n \in N$:

$$[\alpha_i]_{i \in I_n} = \Gamma_n(T_{i'j'}, C_{i'}, C_{j'}, \forall i' \in I_n, \forall j' \in J_n). \tag{3.11}$$

α_i	reduction factor of inlink i, see (3.5)
Γ_n	node model function, it represents the node model for node n
T_{ij}	turn demand from inlink i to outlink j
C_i	capacity of inlink i
C_j	capacity of outlink j
$I_n \subseteq A$	set of inlinks on node n
$J_n \subseteq A$	set of outlinks on node n
N	set of nodes

For each node n in the network, the node model is used to calculate the set of reduction factors for all inlinks on the node, from the set of turn demands on the node as determined by the link model, the capacities of all incoming links on the node and the capacities of all outgoing links on the node. Note that the node model function is defined for the reduction factors on all inlinks; But due to the FIFO rule the reduction factors for all turns from the same inlink are the same. There exist different kinds of node models, the one within STAQ is adopted from Tampère et al [16]. This specific node model and the corresponding algorithm are explained in detail in subsection 3.2.2.

It can be observed from respectively equations (3.11), (3.6) and (3.10) that indeed the reduction factors are depending on the turn demands, while the turn demands on their turn are depending on the reduction factors. As mentioned before, this problem, which is solved within the network loading submodel of STAQ-squeezing, is a fixed point problem. Following Bliemer et al. this problem can be formally defined as follows [3]:

$$\alpha = \Gamma(T|C) = \Gamma(\Upsilon(\alpha|Q)|C) = g(\alpha|Q, C). \tag{3.12}$$

Γ	node model function, see (3.11)
Υ	turn demand function, see (3.6)
g	composite function $\Gamma \circ \Upsilon$
α	vector of reduction factors on all turns
T	vector of turn demands on all turns
Q	vector of path demands on all paths
C	vector of capacities on all links

The vector of reduction factors α^* that satisfies $\alpha^* = g(\alpha^*|Q, C)$ is a solution to this fixed point problem.

In algorithm 1 it is shown how this fixed point problem within the network loading submodel of STAQ-squeezing without junction modelling is solved. The input of this algorithm are the path demands Q, which are given by the route choice submodel and the capacities C which are known from the given network. The output are the flows on all links in the network and the reduction factors on all turns in the network.

Algorithm 1 Capacity constrained network loading algorithm

INPUT: Q, C ▷ path demands, link capacities
OUTPUT: y, α ▷ link flows, reduction factors

STEP 0: initialization

Assume an empty network.
Initialize all reduction factors $\alpha^{(0)} = 1$.

STEP 1: calculate initial link flows

Calculate the turn demands ${T_{ij}}^{(0)}$ applying equation (3.6), using reduction factors $\alpha^{(0)}$ and path demand Q.
Calculate the link flows ${y_a}^{(0)}$ applying equations (3.7) and (3.9), using reduction factors $\alpha^{(0)}$ and path demand Q.
Set $l := 1$. ▷ iteration number

STEP 2: determine potentially congested links

Determine the set of potentially congested links using (3.13).
Determine the corresponding turn demands ${\tilde{T}_{ij}}^{(l-1)}$ using (3.15).

STEP 3: compute reduction factors

Calculate the reduction factors $\tilde{\alpha}^{(l)}$, applying the node model in (3.11), using turn demands ${\tilde{T}_{ij}}^{(l-1)}$ and link capacities C. For details, see section 3.2.2.

STEP 4: compute turn demands

Calculate the the turn demands ${\tilde{T}_{ij}}^{(l)}$ applying equation (3.6), using reduction factors $\tilde{\alpha}^{(l)}$ and path demand Q.

STEP 5: convergence check

Converged if: $\frac{1}{|\tilde{A}|}\|\tilde{\alpha}^{(l)} - \tilde{\alpha}^{(l-1)}\| < \epsilon_1$ for some $\epsilon_1 > 0$, go to STEP 6.
Otherwise, set $l := l + 1$ and return to STEP 3.

STEP 6: update the link flows on potentially congested links

Update the link flows $\tilde{y}_a$ applying equations (3.7) and (3.9), using the reduction factors $\tilde{\alpha}^{(l)}$ and path demand Q.

The algorithm starts with an initialization. All reduction factors are set to one. Using these initial reduction factors, initial turn demands and link flows are calculated using equation (3.6), (3.7) and (3.9). Then it is determined which links are potentially congested links. For each link the ratio between the link flow and the capacity is calculated. When a link has a flow/capacity ratio larger than one, this link is a potential bottleneck. All turns into this link are therefore potentially blocked, hence all corresponding inlinks are potentially congested. The set of potentially congested[4] links is formally defined as follows:

$$\tilde{A} = \{i \in I_n,\ n \in N \mid t_{ij} > 0,\ y_j > C_j,\ j \in J_n\}. \tag{3.13}$$

$\tilde{A} \subseteq A$	set of potentially congested links
t_{ij}	turn flow from inlink i to outlink j
y_j	flow on outlink j
C_j	capacity of outlink j
$I_n \subseteq A$	set of inlinks on node n
$J_n \subseteq A$	set of outlinks on node n

[4]Note that a turn into a potential bottleneck may also block other turns originating from the same inlink due to the FIFO assumption.

Only the reduction factors of the turns starting from a potentially congested link have to be considered. All other reduction factors remain fixed on their initial value of one; For these turns, it is known that all flow can fit trough. The reduced vector of considered reduction factors and the corresponding reduced vector of considered turn demands become:

$$\tilde{\alpha} = [\alpha_i]_{\{i \in \tilde{A}_n,\ n \in N\}}, \tag{3.14}$$

$$\tilde{T} = [T_{ij}]_{\{i \in \tilde{A}_n,\ j \in J_n,\ n \in N\}}. \tag{3.15}$$

$\tilde{\alpha}$	reduced vector of reduction factors
α_i	reduction factor of inlink i
$\tilde{T}$	reduced vector of turn demands
T_{ij}	turn demand from inlink i to outlink j
$\tilde{A}_n \subset \tilde{A}$	set of potentially congested links on node n

Now the algorithm iteratively runs the node model and the link model to solve for respectively $\tilde{\alpha}$ and $\tilde{T}$. The vector of reduced reduction factors is calculated using equation (3.11) and the vector of reduced turn demands is calculated using equation (3.6). These iterations are performed till the the reduction factors and hence the turn demands do not change much any more between iterations. In that case a fixed point has been found. Then finally the link model updates the link flows on the potentially congested links, using equations (3.7) and (3.9). Given these link flows, the travel time calculator can calculate the corresponding travel times. This travel time calculation is discussed in section 4.4.1.

3.2.2 The node model

Within algorithm 1, in STEP 3 the node model function as defined in equation (3.11) is used to determine the reduction factors. But, as mentioned before, this node model function is used to represent the node model. The exact node model algorithm within STAQ is explained in detail in this section.

Recall that a node model determines the reduction factors on a node, given the capacities of all links on the node and the turn demands (as determined by the link model) of all turns on the node. So note that the turn demands remain fixed during a run of the node model. Within the network loading submodel of STAQ a consistent set of link flows is sought for, which do not exceed the capacities of the links. If there are, in a congested situation, more inlinks competing for the limited supply of an outlink, the node model within the network loading submodel of STAQ determines how this limited supply is divided over the competing links. So in fact the node model determines the flows on all turns on a node. Then from these turn flows and the given turn demands on the node, the reduction factors on the node can then be determined using relation (3.4).

There exist different kinds of node models[5]. The node model within STAQ is a macroscopic first order node model. STAQ only considers traffic flows and traffic

[5]Smits et al. [14] give a good overview of the existing node models and Wright et al. [21] recently published a new first order node model.

flow characteristics, therefore it's scale is macroscopic[6]. Besides STAQ has a first order link model[7], and as such a first order node model suffices. Tampère et al. [16] describe a set of generic requirements for first order macroscopic node models. They define that within each first order macroscopic node model the optimization problem as formulated in (3.16) till (3.22) has to be solved.

For a given $n \in N$, and given $R_j,\ Y_i$ and T_{ij}:

$$\max_{t_{ij}} \Big(\sum_{i \in I_n} \sum_{j \in J_n} t_{ij} \Big) \qquad \text{subject to:} \tag{3.16}$$

$$y_j = \sum_{i \in I_n} t_{ij} \leq R_j, \quad \forall j \in J_n. \tag{3.17}$$

$$y_i = \sum_{j \in J_n} t_{ij} \leq Y_i, \quad \forall i \in I_n. \tag{3.18}$$

$$t_{ij} \geq 0, \quad \forall i \in I_n,\ j \in J_n. \tag{3.19}$$

$$f_{ij} = \frac{T_{ij}}{Y_i} = \frac{t_{ij}}{y_i} \quad \forall i \in I_n,\ j \in J_n. \tag{3.20}$$

$$\text{if } \exists i \in I_n \mid y_i < Y_i, \text{then } y_i \text{ invariant to } Y_i \rightarrow C_i. \tag{3.21}$$

$$\text{node supply constraints.} \tag{3.22}$$

$$\text{SCIR constraints.} \tag{3.23}$$

t_{ij}	turn flow from inlink i to outlink j
y_j	flow on outlink j
R_j	supply on outlink j
Y_i	demand on inlink i
f_{ij}	turning fraction from inlink i to outlink j
T_{ij}	turn demand from inlink i to outlink j
y_i	flow on inlink i
C_i	capacity on inlink i
I_n	set of inlinks on node n
J_n	set of outlinks on node n

As we shall see, this optimization problem (3.16) to (3.22) is solved iteratively by the node model algorithm within STAQ (see algorithm 2). Indeed the variables within this optimization problem are the turn flows. Note that the optimization problem is defined on node level. So it has to be solved for each node in the network, to determine all turn flows in the network. The turn demands T_{ij} and hence also the link demands Y_i (see equation 3.8) are given from the link model

[6] Microscopic traffic flow models simulate single vehicles.
[7] In a first order link model only one independent variable (flow) is considered.

and the supply R_j, equals for STAQ-squeezing without junction modelling the capacity C_j, which is known from the given network. The objective function (3.16) is to maximize the sum of all turn flows. Turn flows are maximized because drivers will always try to move whenever possible. So each turn flow will increase till it is actively restricted by one of the constraints. Besides the supply constraints, which were mentioned before (see equation 3.3), Tampère et al. describe more requirements which should be fulfilled by the node model in order to find a realistic and consistent set of link flows. All these requirements are explained below, following the paper of Tampère et al. [16].

The first two set of constraints (3.17) and (3.18) are respectively the supply and the demand constraints. The supply constraints R_j describe the maximum inflow that outlink j could possibly receive, when there are considered no constraints on the inflow of link j. The demand constraints Y_i describe on their turn the maximum flow that inlink i could possibly send, when there are considered no constraints on the outflow of link i. As mentioned before, the supply constraints R_j are in this study defined by the capacity of the outlink C_j. It can be concluded that the supply and demand constraints ensure that the flow on a link does not exceed the supply of the link respectively the flow demand for the link. The third set of constraints (3.19) are the non-negativity constraints. Traffic never flows backwards. Therefore this constraint ensures that the link flows never attain a negative value. Then the fourth set of constraints (3.20) are the conservation of turning fractions (CFT) constraints. These constraints describe that vehicles should exit a link in a FIFO sequence[8]. The total demand of an inlink can be split into turn demands to all considered outlinks. To ensure that a constraint on one of these turn demands also restricts the other turn demands, as required by the FIFO assumption, the turning fractions f_{ij} in the turn demand should be conserved in the resulting flows. This is ensured by the CFT constraints. The fifth requirement[9] (3.21) describes the invariance principle. This principle states that flows should be invariant during an infinitesimal time step if the demand and supply constraints are constant. It is known from traffic flow dynamics that the demand of a link in a congested regime increases after some infinitesimally small time step to the link capacity C_i. When the node model would predict a different value for y_i due to this change from Y_i to C_i, it would contradict its initial solution. Therefore the invariance principle ensures that when a link is in a congested regime, the corresponding link flow is invariant under the replacement of Y_i by C_i. Note that the flow on link i equals the sum of the flows on all turns starting from inlink i, so indeed this requirement is an constraint on the turn demands t_{ij}.

Tampère et al. [16] describe that the supply constraints, demand constraints, non-negativity constraints, CFT constraints and the invariance principle are generic requirements; They hold for all types of macroscopic first order node models. But they define that the next two constraints, the supply constraint interaction rule constraints (SCIR) and the node supply constraints, are specific for the type of macroscopic first order node model that is considered. A node typically imposes some node supply constraints (3.22). These constraints express the effects of limited supply on the node (junction) due to conflict points and

[8]The FIFO-assumption has been explained in subsection 3.2.1.

[9]Note that this requirement is not a constraint.

traffic regulation. In this study no junction modelling is considered, therefore in this study within the node model of STAQ-squeezing it is assumed that there are no node supply constraints. The SCIR constraints (3.23) describe the distribution of the supply. It is known that drivers are selfish and seek for a user optimum. Without the specification of SCIR constraints cooperative non-selfish behaviour would be assumed, because then the distribution of the supply would be only determined by the maximization of the total flow. The SCIR constraints define how the limited supply is distributed over the competing flows and how the supply constraints interact with each other, herein reflecting the aggregate driving behaviour at a congested intersection.

The SCIR within STAQ is the oriented capacity proportional distribution. This SCIR is adopted from Tampère et al. [16]. So it can be concluded that the node model within STAQ is a first order marcroscopic node model with an oriented capacity proportional distribution. The distribution scheme of the oriented capacity proportional distribution is based on directional capacities C_{ij}:

$$C_{ij} = f_{ij} C_i = \frac{T_{ij}}{\sum_{i \in I_i} T_{ij}} C_i, \quad \forall i \in I_n, j \in J_n. \tag{3.24}$$

C_{ij}	directional capacity per turn
f_{ij}	turning fraction for the turn from inlink i to outlink j
T_{ij}	turn demand from inlink i to outlink j
C_i	capacity of link i
J_i	set of outlinks considered by inlink i
I_n	set of inlinks on node n
J_n	set of outlinks on node n

The available supply of an outlink j is divided over the competing inlinks i proportional to their directional capacities C_{ij} as defined in the distribution scheme above. Hence the rightful share of the supply for each competing inlink i is depending on its capacity and its turning fraction in relation to the capacities and turning factions of the other competing inlinks. In their paper [16], Tampère et al. give a behavioural interpretation for oriented capacities as basis for the distribution scheme.

Besides the oriented capacities for all turns, more information is needed to be able to determine the distribution of the available supplies in a congested situation; It should also be known which outlinks are bottlenecks and which inlinks are competing for the limited supply of these outlinks. Actually, in the case of active supply constraints, the SCIR has to find a consistent answer to the following two questions[10]:

- For each inlink i: Which constraint is most restrictive for this inlink?
  The maximization of turn flows in (3.16) implies that each turn flow is restricted by either a demand constraint or a supply constraint. Furthermore, due to the FIFO assumption, the turn flows starting from the same inlink are mutually coupled. Therefore for each inlink i exactly one constraint

[10] For inlinks $i \in I_n$ and supply constraints R_j on the outlinks $j \in J_n$.

can be identified as the most restrictive one. So each inlink i belongs to exactly one of the following sets:

$$\begin{aligned} i \in W_i &\iff y_i = Y_i, \\ i \in U_j &\iff j = j^*(i) \text{ and } y_i < Y_i. \end{aligned} \tag{3.25}$$

W_i	set of inlinks constrained by Y_i
U_j	set of inlinks constrained by R_j
Y_i	demand on inlink i
R_j	supply of outlink j
y_i	flow on inlink i
$j^*(i)$	outlink j with the most restrictive supply constraint on inlink i

The composition of these sets U and W is depending on how the available supply is distributed over the competing flows.

- For each supply constraint R_j: how is the supply distributed to each one of the competing turns towards j?
  For each supply constraint the supply distribution scheme of the SCIR describes how the limited supply is distributed. This scheme accounts for the aggregate driver behaviour and is depending on the composition of the sets U and W. Within the node model of STAQ the distribution is based on the directional capacities as defined in (3.24).

Note that these two questions are related to each other. Whether the flow on a link is limited by some supply constraint or not, depends on the share of the supply that is distributed to this link. But the distribution of the supply is on its turn depending on the composition of the sets U and W, hence on which flows are limited by which constraint. Therefore the node model algorithm works iteratively, as will be explained below.

An important observation is that when the flow on an inlink i on node n is constrained by the supply of the most restrictive outlink $R_{j^*(i)}$, then due to the FIFO assumption, inlink i uses less than its rightful share of the supplies R_j of the other outlinks j on node n it was competing for. Therefore, the SCIR first subtracts the claims from all non-competing inlinks ($i \notin U_j$), which are less then their rightful share, from the supply of an outlink R_j (see line 27 of algorithm 2). Then the reduced supply is distributed among the competing inlinks ($i \in U_j$), proportional to their oriented capacity. In this way the full supply R_j of each outlink is used. The reduced supply is defined as follows:

$$\tilde{R}_j = R_j - \sum_{i \notin U_j} t_{ij}. \tag{3.26}$$

$\tilde{R}_j$	reduced supply on outlink j
R_j	supply of outlink j
U_j	set of inlinks $i \in I_n$ constrained by R_j
t_{ij}	turn flow from inlink i to outlink j

Note that also the turn demands from demand constrained inlinks are subtracted from the supply of the considered outlink to obtain the reduced supply.

In algorithm 2 it is shown how the node model within STAQ, which is adopted from Tampère et al. [16], finds the turn flows on a node, given the capacities of the inlinks, the capacities of the outlinks and the turn demands from the link model. Within the algorithm the turn flows are determined as follows:

$$t_{ij} = \begin{cases} T_{ij} & \forall i \mid W_i \neq \emptyset, \\ \beta_{j^*(i)} C_{ij} & \forall i \in U_{j^*(i)}, \end{cases} \tag{3.27}$$

where:

$$\beta_j = \begin{cases} \frac{\tilde{R}_j}{\sum\limits_{i \in U_j} C_{ij}} & \text{if } U_j \neq \emptyset, \\ 1 & \text{if } U_j = \emptyset, \end{cases} \tag{3.28}$$

$$j^*(i) = \underset{j \mid T_{ij} > 0}{\operatorname{argmin}} \beta_j. \tag{3.29}$$

t_{ij}	turn flow from inlink i to outlink j
T_{ij}	turn demand from inlink i to outlink j
β_j	level of reduction for outlink j
C_{ij}	directional capacity from inlink i to outlink j
$j^*(i)$	outlink j with the most restrictive supply constraint on inlink i
W_i	set of all inlinks i constrained by Y_i
U_j	set of inlinks constrained by R_j
$\tilde{R}_j$	reduced supply on outlink j

So when a turn is constrained by the demand of the inlink, the turn flow of this turn is equal to the turn demand. But when a turn is constrained by the supply of the most restrictive outlink corresponding to its inlink, this turn obtains a share of the reduced supply which is proportional to its oriented capacity; This can be seen from equations (3.27) and (3.28):

$$\beta_{j^*(i)} C_{ij} = \frac{C_{ij}}{\sum\limits_{i \in U_j} C_{ij}} \tilde{R}_j, \quad \forall i \in U_{j^*(i)}. \tag{3.30}$$

β_j	level of reduction for outlink j
$j^*(i)$	outlink j with the most restrictive supply constraint on inlink i
C_{ij}	directional capacity from inlink i to outlink j
U_j	set of inlinks constrained by R_j
$\tilde{R}_j$	reduced supply on outlink j

Note that the turn flows in (3.27) are iteratively determined. The level of reduction β_j and the sets U_j, which are required to determine the turn flows, are mutually depended. Equation (3.29) implicitly defines the sets W and U, and thereby requires the level of reduction. But to calculate the level of reduction following equation (3.28), the sets U are required. This is why algorithm 2 solves the problem iteratively. From equations (3.26) till (3.29) it can be deduced that:

$$\beta_j = \frac{\tilde{R}_j}{\sum_{i \in U_j} C_{ij}} = \frac{R_j - \sum_{i \mid W_i \neq \emptyset} T_{ij} - \sum_{j' \neq j} \sum_{i \in U_{j'}} \beta_{j'} C_{ij}}{\sum_{i \in U_j} C_{ij}}. \tag{3.31}$$

β_j	level of reduction for outlink j
$\tilde{R}_j$	reduced supply on outlink j
C_{ij}	directional capacity from inlink i to outlink j
R_j	supply on outlink j
W_i	set of all inlinks i constrained by Y_i
T_{ij}	turn demand from inlink i to outlink j
U_j	set of inlinks constrained by R_j

From (3.29) and (3.31) it can be seen that the smallest level of reduction can be directly calculated from the input of the node model; There exist no turns with positive turn demand to the most restrictive outlink $\hat{j}$, for which the corresponding inlink is restricted more by another outlink. Therefore, in the calculation of the smallest level of reduction, the last term in the numerator of equation (3.31) can be omitted:

$$\beta_{\hat{j}} = \frac{R_j - \sum_{i|W_i \neq \emptyset} T_{ij}}{\sum_{i \in U_j} C_{ij}}, \quad \forall j \in J_n. \tag{3.32}$$

β_j	level of reduction for outlink j
$\tilde{R}_j$	reduced supply on outlink j
C_{ij}	directional capacity from inlink i to outlink j
R_j	supply on outlink j
W_i	set of all inlinks i constrained by Y_i
T_{ij}	turn demand from inlink i to outlink j
$\hat{j}$	most restrictive outlink

In general it holds that in the calculation of the level of reduction of an outlink, only the interaction from outlinks with a smaller level of reduction has to be considered. So from the smallest level of reduction the second smallest level of reduction can be calculated and so on. In this way the algorithm iteratively finds the exact solution for all β_j and U_j and thereby determines all turn flows t_{ij}.

