Gravity model parameter calibration for large scale strategic transport models.

Mark Pots

Company supervisors: Bastiaan Possel, Luuk Brederode and Nico Aardoom

University supervisor: Peter J.C. Dickinson

Graduation supervisor: Marc Uetz

Chair: Discrete Mathematics and Mathematical Programming (DMMP)
Applied Mathematics

October 29, 2018

adviseurs mobiliteit
Goudappel Coffeng

UNIVERSITY OF TWENTE.

Abstract

This study considers the calibration of the lognormal cost function parameters within the simultaneous gravity model for large scale strategic transport models. The parameters are calibrated based on observed trip length distributions and modal split fractions. First the full trip distribution used at Goudappel Coffeng that consists of a stratification in trip purposes and sub purposes is described in detail. A new calibration approach is investigated that proposes a triproportional fitting procedure over the current biproportional fitting procedure. The triproportional fitting procedure solves the gravity model equations while simultaneously fitting the trip distribution on modal split. This simplifies the calibration eliminating the modal split constraint and its associated lognormal parameters. Under the new approach the calibration algorithms' running times are considerably reduced.

The calibration problem for this new approach then is formulated as a bilevel optimization problem. In this formulation the higher-level optimization task is the choice of parameters and the lower-level optimization task is to maximize entropy subject to trip end and modal split constraints, given the choice of (behavioural) parameters. This lower-level optimization task is solved by the triproportional fitting procedure. A BFGS Quasi Newton method was developed to solve the higher-level optimization. It is compared with the currently used hillclimbing algorithm at Goudappel Coffeng. This new method converges more reliably to the local minimum albeit much slower than the hillclimbing algorithm. Further two methods to compute the gradient required in the BFGS method are considered. The first is a simple finite differences approach. The second method is an analytical adjoint method. The second method is observed to be significantly faster for smaller to medium scale models ($\leq$ 3300 zones), however occasionally suffers from a singular linear system.

Contents

Notation

Globally used indices:

i	a zone of origin
j	a zone of destination
m	a mode, e.g. $m \in \{car, bike, public\ transit\}$
u	a user class, e.g. $u \in \{\text{co (car owners)}, \text{nco (non car owners)}\}$
k	a cost bin

Globally used Variables:

t_{ijmu}	number of trips made from zone i to zone j using mode m by trip makers in user class u
$\boldsymbol{T}^m$	OD matrix for mode m
$\boldsymbol{T}$	either denotes a full trip distribution i.e. a distribution of T over all the t_{ijmu}'s
	or the OD matrix aggregated over all modes: $\boldsymbol{T} = \sum_m \boldsymbol{T}^m$
$P_{i(u)}$	production value of zone i i.e. the observed number of trips departing from zone i (made by user class u)
$A_{j(u)}$	attraction value of zone j i.e. the observed number of trips arriving in zone j
$O_{i(u)}$	production balancing factor of zone i
$D_{j(u)}$	attraction balancing factor of zone j
$F^{mu}(c_{ijm})$	cost function to be calibrated, models the willingness to travel using mode m at generalized costs c by a trip maker from user class u
c_{ijm}	(generalized) cost to travel from zone i to zone j using mode m
α_{mu}	(lognormal) cost function parameter for mode m and user class u to be estimated in the calibration or balancing factor for mode m and user class u
β_{mu}	(lognormal) cost function parameter for mode m and user class u to be estimated in the calibration
$\widehat{d}_{muk}$	observed number of trips made by mode m by user class u of a cost in bin k
d_{muk}	modelled number of trips made by mode m by user class u of a cost in bin k
$\widehat{MS}_{mu}$	the observed fraction of trips made using mode m by user class u
MS_{mu}	the modelled fraction of trips made using mode m by user class u

1 Introduction

This thesis is the final result of my graduation research at Goudappel Coffeng. The goal of this research was to improve the automatic parameter calibration of the gravity model that exists within the transport planning software package Omnitrans. The emphasis is on gravity models within large scale strategic traffic models. The gravity model for trip distribution is based upon Newton's gravity model. It generates a trip distribution over the zonal grid of the strategic traffic model. The most important element that determines this distribution of trips is a cost function inside the gravity model. The calibration considers the parameters of this cost function.

This section gives an introduction to the framework and context in which this research is done. Section 1.1 describes some basic trip properties which form the dimensions of the trip distribution resulting from the gravity model. Section 1.2 describes the four-step traffic model, which is the larger transport modelling context within which the gravity model is an important component.

1.1 Trip numbers

The strategic traffic model we deal with considers a study area which is assumed to be partitioned into a set of n zones i.e. a zonal grid. In the trip distribution model we are concerned with the origin and destination of trips as well as the mode by which the trip is made. We use i to denote a zone of origin from which a trip starts and j to denote a zone at which a trip ends. We use the index m to denote a mode of transport by which a trip is made. Examples of modes of transports include car, bike or public transit.

We have described three properties a trip has in the trip distribution model so far: a zone of origin, a zone of destination and the mode of transport by which it is made. Another trip property we are interested in is the person who makes the trip. We can partition the population of trip makers into groups so that in a group all the trip makers share one or more of the same properties. These groups we will call user classes and we use u to denote a particular user class. We can for example divide trip makers in the user classes of trip makers who own a car and trip makers who do not own a car: $u \in \{co, nco\}$.

With these indices we can specify the number of trips with certain properties. By the trip number variable t_{ijmu} we denote the number of trips that are made from zone i to zone j by a trip maker from user class u using m as a mode of transport for each pair of zones (i, j), mode m and user class u. Together all the t_{ijmu}'s form the trip distribution.

1.2 The four-step traffic model

The traffic and transportation modelling and forecasting done at Goudappel Coffeng, is usually based on the classical four-step traffic forecasting model. This model describes in four steps the process of modelling the movements of traffic on a network. The modelling is done for a particular real world study area. The study area itself is modelled by a zonal grid and a network representing roads and traffic infrastructure connecting the zones of the grid. The four steps are illustrated in figure 1 and are:

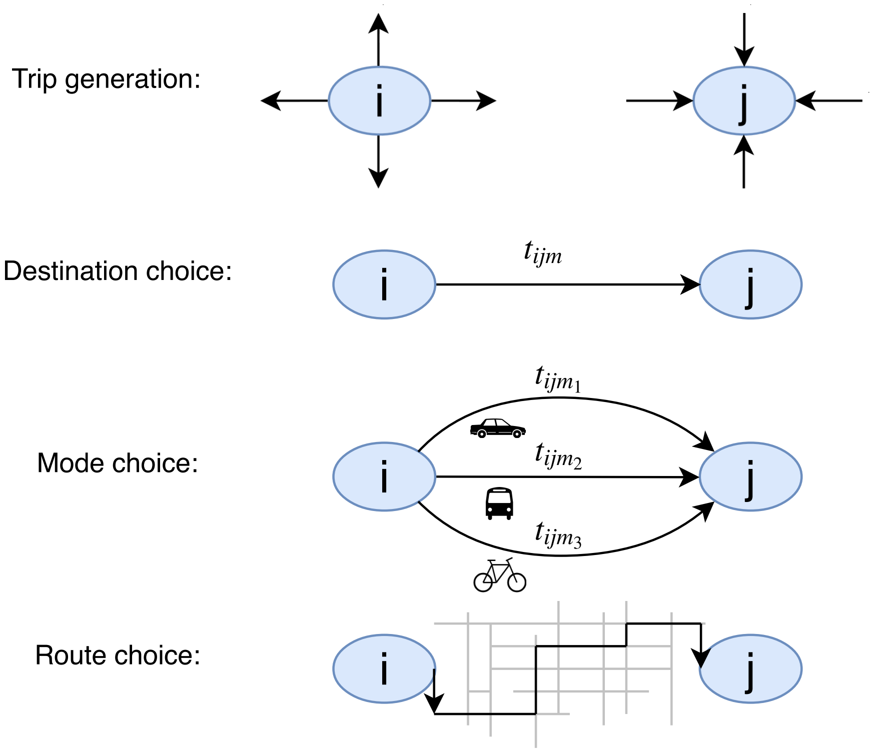


Figure 1: Steps of the four-step traffic model.

One - Trip generation:
The objective of the first step is to determine for each zone how many trips should have their origin (departures) and destination (arrivals) in that zone. The number of trips that have their origin in a particular zone is called the production of that zone while the number of trips that have their destination in that zone is called the attraction.

Two - Destination choice:
The second step matches the production and attraction values to get a trip distribution specifying the number of trips that go from each origin zone to each destination zone. Other than the production and attraction value of an origin-destination pair the main factor determining this trip distribution is travel impedance meaning roughly speaking the time and distance it takes to travel the trip.

Three - Mode choice:
The third step of mode choice, also called modal, split further refines the trip distribution resulting from the second step by determining for each trip which mode of transport is used.

Four - Route choice:
Having specified a origin-destination pair and mode of transport for each trip, the last step of the model is concerned with the route each trip takes in the network. Another name for this step commonly used is the traffic assignment step since the step assigns traffic to the network. The traffic assignment is an important step in the four-step model. As it calculates the load on the network which gives an estimation of the number of travelers and travelling time on real traffic facilities such as roads and railways. These are important to parties

such as traffic engineers and municipalities.

Usually at Goudappel Coffeng, and so is the case in this thesis, steps two and three, destination choice and mode choice, are combined in one trip distribution model which does both steps simultaneously. For this reason the specific trip distribution model we will be considering, which is a gravity model, is called a simultaneous gravity model.

This thesis will not explain the first step of trip generation other than mentioning that both production and attraction values are generated in an exogenous model based on socio-economic data. Therefore we assume the production and attraction values from this step to be readily available data in the next step of trip distribution. For a more thorough explanation on the trip generation and traffic assignment phase and a more through treatment of the four step model in general we refer to Ortur and Willumsen[5].

The goal of this research is to improve the calibration of cost function parameters of the simultaneous gravity model for large-scale (thousands of zones) strategic traffic models. Performance measures were established on which the developed methods were tested on. In particular we implemented gradient based methods and compared their performance with the original hillclimbing algorithm.

Structure of the remaining thesis

Chapter 2 introduces the reader to the gravity model of traffic by describing a basic doubly constrained simultaneous gravity model. In chapter 3 the full trip distribution model is described after which the actual act of calibrating it is discussed. An alternative calibration approach is discussed and chosen instead of the currently used approach within Goudappel Coffeng. In chapter 4 the problem of calibrating the trip distribution model by the new approach is cast into a mathematical optimization formulation. Further performance measures are identified by which potential methods can be judged. Chapter 5 discusses the currently used hillclimbing algorithm after which various potential alternative optimization algorithms are considered. A choice is made for gradient based methods. The details of a particular gradient based method which is the BFGS method, a so called Quasi Newton method, are further discussed in chapter 6. Chapter 7 discusses the results obtained from tests on these methods as well as results from comparing two different routines to calculate the gradient. Finally chapter 8 states conclusions and recommendations that follow from this research.