Algorithm 2 Node model algorithm - for node n

INPUT:
$C_i, C_j \mid \forall i \in I_n, j \in J_n$ ▷ link capacities
$T_{ij} = T_{ij}^{(l-1)} \mid \forall i \in I_n, j \in J_n$ ▷ turn demands from link model
$Y_i = Y_i^{(l-1)}, Y_j = Y_j^{(l-1)} \mid \forall i \in I_n, j \in J_n$ ▷ link demands from link model
OUTPUT:
$t_{ij} \mid \forall i \in I_n, j \in J_n$ ▷ turn flows
STEP 1: initialize ▷ initial iteration number
for all i, j **do**
    $\tilde{R}_j^{(0)} = C_j$ ▷ initial supply constraint
    $U_j{}^{(0)} = \{i \mid T_{ij} > 0\}$ ▷ initial set of competing inlinks per outlink
    $J^{(0)} = \{j \mid Y_j > 0\}$ ▷ initial set of considered outlinks

STEP 2: determine oriented capacities

for all $i \mid Y_i > 0$ **do**
  for all j **do**
    $C_{ij} = \frac{T_{ij}}{Y_i} C_i, \forall j$ ▷ calculate oriented capacities

STEP 3: determine most restrictive constraint

for all $j \in J$: **do**
  $\beta_j = \frac{\tilde{R}_j}{\sum_{i \in U_j} C_{ij}}$ ▷ calculate level of reduction for outlink j
  $\hat{j} = \text{argmin}_{j \in J} \beta_j$ ▷ determine most restrictive outlink

STEP 4: determine flows of set $U_{\hat{j}}$ and recalculate $\tilde{R}_j$

if $\exists i \in U_{\hat{j}} \mid Y_i \leq \beta_{\hat{j}} C_i$ **then** ▷ if $\exists i \in U_{\hat{j}}$ demand constrained
  for all $i \in U_{\hat{j}} \mid Y_i \leq \beta_{\hat{j}} C_i$ **do**
    $t_{ij} = T_{ij}, \quad \forall j$ ▷ calculate link flow
    for all $j \in J$ **do**
      $\tilde{R}_j^{(k+1)} = \tilde{R}_j - T_{ij}$ ▷ remove used supply from constraint
      $U_j^{(k+1)} = U_j \setminus \{i\}$ ▷ remove inlink from U_j
      if $U_j^{(k+1)} = \emptyset$ **then**
        $\beta_j = 1$ ▷ set definitive β_j
        $U_j = \emptyset$ ▷ set definitive U_j
        $J^{(k+1)} = J^k \setminus \{j\}$ ▷ remove outlink from considered set
else if $Y_i > \beta_{\hat{j}} C_i \quad \forall i \in U_{\hat{j}}$ **then** ▷ if $\forall i \in U_{\hat{j}}$ constrained by $\hat{j}$
  for all $i \in U_{\hat{j}}$ **do**
    $t_{ij} = \beta_{\hat{j}} C_{ij} \quad \forall j$ ▷ calculate link flow
    for all $j \in J$ **do**
      $\tilde{R}_j^{(k+1)} = \tilde{R}_j - \beta_{\hat{j}} C_{ij}$ ▷ remove used supply from constraint
      if $j \neq \hat{j}$ **then**
        $U_j^{(k+1)} = U_j \setminus U_{\hat{j}}$ ▷ remove inlinks from U_j
        if $U_j^{(k+1)} = \emptyset$ **then**
          $\beta_j = 1$ ▷ set definitive β_j
          $U_j = \emptyset$ ▷ set definitive U_j
          $J^{(k+1)} = J^k \setminus \{j\}$ ▷ remove outlink from considered set
      else if $j = \hat{j}$ **then**
        $\beta_j = \beta_{\hat{j}}$ ▷ set definitive β_j
        $U_j = U_{\hat{j}}$ ▷ set definitive U_j
        $J^{(k+1)} = J^k \setminus \{\hat{j}\}$ ▷ remove outlink from considered set

STEP 5: Stop criterion

      if $J^{(k+1)} = \emptyset$ **then**
        Stop ▷ algorithm ends
      else
        return to STEP 3 ▷ new iteration

Finally, from the turn flows on a node as calculated by algorithm 2, the reduction factors for all turns on the node (STEP 3 algorithm 1) can be determined using relation (3.4):

$$\alpha_{ij}{}^{(l)} = \frac{t_{ij}}{T_{ij}{}^{(l-1)}}, \quad \forall i \in I,\ \forall j \in J. \tag{3.33}$$

$\alpha_{ij}{}^{(l)}$	reduction factor from inlink i to outlink j; in iteration (l) of the network loading algorithm
t_{ij}	turn flow from inlink i to outlink j; determined by the node model algorithm, in iteration (l) of the network loading algorithm
$T_{ij}{}^{(l-1)}$	turn demand from inlink i to outlink j; determined by the link model, in iteration $(l-1)$ of the network loading algorithm

Note that, due to the FIFO assumption, the reduction factors for all turns which start from the same inlink are the same. In their paper Tampère et al. [16] prove that algorithm 2 indeed satisfies the general requirements for first order macroscopic node models as defined in (3.16) till (3.23).

3.2.3 Route choice submodel

In the previous two subsections, the network loading submodel of STAQ-squeezing without junction modelling (subsection 3.2.1) and its corresponding node model (subsection 3.2.2) are discussed. Recall that each assignment model consists of both a network loading submodel and a route choice submodel. However, Brederode et al. [4] describe that the advantages of the assignment model STAQ as compared to a traditional STA model, are derived from its network loading submodel. As such, the route choice submodel of STAQ is interchangeable. The route choice submodel as used within this study is adopted from Brederode et al. [4]. This route choice submodel consists of three components (see figure 3.1):

- The route set generator
- The route choice model
- The route demand calculator

These three components are shortly explained in this section. It is important to realize that concerning the route choice, there is a difference depending whether the traffic assignment is a one shot model or an equilibrium model. In an equilibrium model the route set generator is run once at the beginning of the assignment step, but in each iteration, for computing an (approximate) user equilibrium, the route choice and the route demand are calculated anew based on the actual path travel times in that iteration (see figure 1.8). However in a one-shot model, such as the variant of STAQ-squeezing as used within this study[11], the route set, route choice (depending on the free-flow travel time) and the route demand are only computed once per traffic assignment (see figure 3.1).

[11] Recall that within this study the route fractions are considered to be fixed. As such the considered assignment model is not an equilibrium model but a one-shot model.

The first component of the route choice submodel is the route set generator. The route set generator determines a set of potential routes for each OD-pair [7]. First, the route set generator calculates the shortest path for all OD-pairs using the Dijkstra algorithm on the given network. Then, to generate alternative routes, a random sampling process using a gamma distribution is applied. Herein free-flow travel times are assumed. Finally, a route filter is used to reduce route overlap, to remove irrelevant routes and to restrict the size of the set of potential routes [4]. Note that the route set generator is only needed within the first run of the route choice submodel.

From the route sets as determined by the route set generator and given the travel times, the route choice model, which is the second component within the route choice submodel, determines the corresponding route fractions. In this study the route choice is stochastic; The trips are assigned to the potential routes using probabilities. These probabilities, which are called route fractions, represent the likelihood of choosing a certain path and are determined based on the generalized costs. To calculate the route fractions, the multinomial logit (MNL) model is used. This leads to the following definition of route fractions [4]:

$$\psi_p^{rs} = \frac{e^{-\mu c_p}}{\sum_{p' \in P_{rs}} e^{-\mu c_{p'}}}. \tag{3.34}$$

ψ_p^{rs}	fraction of OD-demand rs on path p
μ	scale parameter
c_p	generalized costs on path p
P_{rs}	set of paths p between OD-pair rs

Note that there are multiple paths considered in one trip assignment; Not all drivers take the route with the smallest generalized cost[12], also other routes are used, only with a smaller probability. In this way variations in driver perceptions are modelled. The scale parameter describes the degree of the perception errors of travellers on route travel times. It can be seen that if the scale parameter goes to infinity, $\mu \to \infty$, then only the route with the smallest generalized costs is used, while if the scale parameter is goes to zero, $\mu \to 0$, then all demand is equally distributed over the potential paths. The route choice is conditional on the path set as determined by the route set generator.

Finally, from the route fractions as determined by the route choice model, the route demand calculator, which is the third component of the route choice submodel, calculates the corresponding path demands, using the following relation[13]:

$$Q_p = \sum_{rs \in RS} \psi_p^{rs} D_{rs}. \tag{3.35}$$

Q_p	demand on path p
ψ_p^{rs}	fraction of OD-demand rs on path p
D_{rs}	demand from origin r to destination s

[12] Recall that the generalized costs are assumed to consist of travel times only.

[13] Note that each path p only runs between one OD-pair. Hence only for one OD-pair rs the route fraction ψ_p^{rs} is non zero.

Because in this study fixed route fractions and hence fixed path demands are assumed, the route choice submodel is only run once. But when STAQ is used as an equilibrium model, iterations are preformed between the route choice submodel and the network loading submodel. Then, to check for convergence a relative duality gap is used. And to speed up convergence the route demands are averaged over iterations. For details the reader is referred to Bliemer et al. [3] and Brederode et al. [4]. After convergence in the equilibrium model a stochastic user equilibrium (SUE) has been approximated. Recall that for STAQ-queueing such a user equilibrium cannot always be reached [6].

3.2.4 Dogbone example

Now that both the network loading submodel of STAQ-squeezing without junction modelling and the considered route choice submodel are discussed, in this subsection an example of an assignment with the assignment model STAQ-squeezing with fixed route choice and no junction modelling is shown on a relative simple network. The considered network is called the Dogbone network. This network is shown in figure 3.3. It has two origins, two destinations and seven links. In the figure for each link the link number and the capacity of the link (veh/h) are given. $R = \{1, 3\}$ is the set of origins and $S = \{2, 4\}$ is the set of destinations of the network. Hence the OD-matrix for the Dogbone network has the following form:

$$\mathbf{D} = \begin{bmatrix} D_{12} & D_{14} \\ D_{32} & D_{34} \end{bmatrix}. \tag{3.36}$$

Note that there is no route choice in this simple Dogbone network. For each OD-pair, there is only one route possible. Hence the route choice within the Dogbone network is inherently fixed; No route choice model is needed to determine the route fractions. In figure 3.4 the four paths in the Dogbone network are indicated.

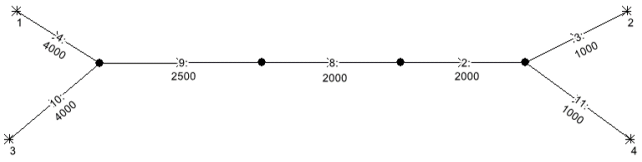


Figure 3.3: The Dogbone network.

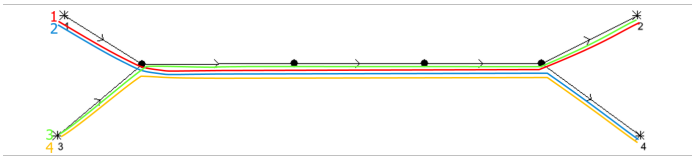


Figure 3.4: Paths in the Dogbone network.

Each path is used by one of the four OD-pairs. So it can be concluded that for the Dogbone network the OD-demand D_{rs} is equal to the demand of the

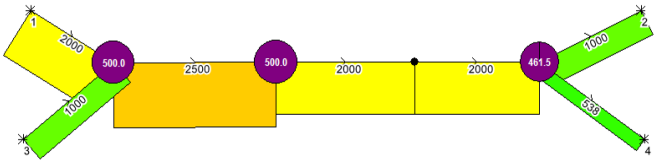


Figure 3.5: Resulting flows and vertical queues on the Dogbone network.

corresponding path Q_p. Formally the following route fractions ψ_p^{rs} can be defined:

$$\psi_1^{12} = 1,\ \psi_2^{14} = 1,\ \psi_3^{32} = 1,\ \psi_4^{34} = 1. \tag{3.37}$$

Suppose that the following OD-matrix is assigned to the Dogbone network by the assignment model STAQ-squeezing with fixed route choice and no junction modelling:

$$\mathbf{D} = \begin{bmatrix} 1500 & 500 \\ 500 & 500 \end{bmatrix}. \tag{3.38}$$

Following equation (3.35) and using the route fractions as given in (3.37), the path demands corresponding to the given OD-demands are:

$$Q_1 = 1500,\ Q_2 = 500,\ Q_3 = 500,\ Q_4 = 500. \tag{3.39}$$

Given these path demands, the network loading submodel, can calculate the corresponding flows on the links and the vertical queues on the bottlenecks in the network. These assignment results are shown in figure 3.5. To obtain these results, the network loading algorithm, algorithm 1 is applied. This algorithm iteratively solves for the reduction factors and the turn demands, till a fixed point is reached. The following reduction factors and turn demands are iteratively found by the network loading algorithm from the considered fixed path demands (3.39) on the Dogbone network[14]:

↓turns	$\boldsymbol{\alpha^{(0)}}$	$\boldsymbol{T^{(0)}}$	$\boldsymbol{\alpha^{(1)}}$	$\boldsymbol{T^{(1)}}$	$\boldsymbol{\alpha^{(2)}}$	$\boldsymbol{T^{(2)}}$	$\boldsymbol{\alpha^{(3)}}$
49	1	2000	$\frac{3}{4}$	1500	$\frac{3}{4}$	1500	$\frac{3}{4}$
109	1	1000	1	1000	1	1000	1
98	1	3000	$\frac{4}{5}$	2500	$\frac{4}{5}$	2500	$\frac{4}{5}$
82	1	3000	1	2000	1	2000	1
23	1	2000	$\frac{1}{2}$	1300	$\frac{10}{13}$	1300	$\frac{10}{13}$
211	1	1000	$\frac{1}{2}$	700	$\frac{10}{13}$	700	$\frac{10}{13}$

It can be seen that a fixed point is already reached after the second iteration; Within the next iteration the vector of reduction factors does not change any more. Hence it turns out that the fixed point problem for the simple Dogbone network is relatively easy to solve; For other networks it could be necessary to perform much more iterations before a fixed point is reached.

From the resulting vector of reduction factors, the network loading algorithm determines the link flows using equations (3.7) and (3.9). The reader can check

[14]The reader can verify these results by applying the network loading algorithm (1) and the embedded node model algorithm (2) on the path demands as given in (3.39).

that these link flows indeed are the flows as shown in figure 3.5. Also the vertical queues as shown in this figure can be verified. To do so, recall that all demand on a link that exceeds the maximum capacity of this link, forms a vertical point queue on the upstream node of this link. So for example, on link 9 the link demand is 3000 vehicles per hour (see equation (3.8)), but the capacity of this link is only 2500 vehicles per hour. Therefore a vertical queue of 500 vehicles per hour can be found on the upstream node of link 9.

3.3 Constraints

In the previous section the lower level assignment model is discussed in detail. The reader is now familiar with reduction factors and knows how they are determined by the lower level assignment model. Therefore in this section the traffic regime constraints within the matrix estimation problem for STAQ-squeezing (3.1) can be formulated explicitly.

3.3.1 Traffic regime constraints

As mentioned in section 3.1, within the proposed matrix estimation method for STAQ, it is assumed that it is possible to determine for each link whether this link is in a free-flow traffic regime or in a congested traffic regime. So for each outlink j an indicator χ_j can be defined, which indicates whether the outlink is actively constraining inflow or not:

$$\chi_j = \begin{cases} 1 & \text{if outlink } j \text{ is actively constraining inflow.} \\ 0 & \text{if outlink } j \text{ is not actively constraining inflow.} \end{cases} \tag{3.40}$$

In practice these χ_j-indicators are set by the decision maker, based on available road information. This traffic regime information is used to define traffic regime constraints. Within the matrix estimation problem there is sought for an optimal OD-matrix. The traffic regime constraints are added, to ensure that this OD-matrix can only be chosen such that all links remain within the right traffic regime. This means that all free-flow links, for which $\chi_j = 0$, remain free-flow and that all congested links, for which $\chi_j = 1$, remain congested. The traffic constraints are defined as follows:

$$\chi_j = \begin{cases} 1 \quad \text{then} \quad \sum_{p \in P_j} \psi_p^{rs} D_{rs} \prod_{ij \in \tilde{IJ}_{jp}} \alpha_{ij}(D) \geq C_j, \\ 0 \quad \text{then} \quad \sum_{p \in P_j} \psi_p^{rs} D_{rs} \prod_{ij \in \tilde{IJ}_{jp}} \alpha_{ij}(D) \leq C_j, \end{cases} \qquad \forall j \in \tilde{J}. \tag{3.41}$$

D_{rs}	demand from origin r to destination s
ψ_p^{rs}	fraction of OD-demand rs on path p
$\alpha_{ij}(D)$	reduction factor from inlink i to outlink j for OD-vector D
C_j	capacity of outlink j
χ_j	$\in \{1, 0\}$, indicates if outlink j is actively constraining inflow
P_j	set of paths using outlink j
$\tilde{IJ}_{jp}$	set of turns on path p travelling from origin to outlink j, excluding the last turn to link j
$\tilde{J}$	set of outlinks j which can actively constrain flow

These constraints describe that when an outlink is in a congested regime, the total reduced traffic demand of all paths using this outlink is more than the capacity of this outlink, hence the link is actively constraining inflow. And when an outlink is in a free-flow regime, the total reduced traffic demand of all paths using this outlink is less than the capacity of this outlink, hence the link is not actively constraining inflow[15]. Note that the traffic regime constraints are only known explicitly for a certain OD-vector after a lower level assignment of this OD-vector; Only then the reduction factors corresponding to this OD-vector are known. All traffic regime constraints form, together with the constraint that the OD-demands should be non-negative, the feasible set of solutions.

Frederix et al. [8] discuss the importance of including traffic regime constraints. They explain that the relationship between link flows and OD-demands behaves non-monotonic in the transition from a free-flow regime to a congested regime; In a free-flow regime the link flows are an increasing function of the passing OD-flows, whereas in a congested regime, the link flows are determined by the bottleneck capacities and therefore are insensitive to the passing OD-flows. As such, the sensitivity of the link flows to changes in the OD-demands is different in both regimes, not only in magnitude but also in sign. Therefore the transition from a free-flow to a congested regime (or the other way around) forms a source of local minima. Jumps between different local minima can be avoided, by choosing only OD-matrices D for which the corresponding link flows are in the same traffic regime as the actual traffic regime.

3.3.2 Dogbone example

As an example, the traffic regime constraints for the Dogbone network are derived. The Dogbone example has been introduced in section 3.2.4. From the link capacities of the Dogbone network as given in figure 3.3, it can be seen that there are four links in the Dogbone network which can actively constrain flow. This are link 3, 11, 8 and 9. Therefore, in the Dogbone network, the indicator set $\chi = \{\chi_3, \chi_{11}, \chi_8, \chi_9\}$ defines which outlinks are able to actively constrain flow. From (3.41) for the Dogbone network the following traffic regime constraints can be defined:

$$\chi_3 = \begin{cases} 1 & \text{then} \quad \alpha_{4,9}(D)\alpha_{9,8}(D)D_{12} + \alpha_{10,9}(D)\alpha_{9,8}(D)D_{32} \geq 1000. \\ 0 & \text{then} \quad \alpha_{4,9}(D)\alpha_{9,8}(D)D_{12} + \alpha_{10,9}(D)\alpha_{9,8}(D)D_{32} \leq 1000. \end{cases}$$

$$\chi_{11} = \begin{cases} 1 & \text{then} \quad \alpha_{4,9}(D)\alpha_{9,8}(D)D_{14} + \alpha_{10,9}(D)\alpha_{9,8}(D)D_{34} \geq 1000. \\ 0 & \text{then} \quad \alpha_{4,9}(D)\alpha_{9,8}(D)D_{14} + \alpha_{10,9}(D)\alpha_{9,8}(D)D_{34} \leq 1000. \end{cases}$$

$$\chi_8 = \begin{cases} 1 & \text{then} \quad \alpha_{4,9}(D)(D_{12} + D_{14}) + \alpha_{10,9}(D)(D_{32} + D_{34}) \geq 2000. \\ 0 & \text{then} \quad \alpha_{4,9}(D)(D_{12} + D_{14}) + \alpha_{10,9}(D)(D_{32} + D_{34}) \leq 2000. \end{cases}$$

$$\chi_9 = \begin{cases} 1 & \text{then} \quad D_{12} + D_{14} + D_{32} + D_{34} \geq 2500. \\ 0 & \text{then} \quad D_{12} + D_{14} + D_{32} + D_{34} \leq 2500. \end{cases}$$

For the Dogbone network the relation between the traffic regimes and the path demands, which equal the OD-demands, can be visualized. This is shown in

[15] It can be seen from equations (3.6), (3.8) and (3.35) that the left side of the inequalities in (3.41) equals the total demand on outlink j (Y_j).