2 The simultaneous gravity model

The purpose of this chapter is to review the simultaneous gravity model within the four-step model. The simultaneous gravity model is the key building stone of the full trip distribution models that we review in chapter 3. Section 2.1 introduces a doubly constrained simultaneous gravity model for uniform trip makers. Section 2.2 discusses the choice of deterrence function that can be made inside the gravity model. The calibration of this deterrence function is the main research subject. Section 2.3 presents a mathematical derivation of the gravity model. Section 2.4 describes the solution method for the gravity model which is the biproportional fitting procedure and reviews some of its properties.

2.1 Doubly constrained simultaneous gravity model

This section describes the doubly constrained simultaneous gravity model. Wilson[15] founded the gravity model approach to trip distribution modelling. To simplify matters and to cater for the reader with no background knowledge on gravity models we first introduce a simultaneous gravity model in which the population of trip makers is assumed to be uniform. Therefore the index u for user classes is omitted for now and the model discussed here computes the trip distribution t_{ijm}. The model computes a trip distribution based on trip-end values, generalized costs and a definition of the deterrence functions.

2.1.1 Trip-end constraints

The trip-ends consist of production and attraction values obtained from the trip generation step. Denote by P_i the production value of zone i and denote by A_i its attraction value. As mentioned before: the production value of a zone represents the number of trips that should originate in that zone while the attraction value represents the number of trips that should have their destination in that zone. This is encapsulated by the trip-end constraints:

$$\sum_{m,j} t_{ijm} = P_i \qquad \text{for each } i \tag{1}$$

$$\sum_{m,i} t_{ijm} = A_j \qquad \text{for each } j$$

The gravity model is called doubly constrained if both production and attraction constraints are required to be met. Singly constrained models only require one constraint type to be met. The trip-end data can also be omitted or simply unavailable in which case the model is called unconstrained. So the choice of gravity model in terms of it being singly (origin or destination based), doubly or unconstrained depends on the level of knowledge in trip-ends.

2.1.2 Balancing trip-end values

Denote by T the total number of trips modelled i.e. the sum of all the t_{ijm}. It follows from the trip-end constraints (1) that:

$$\sum_i P_i = \sum_i \sum_{m,j} t_{ijm} = T = \sum_j \sum_{m,i} t_{ijm} = \sum_j A_j$$

However, the left- and right-hand side can be unequal in case the production model and attraction model (together forming the step of trip generation) are inconsistent[1]. This inconsistency can be corrected by a simple preprocessing step called balancing. Balancing simply means to either scale the production values such that

[1] In practice this depends on the time period that is modelled for. For a time period of a full day the productions and attractions should be approximately equal. This is for example not the case for the period of rush hour.

their sum matches the sum of attraction values or to scale the attraction values to match the total production sum. Which sum we hold as valid is determined by comparing the reliability of the models for productions and attractions. For example in case we trust the production values more we scale the attraction values towards adjusted values $A_j^{'}$ by:

$$A_j^{'} = \left(\frac{\sum_i P_i}{\sum_i A_i} \right) \cdot A_j$$

2.1.3 Generalized costs

Intuitively one would in general expect more people to travel between two locations given it is easier to make the trip. This is where the second main input to the gravity model of generalized travelling costs come in to play. Denote by c_{ijm} the generalized cost to travel from zone i to zone j using m as a mode of transport. Practically speaking the units of these costs can be anything from time, distance, energy or money hence they are called generalized costs. The generalized costs can also be a linear combination of these numbers by applying appropriate conversion coefficients. For example it is possible to express generalized costs in purely monetary units from travel time and travel distance in case we have knowledge of the value of time and fuel costs.

2.1.4 The gravity equation

The gravity equation can now be introduced by:

$$t_{ijm} = p_i a_j P_i A_j F^m(c_{ijm}) \tag{2}$$

Here $p_i \geq 0$ and $a_j \geq 0$ are called balancing factors which need to be determined so that the trip-end constraints hold. The cost function $F^m(.)$ is mode specific and acts on the generalized costs to model the willingness to travel at a certain (generalized) cost. The cost functions are also called deterrence functions.

We can rewrite the gravity equation (2) in a simpler form by a change of variables: $O_i = p_i \cdot P_i$, $D_j = a_j \cdot A_j$:

$$t_{ijm} = O_i D_j F^m(c_{ijm}) \tag{3}$$

We will actually use this equation as the gravity equation and by balancing factors we will mean these O_i's and D_j's. However the form of equation (2) is useful to observe three assumptions that are built into the gravity model. Namely that the number of trips between a zone of origin i and a destination zone j made by mode m is proportional to the observed number of departures P_i from the origin zone i, the observed number of arrivals A_j to the destination zone j and also is proportional to a cost function factor $F^m(c_{ijm})$ representing the willingness to travel at the cost to travel from zone i to zone j by mode of transport m.

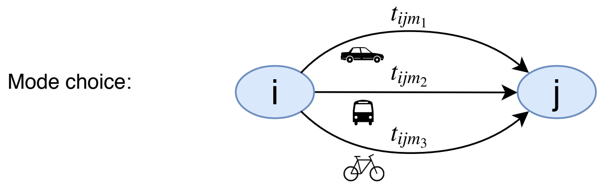


Figure 2: The simultaneous gravity model determines both destination choice and mode choice.

2.1.5 Resemblance with Newton's law

The gravity model of traffic is similar to Newton's law of universal gravitation therefore lending its name from it. Newton's gravitational law states that the gravitational pull between two objects is proportional to the mass of the first and second object and also is proportional to the inverse square of the distance between the objects, see figure 3. The gravity model of traffic usually uses a more general cost function however $F^m(c_{ijm}) = c_{ijm}^{-2}$ has been used in some models as a cost function. One way in which the analogy between Newton's law and the gravity model of traffic breaks down is that in Newton's law the calculated force works bidirectional i.e. the force object one exerts on object two F_2 is equal to the force object two exerts on object one F_1 while we do not have $t_{ijm} = t_{jim}$ necessarily. Since the costs to travel on the way in c_{ijm} and on the way out c_{jim} do not have to be equal and neither do the production and attraction value pairs.

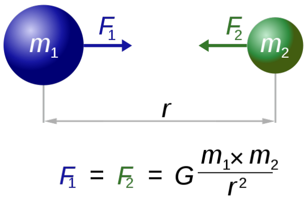


Figure 3: Newton's law of universal gravitation.

2.2 Deterrence functions

Different choices of cost functions $F^m(c_{ijm})$ are available. The choice of the cost function has a large influence on the trip distribution. The deterrence function values $F^m(c_{ijm})$ directly determine the relative distribution of the total trips over the different modes of transport i.e. the modal split between a zone-pair (i, j). To illustrate this, suppose we have a model with three modes m_1, m_2 and m_3. Then by the gravity equation (3) we have after cancelling out common factors O_i and D_j:

$$\frac{t_{ijm}}{t_{ijm_1} + t_{ijm_2} + t_{ijm_3}} = \frac{F^m(c_{ijm})}{F^{m_1}(c_{ijm_1}) + F^{m_2}(c_{ijm_2}) + F^{m_3}(c_{ijm_3})}, \quad for\ m \in \{m_1, m_2, m_3\}$$

2.2.1 Exponential deterrence function

The most standard deterrence function is the exponential deterrence function:

$$F^m(c_{ijm}) = e^{\beta_m c_{ijm}} \tag{4}$$

Here for each mode m the $\beta_m < 0$ is a calibration parameter that can be used to influence the trip distribution outcome. Picking a more negative β_m has the effect of modelling shorter trips for mode m (more trips of smaller generalized costs).

2.2.2 Lognormal deterrence function

In the case of a lognormal deterrence function two calibration parameters need to be specified: for each mode m we have again a $\beta_m < 0$ but now additionally we require for each mode an $\alpha_m > 0$:

$$F^m(c_{ijm}) = \alpha_m \cdot e^{\beta_m ln^2(c_{ijm}+1)} \tag{5}$$

The β_m parameters are similar to those of the exponential function. The newly introduced parameters α_m's allow for scaling of the deterrence functions.

The only difference between the lognormal and the exponential deterrence function, other than this scaling property, is that the costs are transformed by $c_{ijm} \to ln^2(c_{ijm} + 1)$. Figure 4 illustrates the shape of the lognormal distribution function for different parameter value choices.

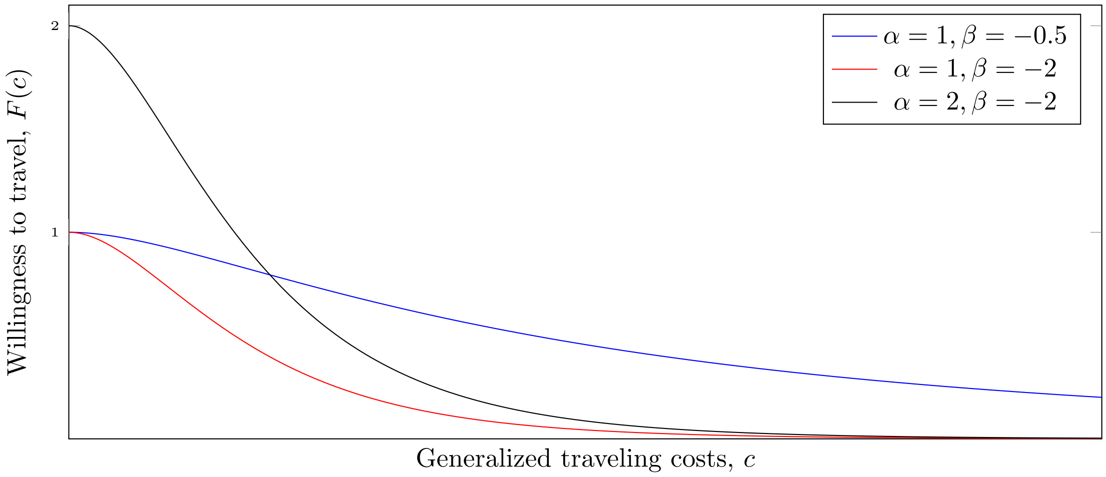


Figure 4: Lognormal deterrence functions for different values of parameters α and β.

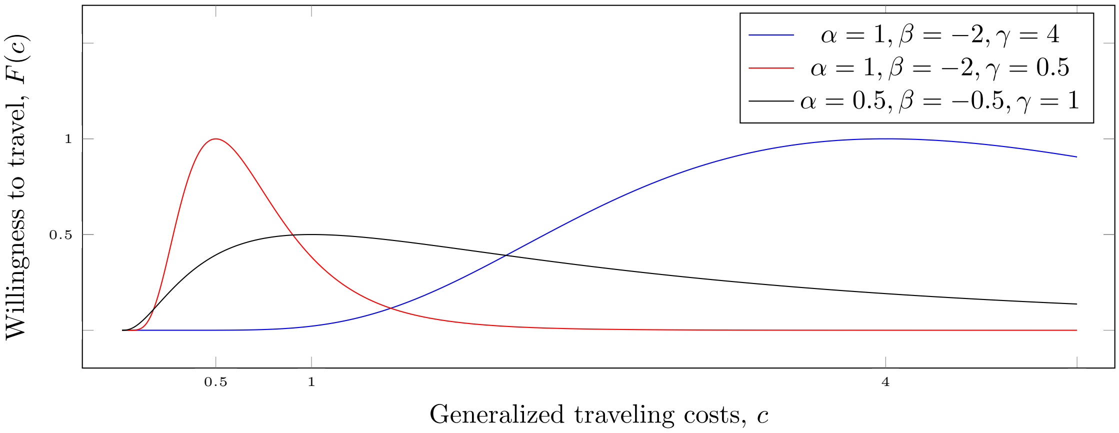


Figure 5: Top-lognormal deterrence functions for different values of parameters α, β and γ.

2.2.3 Top-lognormal deterrence function

A top-lognormal deterrence function is specified by an additional parameter $\gamma_m > 0$:

$$F^m(c_{ijm}) = \alpha_m \cdot e^{\beta_m ln^2(\frac{c_{ijm}}{\gamma})}$$

The maximum willingness to travel for the top-lognormal function type then occurs at costs γ_m, as illustrated in figure 5. A top-lognormal distribution function is mostly applied in unimodal strategic traffic models to model peak attractiveness of a mode of transport at costs other than zero (the peak for lognormal functions). For multimodal models, which this research focuses on, the simpler lognormal deterrence functions are sufficient in ensuring peak attractiveness of a mode is achieved at a target cost interval due to the competitiveness, regulated by the β_m parameters, between different modes.

2.2.4 Discrete deterrence function

Another possibility for a deterrence function is the discrete deterrence function. A discrete cost function is specified by n_c cost bin intervals $\bar{c}_k = [c_k, c_{k+1}]$, $k = 1, \ldots, n_c$ and constant values F_k^m that the function takes on over these intervals:

$$F^m(c_{ijm}) = \begin{cases} F_1^m, & if\ c_{ijm} \in \bar{c}_1 \\ F_2^m, & if\ c_{ijm} \in \bar{c}_2 \\ \vdots \\ F_{n_c}^m, & if\ c_{ijm} \in \bar{c}_{n_c} \end{cases}$$

An advantage of using a discrete cost function is that it allows for more flexibility in terms of generating a trip distribution. Too much flexibility can however lead to bad modelling practice. In the real world one expects the willingness to travel to always decrease for increasing travelling costs. As the deterrence function models this willingness to travel one should restrict the choice to monotonically decreasing discrete functions[2]. Figure 6 shows two examples of such a cost function.

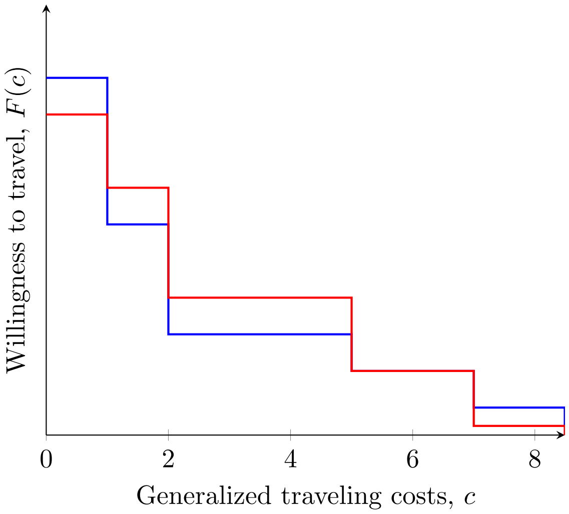


Figure 6: Two discrete distribution functions defined for the same set of cost bins

2.3 Gravity model derivation

In this section we present the mathematical basis for the choice of a gravity model as the trip distribution model. Gravity models have the property that their solution maximizes a quantity called entropy. This was first shown by Wilson[15]. Wilson derives the entropy maximizing property for a unimodal doubly constrained gravity model. Here we apply his approach for the multimodal gravity model described in section 2.1 that has an exponential cost function (4).

[2] For a unimodal traffic model one can, for the same reason a top-lognormal function is used, allow for unimodal discrete functions i.e. a discrete function $F(c)$ nondecreasing for $c \leq m$ and nonincreasing for $c \geq m$ for some $m \in \mathbb{R}$.

The gravity model aims at finding a trip distribution that is the optimum of a certain optimization problem. Namely it aims to find a trip distribution subject to the trip-end constraints (1) and an additional set of cost constraints that maximizes a quantity called entropy. The entropy $entropy(\boldsymbol{T})$ of a trip distribution $\boldsymbol{T}$ is a measure for the probability of the trip distribution occurring and is defined as:

$$entropy(\boldsymbol{T}) = \frac{T!}{\prod\limits_{i,j,m} t_{ijm}!}$$

Where T is again the total number of trips modelled. To make the maximization easier we apply Stirling's approximation $ln(N!) \approx N \cdot ln(N) - N$, we instead maximize the following function:

$$e(\boldsymbol{T}) = -\sum_{i,j,m} t_{ijm} \cdot ln(t_{ijm})$$

The additional set of cost constraints is:

$$\sum_{i,j} t_{ijm} c_{ijm} = C_m, \quad \forall m$$

This constraint states that the total amount of capital or budget spent in the region of interest on trips made by mode m is a fixed amount C_m. However these constants are assumed to be unknown.

Summarizing, the optimization problem is:

$$\max_{\boldsymbol{T}} -\sum_{i,j,m} t_{ijm} \cdot ln(t_{ijm}) \tag{6}$$

such that:

$$\sum_{j,m} t_{ijm} = P_i, \quad \forall i \tag{7}$$

$$\sum_{i,m} t_{ijm} = A_j, \quad \forall j$$

$$\sum_{i,j} t_{ijm} c_{ijm} = C_m, \quad \forall m$$

We now apply the method of Lagrange multipliers, with multipliers $\lambda_i^{(1)}$, $\lambda_i^{(2)}$ and β_m corresponding to these constraints, to obtain necessary conditions for local optima:

$$\begin{aligned} L = e(\boldsymbol{T}) &+ \sum_i \lambda_i^{(1)}(P_i - \sum_{j,m} t_{ijm}) \\ &+ \sum_j \lambda_j^{(2)}(A_j - \sum_{i,m} t_{ijm}) \\ &+ \sum_m \beta_m(C_m - \sum_{i,j} t_{ijm} c_{ijm}) \end{aligned}$$

Now we look for stationary points of the Lagrangian. The partial derivatives of the Lagrangian with respect to the trip numbers satisfy:

$$\frac{\partial L(\boldsymbol{T})}{\partial t_{ijm}} = ln(t_{ijm}) - \lambda_i^{(1)} - \lambda_j^{(2)} - \beta_m c_{ijm}, \quad \forall i, j, m$$

This partial derivative should vanish which happens exactly if:

$$t_{ijm} = e^{-\lambda_i^{(1)} - \lambda_j^{(2)} - \beta_m c_{ijm}}, \quad \forall i, j, m$$

To obtain the form of the gravity equation (3) we only have to apply two changes in variables:

$$O_i = e^{-\lambda_i^{(1)}}, \quad \forall i$$
$$D_j = e^{-\lambda_j^{(2)}}, \quad \forall j$$

Giving us:

$$t_{ijm} = O_i D_j e^{-\beta_m c_{ijm}}, \quad \forall i, j, m$$

Note that the partial derivatives of the Lagrangian with respect to the multipliers give us back the constraints (7).

As the objective function (6) is a concave function and the equality constraints (7) are affine functions the necessary conditions are also sufficient for a local optimum. In fact the concavity of the objective function also implies that any local maximum must be a global maximum as well. It follows that a trip distribution satisfying the gravity equation (3) maximizes entropy.

2.4 Solving the gravity model

2.4.1 OD matrices and skim matrices

Before presenting the solution method to the gravity model we first introduce some convenient matrix notation. It is convenient to think of a trip distribution as a set of matrices, specifically origin-destination matrices (OD matrices) or trip tables. For each mode m we define its OD matrix as the square matrix $\boldsymbol{T}^m \in \mathbb{R}^{nxn}_{\geq 0}$ where t_{ijm} is its $(i,j)^{th}$ entry.

In the same way define generalized cost matrices $\boldsymbol{C}^m \in \mathbb{R}^{nxn}_{\geq 0}$ for each mode m with c_{ijm} as the $(i,j)^{th}$ entry of $\boldsymbol{C}^m$. Generalized costs matrices are also called skim matrices.

2.4.2 Solution method

The basic doubly constrained simultaneous gravity model described so far can be formulated as:

$$t_{ijm} = O_i D_j F^m(c_{ijm})$$

$$\sum_{m,j} t_{ijm} = P_i \qquad \text{for each } i$$

$$\sum_{m,i} t_{ijm} = A_j \qquad \text{for each } j$$

$$O_i, D_i \geq 0 \qquad \text{for each } i \tag{8}$$

To find a trip distribution that satisfies the gravity model the Furness method is used. In mathematics, the Furness method is better known as IPF (iterative proportional fitting) or matrix raking. The Furness method is described as a series of iterations in which in the odd iterations we scale for each zone the current number of departures towards the target production value and in the even iterations we scale for each zone the current number of arrivals towards the target attraction value. This fitting procedure is started from an initial trip distribution where the trip numbers are equal to the product of the three proportionality factors: $t^0_{ijm} = P_i A_j F^m(c_{ijm})$ so the starting solution is $O_i = P_i$ and $D_i = A_i$ for each zone i.