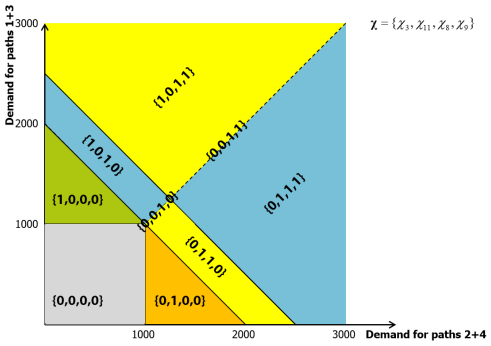


Figure 3.6: Possible traffic regimes for the Dogbone network.

figure 3.6. Note that such a visualization of the traffic regimes is not possible in general. Because the traffic regime constraints form, together with the constraint that all OD-demands should be non-negative, the feasible set of solutions, it can be concluded that figure 3.6 actually visualizes for the Dogbone network for each possible χ the set of feasible solutions. It can be seen that not all combinations of the four indicators exist in practice. This can be easily explained by the capacities of the links in the network (see figure 3.3). First of all it is not possible that both $\chi_3 = 1$ and $\chi_{11} = 1$, because the capacity of link 2 is only 2000. Secondly it is not possible that $\chi_9 = 1$ and $\chi_8 = 0$, because the capacity of link 9 is 2500 while the capacity of link 8 is 2000.

3.4 Uniqueness of the solution

In the previous sections, the upper level problem, lower level problem and the constraints of the considered bi-level matrix estimation problem for STAQ-squeezing with fixed route choice and no junction modelling (3.1) are discussed. In this final section the uniqueness of the solution to the given upper level minimization problem is discussed. To do so, in the next subsections the three objective parts, f_1, f_2 and f_3 are first analysed separately.

3.4.1 Uniqueness of the solution (f_1)

In this subsection only the first part of the given upper level objective function is considered:

$$\begin{aligned} &\min_D f_1 = \min_D \sum_{rs \in RS} \left(D_{rs} - D^0_{rs}\right)^2 \\ &\text{s.t. } D \geq 0 \\ &\chi_j = \begin{cases} 1 & \text{then} \quad \sum\limits_{p \in P_j} \psi^{rs}_p D^k_{rs} \prod\limits_{ij \in \tilde{I}\tilde{J}_{jp}} \alpha_{ij}(D) \geq C_j, \\ 0 & \text{then} \quad \sum\limits_{p \in P_j} \psi^{rs}_p D^k_{rs} \prod\limits_{ij \in \tilde{I}\tilde{J}_{jp}} \alpha_{ij}(D) \leq C_j, \end{cases} \qquad \forall j \in \tilde{J}. \end{aligned} \tag{3.42}$$

This problem is obtained when w_1 is set to one and w_2 and w_3 are set to zero in the given matrix estimation problem (3.1). It is analysed whether this

minimization problem has a unique solution. The given objective function f_1 is a strictly convex function (see appendix C). It can be easily seen that if the OD-demands are set equal to the prior OD-demands, so if D equals D^0, the function f_1 has a minimum function value of zero. Of course, this minimum function value can only be reached if the minimizer, D^0, is a feasible solution[16]. Since $f(D) > f(D^0)$ for all $D \neq D^0$, it can be concluded that if D^0 is feasible, then it is the unique solution to the given minimization problem. But also if the prior OD-matrix D^0 is not feasible, it is possible to find the minimum of the optimization problem in (3.42). However, in this case the corresponding solution is possibly non-unique. The given set of constraints is closed and continuous in D. Besides it can be seen that if $||D|| \to \infty$, then $f_1(D) \to \infty$. As such the minimum of the given optimization problem exists. But for a strictly convex objective function, only on a convex set the corresponding solution can be proven to be unique. The set of constraints in the given minimization problem is non-convex, because the traffic regime constraints are non-linear. So when the prior OD-matrix D^0 is not feasible, the solutions to (3.42) are possibly non-unique. Note that in this case the minimum function value is also bigger than zero.

3.4.2 Uniqueness of the solution (f_2)

In this subsection only the second part of the given upper level objective function is considered. The variant of STAQ-squeezing as considered within this study, is denoted by STAQ-squeezing*:

$$\begin{aligned} &\min_D \ f_2 = \min_D \sum_{a \in \tilde{A}} (y_a(D) - \tilde{y}_a)^2 \\ &\text{s.t. } y_a(D) = \text{STAQ-squeezing}^*(D) \\ &\quad D \geq 0 \\ &\quad \chi_j = \begin{cases} 1 \quad \text{then} \quad \sum_{p \in P_j} \psi_p^{rs} D_{rs}^k \prod_{ij \in \tilde{I}J_{jp}} \alpha_{ij}(D) \geq C_j, \\ 0 \quad \text{then} \quad \sum_{p \in P_j} \psi_p^{rs} D_{rs}^k \prod_{ij \in \tilde{I}J_{jp}} \alpha_{ij}(D) \leq C_j, \end{cases} \qquad \forall j \in \tilde{J}. \end{aligned} \tag{3.43}$$

This problem is obtained when w_2 is set to one and w_1 and w_3 are set to zero in the given matrix estimation problem (3.1). It is analysed whether this minimization problem has a unique solution. It can be easily seen that when the flow on each link equals the count value on that link, then the minimum function value of zero is reached. So only if there exists a feasible OD-matrix D for which $y_a(D) = \tilde{y}_a$ for all $a \in \tilde{A}$, the minimum function value zero is attained. However it is always possible to find an OD-matrix D which minimizes (3.43).

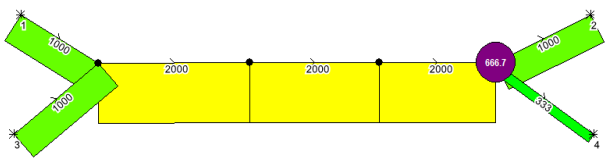


Figure 3.7: Link flows real situation.

[16]This feasibility should be checked with respect to the traffic regime constraints.

To gain some insight, first a case within the Dogbone network is analysed. The Dogbone network has been introduced in section (3.2.4). Consider in the Dogbone network the following OD-demands:

$$D_{\text{real}} = \begin{bmatrix} 750 & 250 \\ 750 & 250 \end{bmatrix}. \tag{3.44}$$

In figure 3.7 the link flows in the Dogbone network corresponding to these OD-demands are shown, as generated by the assignment model STAQ-squeezing*. It can be seen that only link 3 is actively constraining flow. Hence the traffic regime in this situation is $\chi = \{1, 0, 0, 0\}$. Now suppose, that the link flows as shown in figure 3.7 are available as count information. For this case in the Dogbone network, the considered matrix estimation problem (3.43) gets the following form:

$$\begin{aligned} \min_{D} \; & (y_4(D) \; - 1000)^2 + (y_{10}(D) - 1000)^2 + (y_9(D) - 2000)^2 + \\ & (y_2(D) \; - 2000)^2 + (y_3(D) \; - 1000)^2 + (y_8(D) - 2000)^2 + \\ & (y_{11}(D) - \; 333)^2 \\ \text{s.t. } & y_a(D) = \text{STAQ-squeezing}^*(D) \\ & D \geq 0 \\ & D_{12} + D_{14} + D_{32} + D_{34} \leq 2000, \\ & D_{12} + D_{32} \geq 1000, \\ & D_{14} + D_{34} \leq 1000. \end{aligned} \tag{3.45}$$

By construction, the OD-matrix D_{real} is a minimizer to (3.45). Assigning D_{real} to the Dogbone network using STAQ-squeezing* results in link flows which perfectly match with the available counts. Furthermore, because the matched counts perfectly reflect the given traffic regime $\chi = \{1, 0, 0, 0\}$, also the traffic regime constraints and the non-negativity constraints are satisfied. The question is if the matrix D_{real} is a unique minimizer, or if there exist more solutions to the given minimization problem (3.45). Note that D_{real} results in an minimum objective function value of zero. So to minimize the squared differences between the link flows and the link counts, an OD-matrix D should satisfy the following relations:

$$\begin{aligned} y_4(D) &= D_{12} + D_{14} = 1000, \\ y_{10}(D) &= D_{32} + D_{34} = 1000, \\ y_9(D) &= y_8(D) = y_2(D) = D_{12} + D_{14} + D_{32} + D_{34} = 2000, \\ y_3(D) &= (D_{12} + D_{32}) * \alpha_{2,3} = 1000, \\ y_{11}(D) &= (D_{14} + D_{34}) * \alpha_{2,11} = 333. \end{aligned} \tag{3.46}$$

It can be seen that the first two equations directly give information about the sum of the OD-demands: $D_{12} + D_{14}$ and $D_{32} + D_{34}$, and from the last three equations, the sum of the OD-demands $D_{12} + D_{32}$ and the sum of the OD-demands $D_{14} + D_{34}$ can be deduced. To do so, note that the turns 2-3 and 2-11 form a diverge. So following the FIFO rule in the node model, it should hold that: $\alpha_{2,3} = \alpha_{2,11} = \alpha_2$. Using this rule and the last three equations, it can be found that $\alpha_2 = 0.667$, $D_{12} + D_{32} = 1500$ and $D_{14} + D_{34} = 500$. Hence, it can

be concluded that it is possible to deduce from the available count information the row sum of each row and the column sum of each column of an OD-matrix D that minimizes (3.45):

$$\mathbf{D} = \begin{array}{ccc} \mathbf{D_{12}} & \mathbf{D_{14}} & \textit{1000} \\ \mathbf{D_{32}} & \mathbf{D_{34}} & \textit{1000}. \\ \textit{1500} & \textit{500} & \end{array} \tag{3.47}$$

Note that there exist different OD-matrices D which satisfy these given row sums and the column sums. The matrix D_{real} is one of the possible solutions, but for example matrix $\hat{D}$ with $\hat{D}_{12} = 600$, $\hat{D}_{32} = 900$, $\hat{D}_{14} = 400$ and $\hat{D}_{34} = 100$ also suffices. When it is assumed that the OD-demands can only take integer values, then there are already 501 different solutions possible. This can be seen by setting one of the OD-demands to x and express the other OD-demands in x:

$$\mathbf{D} = \begin{array}{cc} (\mathbf{x + 500}) & (\mathbf{500 - x}) \\ (\mathbf{1000 - x}) & (\mathbf{x}) \\ \textit{1500} & \textit{500} \end{array} \begin{array}{c} \textit{1000} \\ \textit{1000}. \\ \end{array} \tag{3.48}$$

Because x can take values between 0 and 500, there are 501 different OD-matrices D possible. It can be seen that all 501 OD-matrices satisfy the traffic regime constraints as shown in (3.45). Therefore in this case all possible OD-matrices D are also feasible OD-matrices. This is not surprising; the counts which are matched by these solutions perfectly reflect the given real traffic regime $\chi = \{1, 0, 0, 0\}$. So it can be concluded that D_{real} is not a unique minimizer of (3.45).

In the considered example, counts are available on all links. In practice this will never be the case. When there are less counts available, the set of possible solutions is possibly larger. However note that not all these extra solutions are also feasible solutions with respect to the traffic regime constraints. Furthermore it is important to realize that the placement of the available counts is more important then the number of available counts for the amount information which can be deduced from the counts[17]. In this example the best case possible with 2 counts and the worst case possible with 6 counts give the same amount of information. Consider the same case as above. For the situation with two counts, most information can be deduced if these counts are positioned on link 4 and link 9. For the situation with six counts, least information can be deduced if these counts are positioned on link 4, 10, 9, 8, 2 and 3. This best case for two counts and worst case for six counts give the same amount of information about OD-matrix D. For both cases it can be deduced that the total OD-demand is 2000 and:

$$\mathbf{D} = \begin{array}{ccc} \mathbf{D_{12}} & \mathbf{D_{14}} & \textit{1000} \\ \mathbf{D_{32}} & \mathbf{D_{34}} & \textit{1000}. \\ ? & ? & \end{array} \tag{3.49}$$

A final remark is that counts are measurements, so in practice it is possible that counts are mutually inconsistent. In that case, the minimum function value of zero cannot be reached.

[17] The maximization of information with a minimum number of sensors is a well-known problem, because the placement of sensors is costly.

From the given example in the Dogbone network it can be concluded that the solutions to (3.43) are possibly non-unique. However the considered example within the Dogbone network turns out to be very specific. A similar uniqueness analysis is not always possible. Already for more congested situations in the Dogbone network, it is not possible to draw conclusions about the uniqueness of the solutions to the upper level optimization problem any more. To see this, note that to find all OD-matrices D for which the corresponding link flows perfectly match the given counts, the following problem is solved:

$$\begin{aligned} y(D) = B(D)P(D)D &= \tilde{y} \\ D &\geq 0 \end{aligned} \tag{3.50}$$

$y(D)$	vector of estimated link flows
$\boldsymbol{B}(D)$	crossing fraction matrix
$\boldsymbol{P}(D)$	route fraction matrix
D	vector of OD-demands
$\tilde{y}$	set of links with count information

For the example in the Dogbone network as discussed above, this system of equations reads[18]:

$$\begin{pmatrix} 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ \alpha_2(D) & 0 & \alpha_2(D) & 0 \\ 0 & \alpha_2(D) & 0 & \alpha_2(D) \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} D_{12} \\ D_{14} \\ D_{32} \\ D_{34} \end{pmatrix} = \begin{pmatrix} 1000 \\ 1000 \\ 2000 \\ 2000 \\ 2000 \\ 1000 \\ 333 \end{pmatrix} \tag{3.51}$$

And for a situation in the Dogbone network with three bottlenecks, this system or equations reads[19]:

$$\begin{pmatrix} \hat{\alpha}_4^1(D) & \hat{\alpha}_4^2(D) & 0 & 0 \\ 0 & 0 & \hat{\alpha}_{10}^3(D) & \hat{\alpha}_{10}^4(D) \\ \hat{\alpha}_9^1(D) & \hat{\alpha}_9^2(D) & \hat{\alpha}_9^3(D) & \hat{\alpha}_9^4(D) \\ \hat{\alpha}_8^1(D) & \hat{\alpha}_8^2(D) & \hat{\alpha}_8^3(D) & \hat{\alpha}_8^4(D) \\ \hat{\alpha}_2^1(D) & \hat{\alpha}_2^2(D) & \hat{\alpha}_2^3(D) & \hat{\alpha}_2^4(D) \\ \hat{\alpha}_3^1(D) & 0 & \hat{\alpha}_3^3(D) & 0 \\ 0 & \hat{\alpha}_{11}^2(D) & 0 & \hat{\alpha}_{11}^4(D) \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} D_{12} \\ D_{14} \\ D_{32} \\ D_{34} \end{pmatrix} = \begin{pmatrix} \tilde{y}_4 \\ \tilde{y}_{10} \\ \tilde{y}_9 \\ \tilde{y}_8 \\ \tilde{y}_2 \\ \tilde{y}_3 \\ \tilde{y}_{11} \end{pmatrix} \tag{3.52}$$

$$\begin{aligned} \hat{\alpha}_4^1(D),\ \hat{\alpha}_4^2(D) &= \alpha_{4,9}(D) \\ \hat{\alpha}_{10}^3(D),\ \hat{\alpha}_{10}^4(D) &= \alpha_{10,9}(D) \\ \hat{\alpha}_9^1(D),\ \hat{\alpha}_9^2(D),\ \hat{\alpha}_8^1(D),\ \hat{\alpha}_8^2(D),\ \hat{\alpha}_2^1(D),\ \hat{\alpha}_2^2(D) &= \alpha_{4,9}(D)\alpha_{9,8}(D) \\ \hat{\alpha}_9^3(D),\ \hat{\alpha}_9^4(D),\ \hat{\alpha}_8^3(D),\ \hat{\alpha}_8^4(D),\ \hat{\alpha}_2^3(D),\ \hat{\alpha}_2^4(D) &= \alpha_{10,9}(D)\alpha_{9,8}(D) \\ \hat{\alpha}_3^1(D),\ \hat{\alpha}_{11}^2(D) &= \alpha_{4,9}(D)\alpha_{9,8}(D)\alpha_2(D) \\ \hat{\alpha}_3^3(D),\ \hat{\alpha}_{11}^4(D) &= \alpha_{10,9}(D)\alpha_{9,8}(D)\alpha_2(D) \end{aligned}$$

[18] Note that indeed this system of equations is the same as the system of equations in (3.46).
[19] Recall from (3.5) that $\alpha_{2,3}(D) = \alpha_{2,11}(D) = \alpha_2(D)$.

The system of equations in (3.51) contains four variables, but only three independent equations. As such, as has been shown before, it can be concluded that the corresponding solution is non-unique. But for the system of equations in (3.52) it is not possible to determine the number of independent equations; The path based reduction factors are not known explicitly[20]. Therefore it is not possible to draw conclusions about the uniqueness of the solutions to the system of equations as given in (3.52).

Summarizing it can be concluded that it is difficult to draw apriori general conclusions about the uniqueness of the solutions to (3.43). However from the given example on the Dogbone network, it can be concluded that the solutions to (3.43) are possibly non-unique. In appendix A.2 the uniqueness of the solutions to f_2 for a traditional STA model are discussed. Note that STAQ reduces to a traditional STA model when the crossing fraction matrix equals the link path incidence matrix.

3.4.3 Uniqueness of the solution (f_3)

In this subsection the third part of the given upper level objective function is considered:

$$\begin{aligned}
&\min_D f_3 = \min_D \sum_{p\in\tilde{P}} (\tau_p(D) - \tilde{\tau}_p)^2 \\
&\text{s.t. } \tau_p(D) = \text{STAQ-squeezing}^*(D) \\
&\quad D \geq 0 \\
&\quad \chi_j = \begin{cases} 1 \quad \text{then} \quad \sum\limits_{p\in P_j} \psi_p^{rs} D_{rs}^k \prod\limits_{ij\in\tilde{IJ}_{jp}} \alpha_{ij}(D) \geq C_j, \\ 0 \quad \text{then} \quad \sum\limits_{p\in P_j} \psi_p^{rs} D_{rs}^k \prod\limits_{ij\in\tilde{IJ}_{jp}} \alpha_{ij}(D) \leq C_j, \end{cases} \qquad \forall j \in \tilde{J}.
\end{aligned} \tag{3.53}$$

This problem is obtained when w_3 is set to one and w_1 and w_2 are set to zero in the given matrix estimation problem (3.1). It is analysed whether this minimization problem has a unique solution. It can be seen that when the queueing delay on each path equals the measured queueing delay on that path, then the minimum function value of zero is reached. So only if there exists a feasible OD-matrix D for which $\tau_p(D) = \tilde{\tau}_p$ for all $a \in \tilde{A}$, the minimum function value zero is attained. However it is always possible to find an OD-matrix D which minimizes (3.53).

In section 4.4.1 it is explained how STAQ-squeezing* calculates the average queueing delays on a path. The following formula is given:

$$\tau_p(D) = \frac{T}{2}\left(\frac{1}{\prod\limits_{ij\in IJ_p} \alpha_{ij}(D)} - 1\right), \quad \forall p \in P. \tag{3.54}$$

Here T is the considered study period duration and IJ_p is the set of turns used by path p travelling from origin to destination. It can be seen that the

[20] Recall that the reduction factors are only known after a lower level assignment for the corresponding OD-vector D.

queueing delay on a path p is depending on the product of all reduction factors on that path. When the total reduction factor on a path p equals 1, there are no bottlenecks, hence there is no queueing delay on path p. But when the total reduction factor on path p has a value $\in (0, 1)$ then there are bottleneck(s) on path p in the network, and as such there is a queueing delay on path p. So it can be concluded that a measured average queueing delay $\tilde{\tau}_p$ gives information about the total reduction factor on a path p. As mentioned earlier, in practice not the average path queueing delays but the average travel times on a path are measured. The average queueing delay on a path can be easily determined from the average path travel time by subtracting the free-flow path travel time.