In terms of the OD matrices $\boldsymbol{T}^m$ the method can be described as in the odd iterations scaling the row sums of the aggregated matrix $\boldsymbol{T} = \sum_m \boldsymbol{T}^m$ towards the target production values P_i and in the even iterations scaling the column sums of $\boldsymbol{T}$ towards the attraction values A_j. Note that the row sums always match the production values after an odd iteration, and similarly the column sums precisely match the attraction values after an even iteration. Algorithm 1 shows the pseudocode of this biproportional fitting procedure.

Algorithm 1 Biproportional fitting

```
for each zone $i$ do
    $O_i \leftarrow P_i$
    $D_i \leftarrow A_i$
end for
for each mode $m$ do
    for each od pair $(i, j)$ do
        $t_{ijm} \leftarrow O_i \cdot D_j \cdot F^m(c_{ijm})$
    end for
end for
while not converged do
    for each zone $i$ do
        if $\sum_{j,m} t_{ijm} > 0$ then f $\leftarrow \left(\frac{P_i}{\sum_{j,m} t_{ijm}}\right)$ else $f \leftarrow 0$
        $O_i \leftarrow f \cdot O_i$
        for each zone $j$ do
            for each mode $m$ do
                $t_{ijm} \leftarrow f \cdot t_{ijm}$
            end for
        end for
    end for
    for each zone $j$ do
        if $\sum_{i,m} t_{ijm} > 0$ then f $\leftarrow \left(\frac{A_j}{\sum_{i,m} t_{ijm}}\right)$ else $f \leftarrow 0$
        $D_j \leftarrow f \cdot D_j$
        for each zone $i$ do
            for each mode $m$ do
                $t_{ijm} \leftarrow f \cdot t_{ijm}$
            end for
        end for
    end for
end while
```

To illustrate the concepts discussed so far we now introduce a simple gravity model instance[3] and show how IPF solves it.

Example 2.1. We consider a small instance with two modes and three zones. The production and attraction values for the three zones are assumed to be given by:

$$\boldsymbol{P} = \begin{bmatrix} 80 \\ 50 \\ 20 \end{bmatrix}, \qquad \boldsymbol{A} = \begin{bmatrix} 20 & 30 & 100 \end{bmatrix}$$

Given generalized costs defined by the following skim matrices:

[3] By gravity model instance we mean a combination of trip-end values, skim matrices and deterrence function choice.

$$\boldsymbol{C}^{car} = \begin{bmatrix} 5 & 1 & 2 \\ 1 & 8 & 2 \\ 1 & 4 & 2 \end{bmatrix}, \qquad \boldsymbol{C}^{bike} = \begin{bmatrix} 2 & 6 & 3 \\ 6 & 5 & 5 \\ 2 & 1 & 4 \end{bmatrix}$$

Assuming lognormal deterrence functions with parameters specified by:

$$(\alpha_{car}, \alpha_{bike}) = (2, 1), \quad \text{and}$$

$$(\beta_{car}, \beta_{bike}) = (-0.5, -1)$$

Then the initial trip distribution can be calculated via $t^0_{ijm} = P_i F^m(c_{ijm}) A_j$ (this corresponds with initial balancing factors $O_i = P_i$ and $D_i = A_i$, see algorithm 1):

$$\boldsymbol{T}^0_{car} = \begin{bmatrix} 642.7 & 3775 & 8750.5 \\ 1572.9 & 268.4 & 5469.1 \\ 629.2 & 328.6 & 2187.6 \end{bmatrix}, \qquad \boldsymbol{T}^0_{bike} = \begin{bmatrix} 64.5 & 1484.4 & 2392.9 \\ 618.5 & 12.0 & 1495.5 \\ 247.4 & 45.0 & 598.2 \end{bmatrix}$$

The first balancing step we scale the rows towards the target production values, this is done by multiplying row factors $\boldsymbol{f}$

$$\boldsymbol{f} = \begin{bmatrix} 0.0047 \\ 0.0053 \\ 0.0050 \end{bmatrix}$$

to the rows of the OD matrices sucht that the resulting row sums match the production values:

$$\boldsymbol{T}^1_{car} = \begin{bmatrix} 3.005 & 17.650 & 40.914 \\ 8.334 & 1.422 & 28.979 \\ 3.118 & 1.629 & 10.841 \end{bmatrix}, \qquad \boldsymbol{T}^1_{bike} = \begin{bmatrix} 0.302 & 6.941 & 11.188 \\ 3.277 & 0.064 & 7.924 \\ 1.226 & 0.223 & 2.964 \end{bmatrix}$$

The row factors of $\boldsymbol{f}$ are also multiplied with the production balancing factors to update them:

$$\boldsymbol{O} = \begin{bmatrix} 0.3740 \\ 0.2649 \\ 0.0991 \end{bmatrix}$$

The next step consists of scaling the columns towards the target attraction values, we get column factors

$$\boldsymbol{f} = \begin{bmatrix} 1.0383 & 1.0742 & 0.9727 \end{bmatrix}$$

Which update the trip distribution such that the target attraction values are met:

$$\boldsymbol{T}^2_{car} = \begin{bmatrix} 3.120 & 18.960 & 39.796 \\ 8.654 & 1.5276 & 28.187 \\ 3.237 & 1.7493 & 10.544 \end{bmatrix}, \qquad \boldsymbol{T}^2_{bike} = \begin{bmatrix} 0.3133 & 7.4554 & 10.882 \\ 3.4028 & 0.0683 & 7.7077 \\ 1.2729 & 0.2395 & 2.8834 \end{bmatrix}$$

The attraction balancing factors become:

$$\boldsymbol{D} = \begin{bmatrix} 20.766 & 32.226 & 97.267 \end{bmatrix}$$

After just two balancing steps the resulting distribution is already matching the trip-ends quite closely. Continuing the balancing process, the distribution would get closer and closer towards the exact solution.

□

2.4.3 Solution properties and convergence of the biproportional fitting procedure:

Uniqueness

Note that given feasible balancing factors exist, they are not unique. Assuming $(\boldsymbol{O}, \boldsymbol{D})$ is a solution to the trip end constraints in (8) we clearly have for any $\lambda > 0$ that $(\frac{1}{\lambda} \cdot \boldsymbol{O}, \lambda \cdot \boldsymbol{D})$ satisfies the trip-end constraints as well. However given a solution exists the balancing factors are unique up to this constant factor i.e. we only have $\lambda > 0$ as a degree of freedom. This is proved in theorem 2.1. In what follows the balancingfactors $\boldsymbol{O}$, $\boldsymbol{D}$ are represented either by a column and row vector or $\boldsymbol{O}$, $\boldsymbol{D}$ represent diagonal matrices with diagonal entries e.g. $O_{ii} = O_i$, depending on context.

Theorem 2.1. Uniqueness of trip distribution and balancingfactors.

Let $\boldsymbol{M} \in \mathbb{R}^{mxn}_{\geq 0}$ be a matrix containing no zero rows or columns. Suppose we have production and attraction values $\boldsymbol{P} \in \mathbb{R}^m_{>0}$, $\boldsymbol{A}^T \in \mathbb{R}^n_{>0}$. Suppose we have two sets of production and attraction balancing factors $\boldsymbol{O}, \tilde{\boldsymbol{O}} \in \mathbb{R}^{mxm}_{>0}$ and $\boldsymbol{D}, \tilde{\boldsymbol{D}} \in \mathbb{R}^{nxn}_{>0}$ such that both the matrices $\boldsymbol{T} = \boldsymbol{OMD}$ and $\tilde{\boldsymbol{T}} = \tilde{\boldsymbol{O}}\boldsymbol{M}\tilde{\boldsymbol{D}}$ satisfy the productions $\boldsymbol{P}$ and attractions $\boldsymbol{A}$. Then the following two statements hold:

i): The balanced matrices are equal i.e. $\boldsymbol{T} = \tilde{\boldsymbol{T}}$
ii): There exists a constant $\lambda > 0$ s.t. $\tilde{\boldsymbol{O}} = \frac{1}{\lambda} \cdot \boldsymbol{O}$ and $\tilde{\boldsymbol{D}} = \lambda \cdot \boldsymbol{D}$

Proof:
Statement (i) follows directly from theorem 4 in Rothblum[12]. Then with (i) holding (ii) can be proved quite easily: Since $\boldsymbol{OMD} = \tilde{\boldsymbol{O}}\boldsymbol{M}\tilde{\boldsymbol{D}}$ we must have $\boldsymbol{OD} = \tilde{\boldsymbol{O}}\tilde{\boldsymbol{D}}$ (here $\boldsymbol{O}$, $\tilde{\boldsymbol{O}}$ represent columns and $\boldsymbol{D}$, $\tilde{\boldsymbol{D}}$ rows). Then the jth columns of these must be equal i.e. $D_j \cdot \boldsymbol{O} = \tilde{D}_j \cdot \tilde{\boldsymbol{O}}$. Since the balancingfactors are positive we can safely take $\lambda = \frac{\tilde{D}_j}{D_j} > 0$. Comparing the ith rows of $\boldsymbol{OD} = \tilde{\boldsymbol{O}}\tilde{\boldsymbol{D}}$ confirms that $\tilde{\boldsymbol{D}} = \lambda \cdot \boldsymbol{D}$ holds as well for this λ.

□

To apply the theorem to the initial matrix in the biproportional fitting algorithm one should set $\boldsymbol{M} = \sum_m F^m(\boldsymbol{C}^m)$. Theorem 2.1 allows the matrix $\boldsymbol{M}$ to contain zero entries however its rows and columns should not vanish. The other restriction is that the production and attraction values should be strictly positive. For the gravity models considered these restrictions were usually violated. However the theorem is still applicable as the instances can be reduced by eliminating the zero rows and columns as well as the rows and columns of zero production and attraction.

Existence of solution and convergence of biproportional fitting procedure

In example 2.1 it was shown the biproportional fitting procedure converged for the given instance. An important question that arises is whether this is the case for every instance. The answer is no as the following counterexample taken from Pukelsheim[11] proves.

Example 2.2. Consider the following unimodal (of one mode) instance with two zones and the following trip-ends:

$$\boldsymbol{P} = \begin{bmatrix} 4 \\ 2 \end{bmatrix}, \qquad \boldsymbol{A} = \begin{bmatrix} 2 & 4 \end{bmatrix}$$

Suppose the costs and cost function lead to the initial OD matrix:

$$\boldsymbol{T}^0 = \begin{bmatrix} 30 & 0 \\ 10 & 20 \end{bmatrix}$$

The fact that $t_{12} = 0$ could have as an explanation that zone 2 is unreachable from zone 1 which would have been modelled by setting $c_{12} = \infty$. One can show that the biproportional fitting procedure does not converge in this case but "oscillates between two distinct accumulation points"[11]:

$$\lim_{t=1,3,\ldots} \boldsymbol{T}^t = \begin{bmatrix} 4 & 0 \\ 0 & 2 \end{bmatrix}, \quad \text{and} \quad \lim_{t=2,4,\ldots} \boldsymbol{T}^t = \begin{bmatrix} 2 & 0 \\ 0 & 4 \end{bmatrix}$$

□

Pukelsheim[11] also establishes necessary and sufficient conditions for convergence of the biproportional fitting procedure. His analysis and theorem make use of the so called L_1-error. The L_1-error for the OD matrix in the k_{th} iteration $\boldsymbol{T}^k$ denoted by $f(k)$ is calculated by:

$$f(k) = \frac{1}{2}\sum_i \left| \sum_j t_{ij}^k - P_i \right| + \frac{1}{2}\sum_j \left| \sum_i t_{ij}^k - A_j \right|$$

A column j of the matrix $\boldsymbol{T}^0$ is said to be connected to a row i of that matrix if $t_{ij}^0 > 0$. For a subset of rows I define $J(I)$ to be the subset of columns connected to I in the initial OD matrix $\boldsymbol{T}^0$ i.e. $J(I)$ contains all columns containing a positive entry in some row of I. Pukelsheim's main result can now be stated:

Theorem 2.2. Convergence of the biproportional fitting procedure (Pukelsheim[11], 2009)

For an initial matrix $\boldsymbol{T}^0 \in \mathbb{R}^{mxn}_{\geq 0}$ with no zero rows or columns, production and attraction values $\boldsymbol{P} \in \mathbb{R}^m_{>0}$, $\boldsymbol{A}^T \in \mathbb{R}^n_{>0}$ we have for the limit of the L_1-error during the biproportional fitting procedure:

$$\lim_{k\to\infty} f(k) = \max_{I\subseteq\{1,\ldots,m\}} \left(\sum_{i\in I} P_i - \sum_{j\in J(I)} A_j \right) \tag{9}$$

Moreover, the biproportional fitting procedure converges if and only if this limit is zero.

Again we can simply reduce gravity model instances by eliminating zero rows and columns as well as the rows and columns of zero production and attraction so that the theorem essentially applies to all instances of interest to us.

Some expressions in the maximum of (9) can be computed to possibly detect non-feasibility without actually running the biproportional fitting procedure. However, for a general instance, theorem 2.2 is of little use in proving the biproportional fitting procedure will converge. As determining whether the maximum expression in (9) is greater than zero has a running time of $O(2^m)$. Therefore deciding in general whether feasible balancing factors exist is likely best done by just running the biproportional fitting procedure. In this research we found that the fitting procedures converged for each gravity model instance encountered. This suggests that nonconvergence is rare at least in our specific setting.

Purpose	Sub purpose	Mode-user class combinations	Productions	Attractions
Work	Home → Work:	All modes	$\boldsymbol{P}^{co}$, $\boldsymbol{P}^{nco}$	$\boldsymbol{A}$
	Work → Home:	All modes	$\boldsymbol{P}$	$\boldsymbol{A}^{co}$, $\boldsymbol{A}^{nco}$
Business	Home → Business:	All modes	$\boldsymbol{P}^{co}$, $\boldsymbol{P}^{nco}$	$\boldsymbol{A}$
	Business → Home:	All modes	$\boldsymbol{P}$	$\boldsymbol{A}^{co}$, $\boldsymbol{A}^{nco}$
	Business (home unrelated) - co:	co modes	$\boldsymbol{P}^{co}$	$\boldsymbol{A}^{co}$
	Business (home unrelated) - nco:	nco modes	$\boldsymbol{P}^{nco}$	$\boldsymbol{A}^{nco}$
Education	Home → Education:	All modes	$\boldsymbol{P}^{co}$, $\boldsymbol{P}^{nco}$	$\boldsymbol{A}$
	Education → Home:	All modes	$\boldsymbol{P}$	$\boldsymbol{A}^{co}$, $\boldsymbol{A}^{nco}$
Stores	Home → Stores:	All modes	$\boldsymbol{P}^{co}$, $\boldsymbol{P}^{nco}$	$\boldsymbol{A}$
	Stores → Home:	All modes	$\boldsymbol{P}$	$\boldsymbol{A}^{co}$, $\boldsymbol{A}^{nco}$
Other	Home → Other - co:	co modes	$\boldsymbol{P}^{co}$	$\boldsymbol{A}^{co}$
	Home → Other - nco:	nco modes	$\boldsymbol{P}^{nco}$	$\boldsymbol{A}^{nco}$

Table 1: Summary of full trip distribution model for the strategic traffic models considered. Most multi-modal models used within Goudappel Coffeng have a similar structure.

3 Calibration of the simultaneous gravity model

This chapter gives a detailed explanation of the trip distribution model and the process of calibrating it which is the focus of this thesis. Section 3.1 first describes the trip distribution model for all purposes and user classes. After that we turn to the main research problem of this thesis, namely the calibration of the gravity model parameters. Section 3.2 discusses within which context the calibration is done and its purpose. Section 3.3 discusses the criteria upon which the calibration process is based. Namely observed trip length distributions and from these derived observed modal splits. Section 3.4 discusses how the modelled counterparts to these observed distributions are calculated. Then section 3.5 first explains the earlier approach to calibrating the lognormal distribution function at Goudappel Coffeng after which a new approach is presented and discussed in section 3.6.

3.1 Full trip distribution model

This subsection describes the full trip distribution model used within Goudappel Coffeng which includes user classes and a stratification in trip purposes.

3.1.1 Trip purposes and sub purposes

In the real world trips are made for different purposes. For example trips made by people commuting can be labeled with the Work purpose. A trip of a purpose here is further divided into sub purposes which often classify the direction of the trip. In the literature Abdel-Aal[2] also uses purposes in the context of gravity model calibration. The first and second column of table 1 (page 20) show for the models we will be concerned with the stratification in purposes and sub purposes. Each sub purpose is modelled by a separate gravity model.

3.1.2 User classes

User classes add another dimension to the trip distribution model and add another index u to the trip number variables: t_{ijmu}. Instead of estimating an OD matrix $\boldsymbol{T}^m$ for each mode m the gravity model now estimates an OD matrix $\boldsymbol{T}^{mu}$ for each mode-user class pair (m, u). The way in which user classes are exactly embedded into the gravity model depends on the sub purpose data. We distinguish two cases: In case of symmetric productions and attractions, by which we mean the number of production and attraction values are equal, the sub purpose is simply modelled by the gravity model already encountered. In case of asymmetric productions and attractions the gravity model is extended in an appropriate way. We now discuss these two cases in more detail and in relation to table 1.

Asymmetric productions and attractions

We illustrate the two different gravity models with asymmetric productions and attractions by considering the Work purpose. As shown in table 1 the Work purpose has two sub purposes which represent the directions 'Home → Work' and 'Work → Home'. First we consider the 'Home → Work' direction. For this direction more refined data is given regarding its production values: they are distributed over car owners and non car owners, however the attraction side is aggregated. Trips with the 'Home → Work' direction are modelled by a gravity model of the form:

$$t_{ijmu} = O_{iu} D_j F^{mu}(c_{ijm}), \quad \forall i, j, m, u$$

s.t.

$$\sum_{j,m} t_{ijmu} = P_{iu}, \quad \forall i, u$$

$$\sum_{i,m,u} t_{ijmu} = A_j, \quad \forall j \tag{10}$$

Trips of the reverse 'Work → Home' direction are modelled by a separate gravity model of the form:

$$t_{ijmu} = O_i D_{ju} F^{mu}(c_{ijm}), \quad \forall i, j, m, u$$

s.t.

$$\sum_{j,m,u} t_{ijmu} = P_i, \quad \forall i$$

$$\sum_{i,m} t_{ijmu} = A_{ju}, \quad \forall j, u \tag{11}$$

The Education and Stores purpose as well as the two sub purposes taken together of the Business purpose are modelled the same way as the Work purpose just described. Each of these have a sub purpose for home based trips, meaning the origins of the trips represent homes while the destinations represent places related to the trips purpose e.g. offices, stores or schools, and each has a reversed direction sub purpose for which the role of origins and destinations are interchanged. The home based sub purpose has distinct production values for both user classes P_i^{co}, P_i^{nco} but single aggregated attraction values A_j. For the sub purpose of the reverse direction the reverse statement holds.

The solution procedure to these gravity models with asymmetric production and attraction values is described in algorithm 2 and is essentially still the biproportional fitting procedure of algorithm 1. In case of user class specific production values we just get double the rows for the aggregate matrix $\boldsymbol{T}$ if we define the OD matrices $\boldsymbol{T}^m$ as stacking the matrices $\boldsymbol{T}^{mu}$ vertically i.e.

$$\boldsymbol{T}^m = \left[\begin{array}{c} \boldsymbol{T}^{m,co} \\ \hdashline \boldsymbol{T}^{m,nco} \end{array}\right]$$

Similarly in case of user class specific attraction values the aggregate matrix $\boldsymbol{T}$ has double the columns with OD matrices $\boldsymbol{T}^m = [\boldsymbol{T}^{m,co} \;\vdots\; \boldsymbol{T}^{m,nco}]$. So the problem algorithm 2 solves is still the doubly constrained gravity model of chapter 2 and it is still the biproportional fitting algorithm. Therefore the solution and convergence properties discussed in section 2.4.3 also hold for these extensions to user classes.

Algorithm 2 Biproportional fitting with user class specific production values and aggregate attraction values for two user classes $u \in \{co, nco\}$

.

```
for each zone $i$ do
    $O_{iu} \leftarrow P_{iu}$
    $D_i \leftarrow A_i$
end for
for each mode-user class pair $(m, u)$ do
    for each od pair $(i, j)$ do
        $t_{ijmu} \leftarrow O_{iu} \cdot D_j \cdot F^{mu}(c_{ijm})$
    end for
end for
while not converged do
    for each zone $j$ do
        if $\sum_{i,m,u} t_{ijmu} > 0$ then f $\leftarrow \left(\frac{A_j}{\sum_{i,m,u} t_{ijm}}\right)$ else $f \leftarrow 0$
        $D_j \leftarrow f \cdot D_j$
        for each zone $i$ do
            for each mode-user class pair $(m, u)$ do
                $t_{ijmu} \leftarrow f \cdot t_{ijmu}$
            end for
        end for
    end for
    for each user class $u \in \{co, nco\}$ do
        for each zone $i$ do
            if $\sum_{j,m} t_{ijmu} > 0$ then f $\leftarrow \left(\frac{P_{iu}}{\sum_{j,m} t_{ijmu}}\right)$ else $f \leftarrow 0$
            $O_{iu} \leftarrow f \cdot O_{iu}$
            for each zone $j$ do
                for each mode $m$ do
                    $t_{ijmu} \leftarrow f \cdot t_{ijmu}$
                end for
            end for
        end for
    end for
end while
```

Here the production balancing steps are done last in the while loop. This is done because we are more confident in the accuracy of the production values (as the productions are known separately per user class[4]). Balancing the productions last has the result that the production constraints are met exactly. For the same reason in the balancing preprocessing step the attraction values should be balanced towards the total of productions.