Again, to gain some insight, the given minimization problem is analysed on the Dogbone network. In the Dogbone network there exist four paths, so when the average travel times are known on all four paths, f_3 in (3.53) consists of four terms. In figure 3.3 it can be seen that the Dogbone network has four potentially blocking links. As such, there are five turns in the Dogbone network that can have a reduction factor which does not equal 1. This are: $\alpha_{4,9}$, $\alpha_{10,9}$, $\alpha_{9,8}$, $\alpha_{2,3}$ and $\alpha_{2,11}$. Note that, because of the FIFO-rule (see equation (3.5)), it holds that $\alpha_{2,3} = \alpha_{2,11} = \alpha_2$. Therefore, using (3.54), for the Dogbone network the following formulas for the path queueing delays can be derived:

$$\begin{aligned}
\tau_1(D) &= \frac{T}{2}\left(\frac{1}{\alpha_{4,9}(D)\alpha_{9,8}(D)\alpha_2(D)} - 1\right),\\
\tau_2(D) &= \frac{T}{2}\left(\frac{1}{\alpha_{4,9}(D)\alpha_{9,8}(D)\alpha_2(D)} - 1\right),\\
\tau_3(D) &= \frac{T}{2}\left(\frac{1}{\alpha_{10,9}(D)\alpha_{9,8}(D)\alpha_2(D)} - 1\right),\\
\tau_4(D) &= \frac{T}{2}\left(\frac{1}{\alpha_{10,9}(D)\alpha_{9,8}(D)\alpha_2(D)} - 1\right).
\end{aligned} \tag{3.55}$$

To minimize the upper level problem in (3.53), an OD-matrix D is sought for, such that $\tau_p(D) = \tilde{\tau}_p$ for all paths with measurements. So when there are measurements available on all paths in the Dogbone network, using (3.55), the following information can be deduced for the OD-matrix D in the Dogbone network that minimizes (3.53):

$$\begin{aligned}
\alpha_{4,9}(D)\alpha_{9,8}(D)\alpha_2(D) &= \frac{T}{2\tilde{\tau}_1 + T},\\
\alpha_{4,9}(D)\alpha_{9,8}(D)\alpha_2(D) &= \frac{T}{2\tilde{\tau}_2 + T},\\
\alpha_{10,9}(D)\alpha_{9,8}(D)\alpha_2(D) &= \frac{T}{2\tilde{\tau}_3 + T},\\
\alpha_{10,9}(D)\alpha_{9,8}(D)\alpha_2(D) &= \frac{T}{2\tilde{\tau}_4 + T}.
\end{aligned} \tag{3.56}$$

Note that the same information can be deduced from a measurement on path 1 as from a measurement on path 2. The same holds for path 3 and 4. And if $\alpha_{4,9}(D) = \alpha_{10,9}(D)$, all measurements give the same information.

Now an example case in the Dogbone network is studied. Suppose that the considered study period in the Dogbone network is one hour and the measured queueing delays on the paths in the Dogbone network all equal fifteen minutes. Furthermore the traffic regime is assumed to be $\chi = \{1, 0, 0, 0\}$. Then the considered matrix estimation problem (3.53) gets the following form:

$$\begin{aligned} \min_{D} \; & (\tau_1(D) - 0.25)^2 + (\tau_2(D) - 0.25)^2 + (\tau_3(D) - 0.25)^2 + \\ & (\tau_4(D) - 0.25)^2 \\ \text{s.t. } & \tau_p(D) = \text{STAQ-squeezing}^*(D) \\ & D \geq 0 \\ & D_{12} + D_{14} + D_{32} + D_{34} \leq 2000, \\ & D_{12} + D_{32} \geq 1000, \\ & D_{14} + D_{34} \leq 1000. \end{aligned} \tag{3.57}$$

Using (3.56), for this case in the Dogbone network the following information can be deduced:

$$\begin{aligned} \alpha_{4,9}(D)\alpha_{9,8}(D)\alpha_2(D) &= \frac{2}{3} \approx 0,667, \\ \alpha_{10,9}(D)\alpha_{9,8}(D)\alpha_2(D) &= \frac{2}{3} \approx 0,667. \end{aligned} \tag{3.58}$$

And from the given traffic regime it can be seen that in this case $\alpha_{4,9}(D) = \alpha_{10,9}(D) = \alpha_{9,8}(D) = 1$. As such, for the given traffic regime (3.58) reduces to:

$$\alpha_2(D) = \frac{2}{3} \approx 0,667. \tag{3.59}$$

So it can be seen that for the given traffic regime (3.58) results in a relative simple set of equations[21]; There are no products of turn based reduction factors left. It can be shown that there exist different OD-matrices D which minimize the matrix estimation problem as given in (3.57). To do so, note that the link flows as shown in figure 3.7 result in reduction factors which satisfy (3.59) and comply with the given traffic regime. Hence it can be concluded that the OD-matrix D_{real} minimizes (3.57). But as discussed in section 3.4.2, there are 501 other OD-matrices which result in the same flows on the Dogbone network. As such, all these 501 different OD-matrices minimize the given matrix estimation problem (3.57). Therefore it can be concluded that also the solutions to the upper level minimization problem as given in (3.53) are possibly non-unique.

From the considered example in the Dogbone network, it can be seen that solving the upper level minimization problem in (3.57) is related to solving the upper level minimization problem in (3.45); The flows on all links in a network implicitly imply the travel times and hence queueing delays (expressed in reduction factors) on the paths in this network. Furthermore note that if we consider an objective function in which both f_2 and f_3 are included, then the measured path queueing delays add information to the problem as defined in (3.50); The given products of reduction factors (partly) specify the unknown crossing fraction matrix.

[21] For example for the traffic regime $\chi = \{1, 0, 1, 1\}$, the set of equations remains as in (3.58).

3.4.4 Conclusion

From the previous three sections it can be concluded that if we use exclusively one of the three parts of the upper level objective function in (3.1) only the solution to the first part is possibly unique. When the prior OD-matrix D^0 satisfies the traffic regime constraints, this OD-matrix is the unique solution to f_1. For the second part of the upper level objective function, it turns out to be difficult to draw general conclusions on the uniqueness of the solutions. But the given example in the Dogbone network shows that the solutions to f_2 are possibly non-unique. The same can be concluded for the solutions to the third part of the objective function; Also the solutions to f_3 are possibly non-unique. Therefore in practice it is required to put always some weight on the first part of the upper level objective function f_1. However note that this will not always guarantee a unique solution.

Chapter 4

Solution method

In chapter 3 the matrix estimation problem for STAQ-squeezing with fixed route choice and no junction modelling as considered within this study has been formulated and discussed. In this chapter it is described in detail how this problem is solved, using the matrix estimation method for STAQ as developed by Brederode et al. [5].

> Analysing the proposed matrix estimation method [5] resulted in some remarks, which are given in a gray text box. Those remarks are important in the further development of the given matrix estimation method.

4.1 General idea

Below the matrix estimation problem as considered within this study is shown[1]. Again the considered STAQ variant, STAQ-squeezing with fixed route choice and no junction modelling, is denoted by STAQ-squeezing*.

$$\begin{aligned} \min_D F(D) &= \min_D w_1 f_1(D, D^0) + w_2 f_2(y, \tilde{y}) + w_3 f_3(\tau, \tilde{\tau}) \\ \text{s.t. } & y, \tau = \text{STAQ-squeezing}^*(D) \\ & D \geq 0 \\ & \chi_j = \begin{cases} 1 \quad \text{then} \quad \sum_{p \in P_j} \psi_p^{rs} D_{rs} \prod_{ij \in \tilde{I}\tilde{J}_{jp}} \alpha_{ij}(D) \geq C_j, \\ 0 \quad \text{then} \quad \sum_{p \in P_j} \psi_p^{rs} D_{rs} \prod_{ij \in \tilde{I}\tilde{J}_{jp}} \alpha_{ij}(D) \leq C_j, \end{cases} \qquad \forall j \in \tilde{J}. \end{aligned} \tag{4.1}$$

It can be seen that the link flows y and the path queueing delays τ in the upper level minimization problem are depending on the OD-vector D. The lower level assignment model STAQ-squeezing* can calculate for every possible OD-vector D the flows y_a on all links a and the path queueing delays τ_p on all paths p. But considering STAQ, it is not possible to determine explicit relationships $y(D)$

[1] This problem has been defined in chapter 3. See equation (3.1) for the matrix estimation problem and equation (3.41) for the corresponding traffic regime constraints.

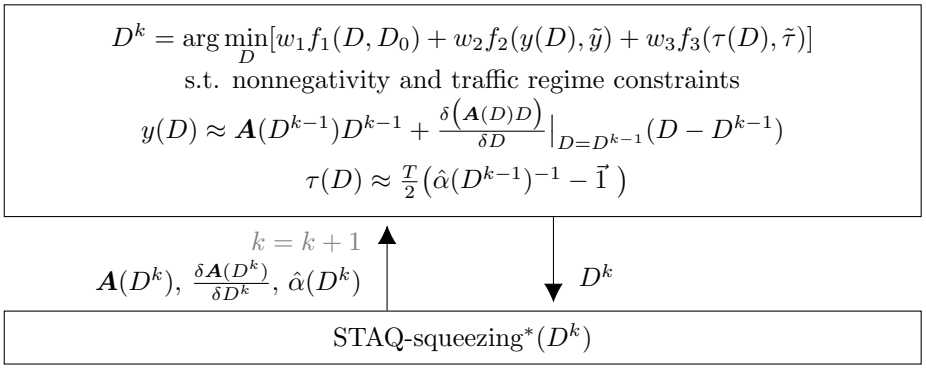


Figure 4.1: Proposed solution approach for STAQ.

and $\tau(D)$. As such, the given matrix estimation problem for STAQ-squeezing* is a bi-level optimization problem. The upper level minimization problem is constrained by the lower level assignment problem. In chapter 2 it has been described, that to solve this bi-level matrix estimation problem, Brederode et al [5] propose a heuristic algorithm that iteratively assigns the OD-vector from the upper level into the lower level and then solves the upper level optimization problem using information from the lower level to approximate the relationship between the OD-demands and the link flows $y(D)$ and the relationship between the OD-demands and the path queueing delays $\tau(D)$. This solution approach is shown in figure 4.1. By approximating the link flows and path queueing delays in the upper level problem, less lower level assignments are required, which possibly speeds up the optimization process.

Remark 1: But note that the approximated link flows and path queueing delays might differ significantly from the actual link flows and path queueing delays as can be calculated by the lower level assignment model STAQ-squeezing*. Therefore, it could be better to perform only one or a few steps within the upper level optimization problem (instead of solving it to optimality) before a new lower level assignment is performed[a].

[a] Another possibility is to put more weight on the prior part of the objective function.

Recall from section 2.1 that the proposed solution approach as shown in figure 4.1 is adopted from the conventional matrix estimation method for traditional STA models [8]. However, within the conventional matrix estimation problem no path queueing delays[2] and no traffic regime constraints are considered. And more importantly, Brederode et al. [5] have introduced a different way of approximating the link flows in the upper level optimization problem. Not only the assignment matrix, but also the sensitivity of the assignment matrix to changes in the OD-demands is taken into account in the proposed first order Taylor approximation of the link flows.

[2] The calculation and approximation of the queueing delays are discussed in section 4.4.

In the next section the upper level approximation of the link flows is explained in detail. Subsequently in section 4.3 it is described how the required sensitivity of the assignment matrix to changes in the OD-demands is determined from the lower level assignment model STAQ-squeezing*. Then in section 4.4 the calculation and the upper level approximation of the path queueing delays are discussed. And finally in section 4.5 it is explained how the traffic regime constraints are taken into account in the given upper level optimization problem.

4.2 Approximation of the link flows

In this section the first order Taylor approximations of the link flows as considered within this study are explained in detail. Besides an example is given to show why the conventional method to approximate the link flows is not suitable for STAQ.

4.2.1 Assignment matrix

To approximate the link flows Brederode et al. [5] propose a first order Taylor approximation around the previous upper level solution (see subsection 2.1.2). Note that this previous solution vector has been assigned to the network in the lower level. As such the link flows corresponding to this OD-matrix are known. To formulate the considered first order Taylor approximations, first an expression for the relationship between an assigned OD-vector and the resulting link flows is deduced, considering the lower level assignment model STAQ-squeezing*. Recall from section 1.3 that this relationship can be described by an assignment matrix, which can be subdivided in a crossing fraction and a route fraction matrix. For the assignment model STAQ, both these matrices are normally depending on the corresponding OD-matrix D. However within this study a fixed route choice and hence a fixed route fraction matrix is assumed. So considering STAQ-squeezing* it holds that:

$$y(D) = \boldsymbol{A}(D)D = \boldsymbol{B}(D)\boldsymbol{P}D. \tag{4.2}$$

D	vector of OD-demands, $\lvert R \times S \rvert$
$\boldsymbol{A}(D)$	assignment matrix, $\lvert \tilde{A} \rvert \times \lvert R \times S \rvert$
$\boldsymbol{B}(D)$	crossing fraction matrix, $\lvert \tilde{A} \rvert \times \lvert P \rvert$
$\boldsymbol{P}$	route fraction matrix, $\lvert P \rvert \times \lvert R \times S \rvert$

The elements of the crossing fraction matrix express the proportion of a route flow that passes a link. For STAQ-squeezing* these elements equal the path based reduction factors $\hat{\alpha}_a^p$, which can be determined from the turn based reduction factors as calculated by the network loading submodel of STAQ-squeezing* (3.10). And the elements of the route fraction matrix express the proportion of an OD-flow choosing a certain route. So this are the fixed route fractions ψ_p^{rs}, as determined by the route choice submodel (3.34). It can be concluded that for STAQ-squeezing* the elements of the assignment matrix $\boldsymbol{A}(D)$ can be described as follows:

$$A_a^{rs}(D) = \sum_{p \in P_a} \hat{\alpha}_a^p(D)\psi_p^{rs}. \tag{4.3}$$

$A_a^{rs}(D)$	fraction of demand from OD-pair rs that flows over link a
ψ_p^{rs}	fraction of demand from OD-pair rs that uses path p
$\hat{\alpha}_{ap}(D)$	reduction factor on path p till link a
$P_a \subseteq P$	set of paths that use link a

From (4.2) and (4.3) and (3.35) it can be seen that considering STAQ-squeezing* the flow on a link a is defined as follows[3]:

$$\begin{aligned} y_a(D) &= \sum_{rs \in RS} \sum_{p \in P_a} \hat{\alpha}_a^p(D) \psi_p^{rs} D_{rs}, \\ &= \sum_{p \in P_a} \hat{\alpha}_a^p(D) Q_p. \end{aligned} \tag{4.4}$$

$y_a(D)$	flow on link a for OD-vector D^* as calculated by STAQ-squeezing*
ψ_p^{rs}	fraction of demand from OD-pair rs that use path p
$\hat{\alpha}_{ap}(D)$	reduction factor on path p till link a for OD-vector D
Q_p	demand on path p for OD-vector D

So for an OD-vector assigned in the lower level the corresponding flows on the links can be calculated from the path based reduction factors as determined by the lower level assignment model STAQ-squeezing* and the fixed route fractions.

4.2.2 Approximation of the link flows

Given equation (4.4) the first order Taylor approximation of the link flows as considered within this study can be explicitly formulated. A first order Taylor approximation of the flow on link a for OD-vector D around the previous upper level solution D^{k-1} is defined as follows:

$$\begin{aligned} &y_a(D) = y_a(D^{k-1}) + y_a'(D^{k-1})(D - D^{k-1}) + o\big(||D - D^{k-1}||\big), \\ &y_a(D) \approx y_a(D^{k-1}) + y_a'(D^{k-1})(D - D^{k-1}). \end{aligned}$$

Where using (4.4) gives:

$$\begin{aligned} y_a(D) &\approx \sum_{rs \in RS} \sum_{p \in P_a} \hat{\alpha}_a^p(D^{k-1}) \psi_p^{rs} D_{rs}^{k-1} + \sum_{rs \in RS} \frac{\delta y_a(D)}{\delta D_{rs}}\bigg|_{D=D^{k-1}} (D_{rs} - D_{rs}^{k-1}), \\ &= \sum_{rs \in RS} \sum_{p \in P_a} \hat{\alpha}_a^p(D^{k-1}) \psi_p^{rs} D_{rs}^{k-1} \\ &\quad + \sum_{rs \in RS} \frac{\delta\Big(\sum_{rs' \in RS} \sum_{p \in P_a} \hat{\alpha}_a^p(D) \psi_p^{rs'} D_{rs'}\Big)}{\delta D_{rs}}\bigg|_{D=D^{k-1}} (D_{rs} - D_{rs}^{k-1}), \\ &= \sum_{rs \in RS} \sum_{p \in P_a} \hat{\alpha}_a^p(D^{k-1}) \psi_p^{rs} D_{rs}^{k-1} \\ &\quad + \sum_{rs \in RS} \Big(\sum_{rs' \in RS} \sum_{p \in P_a} \frac{\delta \hat{\alpha}_a^p(D)}{\delta D_{rs}}\bigg|_{D=D^{k-1}} \psi_p^{rs'} D_{rs'}^{k-1} \Big)(D_{rs} - D_{rs}^{k-1}) \\ &\quad + \sum_{rs \in RS} \Big(\sum_{p \in P_a} \hat{\alpha}_a^p(D^{k-1}) \psi_p^{rs} \Big)(D_{rs} - D_{rs}^{k-1}), \end{aligned}$$

[3] Note that (4.4) can also be deduced from the definitions of link flow (3.9), turn flow (3.7) and path demand (3.35) as given in subsection 3.2.1.

$$
\begin{aligned}
&= \sum_{rs\in RS}\sum_{p\in P_a} \hat{\alpha}_a^p(D^{k-1})\psi_p^{rs} D_{rs}^{k-1} + \sum_{rs\in RS}\sum_{p\in P_a} \hat{\alpha}_a^p(D^{k-1})\psi_p^{rs}(D_{rs} - D_{rs}^{k-1}) \\
&\quad + \sum_{rs\in RS}\Big(\sum_{rs'\in RS}\sum_{p\in P_a} \frac{\delta\hat{\alpha}_a^p(D)}{\delta D_{rs}}\Big|_{D=D^{k-1}} \psi_p^{rs'} D_{rs'}^{k-1}\Big)(D_{rs} - D_{rs}^{k-1}), \\
&= \sum_{rs\in RS}\sum_{p\in P_a} \hat{\alpha}_a^p(D^{k-1})\psi_p^{rs} D_{rs} \\
&\quad + \sum_{rs\in RS}\Big(\sum_{rs'\in RS}\sum_{p\in P_a} \frac{\delta\hat{\alpha}_a^p(D)}{\delta D_{rs}}\Big|_{D=D^{k-1}} \psi_p^{rs'} D_{rs'}^{k-1}\Big)(D_{rs} - D_{rs}^{k-1}). \qquad (4.5)
\end{aligned}
$$

This derivation of this first order Taylor approximation is adopted from Frederix et al. [8]. It can be seen that to approximate the link flows using (4.5), the previous upper level solution, the path based reduction factors[4] and the partial derivatives of the path based reduction factors to the OD-demands[4] are required. In equation (3.10) it is shown that these path based reduction factors can be easily determined from the turn based reduction factors as given by the network loading submodel of STAQ-squeezing*. More difficult is the approximation of the sensitivity of the path based reduction factors to the OD-demands. How these sensitivities are determined within the lower level assignment model is described in section 4.3.

Remark 2: Note that the approximation of the link flows as given in (4.5) is defined for STAQ-squeezing*. When STAQ-squeezing with a variable route choice is considered, which is a future goal, then the route fractions are not fixed, but depending on the given OD-matrix D:

$$y_a(D) = \sum_{rs\in RS}\sum_{p\in P_a} \hat{\alpha}_a^p(D)\psi_p^{rs}(D)D_{rs}.$$

So in this case, within the partial derivatives of the link flow to the OD-demands also the sensitivity of the route fractions to the OD-demands should be taken into account. It should be investigated if it is possible to approximate these sensitivities within a feasible computation time[a].

[a] From a practical point of view it could be studied if it is really needed to include sensitivities of the route fractions.

The difference between the approximation of the link flows as used in conventional matrix estimation methods for traditional STA models (see section 2.1.1) and the first order Taylor approximation of the link flows as given in (4.5) is the second term in equation (4.5). This term incorporates the sensitivity of the assignment matrix to changes in the OD-demands. Using the first order Taylor approximation of the link flows, the first order response of the lower level problem is taken into account in the upper level problem [8]. Note that secondary and higher order interaction effects are omitted. As such there is not accounted for the fact that when simultaneously changing multiple elements in the OD-vector D, the effect on the assignment matrix might not be simply the sum of the effects of changing D sequentially per OD-pair [5].

[4] As calculated in the previous lower level assignment STAQ-squeezing*(D^{k-1}).

4.2.3 Corridor example

In this subsection it is shown by example why the conventional method to approximate the link flows is not suitable for STAQ. Consider the corridor network in figure 4.2. The link capacities are given above the links. There is only one count location, on link n. Now suppose that $D_{rs}^{k-1} = 160$ veh/h is the previous upper level solution.

Figure 4.2: Corridor network.