Symmetric productions and attractions

For the Other purpose only home based trips are modelled. However both user class specific production values and user class specific attraction values are available for these trips and so two doubly constrained gravity

[4] Also because the production data for the sub purposes 'Home → Work/Business/..' are more reliable in itself. In the other direction the attraction data for the sub purposes 'Work/Business/.. → Home' are more reliable itself. In general the home-based trip-ends are the most reliable.

models are used. One for the car owner user class and one for the non car owner user class. These gravity models are therefore entirely equivalent to the gravity model discussed in chapter 2. For the Business purpose we have two extra sub purposes that model nonhome related trips i.e. business trips for which neither the origin or destination represents a home. Each of these sub purposes models a user class separately similar to the Other purpose.

OD matrix aggregation

We have described how each sub purpose within a trip purpose is modelled by a separate gravity model. However in the rest of this thesis we are interested in the aggregate trip numbers for the trip purposes and so the outcomes of the separate gravity models are aggregated again. Thus for each purpose the OD matrices $\boldsymbol{T}^{mu}$ are equal to the sum of the OD matrices of its respective sub purposes.

3.2 Calibration of deterrence function behavioral parameters

The calibration of the trip distribution model deals with the choice of parameters β_{mu} which appear in the exponential and lognormal cost functions inside the gravity model and additionally α_{mu} parameters in case of lognormal cost functions. These parameters are called behavioral parameters since they specify behavior of trip makers. The β_{mu}'s model the propensity to make less costly trips for all mode-user class combinations while the α_{mu}'s influence for each user class what percentage of its trip makers use a certain mode of transport i.e. the modal split of the user class. The behavior we try to encapsulate in these parameters is dependent on trip purpose. For example we generally expect trips to school or the shop to be shorter than the commute to work. Therefore the trip distribution model of each purpose is calibrated separately. However the gravity models of the sub purposes that constitute the trip distribution of a purpose share the same set of behavioral parameters. At Goudappel Coffeng the cost function used for the strategic traffic models is usually the lognormal one. The lognormal deterrence function in the case of user classes is of the form:

$$F^{mu}(c_{ijm}) = \alpha_{mu} \cdot e^{\beta_{mu} ln^2(c_{ijm}+1)} \tag{12}$$

In this thesis we also assume this lognormal deterrence function so the calibration focuses on both the β_{mu}'s and α_{mu}'s parameters. Note that in (12) the user class index u is omitted in c_{ijm} as generalized costs are the same for different trip makers.

Transferability of parameters

The production and attractions balancing factors O_i and D_j act upon a subset of cells (rows and columns) of the OD matrices while the behavioral parameters α_{mu} and β_{mu} act upon the whole matrices. The latter parameters can be called transferrable parameters. To explain why: suppose we have successfully calibrated the behavioural parameters. The main use of these calibrated behavioural parameters then is to serve as an input for four-step models that consider the same study area but consider different trip-end values or skim matrices in the gravity model. This is done to for example simulate the effect of changes in infrastructure and or zonal characteristics on traffic movement patterns. Changes in infrastructure are modelled by adjusting the road network or cost skim matrices and changes in zonal characteristics are modelled by adjusting the socio-economic data which changes the trip-end values of zones.

The production and attraction balancing factors are not transferable to other modelling contexts, as they are naturally sensitive to their associated production and attraction values, while the behavioral parameters in the calibration are mainly sensitive to empirical data outside the gravity model such as the observed trip length distributions discussed in the next section.

3.3 Calibration criteria

The calibration of the behavioral parameters is based on both empirical trip length distributions and modal splits. First we describe what these observed trip length distributions are and how observed modal splits are derived from them. Then we show how the count numbers on which the observed trip length distributions are based can be used to construct confidence intervals.

3.3.1 Empirical trip length distributions and modal splits

For each of the traveling purposes we have for each mode-user class combination (m, u) an empirical trip length distribution $\widehat{\boldsymbol{d}}_{mu}$ available. To obtain such distributions surveys are done for the study area in which respondents were asked to keep a travel diary, for a randomly selected week of the year, in which they keep track of all the trips they made that week. From this data, the number of trips per purpose, mode, user class and distance class was derived and scaled to account for the sampling ratio. In figure 7 we see, in blue, such a resulting distribution.

From the observed trip length distribution we can, given a user class u, derive for a mode m its observed modal split percentage $\widehat{MS}_{mu}$ within that user class by:

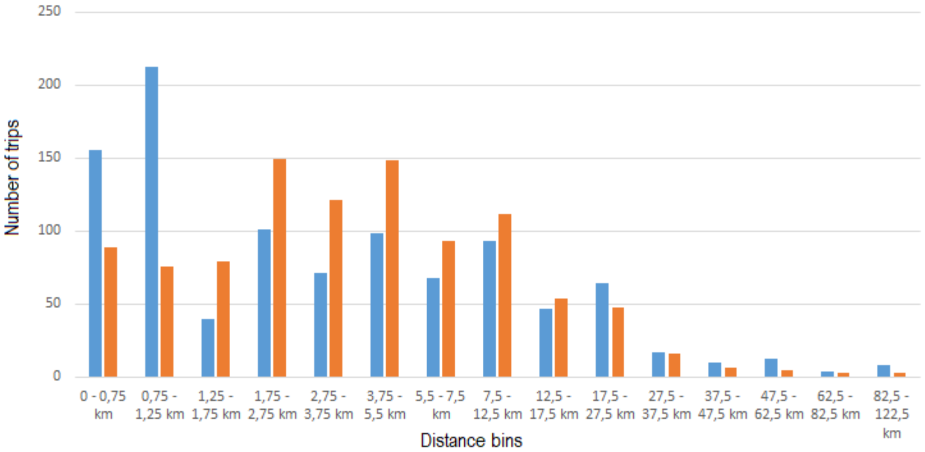


Figure 7: An observed (blue) and modelled (red) trip length distribution for the purpose 'Work', mode car and user class car owners.

$$\widehat{MS}_{mu} = \frac{\sum_k \widehat{d}_{muk}}{\sum_{\tilde{m},k} \widehat{d}_{\tilde{m}uk}}$$

Both modelled trip length distributions and modelled modal splits are compared to their observed counterparts. In section 3.4 we describe how modelled trip length distributions and modelled modal splits are computed from the OD matrices.

3.3.2 Approximate confidence intervals

This section describes how approximate confidence intervals can be constructed for the observed trip length distributions in case count numbers for these are available.

Let N be the total number of counted trips in a survey for some mode-user class combination (m, u). Of these N counts denote by $\widehat{k_i}$ the number of trips with a length in the i^{th} distance bin. Denote by p_i the true fraction of trips made with a length in the i^{th} distance bin. Then for each bin k the fraction $\widehat{p_i} = \frac{\widehat{k_i}}{N}$ is a point estimate of p_i. We now provide a measure of the accuracy of these point estimates in the form of approximate confidence intervals for the p_k. For more information regarding the construction of binomial confidence intervals we refer to Wallis[14].

From the theory of confidence intervals it follows that an approximate $100(1-\alpha)\%$ - confidence interval for p_i is:

$$p_i \in \left[\widehat{p_i} - z_{\alpha/2}\sqrt{\frac{\widehat{p_i}(1-\widehat{p_i})}{N}}, \widehat{p_i} + z_{\alpha/2}\sqrt{\frac{\widehat{p_i}(1-\widehat{p_i})}{N}}\right]$$

Where α is the desired significance level and $z_{\alpha/2}$ is the (1 - $\alpha/2$)-percentile of the standard normal distribution. In case $k_i = 0$ the so called rule of three from statistics can be used which assigns the interval $[0, \frac{3}{N}]$ to p_i. The confidence intervals can be interpreted in the following way: For a significance level of α we can expect the constructed intervals to contain the true fraction of trips p_i approximately $100(1-\alpha)\%$ of the time. The relative error r_i can then be calculated by:

$$r_i = \frac{z_{\alpha/2}\sqrt{\frac{\widehat{p_i}(1-\widehat{p_i})}{N}}}{\widehat{p_i}} = z_{\alpha/2}\sqrt{\frac{1}{\widehat{k_i}} - \frac{1}{N}} \approx \frac{z_{\alpha/2}}{\sqrt{\widehat{k_i}}}, \quad \textit{for large } N \tag{13}$$

Thus the width of the confidence intervals is inversely proportional to the square root of the number of observations. The confidence intervals can be plotted as error bars around the observed trip length distributions and form a visual guideline when inspecting the fit of the model to the observed distributions. Another use of these confidence intervals is to incorporate them directly in the objective function by defining weights for the objective function for each distance bin that are inversely proportional to the width of the corresponding confidence interval. So in fact these weights are proportional to the square root of the number of observations.

Overestimation of counts

In general there will be a dependency between the trips a survey respondent makes in the same day. The most important dependency is the one resulting from tours i.e. back-and-forth displacements. In the worst case all observations are tours and then the number of observations is overestimated by a factor of 2, leading to an overestimation in confidence by a factor of $\sqrt{2}$. However the extent to which this is a problem depends upon the time period for which the strategic traffic model is used. For a complete day many tours can be expected to be in the data while one would expect less tours for survey data specific to the morning commute.

3.3.3 Normalization of trip distributions by binwidth

The bins on which the trip distributions are defined are of varying widths. This leads to biased trip length distributions as wider bins are more likely to have more trip counts because a given trip is more likely to fall into a wider bin. To interpret the trip distributions more fairly we can normalize them by bin width. In figure 8 an observed and modelled normalized trip distribution are shown together with confidence bounds, in dotted marker style, as constructed in section 3.3.2. Here the normalized trip numbers are plotted against the bin midpoints to give a better sense of how many trips are made of a certain length.