Assigning this OD-matrix[5] to the given network in the lower level using STAQ-squeezing*, results in the reduction factors $\alpha_l(D^{k-1}) = 0.5$ and $\alpha_m(D^{k-1}) = 1$. Given these reduction factors, the flow on link n corresponding to the solution D^{k-1} can be calculated from equation (4.4):

$$\begin{aligned} y_n(D^{k-1}) &= \hat{\alpha}_n^p(D^{k-1})\psi_p^{rs}D_{rs}^{k-1}, \\ &= \alpha_l(D^{k-1})\alpha_m(D^{k-1})\psi_p^{rs}D_{rs}^{k-1}, \\ &= 0.5 * 1 * 1 * 160 \text{ veh/h} = 80 \text{ veh/h}. \end{aligned} \tag{4.6}$$

where the second equality follows from equation (3.10). So it can be seen that in iteration k-1 the flow on link n does not yet match the given count value of 100 veh/h. Therefore in the next iteration, in iteration k, there is sought for a more suitable OD-matrix D^k. Using the conventional way to approximate the link flows as given in equation (2.2), in the upper level optimization problem there is sought for a feasible OD-matrix for which it holds that:

$$\begin{aligned} y_n(D^k) \approx \hat{\alpha}_n^p(D^{k-1})\psi_p^{rs}D_{rs}^k &= \tilde{y}_n, \\ 0.5 * 1 * D_{rs}^k &= 100 \text{ veh/h}. \end{aligned} \tag{4.7}$$

Note that $D_{rs}^k = 200$ veh/h, solves the problem as given in (4.7). However, when this solution to the given upper level problem is assigned to the network using STAQ-squeezing*, it turns out that the corresponding actual reduction factor, $\alpha_l(D^k)$ equals 0.4. As such, the actual flow on link n corresponding to D^k equals:

$$\begin{aligned} y_n(D^k) &= \hat{\alpha}_n^p(D^k)\psi_p^{rs}D_{rs}^k, \\ &= 0.4 * 1 * 200 \text{ veh/h} = 80 \text{ veh/h}. \end{aligned} \tag{4.8}$$

So increasing the demand on OD-pair D_{rs}, has no effect on the flow on link n; Only the reduction factor on the turn from inlink l to outlink m changes. Considering STAQ-squeezing* the flow on link m will never exceed the capacity of 80 veh/h. Hence for STAQ-squeezing* it is not possible to match the given count on link n. But using the conventional approximation (4.7) of the link flows,

[5]Note that in this simple network the OD-matrix consists of only one OD-pair.

it seems as if this count value can be reached. This is because this approximation does not account appropriately for capacity constraints. The first order Taylor approximation of the link flows better accounts for capacity constraints; The sensitivity of the assignment matrix to changes in the OD-demands is taken into account. Using this approximation, the approximated flows on link m and n do not exceed the bottleneck capacity of 80 veh/h. So considering capacity constraints, it is more suitable to approximate the link flows using a first order Taylor approximation[6]. Note that it is not very realistic to place a count location after a bottleneck. It can be easily seen that this count value cannot be reached. This simple example is only used for illustrative purposes. But note that adding one extra inlink to outlink n makes the situation already realistic.

4.3 Lower level information

In subsection 4.2.2 it has been shown that to calculate the first order Taylor approximations of the link flows, the partial derivatives of the path based reduction factors to the OD-demands[4] are required (4.5). Note that actually the sensitivity of the assignment matrix to changes in the OD-demands is required. Given the definition of the assignment matrix for STAQ-squeezing* (4.3) it can be deduced that:

$$\begin{aligned} \frac{\delta A_a^{rs}(D)}{\delta D_{rs'}} &= \frac{\delta}{\delta D_{rs'}} \left(\sum_{p \in P_a} \psi_p^{rs} \hat{\alpha}_a^p(D) \right), \\ &= \sum_{p \in P_a} \psi_p^{rs} \frac{\delta \hat{\alpha}_a^p(D)}{\delta D_{rs'}}. \end{aligned} \tag{4.9}$$

D_{rs}	demand on OD-pair rs
$A_a^{rs}(D)$	fraction of demand from OD-pair rs that flows over link a
ψ_p^{rs}	fraction of demand from OD-pair rs that uses path p
$\hat{\alpha}_a^p(D)$	reduction factor on path p till link a
$P_a \subseteq P$	set of paths that use link a

This expression can be recognized in equation (4.5). So in this study the sensitivity of the assignment matrix to changes in the OD-demands can be calculated given the partial derivatives of the path based reduction factors to the OD-demands and the fixed route fractions. In this section it is described how within the developed matrix estimation method for STAQ-squeezing* the sensitivities of the partial derivatives of the path based reduction factors to the OD-demands are determined from the lower level assignment model.

4.3.1 Sensitivity of the path based reduction factors

Recall from subsection 2.1.2 that it is not possible to explicitly determine the sensitivity of the assignment matrix to changes in the OD-demands. It is only possible to approximate this sensitivity using finite differences by performing $|RS|$ times a lower level assignment. But because this leads to infeasible computation

[6] Recall from subsection 2.1.2 that even for traditional STA models, a first order Taylor approximation of the link flows is theoretically more sound [8].

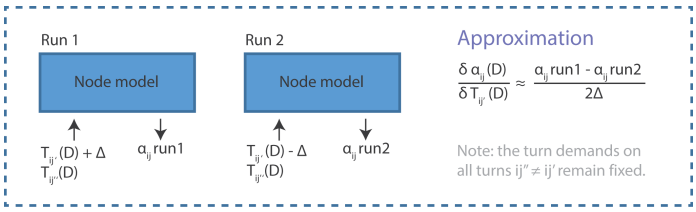


Figure 4.3: Approximation of the path based reduction factors to the turn demands using the node model of STAQ-squeezing*.

times Brederode et al. [5] propose to use the node-model of STAQ-squeezing* to approximate the required partial derivatives of the path based reduction factors to the OD-demands. In figure 4.3 the corresponding solution approach is shown. A point derivative of any reduction factor $\alpha_{ij}(D)$ to any turn demand $T_{ij'}(D)$ is approximated by running the node model twice around the current value of this turn demand $T_{ij'}(D)$: $\forall ij,\ ij'$ on node $n, \forall n \in N$

$$\frac{\delta \alpha_{ij}(D)}{\delta T_{ij'}(D)} \approx \frac{\alpha_{ij}\text{run 1} - \alpha_{ij}\text{run 2}}{2\Delta}. \tag{4.10}$$

$\alpha_{ij}(D)$	reduction factor from inlink i to outlink j
$T_{ij'}(D)$	turn demand from inlink i' to outlink j'
α_{ij}run 1	reduction factor approximated as shown in figure 4.3
α_{ij}run 2	reduction factor approximated as shown in figure 4.3
Δ	finite difference step size

Note that it is assumed that the demands on all other turns $ij'' \neq ij'$ on the considered node n remain fixed. From these sensitivities of the turn based reduction factors to changes in the turn demands, the sensitivities of path based reduction factors to changes in the OD-demands are approximated in a few intermediate steps.

Remark 3: Mathematically seen, the derivative of a reduction factor to a turn demand is not defined; The reduction factors and the turn demands are both depending on the OD-vector D. Besides it is not possible to express the reduction factors as an explicit function of the turn demands. As such this derivative only exists for an OD-vector D of only one OD-pair. See appendix B for a proof. Consequently using the proposed approximations might lead to significant errors in the model.

Given the derivatives of the turn based reduction factors to the turn demands (4.10), the derivatives of the turn based reduction factors to the path demands are determined as follows: $\forall ij$ on node n, $\forall p$ over node n, $\forall n \in N$

$$\frac{\delta \alpha_{ij}(D)}{\delta Q_p(D)} \approx \frac{\delta \alpha_{ij}(D)}{\delta T_{ij'}(D)} \frac{\delta T_{ij'}(D)}{\delta Q_p(D)} = \frac{\delta \alpha_{ij}(D)}{\delta T_{ij'}(D)} \hat{\alpha}_i^p(D). \tag{4.11}$$

$Q_p(D)$	path demand on path p
$\hat{\alpha}_i^p(D)$	reduction factor on path p till link i

Where turn ij' is part of path p and on the same node as turn ij. The equality in (4.11) follows from the definition of turn demand (3.6). Note that the partial derivatives from the turn based reduction factors to the path demands are only defined for paths over the node corresponding to the considered turn. In all other cases they are considered to be zero. From these derivatives of the turn based reduction factors to the path demands (4.11), the derivatives of the turn based reduction factors to the OD-demands are determined as follows: $\forall ij$ on node n, $\forall rs$ with one or more paths $p \in P_{rs}$ over node n, $\forall n \in N$

$$\frac{\delta \alpha_{ij}(D)}{\delta D_{rs}} \approx \sum_{p \in P_{rs}} \frac{\delta \alpha_{ij}(D)}{\delta Q_p(D)} \frac{\delta Q_p(D)}{\delta D_{rs}} = \sum_{p \in P_{rs}} \frac{\delta \alpha_{ij}(D)}{\delta Q_p(D)} \psi_p^{rs}. \tag{4.12}$$

D_{rs}	demand on OD-pair rs
ψ_p^{rs}	fraction of demand from OD-pair rs that use path p
$P_{rs} \subseteq P$	set of paths between OD-pair rs

Here the equality in (4.12) follows from the definition of path demand (3.35). Note that the partial derivatives from the turn based reduction factors to the OD-demands are only defined for OD-pairs which have one or more paths over the node corresponding to the considered turn. In all other cases they are considered to be zero. Finally using the product rule, it can be seen that the derivatives of the path based reduction factors (3.10) to the OD-demands can be determined as follows: $\forall a \in \tilde{A},\ p \in P,\ rs \in RS$

$$\frac{\delta \hat{\alpha}_a^p(D)}{\delta D_{rs}} = \left(\prod_{ij \in IJ_{ap}} \alpha_{ij}(D) \right) \left(\sum_{ij \in IJ_{ap}} \frac{\delta \alpha_{ij}(D) / \delta D_{rs}}{\alpha_{ij}(D)} \right). \tag{4.13}$$

$\hat{\alpha}_a^p(D)$	reduction factor on path p till link a for OD-vector D
$\alpha_{ij}(D)$	reduction factor from inlink i to outlink j for OD-vector D
IJ_{ap}	set of turns used by path p travelling from origin to link a

It should be mentioned that (4.11) and (4.12) are not described in [5], but are deduced from the given Matlab model of the proposed matrix estimation method for STAQ.

4.3.2 Sensitivity of the turn based reduction factors

In this section the sensitivities of the turn based reduction factors to the turn demands as approximated by Brederode et al. [5] (see figure 4.3) are further analysed[7]. By varying the initial turn demand[4] on one turn $T_{ij'}$, while keeping all other turn demands fixed on their initial values, alpha-graphs can be determined which show for a node n the relation between the demand on the turn $T_{ij'}$ and the reduction factors α_n on node n. Such alpha-graphs can be calculated for all turn demands $T_{ij'}$ on a node and for all nodes n in the network. Each alpha-graph shows the sensitivity of a specific turn based reduction factor α_{ij} to the changes in a specific turn demand $T_{ij'}$. Recall that it is assumed that the turn demands on all other turns on the considered node remain fixed.

[7] So in this section remark 3 is neglected.

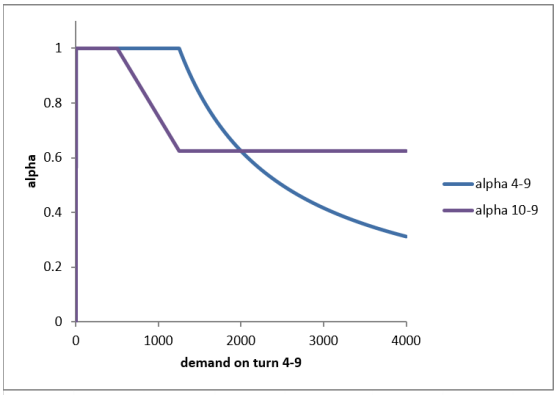


Figure 4.4: Sensitivity of the reduction factors on the first node in the Dogbone network to changes in $T_{4,9}$. Initially $T_{4,9} = T_{10,9} = 2000$.

An example is shown in figure 4.4. In this example the first node within the Dogbone network[8] is considered. This first node has two inlinks, link 4 and link 10, and one outlink, link 9 with a capacity of 2500. The initial turn demands in this example are assumed to be $T_{4,9} = T_{10,9} = 2000$. To determine the sensitivity of the reduction factors $\alpha_{4,9}$ and $\alpha_{10,9}$ to changes in the turn demand $T_{4,9}$, the node model is run for all different possible values of $T_{4,9}$. During all runs the value of $T_{10,9}$ is fixed on its initial value of 2000. Because the capacity of inlink 4 is 4000, the range of the resulting alpha-graphs is $[0, 4000]$.

The form of the alpha-graphs can be explained following the node model solution algorithm (see algorithm 2). It can be seen that both inlinks on the considered node have a directional capacity of 4000, hence both turns have right on half of the available supply of outlink 9, which equals 1250. For $0 \leq T_{4,9} \leq 500$, outlink 9 is not constraining any inflow. So both turns are demand constrained and both reduction factors equal 1. For $501 \leq T_{4,9} \leq 1250$ outlink 9 is constraining the inflow of inlink 10. So the turn 4-9 is demand constrained whereas the turn 10-9 is supply constrained. Inlink 4 claims less than its rightful share of 1250 of the available supply, hence the reduction factor for turn 4-9 equals 1. Inlink 10 obtains all remaining supply $t_{10,9} = 2500 - T_{4,9}$, which is more than its rightful share, but less then the turn demand $T_{10,9}$ of 2000. Therefore the reduction factor for turn 10-9 decreases linearly; Following (3.33) it holds that $\alpha_{10,9} = t_{10,9}/T_{10,9} = (2500 - T_{4,9})/2000$. Finally for $1251 \leq T_{4,9} \leq 4000$, outlink 9 is constraining the inflow of both inlink 4 and inlink 10. So both turns are supply constrained and both reduction factors are less then 1. The reduction factor on turn 4-9 decreases, because $T_{4,9}$ increases; It holds that $\alpha_{4,9} = 1250/T_{4,9}$, hence this reduction factor decreases non-linearly. The reduction factor on turn 10-9 remains $\alpha_{10,9} = 1250/2000 = 0.625$.

Brederode et al. [5] mention that in general, alpha-graphs $\alpha_{ij}(T_{ij'})$ are continuous on its positive domain, can be constructed piecewise and are differentiable almost

[8] See figure 3.3 in section 3.2.4.

everywhere. On each interval of $T_{ij'}$, $\alpha_{ij}(T_{ij'})$ is determined by the same supply and demand constraints and at each non-differentiable point, a switch between active constraints occurs. From this knowledge about the form of the alpha-graphs it can be concluded that the point derivatives which are used to approximate the sensitivity of the turn based reduction factors to the turn demands (4.10), only approximate the slope of the alpha-graph between two non-differential points correctly. Note that herein it is neglected that the point approximation is linear, whereas the alpha-graphs sometimes monotonously increase or decrease non-linearly between two non-differential points. It can be concluded that the point approximations as used in the developed matrix estimation method, are only representative within the turn demand interval in which the initial value of $T_{ij'}$ lays.

Remark 4: It can be concluded that the derivatives of the path based reduction factors to the OD-demands as used within the upper level optimization problem (4.13) are only sufficiently representative, when the OD-vector D in the upper level results in turn demands for which the active demand and supply constraints remain active. In that case the resulting turn demands stay between the same non-differential points of the alpha-graphs and as such the used approximations correctly reflect the corresponding slope of the alpha-graph[a].

[a]Note that when the route choice is considered to be variable, the links will be steered away from becoming (more) actively constraining. This might reduce the number of times the described problem occurs.

The traffic regime constraints solve this problem partially. When an outlink is in a free-flow traffic regime, the corresponding traffic regime constraint ensures that this outlink remains in a free-flow traffic regime and hence all its inlinks remain demand constrained. However when an outlink is in a congested traffic regime, the corresponding traffic regime constraint ensures only that this link remains in a congested traffic regime. It does not ensure that all the supply constrained inlinks remain supply constrained and that all the demand constrained inlinks remain demand constrained. So for turns to a congested outlink it is possible that the approximated point derivatives are not representative enough, because the turn demands corresponding to the OD-vector in the upper level jump over a non-differential point of the alpha-graph.

Consider the example on the first node of the Dogbone network (see figure 4.4). In this example the initial turn demands are assumed to be $T_{4,9} = T_{10,9} = 2000$. So the point derivatives of the turn based reduction factors $\alpha_{10,9}$ and $\alpha_{4,9}$ to changes in the path demand $T_{4,9}$ are determined around the point $T_{4,9} = 2000$, fixing $T_{10,9} = 2000$. Running the node-model twice around this point, results in the following point derivatives:

$$\frac{\delta\alpha_{4,9}(D)}{\delta T_{4,9}} \approx -0,000313 \quad \text{and} \quad \frac{\delta\alpha_{10,9}(D)}{\delta T_{4,9}} = 0. \tag{4.14}$$

Note that in the initial situation outlink 9 is in a congested traffic regime. So the traffic regime constraint corresponding to outlink 9 ensures that the OD-matrix

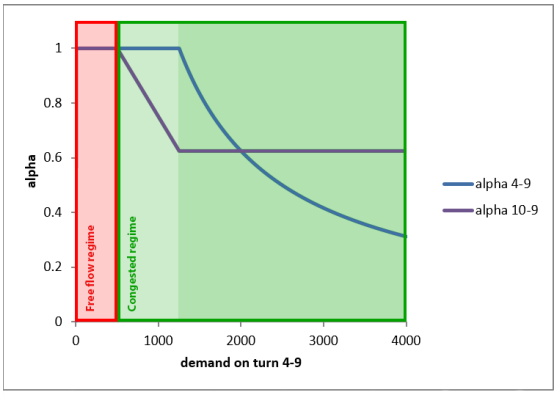


Figure 4.5: The traffic regimes in the alpha graphs for $T_{4,9}$ on the first node in the Dogbone network. Initially $T_{4,9} = T_{10,9} = 2000$.

in the upper level is chosen such that $T_{4,9} + T_{10,9} > 2500$. In figure 4.5 this corresponding area is coloured green. It can be seen that this congested area is split into two intervals by a non-differential point. In the left (light green) area, turn 4-9 is demand constrained and turn 10-9 is supply constrained. Here $501 \leq T_{4,9} \leq 1250$ and $T_{10,9} = 2000$. And in the right (dark green) area, the demand on both turns is supply constrained. This holds for $1251 \leq T_{4,9} \leq 4000$ and $T_{10,9} = 2000$. It can be seen that the point derivatives as given in (4.14) are only representative enough for OD-vectors D in the right (dark green) interval. However in the upper level also OD-vectors corresponding to the left interval can be chosen. For these OD-vectors the point derivatives as given in (4.14) are not sufficient.

Remark 5: It is not possible to formulate constraints on the turn demands in the upper level to ensure that active supply and demand constraints remain active; The relationship between the turn demands and the sets of demand constrained and supply constrained inlinks is determined by the node-model algorithm in the lower level assignment model STAQ-squeezing*. See equation (3.25) in section 3.2.2. Therefore it is not possible to define explicit turn demand constraints to prevent a jump over non differentiable points of the alpha-graphs.

4.4 Approximation of the path queueing delays

In the matrix estimation problem as considered in this study (4.1), not only the differences between the observed and estimated link flows, but also the differences between the observed and estimated path queueing delays are included in the upper level objective function. Recall that the average travel time on a path consists of the free-flow travel time and the queueing delay on that path. But because the free-flow travel time on a path is fixed, only the path queueing delays

are considered in the objective function. In the next subsection the calculation of the average path travel times and hence also the calculation of the average path queueing delays are explained for STAQ-squeezing*.

4.4.1 Calculation of the average path travel time

The derivation of the average path travel time[9] for STAQ-squeezing* as given in this subsection is adopted from Bliemer et al. [3]. For each path p the average path travel time c_p within a given demand period $[0, T]$ can be determined. This is the average travel time of the vehicles departing on path p during time interval $[0, T]$. The average path travel time c_p consists of two terms; It is the sum of the free-flow travel time on path p and the average path queueing delay on path p.

$$c_p(D) = t_p + \tau_p(D), \quad \forall p \in P. \tag{4.15}$$

$c_p(D)$	average travel time path p for OD-vector D
t_p	free-flow travel time path p
$\tau_p(D)$	average queueing delay path p for OD-vector D

The term t_p can be calculated by adding the free-flow travel times of all links in path p. The free-flow travel time of a link a is simply given by its length divided by its maximum speed.

$$c_p(D) = \sum_{a \in p} \frac{L_a}{v_a^{max}} + \tau_p(D), \quad \forall p \in P. \tag{4.16}$$

L_a	length (km) link a
v_a^{max}	maximum speed (km/h) link a

To derive an expression for the term $\tau_p(D)$, first the average queueing delay on link-level is determined. At time 0, there are zero vehicles in the vertical queue of a link a. Therefore the first vehicle departing in time period $[0, T]$ has a delay of zero. At time T, the number of vehicles in the vertical queue of a link a, equals the number of vehicles that have flown into the link, minus the number of vehicles that could exit the link:

$$B_a(D) = y_a(D)T - \alpha_a(D)y_a(D)T = \Big(1 - \alpha_a(D)\Big)y_a(D)T, \quad \forall a \in A. \tag{4.17}$$

From equation (4.4) it can be seen that for STAQ-squeezing* $y_a(D)$ can be written as:

$$y_a(D) = \sum_{p \in P_a} \hat{\alpha}_a^p(D) Q_p, \quad \forall a \in A. \tag{4.18}$$

$B_a(D)$	number of vehicles in the queue on link a at the end of $[0, T]$
$y_a(D)$	inflow (veh/h) on link a
T	time horizon
$\alpha_a(D)$	reduction factor of link a
P_a	set of paths that use link a
Q_p	demand (veh/h) path p
$\hat{\alpha}_a^p(D)$	path-based reduction factor for path p from origin till link a

[9] The travel time calculator within STAQ uses cumulative flow diagrams and not fundamental diagrams to calculate the average path travel times.

Now the delay of the last vehicle departing in time period $[0, T]$ on link a can be computed by dividing the number of vehicles in the queue on link a at time T, by the outflow rate of link a:

$$\frac{B_a(D)}{\alpha_a(D)y_a(D)} = \frac{\Big(1-\alpha_a(D)\Big)y_a(D)T}{\alpha_a(D)y_a(D)} = \frac{(1-\alpha_a(D))}{\alpha_a(D)}T, \quad \forall a \in A. \tag{4.19}$$

Hence the average queueing delay for all vehicles that have entered link a during time period $[0, T]$ is:

$$\frac{T}{2}\frac{\Big(1-\alpha_a(D)\Big)}{\alpha_a(D)}, \quad \forall a \in A. \tag{4.20}$$

Note that this average link queueing delay is based on the number of vehicles that entered a link at time instant T, and not on the total number of vehicles that will eventually pass through the link when all flow demand within period $[0, T]$ exits the network. Therefore, to obtain the average delay that a vehicle travelling along path p encounters on link a, the average link queueing delays are scaled up by a factor $Q_p/y_{ap}(D)$:

$$\begin{aligned} \tau_{ap}(D) &= \frac{Q_p}{y_{ap}(D)}\frac{\Big(1-\alpha_a(D)\Big)}{\alpha_a(D)}\frac{T}{2} = \frac{Q_p}{\hat{\alpha}_a^p(D)Q_p}\Big(\frac{1}{\alpha_a(D)}-1\Big)\frac{T}{2}, \\ &= \frac{1}{\hat{\alpha}_a^p(D)}\Big(\frac{1}{\alpha_a(D)}-1\Big)\frac{T}{2}, \quad \forall a \in A. \end{aligned} \tag{4.21}$$

$\tau_{ap}(D)$	average queueing delay on link a of path p
$y_{ap}(D)$	inflow (veh/h) on link a of vehicles using path p

Here the expression for y_{ap} can be deduced from (4.18). This factor is the ratio between the total number of vehicles that will eventually pass through link a when all path flow demand within period $[0, T]$ exits the network, Q_pT, and the number of vehicles using path p that entered link a at time instant T, $y_{ap}(D)T$. Note that the average link queueing delays are depending on the reduction factors on previous links. Indeed it is known from queueing models that the composition of the vehicles in the queue depends on upstream bottlenecks.

Finally the average path queueing delay $\tau_p(D)$ can be written as the sum of the consecutive average link queueing delays $\tau_{ap}(D)$ on path p:

$$\begin{aligned} \tau_p(D) &= \sum_{a\in p}\tau_{ap}(D) = \sum_{a\in P}\frac{1}{\hat{\alpha}_a^p(D)}\Big(\frac{1}{\alpha_a(D)}-1\Big)\frac{T}{2}, \\ &= \frac{T}{2}\sum_{a\in p}\left(\frac{1}{\hat{\alpha}_a^p(D)\alpha_a(D)}-\frac{1}{\hat{\alpha}_a^p(D)}\right) = \frac{T}{2}\left(\frac{1}{\hat{\alpha}^p(D)}-1\right), \quad \forall p \in P. \end{aligned} \tag{4.22}$$

$\hat{\alpha}_a^p(D)$	path-based reduction factor for path p from origin till link a
$\hat{\alpha}^p(D)$	path-based reduction factor for path p from origin till destination

The last term can be found realizing that the previous term is a telescoping series. Note that if a_2 is the successor of a_1 on path p then it holds that:

$$\frac{1}{\hat{\alpha}^p_{a_1}(D)\alpha_{a_1}(D)} = \frac{1}{\hat{\alpha}^p_{a_2}(D)}. \tag{4.23}$$

Substituting (4.22) into the expression for $\tau_p(D)$ in (4.15), the formula for the average path travel time becomes:

$$c_p(D) = \sum_{a \in p} \frac{L_a}{v_a^{max}} + \frac{T}{2}\left(\frac{1}{\hat{\alpha}^p(D)} - 1\right), \quad \forall p \in P. \tag{4.24}$$

Here the first term represents the the free-flow travel time on path p and second term the average queueing delay on path p.

Remark 6: Note that only the path queueing delay on the total path p with travel time information, $p \in \tilde{P}$, are included in the objective function. See equation (3.2). But in practice there are almost never measured travel times available from origin to destination. However, from equation (4.21) it can be seen that it is possible to determine the queueing delay on a section of a path, by summing only the relevant average link queueing delays. As such, it could be considered to include also sections of paths in the third queueing delay part, f_3, of the upper level objective function[a].

[a]Note that the measured travel time on a section of links might be used by different paths. It should be investigated how this can be taken into account.

4.4.2 Approximation of the average path queueing delays

In the previous subsection, as a part of the average travel time on a path, an expression for the average path queueing delay $\tau_p(D)$ is derived (4.22). For all paths p for which observed travel time information is available, the corresponding path queueing delays are included in the upper level objective function of the matrix estimation problem for STAQ-squeezing* (4.1). Note that these path queueing delays can only be determined after computing a lower level assignment with the assignment model STAQ-squeezing*, because only then the path based reduction factors $\hat{\alpha}^p(D)$ are known (3.10). Therefore, as shown in figure 4.1, also the path queueing delays are approximated in the upper level optimization problem of the developed matrix estimation method for STAQ-squeezing*.

To approximate the path queueing delays, the path based reduction factors from origin till destination as calculated within the previous lower level assignment are used. This leads to the following approximations:

$$\tau_p(D) \approx \frac{T}{2}\left(\frac{1}{\hat{\alpha}^p(D^{k-1})} - 1\right), \quad \forall p \in P. \tag{4.25}$$

T	time horizon
$\tau_p(D)$	average queueing delay on path p
$\hat{\alpha}^p(D)$	path-based reduction factor for path p from origin till destination
D^{k-1}	OD-vector, previous upper level solution

In figure 4.1 these approximations are formulated in vector notation. Given the approximations of the path queueing delays on element level (4.25) the following vector notation is introduced:

$$\tau(D) \approx \frac{T}{2}\big(\hat{\alpha}(D^{k-1})^{-1} - \vec{1}\,\big). \tag{4.26}$$

$\tau(D)$	vector of path queueing delays, of dimension $\lvert\tilde{P}\rvert$
D^{k-1}	OD-vector, previous upper level solution

Where $\hat{\alpha}(D)^{-1}$ is defined as the vector with the elements $\hat{\alpha}^p(D)^{-1}$ for all paths p in $\tilde{P}$. And $\vec{1}$ is defined as the vector of ones with length $|\tilde{P}|$.

> **Remark 7:** Recall that Brederode et al. [5] propose to approximate the link flows using a first order Taylor approximation around the previous upper level solution. It would be more consistent to approximate also the queueing delays using a first order Taylor approximation around the previous upper level solution D^{k-1}. This is possible; The partial derivatives of the queueing delays to the OD-demands at the point D^{k-1} can be determined. The advantage of taking into account the response of the lower level in the upper level approximations has been discussed in subsection 2.1.2.

4.5 Traffic regime constraints

Finally in this last section the traffic regime constraints as considered within the upper level of de developed matrix estimation method for STAQ-squeezing* (4.1) are discussed. Recall from subsection 3.3.1, that to calculate the traffic regime constraints, the reduction factors corresponding to the given OD-matrix D are required. This can be seen in equation (3.41). So also for the traffic regime constraints lower level information is needed to determine them explicitly. In the developed matrix estimation method for STAQ-squeezing* there is chosen to approximate the traffic regime constraints using the reduction factors as calculated within the previous lower level assignment. This leads to the following approximation:

$$\chi_j = \begin{cases} 1 \quad \text{then} \quad \displaystyle\sum_{p\in P_j} \psi_p^{rs} D_{rs} \prod_{ij\in \tilde{I}J_{jp}} \alpha_{ij}(D^{k-1}) \geq \mu_c C_j, \\ 0 \quad \text{then} \quad \displaystyle\sum_{p\in P_j} \psi_p^{rs} D_{rs} \prod_{ij\in \tilde{I}J_{jp}} \alpha_{ij}(D^{k-1}) \leq \mu_f C_j, \end{cases} \qquad \forall j \in \tilde{J}. \tag{4.27}$$

χ_j	$\in \{1, 0\}$, indicates if outlink j is actively constraining inflow
D_{rs}	demand from origin r to destination s
ψ_p^{rs}	fraction of OD-demand rs on path p

$\alpha_{ij}(D)$	reduction factor from inlink i to outlink j for OD-vector D
D^{k-1}	OD-vector, previous upper level solution
C_j	capacity of outlink j
μ_c	safety factor for congested links, $\mu_c > 1$
μ_f	safety factor for free-flow links, $\mu_f < 1$
P_j	set of paths using outlink j
$\tilde{IJ}_{jp}$	set of turns on path p travelling from origin to outlink j, excluding the last turn to link j
$\tilde{J}$	set of outlinks j which can actively constrain flow

Note that there are included safety-factors in (4.27). These factors are introduced to compensate for possible inaccurate approximations. Recall that the traffic regime constraints ensure that the total flow demand on a link j remains either above or below the capacity of link j, depending on whether this link is in a congested or a free-flow traffic regime. However within the upper level optimization problem the total flow demand on link j is approximated (4.27). As such, it can occur that the actual flow demand as calculated by a lower level assignment with the assignment model STAQ-squeezing* turns out to be lower or higher. So it could be that the traffic regime constraints in the upper level seem to be satisfied, but in reality they are violated. To prevent this the link capacities are corrected with a safety factor of $\mu_f < 1$ for free-flow links and $\mu_c > 1$ for congested links. In practice these factors are set by trial and error. For the Dogbone network $\mu_f = 0.99$ and $\mu_c = 1.05$ are used.

Remark 8: Also for the traffic regime constraints it is more consistent to approximate the link demands using a first order Taylor approximation. Note that because there is a similarity between the link flows (4.4) and the link demands[a], the first order Taylor approximations of the link demands can be determined similar to the first order Taylor approximations of the link flows.

[a] See also section 3.2.1 equation (3.8) and equation (3.9).

Chapter 5

Solving the upper level

In the previous two chapters the matrix estimation problem for STAQ-squeezing with fixed route choice and no junction modelling is formulated and the developed solution method for this problem has been described. In this chapter the upper level optimization problem as considered within the developed matrix estimation method for STAQ-squeezing* is further analysed. The goal is to determine whether the `fmincon` interior point algorithm as used within the current *Matlab* model is suitable for this problem, or if a different solver should be used.

5.1 Simplified upper level optimization problem

The developed matrix estimation method for STAQ-squeezing* has been shown in figure 4.1. In this chapter there is referred to the corresponding upper level problem as the simplified upper level optimization problem. The problem is called simplified because the link flows, queueing delays and link demands are approximated instead of determined by the lower level assignment model. The simplified upper level optimization problem in substep $(D^{k-1} \rightarrow D^k)$ reads:

$$
\begin{aligned}
&\min_{D} \; w_1 \sum_{rs \in RS} \left(D_{rs} - D^0_{rs}\right)^2 + w_2 \sum_{a \in \tilde{A}} (\hat{y}_a(D) - \tilde{y}_a)^2 + w_3 \sum_{p \in \tilde{P}} (\hat{\tau}_p(D) - \tilde{\tau}_p)^2. \\
&\quad \hat{y}_a(D) = \sum_{p \in P_a} \hat{\alpha}^p_a(D^{k-1}) \psi^{rs}_p D_{rs} + ... \\
&\qquad\qquad ... \sum_{rs \in RS} \Big(\sum_{rs' \in RS} \sum_{p \in P_a} \frac{\delta \hat{\alpha}^p_a(D)}{\delta D_{rs}}\Big|_{D=D^{k-1}} \psi^{rs'}_p D^{k-1}_{rs'} \Big)(D_{rs} - D^{k-1}_{rs}). \\
&\quad \hat{\tau}_p(D) = \frac{T}{2}\left(\frac{1}{\hat{\alpha}^p(D^{k-1})} - 1\right). \\
&\text{s.t. } D \geq 0. \\
&\quad \chi_j = \begin{cases} 1 & \text{then} \quad \hat{Y}_j(D) \geq \mu_c C_j, \\ 0 & \text{then} \quad \hat{Y}_j(D) \leq \mu_f C_j, \end{cases} \qquad \forall j \in \tilde{J}. \\
&\quad \hat{Y}_j(D) = \sum_{p \in P_j} \psi^{rs}_p D_{rs} \prod_{ij \in \tilde{IJ}_{jp}} \alpha_{ij}(D^{k-1}).
\end{aligned}
\tag{5.1}
$$

$\hat{y}_a(D)$	approximation of the flow on link a
$\hat{\tau}_p(D)$	approximation of the queueing delay on path p
$\hat{Y}_j(D)$	approximation of the total demand on link j

The distance functions as used in the objective function have been introduced in equation (3.2) and the expressions for the approximations of the link flows, queueing delays and traffic regime constraints follow respectively from equations (4.5), (4.25) and (4.27). It can be seen that the objective function as given in (5.1) is a convex quadratic function. Moreover if the weighting factor w_1 is positive, then the objective function is strictly convex and thus the solution D^k of the simplified upper level optimization problem is unique. This is shown in appendix C. Furthermore note that the corresponding constraints are linear inequalities.

In the current *Matlab* implementation of the developed matrix estimation method for STAQ-squeezing*, the `fmincon` interior point algorithm is used to solve the simplified upper level optimization problem. The optimization decision table in the *Matlab* documentation gives an overview of the available solvers in *Matlab* [12]. Given that the simplified upper level optimization problem has a (strictly) convex quadratic objective and linear constraints, this table shows that `quadprog` is the most suitable solver. However, note that the currently used `fmincon` algorithm can also be applied; The objective function is smooth and non-linear. Although the `quadprog` solver is most efficient for the problem as given in (5.1), within this study there is chosen to keep using the current `fmincon` solver. The reason for this choice is that in the analysis of the developed solution method (see chapter 4) it has been concluded that the sensitivities as used within the approximations of the link flows might lead to significant errors (see remark 3). In that case a different approximation of the link flows or even a different heuristic approach could be needed, which influences the optimization problem and hence also the requirements for the solver. The current (suitable) `fmincon` solver is used to investigate this further.

Within `fmincon` there are five different algorithms to choose from. The *Matlab* documentation [12] of `fmincon` describes that both the sqp algorithms and the active set algorithm are only suitable for small to medium sized problems. But the considered matrix estimation problem is a large scale problem; In the *Matlab* implementation of the developed matrix estimation method for STAQ-squeezing* large sparse matrices are used. Only the trust region reflective algorithm and the interior point algorithm can handle such a large scale problem. But because the trust region reflective algorithm only allows (either) bounds or linear equality constraints, the currently used interior point algorithm is indeed most suitable for the given simplified upper level optimization problem. Within the `fmincon` interior point algorithm it is optional to include the Hessian and the gradient. In this study there is decided to include the gradient, because otherwise the `fmincon` solver determines this gradient using finite differences. When the number of OD-pairs becomes large, this slows down the optimization process significantly. In the next section it is shown how the gradient of the simplified upper level optimization problem can be calculated. Note that it would also be possible to include the Hessian matrix.

5.2 Gradient

To determine the gradient of the simplified upper level objective function as shown in (5.1), the three objective parts (f_1, f_2 and f_3) are analysed separately. Note that given the gradients of the three objective parts, the gradient of the simplified objective function can be determined using the weighting factors.

5.2.1 Gradient (f_1)

The partial derivatives of the first (prior) part of the upper level objective function to the OD-demands are given by:

$$\begin{aligned} \frac{\delta f_1}{\delta D_{rs}} &= \frac{\delta}{\delta D_{rs}}\Big(\sum_{rs' \in RS} (D_{rs'} - D^0_{rs'})^2\Big), \\ &= 2(D_{rs} - D^0_{rs}), \quad \forall rs \in RS. \end{aligned} \tag{5.2}$$

So the gradient of the first part of the simplified upper level objective function can be calculated using (5.2).

5.2.2 Gradient (f_2)

The partial derivatives of the second (counts) part of the simplified upper level objective function to the OD-demands are given by:

$$\begin{aligned} \frac{\delta f_2}{\delta D_{rs}} &= \frac{\delta}{\delta D_{rs}}\Big(\sum_{a \in \tilde{A}} (\hat{y}_a(D) - \tilde{y}_a)^2\Big), \\ &= \sum_{a \in \tilde{A}} 2(\hat{y}_a(D) - \tilde{y}_a)\frac{\delta \hat{y}_a(D)}{\delta D_{rs}}, \quad \forall rs \in RS. \end{aligned} \tag{5.3}$$

Here for each link $a \in \tilde{A}$, $\hat{y}_a(D)$ is defined as given in (4.5). Hence the partial derivatives of $\hat{y}_a(D)$ to the OD-demands are defined as follows:

$$\begin{aligned} \frac{\delta \hat{y}_a(D)}{\delta D_{rs}} =& \frac{\delta}{\delta D_{rs}}\Bigg(\sum_{rs' \in RS}\sum_{p \in P_a} \hat{\alpha}^p_a(D^{k-1})\psi^{rs'}_p D_{rs'} + ... \\ & ... \sum_{rs'' \in RS}\Big(\sum_{rs' \in RS}\sum_{p \in P_a} \frac{\delta \hat{\alpha}^p_a(D)}{\delta D_{rs''}}\Big|_{D=D^{k-1}} \psi^{rs'}_p D^{k-1}_{rs'}\Big)(D_{rs''} - D^{k-1}_{rs''})\Bigg), \\ =& \sum_{p \in P_a} \hat{\alpha}^p_a(D^{k-1})\psi^{rs}_p + \sum_{rs' \in RS}\sum_{p \in P_a} \frac{\delta \hat{\alpha}^p_a(D)}{\delta D_{rs}}\Big|_{D=D^{k-1}} \psi^{rs'}_p D^{k-1}_{rs'}, \\ & \forall a \in A, \ \forall rs \in RS. \end{aligned} \tag{5.4}$$

Note that the path based reduction factors and their partial derivatives to the OD-demands for the previous upper level solution D^{k-1} can be determined from the available lower level information, see respectively (3.10) and (4.13). So the gradient of the second part of the simplified upper level objective function can be calculated using (5.3) and (5.4).

5.2.3 Gradient (f_3)

The partial derivatives of the third (path queueing delays) part of the simplified upper level objective function to the OD-demands are given by:

$$\begin{aligned}\frac{\delta f_3}{\delta D_{rs}} &= \frac{\delta}{\delta D_{rs}}\Big(\sum_{p\in\tilde{P}}(\hat{\tau}_p(D)-\tilde{\tau}_p)^2\Big),\\ &= \sum_{p\in\tilde{P}} 2(\hat{\tau}_p(D)-\tilde{\tau}_p)\frac{\delta\hat{\tau}_p(D)}{\delta D_{rs}}, \quad \forall rs\in RS.\end{aligned} \tag{5.5}$$

Here for each path $p \in \tilde{P}$, $\hat{\tau}_p(D)$ is defined as given in (4.25). Hence the partial derivatives of $\hat{\tau}_p(D)$ to the OD-demands are defined as follows:

$$\frac{\delta\hat{\tau}_p(D)}{\delta D_{rs}} = \frac{\delta}{\delta D_{rs}}\frac{T}{2}\left(\frac{1}{\hat{\alpha}_p(D^{k-1})}-1\right) = 0, \quad \forall rs\in RS. \tag{5.6}$$

Note that the approximation of the queueing delay on path p is a constant term which is not depending on the OD-matrix D. So the derivatives of the path queueing delays to the OD-demands all equal zero. And from equation (5.5) it can be seen that also the gradient of the third part of the simplified upper level objective function is a vector of zeros.

It can be concluded that the third part (f_3) of the simplified upper level objective function plays no role in the gradient of the simplified upper level objective function. This shows even more (see also remark 7) the importance of approximating the path queueing delays using a first order Taylor approximation. Using a first order Taylor approximation, the approximation of the path queueing delays is not a constant but a linear equation depending on the OD-matrix D and as such also the third (path queueing delay) part of the objective function has effect on the gradient of the objective function.

Chapter 6

Conclusion

In this chapter the research question as given in chapter 2 is answered. Besides recommendations are given on the further development of the proposed matrix estimation method for STAQ. Finally in the discussion the matrix estimation problem, which has been the central subject of this study, is discussed from a mathematical and practical point of view.