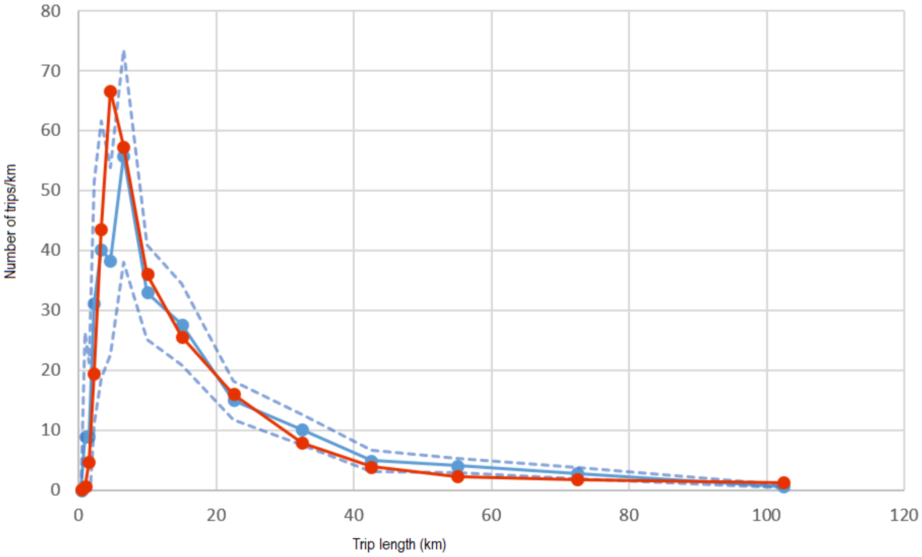
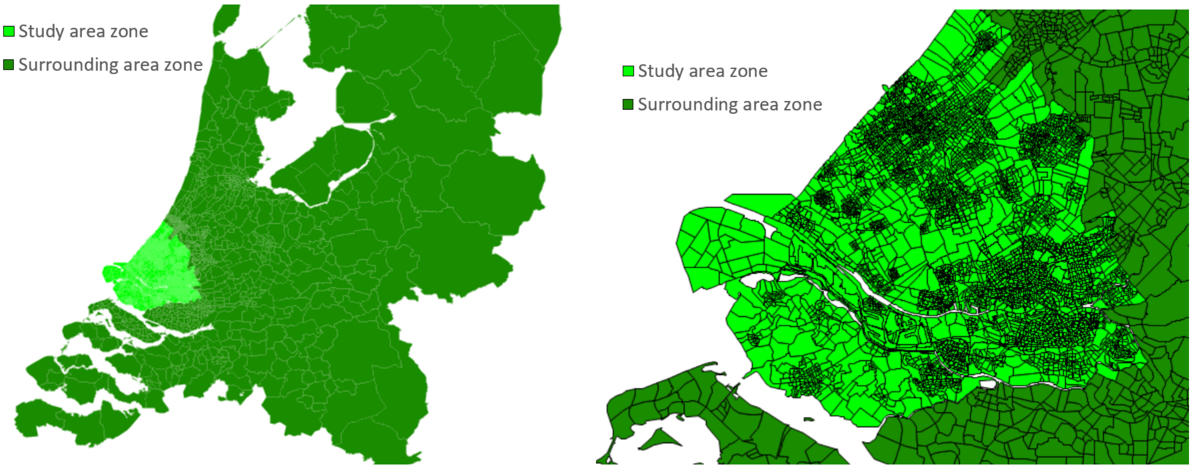


Figure 8: Bin width normalized observed (blue) and modelled (red) trip length distribution with confidence bounds (dotted)

3.4 Modelled trip length distributions and modal splits

Since in reality the study area is not a closed system in terms of traffic displacements the strategic traffic model also considers zones representing the area surrounding the study-area instead of only those making up the study area. To illustrate this figure 9 shows the zonal grid of the study area and surrounding area for the The Hague strategic traffic city model or MRDH model. To save computational effort, generally the size of zones outside the study area increases as the distance to the study area increases.



(a) The Hague city region inside the Netherlands

(b) Subdivision of the region into zones

Figure 9

The classification of zones in study-area and surrounding-area zones also brings about a classification in trips. Namely we distinguish study-area related trips which are trips that either start or end in a study area zone and non study-area related or external trips that both start and end in a zone outside the study area. We need to go one step further by partitioning the study-area related trips into internal, ingoing- and outgoing trips for which we use figure 10 as a definition of these.

The survey counts only include observations of study-area related trips. However these are not counted equally. The number of times a study-area related trip is counted, is equal to the number of times its zone of departure or arrival is a study area zone. So internal trips, in- and outgoing trips, and external trips are counted respectively twice, once and never in the construction of the observed trip length distributions. Now for the purpose of having a fair calibration criterium the modelled trip length distributions (resulting from the gravity models) are computed using the same weights.

Let s_{ijm} denote the length of a trip from zone i to zone j made by mode m and the k's denote length bins (the same bins as those used for the empirical trip length distributions). The zone-to-zone distances logically depend on mode and not on user class. For easier notation in what follows let S_{mk} denote the set of zone-pairs such that the distance between them is in bin k, so $S_{mk} = \{(i,j) \mid s_{ijm} \in s_k\}$. Then the modelled trip length distribution d_{muk} is given by:

$$d_{muk} = 2 \cdot \sum_{(i,j) \in Internal \cap S_{mk}} t_{ijmu} + \sum_{(i,j) \in Ingoing \cap S_{mk}} t_{ijmu} + \sum_{(i,j) \in Outgoing \cap S_{mk}} t_{ijmu}$$

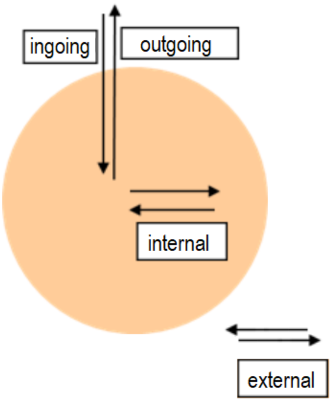


Figure 10: Trips related to the study area: ingoing, outgoing and internal trips versus nonrelated external trips

The modelled modal splits can then be calculated from these trip length distributions by:

$$MS_{mu} = \frac{\sum_k d_{muk}}{\sum_{\tilde{m},k} d_{\tilde{m}uk}} \tag{14}$$

Here we should mention that the trip numbers t_{ijmu} actually denote totals of some trip purpose i.e. they are trip numbers aggregated over the sub purposes of that purpose.

Fitting on relative distributions

For the purpose of calibrating the β_{mu}'s we will be interested in approaching the observed relative frequency trip length distributions. The reason for this is that the total number of trips of the observed and modelled trip length distributions are otherwise not necessarily consistent. As during the calibration the proportions of internal, in- and outgoing and external trips are dependent on the current choice of the calibration parameters and these proportions impact the total numbers of trips in the modelled trip length distributions. This normalization step makes it so we compare the shapes of the observed and modelled trip length distributions.

Feedback loop

It is perhaps interesting to mention now that usually there is an iterative feedback loop between the trip distribution model and the traffic assignment phase in the four-step modelling process. We can imagine costs c_{ijm} and trip lengths s_{ijm} to be arising from the traffic assignment phase as congested traffic routes between zone-pairs would increase the travelling time and possibly eliminate routes. By iterating between the trip distribution and traffic assignment steps the impacts of congestion can be taken into account. This feedback mechanism is however not considered in the calibration as this would be too complicated and too computationally expensive so the calibration is done under static costs and trip lengths.

3.5 Original calibration approach

We now describe the approach that has been used in practice until now at Goudappel Coffeng to automatically calibrate both the α's and β's parameters of the lognormal distribution function (12). Fransen[7] devised this approach illustrated in figure 11. It starts with initial parameters α^0_{mu}'s and β^0_{mu}. The initial α^0_{mu}'s are first adjusted in an initial *change α's step*. Then the algorithm iterates with each iteration consisting of two steps: the *change β's step* and the *change α's step*, respectively. The algorithm stops iterating if either a preset maximum number of iterations is reached or the objective function is smaller than a predefined goal value. Next we describe these two steps.

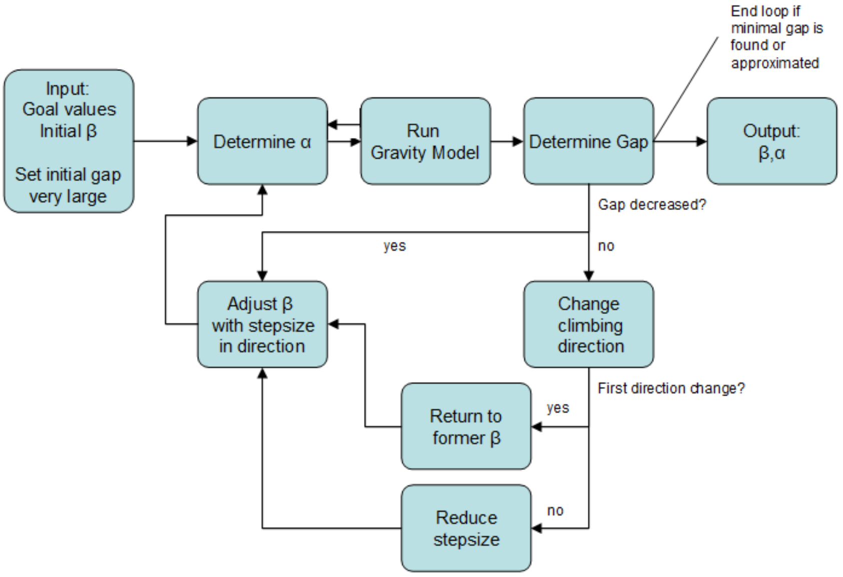

End loop if minimal gap is found or approximated
Input: Goal values Initial β
Set initial gap very large
Determine α
Run Gravity Model
Determine Gap
Output: β,α
Gap decreased?
yes
no
Adjust β with stepsize in direction
Change climbing direction
First direction change?
yes
Return to former β
no
Reduce stepsize

Figure 11: Flowchart of the original automatic calibration approach, taken from Fransen[7]

3.5.1 Change α's step

The goal of this step is to update the α's such that the observed modal split is approximately realized. From the gravity equations in (10) and (11) and the lognormal cost function (12) we see that the α's have a linear influence on the number of trips for each mode. Therefore for each mode-user class pair (m, u) the current α_{mu} is updated to the new α^*_{mu} by multiplying by a scale factor:

$$\alpha^*_{mu} = \frac{\widehat{MS}_{mu}}{MS_{mu}} \alpha_{mu}$$

Where MS_{mu} and $\widehat{MS}_{mu}$ are again the modelled and observed modal split for mode-user class pair (m, u), respectively.