6.1 Conclusion

Using the analysis of the proposed matrix estimation method for STAQ-squeezing* as given in chapter 4 and the analysis of the corresponding simplified upper level optimization problem as described in chapter 5, the given research question can be answered. Below once again this research question is given:

> *Which solver is (most) suitable for the upper level optimization problem within the matrix estimation method for the assignment model STAQ-squeezing as developed by Brederode et al. [5], assuming fixed route choice and no junction modelling?*

Recall that the matrix estimation problem for the assignment model STAQ-squeezing with fixed route choice and no junction modelling is a NP-hard problem. Therefore heuristic methods are used to find suitable solutions for this bi-level problem. Brederode et al. [5] propose a matrix estimation method in which approximations of the link flows and the path queueing delays are used in the upper level problem. As such, contrary to the original upper level optimization problem in each substep $D^{k-1} \rightarrow D^k$, this simplified upper level optimization problem has a (strictly) convex quadratic objective function with linear inequality constraints. From these problem characteristics, in chapter 5 it has been concluded that within *Matlab* the `quadprog` solver can be used best to find the minimum of the given simplified upper level optimization problem. However also the currently used `fmincon` interior point algorithm is suitable.

Because the proposed matrix estimation method for STAQ-squeezing* is implemented in *Matlab*, choosing a build-in solver is most practical. But note that also an open source solver or building an own solver could be considered. However during the project, investigating these options turned out to have no priority.

The `fmincon` interior point algorithm works as desired on small test networks. And at the moment there is worked on the scalability of the implementation, such that it is possible to perform tests on large scale networks. More important to investigate, are the problems with the approximations of the link flows, as identified in the analysis of the proposed matrix estimation method (see chapter 4). Remark 3 describes that the way in which the sensitivity of the assignment matrix to changes in the OD-demands is approximated is mathematically seen not correct. And also remark 4 and remark 5 indicate possible errors in the approximated sensitivities. It still has to be determined whether these errors within the sensitivities as used in the upper level approximations of the link flows are significant on large scale networks. Currently this is the most important question to answer; If the errors are too big[1], it could be desired to adapt the proposed matrix estimation method. Note that this could influence the requirements for the solver.

One possibility to adapt the proposed matrix estimation method would be to use only the assignment matrix and not the sensitivity of the assignment matrix to changes in the OD-demands to approximate the link flows. However in chapter 2 and in the example in section 4.2.3 it has been shown that for STAQ-squeezing, which considers capacity constraints, this conventional approximation method is not suitable. A solution could be to use the conventional method to approximate the link flows, but to include for each link only the paths which have not yet passed a bottleneck[2]. On these free-flow paths a change in path demand results indeed in a change in link flow, such that it is possible to use the conventional approximation method for these paths. It is not possible to approximate the required sensitivities as required within the proposed matrix estimation method using finite differences, because this results in infeasible computation times. The other possibility would be to investigate which other heuristic methods could be used to solve the matrix estimation problem.

6.2 Recommendations

Analysing the proposed matrix estimation method has resulted in some more recommendations for the further development of the given matrix estimation method. These recommendations follow from the remarks as discussed in chapter 4. Note that these remarks focus on improving the current proposed matrix estimation method. So it is assumed that the required sensitivities of the assignment matrix to changes in the OD-demands can be determined as described in subsection 4.3.1.

Observation 1 It might be better to perform only one or a few steps within the simplified upper level optimization problem (in step $D^{k-1} \rightarrow D^k$) instead of solving it to optimality before a new lower level assignment is performed. The approximations of the link flows, path queueing delays and link demands (as needed within the traffic regime constraints) might differ significantly from their actual values as can be determined using the lower level assignment model. Only by performing a lower level assignment it is known whether the steps as taken in the upper level are indeed in the right direction. This has been discussed

[1] It should be checked if there is a significant difference between the sensitivities as determined by using the node model (see figure 4.3) and by using finite differences on lower level assignments.

[2] On all other paths the flow remains fixed on the value as determined by the bottleneck.

in remark 1. More precisely, in each substep $D^{k-1} \rightarrow D^k$ of proposed solution approach as shown in figure 4.1 the approximation $\hat{F}(D)$ of the the objective $F(D)$ is a (strictly) convex approximation of the in general (highly) nonconvex function $F(D)$. So the danger is that even if there is started in step 1 with a vector D^1 near a local minimizer D^* of the original bilevel problem, by solving $D^1 \rightarrow D^2$, the new iterate D^2 could be further away from D^*. So it could be that: $\|D^* - D^1\| < \|D^* - D^2\|$ and even $F(D^*) < F(D^1) < F(D^2)$.

Observation 2 It is important to use also a first order Taylor approximation to approximate the path queueing delays and the traffic regime constraints in the upper level optimization problem. See respectively remark 7 and remark 8. In the current implementation of the proposed matrix estimation method both are approximated using only the assignment matrix from the previous lower level assignment. However in subsection 2.1.2 it has been discussed that it is theoretically more sound to include also the response of the lower level in the upper level optimization problem. As such, similarly as for the link flows, using a first order Taylor approximation to approximate the path queueing delays and the link demands would be more suitable. Note that for the path queueing delays it is extra important to use a first order Taylor approximation; The current approximation of the path queueing delays does not depend on the OD-matrix D at all (see section 5.2.3).

Observation 3 It should be realized that choosing a solver for the upper level problem within the developed matrix estimation method for STAQ-squeezing without route choice and junction modelling is not the final goal; Finally the aim is to solve the upper level problem in the developed matrix estimation method for the full assignment model STAQ. So considering a simplified variant of STAQ is just a intermediate step. In remark 2 has been mentioned that when STAQ-squeezing is used as an equilibrium model, it is questionable if it is still possible to approximate the sensitivity of the assignment matrix to changes in the OD-demands. This is subject of further research. Note that adding junction modelling and considering also the queueing phase, make the optimization problem even more complex.

Finally some minor observations; In remark 6 it has been described that not only the total path queueing delays, but also queueing delays on a section of a path could be included in the objective function. This would be advantageous, because in practice most travel time information is available on sections of links. Furthermore from the uniqueness analysis in section 3.4 it can be concluded that it is required to put always some weight on the prior part (f_1) of the objective function. This is because only the first part of the objective function can guarantee unique minimizers under convexity conditions on the traffic regime constraints. Although this probably works in practice (also when w_2, $w_3 > 0$), note that this will not always guarantee a unique solution.

6.3 Discussion

The matrix estimation problem has been the central subject of this study. In this subsection the matrix estimation process is analysed from a mathematical and practical point of view. In traffic engineering matrix estimation is a well-known problem and it is common to apply a matrix estimation step within the four step traffic model (see figure 1.6). The reason to apply matrix estimation has been discussed in section 1.3.

Recall that the goal of the matrix estimation process is to use additional traffic information, which are observations from practice, to refine the prior OD-matrix as determined in the first three steps of the four step traffic model. The difficulty herein is that there is no one-to-one correspondence between the (prior) OD-matrix and these observations. Given an OD-matrix, which estimates the number of trips between all origin destination combinations in the network, the corresponding link flows, travel times and other relevant characteristics can be predicted using an assignment model. It is important to realize that assignment models are meant to forecast the usage of the network. So they can be applied for example to determine where the bottlenecks or queues in a congested situation possibly will occur. However, it cannot be expected that the results of these assignment models perfectly reflect the situation which can be observed in practice. Therefore, it is questionable if it is valuable to try to match the link flows as predicted by the chosen assignment model, with the link flows as can be observed in practice. This question could be stated as follows: Suppose that the OD-matrix D^0 is a good approximation of the real OD-distribution. How precisely will the outcome $y(D^0)$ (link flows) of the assignment model then describe the real measurable link flow $\tilde{y}$? A relative error

$$\frac{\|y(D^0) - \tilde{y}\|}{\|\tilde{y}\|} \quad \text{of } 10\% \quad \text{or} \quad 40\% \text{ or other?}$$

So can the predicted assignment matrix D^0 or the prediction of the assignment model for the link flows $y(D^0)$ be trusted more?

Another observation is the following non-uniqueness problem; Within the matrix estimation process the input of the assignment model, the prior OD-matrix, is adapted such that it fits as good as possible with the required output, which are observed characteristics like link flows and travel times. So given the required output of the assignment model, there is sought for a suitable input, hence for an OD-matrix that results after assignment in the required output. Recall that given an OD-matrix the assignment model results in one corresponding vector of link flows and one corresponding vector of travel times. However given the flows on the links and/or the travel times on the paths there can be found different OD-matrices which result (after assignment with the considered assignment model) in these observations, assuming that the observations are mutually consistent. This makes it difficult to judge the obtained results. Furthermore note that it would be theoretically more sound, to adapt the model itself and not the input of the model when the output is not as desired. It can be concluded that, from a mathematical and practical point of view, the value of the matrix estimation process is a point of discussion.

Appendix A

The traditional STA model

A.1 Assignment problem

In this appendix the assignment problem for a traditional static traffic assignment (STA) model is discussed. The goal of this appendix is to give the reader insight in the main differences between the STAQ equilibrium model and traditional STA models. Traditional STA models are either unrestrained or capacity restrained static equilibrium assignment models. In an unrestrained or capacity restrained assignment model, there are assumed to be no capacity (or storage) constraints. Hence flows are allowed to exceed link capacities. So within a traditional STA model, a feasible flow, (x, f), $x \in \mathbb{R}^{|E|} \times \mathbb{R}^{|P|}$, does not have to satisfy capacity constraints. In a traditional STA model, for a flow to be feasible, the following conditions must hold [15]:

$$\begin{aligned} \Lambda f &= d \\ \Delta f - x &= 0 \\ f &\geq 0 \end{aligned} \tag{A.1}$$

Λ	path-demand incidence matrix
Δ	path-edge incidence matrix
f	path flow vector, $f = (f_p,\ p \in P)$
f_p	traffic flow (veh/hour) on path p
d	demand vector, $d = (d_w,\ w \in W)$
d_w	traffic demand (veh/hour) from s_w to t_w
x	edge flow vector, $x = (x_e,\ e \in E)$
x_e	traffic flow (veh/hour) on edge e
V	set of vertices
E	set of edges, $e = (i, j),\ i, j \in V$
W	set of OD-pairs $(s_w, t_w),\ s_w, t_w \in V$
P	set of paths

The first condition ensures the conservation of flow. The second condition describes the relationship between path flows and link flows. And the last equation ensures that all path flows are positive. So for a STA model, given

demand d, a feasible flow (x, f) can be defined as:

$$F_d = \{(x, f) \mid (x, f) \text{ satisfies (A.1) } \}. \tag{A.2}$$

d	demand vector
x	edge flow vector
f	path flow vector

Contrary to STA models, STAQ does take into account capacity (and storage) constraints. Hence for STAQ, not only conservation of flow and non-negativity constraints, but also capacity constraints should be satisfied. This makes the assignment problem for STAQ more complex, but also more realistic.

Within an equilibrium model, the link flows in the user equilibrium (UE) should comply with Wardrop's first principle (see section 1.2.1). Wardrop's principle states that a feasible flow $(x, f) \in F_d$ is a UE with respect to the given demand d, if for all $w \in W$ it holds that for any $p, q \in P_w$ we have [15]:

$$f_p > 0 \;\Rightarrow\; \begin{cases} c_p(x) = c_q(x) & \text{if } f_q > 0. \\ c_p(x) \leq c_q(x) & \text{if } f_q = 0. \end{cases} \tag{A.3}$$

P_w	set of all directed paths p connecting s_w to t_w
f_p	traffic flow (veh/hour) on path p
$c_p(x)$	path cost function
x	edge flow vector

Because a traditional STA model assumes no capacity constraints, all link costs[1] are separable (see section 1.2.2); The cost on a link is only depending on the flow on the link itself. Hence for traditional STA models the link and path costs can be formulated as a function of the link flows:

$$\begin{aligned} c_e &= c_e(x_e), \\ c_p(x) &= \sum_{e \in p} c_e(x_e), \quad \forall p \in P. \end{aligned} \tag{A.4}$$

x_e	traffic flow (veh/hour) on edge e
$c_e(x_e)$	edge cost function
$c_p(x)$	path cost function

For STAQ it is not possible to express the link and path costs as in (A.4). Within STAQ, the cost on a link is not only depending on the flow on the link itself, but also on the flows on all other links in the network. So for STAQ $c_e = c_e(x)$.

Assume that the link cost functions $c_e(x_e)$ are continuous and non-decreasing in x_e:

$$c_e(x_e) \leq c_e(x'_e) \quad \text{for} \quad x_e \leq x'_e. \tag{A.5}$$

Beckmann et al. have shown that the UE for a traditional STA model under condition (A.5) can be computed as the solution of the following optimization

[1] Recall that for simplicity in this report the generalized costs are considered to consist of travel times only.

problem [15]:

$$\min_{x,f} \sum_{e \in E} \int_0^{x_e} c_e(\tau)\, d\tau \quad \text{s.t.} \quad (x, f) \in F_d \,. \tag{A.6}$$

Solving this optimization problem, results in link flows and path flows (x, f) corresponding to the given demand vector d. It can be concluded that (A.6) describes the assignment problem for a traditional STA model as an optimization problem.

For STAQ it is not possible to define the assignment problem explicitly as one optimization problem. Due to the capacity constraints within STAQ, flows can be held up on bottlenecks upstream from the considered link. As such path flows are not defined for STAQ[2]. Within STAQ only the inflow an outflow of each link can be determined. The node model, link model and junction model within the network loading submodel of STAQ interact, to determine link flows given the path demands. To do so a fixed point problem has to be solved (see subsection 3.2.1). Consequently it is not possible to define an explicit relationship between path demands and link flows for STAQ. It can be concluded that network loading for a traditional STA models consists of a simple relationship between path flows and link flows, whereas for STAQ a complex algorithm is needed to determine the link flows given the path demands. Therefore, for STAQ it is not possible to combine the route choice and network loading problems into one optimization problem. Using STAQ as an equilibrium model[3], iterations between the route choice submodel and the network loading submodel are preformed to find solutions to the assignment problem for STAQ. After convergence the resulting flows approximate a UE.

A.2 Matrix estimation

For a traditional STA model the matrix estimation problem is formulated as follows (see section 1.3):

$$\begin{aligned} \min_D F(D) &= \alpha f_1(D, D^0) + (1-\alpha) f_2(y, \tilde{y}) \\ \text{s.t. } y &= \text{STA}(D) \\ D &\geq 0 \end{aligned} \tag{A.7}$$

D	vector/matrix of estimated OD-demands
D^0	vector/matrix of prior (modelled) OD-demands
y	vector of estimated link flows
$\tilde{y}$	vector of observed link flows
f_1, f_2	distance functions
α	$\in [0, 1]$ weighting factor

Within this appendix the uniqueness of the solutions to the second part of the upper level objective function in (A.7) is analysed. So the situation is analysed for which α in (A.7) is set to zero.

[2] Instead there is worked with path demands (and reduction factors).

[3] Recall that this is possible in the squeezing-phase but not in the queueing-phase of STAQ (see subsection 1.4.2).

Because STAQ reduces to a STA model when the crossing fraction matrix equals the path-link incidence matrix, this problem is interesting to consider.

In the remainder of this appendix the notation of appendix A.1 is used, which is the most commonly used notation for traditional STA-models. So from now on the vector of OD-demands and the vector of link flows as introduced in (A.7) are denoted differently ($D \equiv d$, $y \equiv x$). Hopefully this causes no confusion. So the matrix estimation problem as considered in this appendix reads:

$$\begin{aligned} &\min_{d \geq 0} f_2(d) := \sum_{e \in \tilde{E}} (x_e(d) - \tilde{x}_e)^2 \\ &\text{s.t. } (f, x) \in F_d \text{ is UE w.r.t. given costs} \end{aligned} \tag{A.8}$$

d	demand vector
x	edge flow vector
f	path flow vector
$x_e(d)$	traffic flow (veh/hour) on edge e for OD-vector d
$\tilde{x}_e$	link flow measurement on link e
$\tilde{E}$	$\subset E$ set of links with a link flow measurement
F_d	set of feasible flows, see A.2

The possible nonuniqueness of minimizers of (A.8) is discussed. To investigate whether there exists an unique OD-vector for which the corresponding path flows match the given observed link flows it is assumed that the flows on all links are known;

Assumption 1: For given $\bar{d} > 0$ the vector $\bar{x}$ is the (unique) link flow UE. Furthermore $\tilde{E} = E$ and $\tilde{x}_e = \bar{x}_e,\ e \in E$.

So it is assumed that $\bar{x}$ is given, such that it is an UE with respect to some c_p and some $\bar{d} \geq 0$. The question is, if there exist different OD-matrices for which the corresponding path flows perfectly match these given edge flows, or if only $\bar{d}$ suffices. To find al possible solutions such that $(\bar{x}, f)$ is a UE with respect to d, the following problem has to be solved[4]:

$$\begin{aligned} \Lambda f &= d \\ \Delta f &= \bar{x} \\ f, d &\geq 0 \end{aligned} \tag{A.9}$$

where only the paths p in Λ and Δ can be taken, such that for $p \in P_w$ we have:

$$c_p(\bar{x}) = \min_{q \in P_w} c_q(\bar{x}). \tag{A.10}$$

Λ	path-demand incidence matrix
Δ	path-edge incidence matrix
$\bar{x}$	observed edge flow vector, $x = (\bar{x}_e,\ e \in E)$
$c_p(x)$	path cost function
P_w	set of all directed paths p connecting s_w to t_w

[4]Note the similarity between (A.9) and (A.1).

Note that if it holds for STAQ that $B(D) = \Delta$, then indeed (3.50) is the same problem as (A.9).

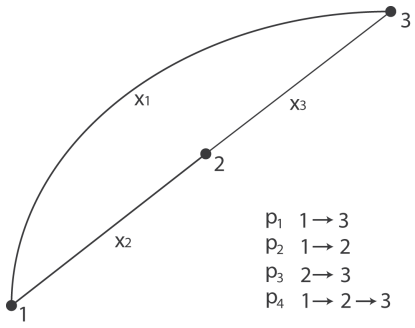


Figure A.1: Example network.

This extra criterion (A.10) is needed, because otherwise not all solutions indeed correspond to a UE. To see this, consider the network as given in figure A.1. If the edge costs in this network are all considered to be one, then $\bar{x} = \{1, 1, 1\}$ is a UE with respect to these edge costs and $\bar{d} = \{1, 1, 1\}$. Given these edge flows, problem (A.9) reads:

$$\begin{aligned} f_1 + f_4 &= d_1 & f_1 &= 1 & f_p \geq 0, \quad \forall p \in P. \\ f_2 &= d_2 & f_2 + f_4 &= 1 & \\ f_3 &= d_3 & f_3 + f_4 &= 1 & \end{aligned} \tag{A.11}$$

Solving this system of equations leads to the following set of solutions:

$$\begin{aligned} 1 \leq d_1 \leq 2, \\ d_2 = 2 - d_1, \\ d_3 = 2 - d_1. \end{aligned} \tag{A.12}$$

But it can be seen that not all these OD-matrices correspond to a UE with respect to $\bar{x}$. When the demand on the first OD-pair is chosen to be bigger than one, $d_1 > 1$, this implies that the demand on path four is bigger than zero, $f_4 > 0$. But in a UE path four cannot be used, because a traveller can do better by changing to path 1:

$$c_4(\bar{x}) = 2 > c_1(\bar{x}) = 1.$$

As such, path four should be set to zero in the path-demand and path-edge incidence matrices to obtain only the solutions which correspond to a UE. In general it holds that in (A.9) only paths should be chosen for which condition (A.10) holds.

A condition similar to (A.10) cannot be formulated for STAQ. The travel times for STAQ can not be described by a monotonously increasing function, but are (in an equilibrium model) determined by the network loading submodel. As such, the generalized costs for a set of given path flows are only known explicitly after performing a network loading step.

In this specific example it can be seen that the solution to (A.11) together with (A.10) is unique. However if in the same network again the edge flows are $\bar{x} = \{1, 1, 1\}$, but the edge cost of link one equals two instead of one, then all solutions d to (A.12) satisfy both (A.11) and (A.10). As such in that case the solution of (A.8) is not unique. Generally the uniqueness of the solutions to the second part of the upper level objective function as given in (A.8), can be analysed using the system of equations (A.9) with extra condition (A.10);

Assumption 2: Let Assumption 1 be satisfied and let the system (A.9) be in reduced form. In other words it is assumed that all columns of Λ and Δ are skipped which belong to a path $q \in P_w$ for some $w \in W$, such that

$$c_q(\bar{x}) > \min_{p \in P_w} c_p(\bar{x}).$$

Lemma 1: *Let Assumption 1 and Assumption 2 hold. Then for all solutions* (d, f) *of the reduced system* (A.9) *the part* d *is a minimizer of* (A.8)*, satisfying* $f_2(d) = 0$.

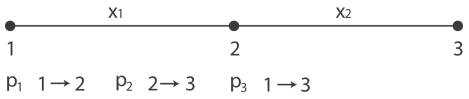


Figure A.2: Example network.

The simplest example where Lemma 1 can be applied to obtain non-unique minimizers is as follows; Let the network be as shown in figure A.2 . The strictly increasing costs $c_e(x_e)$ can be chosen arbitrarily. Furthermore let $\bar{x} = (2, 2) \in F_{\bar{d}}$ be a UE with respect to $\bar{d} = (1, 1, 1)$ and let $\tilde{x} = \bar{x}$ and $\tilde{E} = E$ in (A.8). Then (A.9) reads:

$$\begin{aligned} &f_1 = d_1 \qquad f_1 + f_3 = 2 \qquad f_p \geq 0, \quad \forall p \in P \\ &f_2 = d_2 \qquad f_2 + f_3 = 2 \\ &f_3 = d_3 \end{aligned} \tag{A.13}$$

or equivalently $d_1 + d_3 = 2$, $d_2 + d_3 = 2$. Hence the solution set of (A.13) and thus of (A.8) is given by:

$$d_1 = d_2 = 2 - \alpha, \quad d_3 = \alpha \quad \text{with } 0 \leq \alpha \leq 2. \tag{A.14}$$

For this specific example, the same non-uniqueness occurs when in the lower level of (A.8) STAQ-squeezing is used instead of a traditional STA-model. Consider the problem

$$\begin{aligned} &\min_{d \geq 0} f_2(d) := \sum_{e \in \tilde{E}} (y_e(d) - \tilde{y}_e)^2 \\ &\text{s.t. } y(d) \text{ is the link flow result of STAQ-squeezing} \end{aligned} \tag{A.15}$$

and take the network in figure A.2 with arbitrary link capacities. Applying STAQ-squeezing with $\bar{d} = (1, 1, 1)$ leads to a link flow $y(\bar{d})$. If the observed link

flows are now set to $\tilde{y} = y(\bar{d})$, then obviously for all demands d satisfying (A.14) the inflow demands into all nodes are the same and STAQ-squeezing generates the same flow $y(d) = y(\bar{d}) = \tilde{y}$. As such, all these d's are minimizers of (A.15).

Finally it is interesting to note that the following general non-uniqueness result holds.

Lemma 2: *Let Assumption 1 and Assumption 2 be satisfied and assume that there is a UE $(\bar{f}, \bar{x}) \in F_{\bar{d}}$ (see Assumption 1) with $\bar{f} > 0$. Suppose further that $|W| > |E|$ holds. Then the reduced system* (A.9) *has a solution polytope of dimension ≥ 1.*

W	set of OD-pairs, (s_w, t_w)
E	set of edges
P	set of paths
P_w	set of all directed paths p connecting s_w to t_w

Proof. According to the assumptions the solutions (d, f) of

$$\Lambda f - d = 0, \qquad \Delta f = \bar{x},$$

are given by

$$\begin{pmatrix} d \\ f \end{pmatrix} = \begin{pmatrix} \bar{d} \\ \bar{f} \end{pmatrix} + \ker M, \tag{A.16}$$

where

$$M = \begin{pmatrix} -I & \Lambda \\ 0 & \Delta \end{pmatrix} \quad \text{and} \quad \ker M = \{(d, f) \mid M \begin{pmatrix} d \\ f \end{pmatrix} = 0\}.$$

Here I is the unit matrix of dimension $|W|$. It can be shown that $\ker M \neq \{0\}$ allows non-unique solutions with respect to the d variable. To do so Gauss elimination is applied to M, or in this situation to Δ, and M (or Δ) is transformed into an upper triangular form:

$$\tilde{M} = \begin{pmatrix} -I & \Lambda \\ 0 & U \end{pmatrix} \text{ with } U = \begin{pmatrix} \underline{|\star} & & & \\ & \underline{|\star} & & \\ & & \underline{|\star} & \\ & & & \dots \end{pmatrix},$$

where U has $|P|$ columns and $|E|$ rows and $\star$ denotes the pivot elements. Since it holds that $|P| \geq |W| > |E|$ the matrix U has more columns than rows. Therefore U and thus $\tilde{M}$ must contain at least one non-pivot column q corresponding to a free variable path flow f_q, $q \in P_{w_0}$ for some $w_0 \in W$. Solving the system $\tilde{M}\binom{d}{f} = 0$ wrt. the variable d_{w_0} yields:

$$d_{w_0} = \sum_{p \in P_{w_0}} \Lambda_{w_0,p} f_p.$$

So d_{w_0} contains the free variable f_q with $q \in P_{w_0}$ and represents a solution set of dimension at least one. By the assumption that $\bar{d} > 0$, $\bar{f} > 0$ the solution set of (A.16) allows at least a one dimensional line segment which also satisfies $(d, f) \geq 0$.

Appendix B

Turn based reduction factors

B.1 Existence of the sensitivities to the turn demands

Brederode et al. [5] propose to use the node model within the network loading submodel of STAQ-squeezing* to approximate the partial derivatives of the turn based reduction factors $\alpha_{ij}(D)$ to the turn demands $T_{ij'}(D)$. From equations (3.11), (3.6) and (3.35), it can be seen that both the turn based reduction factors and the turn demands are depending on the OD-vector D. As such the partial derivatives of the turn based reduction factors to the turn demands can be written in the following form:

$$\frac{\delta \alpha_{ij}(D)}{\delta T_{ij'}(D)} \rightarrow \frac{\delta f(x)}{\delta g(x)}. \tag{B.1}$$

$\alpha_{ij}(D)$	reduction factor from inlink i to outlink j
$T_{ij'}(D)$	turn demand from inlink i' to outlink j'

It can be shown that a derivative of the form (B.1) only exist, considering a vector of one variable; To do so suppose $x \in \mathbb{R}$ and $f, g \in C^1$. Then the derivative from $f(x)$ to $g(x)$ at the point $x = \bar{x}$ is defined as:

$$\begin{aligned} \frac{\delta f(x)}{\delta g(x)}\Big|_{x=\bar{x}} &= \lim_{h \to 0} \frac{f(\bar{x}+h) - f(\bar{x})}{g(\bar{x}+h) - g(\bar{x})}, \\ &= \lim_{h \to 0} \frac{f'(\bar{x})h + o(h)}{g'(\bar{x})h + o(h)} = \frac{f'(\bar{x})}{g'(\bar{x})}. \end{aligned} \tag{B.2}$$

But now suppose $x = (x_1, x_2)$, with $x_1, x_2 \in \mathbb{R}$ and $f, g \in C^1$. And let f_{x_1}, f_{x_2}, g_{x_1}, g_{x_2} denote the partial derivatives of f and g. Then the derivative from $f(x)$ to $g(x)$ at the point $x = \bar{x}$ is not defined:

$$\begin{aligned} \frac{\delta f(x)}{\delta g(x)}\Big|_{x=\bar{x}} &= \lim_{h \to 0} \frac{f(\bar{x}+h) - f(\bar{x})}{g(\bar{x}+h) - g(\bar{x})}, \\ &= \lim_{h \to 0} \frac{f_{x_1}(\bar{x})h_1 + f_{x_2}(\bar{x})h_2 + o(|h|)}{g_{x_1}(\bar{x})h_1 + g_{x_2}(\bar{x})h_2 + o(|h|)}. \end{aligned} \tag{B.3}$$

To see this choose for example $h_1 = ch_2,\ c \neq 0$, then:

$$\frac{\delta f(x)}{\delta g(x)}\Big|_{x=\bar{x}} \approx \frac{f_{x_1}(\bar{x})c + f_{x_2}(\bar{x})}{g_{x_1}(\bar{x})c + g_{x_2}(\bar{x})}. \tag{B.4}$$

Which is depending on the value of c. So it can be concluded that a derivative of the form (B.1) only exists for a vector of one variable.

Note that when $f(x)$ in (B.1) can be written in the form $f(x) = k(g(x))$, where $k \in C^1$, then the derivative of $f(x)$ to $g(x)$ at the point $x = \bar{x}$ is defined for a general vector $\bar{x}$:

$$\begin{aligned} \frac{\delta f(x)}{\delta g(x)}\Big|_{x=\bar{x}} &= \lim_{h\to 0} \frac{f(\bar{x}+h) - f(\bar{x})}{g(\bar{x}+h) - g(\bar{x})}, \\ &= \lim_{h\to 0} \frac{k(g(\bar{x}+h)) - k(g(\bar{x}))}{g(\bar{x}+h) - g(\bar{x})} = k'(g(\bar{x})). \end{aligned} \tag{B.5}$$

However, it is not possible to write the reduction factors as an explicit function of the turn demands. Recall form subsection 3.2.1 that the reduction factors are depending on the turn demands and the turn demands are on their turn depending on the reduction factors. To find the turn demands and reduction factors corresponding to a given vector of path demands, within the network loading submodel of STAQ-squeezing* a fixed point problem has to be solved (3.12). So it can be concluded that the derivatives of the turn based reduction factors to the turn demands have form (B.1) and are only defined when the corresponding OD-vector D consists of only one OD-pair.

B.2 Possible model errors

Because the partial derivatives of the turn based reduction factors to the turn demands generally do not exist, using the approximations of the partial derivatives to the turn demands as proposed by Brederode et al. [5] in practice leads to errors in the model. It can be shown that the proposed approximations (see figure 4.3) are only suitable when both the path based reduction factors and the route fractions equal one:

- From the definition of turn demand (3.6) it can be seen that:

$$T_{ij}(D) + \Delta = Q_p(D) + \Delta, \quad \text{for } p \in P_{ij} \iff \hat{\alpha}_i^p(D) = 1. \tag{B.6}$$

$T_{ij}(D)$	turn demand from inlink i to outlink j
$Q_p(D)$	path demand on path p
$\hat{\alpha}_i^p(D)$	reduction factor on path p till link i
$P_{ij} \subseteq P$	set of paths over turn ij

  So a change in turn demand can be directly translated in a change in path demand if the corresponding path based reduction factor equals one ($\hat{\alpha}_i^p = 1$). Note that if path p over turn ij already passed a bottleneck ($\hat{\alpha}_i^p < 1$), then the equality in (B.6) does not hold. In that case, a change in the demand on path p possibly affects the distribution of flow on the

upstream bottleneck, and as such not only the demand on turn ij but also the demand on other turns ij' on the considered node are possibly changed.

- From the definition of path demand (3.35) it can be seen that:

$$Q_p(D) + \Delta = D_{rs} + \Delta, \quad \text{for } p \in P_{rs} \iff \psi_p^{rs} = 1. \tag{B.7}$$

$Q_p(D)$	path demand on path p
D_{rs}	demand on OD-pair rs
ψ_p^{rs}	fraction of demand from OD-pair rs that use path p
$P_{rs} \subseteq P$	set of paths between OD-pair rs

So a change in path demand can be directly translated in a change in OD-demand if the corresponding route fraction equals one ($\psi_p^{rs} = 1$). Note that if the OD-demand of OD-pair rs is divided over different paths ($\psi_p^{rs} < 1$), then the equality in (B.7) does not hold. In that case a change in demand on OD-pair rs does not only affect path p, but also other paths p' between OD-pair rs.

This leads to the conjecture that only when both $\hat{\alpha}_i^p$ and ψ_p^{rs} equal one, a change in turn demand can be directly translated in a change in OD-demand, such that in this case:

$$\frac{\delta \alpha_{ij}(D)}{\delta T_{ij'}(D)} = \frac{\delta \alpha_{ij}(D)}{\delta Q_p(D)} = \frac{\delta \alpha_{ij}(D)}{\delta D_{rs}}. \tag{B.8}$$

$\alpha_{ij}(D)$	reduction factor from inlink i to outlink j
$T_{ij'}(D)$	turn demand from inlink i' to outlink j'

Where turn ij and turn ij' are on the same node. In this case the approximation of the partial derivatives of the turn based reduction factors (see figure 4.3) to the turn demands reads:

$$\frac{\delta \alpha_{ij}(D)}{\delta T_{ij'}(D)} \approx \frac{\alpha_{ij}(D + \Delta e_{rs}) - \alpha_{ij}(D - \Delta e_{rs})}{T_{ij'}(D) + \Delta - (T_{ij'}(D) - \Delta)} = \frac{\alpha_{ij}(D + \Delta e_{rs}) - \alpha_{ij}(D - \Delta e_{rs})}{2\Delta}. \tag{B.9}$$

Here turn ij' lies on path p ($\psi_p^{rs} = 1$) between OD-pair rs. The approximation in (B.9) is a valid approximation. However the reader should realize that a situation in which all the path based reduction factors and all route fractions equal one is never seen in practice. As such, the considered approximations of the path based reduction factors to the OD-demands (4.13) are expected to be incorrect in practice.

Appendix C

Convexity

C.1 Simplified upper level objective function

In this appendix it is shown that the simplified upper level objective function as given in (5.1) is a (strictly) convex function. The considered objective function reads:

$$\begin{aligned}\hat{F}(D) &= w_1\ f_1(D, D^0) + w_2\ f_2(y(D), \tilde{y}) + w_3\ f_3(\tau(D), \tilde{\tau}),\\ &= w_1 \sum_{rs\in RS} (D_{rs} - D^0_{rs})^2 + w_2 \sum_{a\in\tilde{A}} (\hat{y}_a(D) - \tilde{y}_a)^2 + w_3 \sum_{p\in\tilde{P}} (\hat{\tau}_p(D) - \tilde{\tau}_p)^2.\end{aligned} \tag{C.1}$$

Here the approximations of the link flows $\hat{y}_a$ and the queueing delays $\hat{\tau}_p$ are as defined in (4.5) and (4.25). The three objective parts (f_1, f_2 and f_3) are first analysed separately. Note that if all three objective parts are convex, then also the total objective function as given in (C.1) is convex.

Convexity (f_1)
It can be easily seen that the elements of the Hessian matrix of the first (prior) part of the simplified upper level objective function are given by:

$$\frac{\delta^2 f_1}{\delta D_{rs}\delta D_{rs'}} = \frac{\delta^2\big(\sum_{rs\in RS}(D_{rs} - D^0_{rs})^2\big)}{\delta D_{rs}\delta D_{rs'}} = \begin{cases} 0 & \forall rs' \neq rs, \\ 2 & \forall rs' = rs, \end{cases} \qquad \forall rs', rs \in RS.$$

Hence the Hessian matrix $|RS| \times |RS|$ has the following form:

$$H = \begin{bmatrix} 2 & 0 & \dots & 0 \\ 0 & 2 & \dots & \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & 2 \end{bmatrix} = 2I_{|RS|}.$$

This matrix is positive definite and therefore it can be concluded that the first part of the simplified upper level objective function is strictly convex.

Convexity (f_2)
The first order Taylor approximations of the link flows $\hat{y}_a(D)$ as used within the second (counts) part of the simplified upper level objective function (C.1) are linear functions of the form:

$$\hat{y}_a(D) = c_a + b_a^T D \quad \text{with } \; c_a \in \mathbb{R}, \; b_a \in \mathbb{R}^{|RS|}.$$

As such, $(\hat{y}_a(D) - \tilde{y}_a)^2$ is a quadratic function:

$$(\hat{y}_a(D) - \tilde{y}_a)^2 = (b_a^T D)^2 - 2b_a^T D(\tilde{y}_a - c_a) + (\tilde{y}_a - c_a)^2.$$

Where the corresponding Hessian matrix reads $\nabla^2(b_a^T D)^2 = 2b_a b_a^T$. This matrix is positive semidefinite; Indeed it can be seen that:

$$D^T b_a b_a^T D = (b_a^T D)^2 \geq 0 \quad \forall D \in \mathbb{R}^{|RS|} \; .$$

Therefore it can be concluded that the second part of the simplified upper level objective function is convex.

Convexity (f_3)
The approximations of the queueing delays $\hat{\tau}_p(D)$ as used within the third (queueing delays) part of the simplified upper level objective function (5.1) are constants. Therefore in this case the Hessian matrix corresponding to $(\hat{\tau}_p(D) - \tilde{\tau}_p)^2$ equals zero. Note that considering a first order Taylor approximation $\hat{\tau}_a(D)$ of the path queueing delays $\tau_a(D)$, the corresponding Hessian matrix will be positive semidefinite. This can be shown using an argument similar as for the link flows.

Conclusion
It can be concluded that the Hessian of the simplified upper level objective function as given in (C.1) $\nabla^2 \hat{F}(D)$ reads:

$$\nabla^2 \hat{F}(D) = w_1 2I_{|RS|} + w_2 \sum_{a \in \hat{A}} b_a b_a^T.$$

Note that this matrix is positive semidefinite and positive definite if $w_1 > 0$. So in the latter case the function $\hat{F}(D)$ is a strictly convex quadratic function with a unique minimizer D^k. The same holds when a first order Taylor approximation $\hat{\tau}_a(D)$ of the queueing delays $\tau_a(D)$ is used in the third part of the simplified upper level objective function (C.1).

Bibliography

[1] Jonathan F Bard. Some properties of the bilevel programming problem. *Journal of optimization theory and applications*, 68(2):371–378, 1991.

[2] Michiel Bliemer, Mark Raadsen, Luuk Brederode, Michael Bell, Luc Wismans, and Mike Smith. Genetics of traffic assignment models for strategic transport planning. *Transport reviews*, 37(1):56–78, 2017.

[3] Michiel Bliemer, Mark Raadsen, Erik-Sander Smits, Bojian Zhou, and Michael Bell. Quasi-dynamic traffic assignment with residual point queues incorporating a first order node model. *Transportation Research Part B: Methodological*, 68:363–384, 2014.

[4] Luuk Brederode, Adam Pel, Luc Wismans, Erik de Romph, and Serge Hoogendoorn. Static traffic assignment with queuing: model properties and applications. *Transportmetrica A: Transport Science*, pages 1–36, 2018.

[5] Luuk Brederode, Adam John Pel, and Serge Paul Hoogendoorn. Matrix estimation for static traffic assignment models with queuing. In *Paper presented at the 3rd Symposium of the European Association for Research of Transportation*, volume 10, page 12. Leeds UK, 2014.

[6] Luuk Brederode, Adam John Pel, Luc Wismans, and Erik de Romph. Improving convergence of quasi dynamic assignment models. In *6th International Symposium on Dynamic Traffic Assignment*, 2016.

[7] Maria Stella Fiorenzo-Catalano. *Choice set generation in multi-modal transportation networks*. PhD thesis, TU Delft, Delft University of Technology, 2007.

[8] Rodric Frederix, Francesco Viti, and Chris Tampère. Dynamic origin–destination estimation in congested networks: theoretical findings and implications in practice. *Transportmetrica A: Transport Science*, 9(6):494–513, 2013.

[9] Tanja Gellenbeck. Origin-destination matrix estimation with SPSA and STAQ. Master's thesis, University of Twente, 2016.

[10] Michael J Maher, Xiaoyan Zhang, and Dirck Van Vliet. A bi-level programming approach for trip matrix estimation and traffic control problems with stochastic user equilibrium link flows. *Transportation Research Part B: Methodological*, 35(1):23–40, 2001.

[11] Tom V Mathew and K V Krishna Rao. *Introduction to transportation engineering*, chapter 31. fundamental relations of traffic flow, pages 31.1–31.8. NPTEL Web Course, 2007.

[12] MathWorks. R2018a Documentation. https://nl.mathworks.com/help/matlab/. Accessed: 2018-06-28.

[13] Erik-Sander Smits. Origin-destination matrix estimation in omnitrans. Master's thesis, Utrecht University, 2011.

[14] Erik-Sander Smits, Michiel CJ Bliemer, Adam J Pel, and Bart van Arem. A family of macroscopic node models. *Transportation Research Part B: Methodological*, 74:20–39, 2015.

[15] Georg Still. Lectures on parametric optimization: An introduction, March 2018.

[16] Chris Tampère, Ruben Corthout, Dirk Cattrysse, and Lambertus Immers. A generic class of first order node models for dynamic macroscopic simulation of traffic flows. *Transportation Research Part B: Methodological*, 45(1):289–309, 2011.

[17] Hossein Tavana. *Internally-consistent estimation of dynamic network origin-destination flows from intelligent transportation systems data using bi-level optimization.* PhD thesis, The University of Texas at Austin, 2001.

[18] Guangmin Wang, Zhongping Wan, Xianjia Wang, and Yibing Lv. Genetic algorithm based on simplex method for solving linear-quadratic bilevel programming problem. *Computers & Mathematics with Applications*, 56(10):2550–2555, 2008.

[19] John Glen Wardrop and James Ivor Whitehead. Correspondence. some theoretical aspects of road traffic research. *Proceedings of the Institution of Civil Engineers*, 1(5):767–768, 1952.

[20] Allard Wilson. A statistical theory of spatial distribution models. *Transportation research*, 1(3):253–269, 1967.

[21] Matthew A Wright, Gabriel Gomes, Roberto Horowitz, and Alex A Kurzhanskiy. On node models for high-dimensional road networks. *Transportation Research Part B: Methodological*, 105:212–234, 2017.

[22] Hai Yang. Heuristic algorithms for the bilevel origin-destination matrix estimation problem. *Transportation Research Part B: Methodological*, 29(4):231–242, 1995.