Then a new modal split MS^*_{mu} is computed after running the gravity model with the new α^*'s. This process is iterated so that new α^{**}'s are determined from the older α^*'s and factors $\frac{\widehat{MS}_{mu}}{MS^*_{mu}}$ until over all the modes the average relative error between the observed and modelled modal split is smaller than a certain preset percentage.

3.5.2 Change β's step

A hillclimbing algorithm is used in the old approach to update the β's. This original hillclimbing algorithm is described in Fransen[7] and also in section 5.1. Each time a new set of β's are determined the determination of the α's (or *change α's step*) that follows has to be done multiple times until the resulting modal split is sufficiently close to the observed modal split. This means that for a single *Change β's step* multiple gravity model runs need to be done (hence the double arrow between the 'Determine α' and 'Run Gravity Model' blocks in figure 11).

3.6 New calibration approach: modal splits as gravity model constraints

3.6.1 Triproportional fitting procedure

For the following approach we discuss credit is due to Brethouwer[3] who suggested the idea to incorporate the modal split constraints directly into the gravity model when calibrating a lognormal deterrence function. As an example the gravity model for the 'Home → Work' direction in this new approach is of the form:

$$t_{ijmu} = O_{iu} D_j F^{mu}(c_{ijm}), \quad \forall i, j, m, u$$

s.t.

$$\sum_{j,m} t_{ijmu} = P_{iu}, \quad \forall i, u$$

$$\sum_{i,m,u} t_{ijmu} = A_j, \quad \forall j$$

$$MS_{mu} = \widehat{MS}_{mu}, \quad \forall m, u$$

The solution procedure to this gravity model is then to apply a triproportional fitting procedure which includes an extra balancing step for achieving the modal split constraints. This is described in algorithm 3.

Algorithm 3 Triproportional fitting procedure: balancing respectively towards target attraction, production (user class specific, for two user classes $u \in \{co, nco\}$) and modal split values

.

```
for each zone $i$ do
    $O_{iu} \leftarrow P_{iu}$
    $D_i \leftarrow A_i$
end for
for each mode-user class pair $(m, u)$ do
    $\alpha_{mu} \leftarrow \widehat{MS}_{mu}$
end for
for each mode-user class pair $(m, u)$ do
    for each od pair $(i, j)$ do
        $t_{ijmu} \leftarrow \alpha_{mu} \cdot O_{iu} \cdot D_j \cdot F^{mu}(c_{ijm})$
    end for
end for
while not converged do
    for each zone $j$ do
        if $\sum_{i,m,u} t_{ijmu} > 0$ then f $\leftarrow \left( \frac{A_j}{\sum_{i,m,u} t_{ijm}} \right)$ else $f \leftarrow 0$
        $D_j \leftarrow f \cdot D_j$
        for each zone $i$ do
            for each mode-user class pair $(m, u)$ do
                $t_{ijmu} \leftarrow f \cdot t_{ijmu}$
            end for
        end for
    end for
    for each user class $u \in \{co, nco\}$ do
        for each zone $i$ do
            if $\sum_{j,m} t_{ijmu} > 0$ then f $\leftarrow \left( \frac{P_{iu}}{\sum_{j,m} t_{ijmu}} \right)$ else $f \leftarrow 0$
            $O_{iu} \leftarrow f \cdot O_{iu}$
            for each zone $j$ do
                for each mode $m$ do
                    $t_{ijmu} \leftarrow f \cdot t_{ijmu}$
                end for
            end for
        end for
    end for
    for each mode-user class pair $(m, u)$ do
        Calculate $MS_{mu}$ using equation (14).
        if $MS_{mu} > 0$ then $f \leftarrow \left( \frac{\widehat{MS}_{mu}}{MS_{mu}} \right)$ else $f \leftarrow 0$
        $\alpha_{mu} \leftarrow f \cdot \alpha_{mu}$
        for each od pair $(i, j)$ do
            $t_{ijmu} = f \cdot t_{ijmu}$
        end for
    end for
end while
```

The advantage of this new approach is that it requires only a method to calibrate the β_{mu}'s parameter set, as the α_{mu}'s are now a function of the β_{mu}'s. In the original calibration approach α_{mu}'s are arrived at only after multiple *change α's* steps in between the *change β's* steps. The new calibration approach only requires about 20% of the running time of the old calibration approach and achieves the target modal splits exactly.

3.6.2 Discussion: Comparison with old approach

Convergence of the triproportional fitting procedure is like the biproportional fitting procedure not guaranteed but in this research we observed convergence for each of the encountered problem instances. It was also observed that the triproportional fitting procedure converges to the same trip distribution as the biproportional fitting procedure i.e. algorithm 2 for which the lognormal parameters are initially set to the modal split balancing factors obtained in the triproportional fitting procedure. This is also predicted by theorem 2.1. For example, in case of user specific production values, set $\boldsymbol{M}$ in the theorem according to:

$$\boldsymbol{M} = \sum_{m} \left[\begin{array}{c} F^{m,co}(\boldsymbol{C}^{m,co}, \alpha_{m,co}) \\ \hdashline F^{m,nco}(\boldsymbol{C}^{m,nco}, \alpha_{m,nco}) \end{array}\right]$$

Assuming both procedures converged i.e. balancing factors $\boldsymbol{O}_{bi}, \boldsymbol{D}_{bi}$ and $\boldsymbol{O}_{tri}, \boldsymbol{D}_{tri}$ were found such that the converged aggregated OD matrices $\boldsymbol{T}_{bi} = \boldsymbol{O}_{bi}\boldsymbol{M}\boldsymbol{D}_{bi}$ and $\boldsymbol{T}_{tri} = \boldsymbol{O}_{tri}\boldsymbol{M}\boldsymbol{D}_{tri}$ satisfy the productions and attractions. Then by theorem 2.1 these aggregated OD matrices are equal i.e. $\boldsymbol{T}_{bi} = \boldsymbol{T}_{tri}$ and the balancing factors $\boldsymbol{O}_{bi}, \boldsymbol{D}_{bi}$ and $\boldsymbol{O}_{tri}, \boldsymbol{D}_{tri}$ are equal up to constant scaling. Then clearly the disaggregate OD matrices per mode and user class must be the same too.

The importance of these distributions being equal is that the nature of the biproportional gravity model is still intact with the extra balancing step. The modal split balancing step gives a trip distribution satisfying modal split while still maximizing entropy subject to the trip end constraints for the chosen β_{mu} and (resulting) α_{mu}. The α_{mu}'s resulting from the triproportional procedure can therefore still be interpreted as behavioral parameters of the lognormal deterrence function and are still transferable between different models.

3.6.3 Modal split aggregateness

The new calibration approach introduces the modal split constraints directly into the gravity model of a sub purpose. However, in our trip distribution framework the modal split is looked at aggregated over all sub purposes within a purpose. This is why in the old calibration approach the α_{mu}'s for the different sub purposes within a purpose are the same. Fortunately for the models we considered, the disaggregated observed modal splits per sub purpose were close to the observed modal splits of the overarching purpose. Therefore we kept to the approach as described. But we believe that the triproportional approach could be adjusted to work for a single aggregate modal split constraint per purpose by iterating between the following two steps: 1): Balancing the OD matrices on trip end constraints (productions and attractions) for each sub purpose seperately and 2): Jointly balancing (aggregated over all sub purposes) the OD matrices on the modal split constraint. One drawback to this is the larger memory requirement as the OD matrices of all sub purposes would need to be in memory at the same time.

4 Mathematical problem formulation

In this section we give the mathematical problem formulation that we will focus on solving. Specifically we formulate the new triproportional calibration approach discussed in section 3.6 as a bilevel optimization problem. Section 4.1 discusses some potential choices for the calibration objective function after which in section 4.2 the bilevel problem is formulated. Section 4.3 gives the convergence criterion used for the gravity models and explains its importance. Finally section 4.4 gives a list of performance criteria to be taken in mind in the search for potential solution methods.

Some notation:

We first introduce some vector notation for some of the already encountered variables of the trip distribution model. Denote by $\boldsymbol{\beta}$ the full set of parameters β_{mu} of the trip purpose being calibrated at hand. Remember that within a purpose the parameter set of each sub purpose is this set (or a proper subset of this set as logically a user class specific sub purpose inherits only the parameters of its user class). Denote by $\boldsymbol{O}$, $\boldsymbol{D}$, $\boldsymbol{\alpha}$ the full set of production, attraction and modal split balancing factors that relate to the purpose. So each of these sets are the union of the subsets of balancing factors resulting from the IPF procedures for the different sub purpose gravity models. Similarly here by $\boldsymbol{T}$ we denote a full purpose trip distribution i.e. the set of all (disaggregated) trip numbers t_{ijmu} related to the sub purposes of a specific purpose.

4.1 Calibration objective function

In section 3.4 we showed how to derive the modelled trip length distribution from a trip distribution. In the calibration one wants to choose parameters β_{mu} such that the modelled trip length distributions in some sense approach the observed trip length distributions. An objective function that is useful to measure the similarity between two distributions is to take the sum of squared differences:

$$F(\boldsymbol{\beta}) = \sum_{m,u,k} (d^{rel}_{muk} - \widehat{d}^{rel}_{muk})^2 \tag{15}$$

Here d^{rel}_{muk} and $\widehat{d}^{rel}_{muk}$ represent respectively modelled and observed relative trip length distributions both expressed in percentages. They can be calculated by:

$$d^{rel}_{muk} = 100 \cdot \left(\frac{d_{muk}}{\sum_{k'} d_{muk'}} \right) (\%)$$

$$\widehat{d}^{rel}_{muk} = 100 \cdot \left(\frac{\widehat{d}_{muk}}{\sum_{k'} \widehat{d}_{muk'}} \right) (\%)$$

The relative numbers are preferred in (15) since the total number of modelled trips can vary during the calibration as reasoned in section 3.4. The trip length distribution numbers d_{muk} and $\widehat{d}_{muk}$ can either denote absolute trip numbers or these normalized by distance bin width as described in section 3.3.3. Note that the objective function in (15) is written as a function of the parameter set $\boldsymbol{\beta}$ because the modelled trip length distributions are a function of this parameter set. This fact will be made more clear in the next section.

If one is more interested in mileage than trip numbers one can apply a weighting resulting in the following objective function:

$$F(\boldsymbol{\beta}) = \sum_{m,u,k} (m_k(d^{rel}_{muk} - \widehat{d}^{rel}_{muk}))^2$$

Here the bin midpoints m_k serve as weights with the goal of better fitting on distance bins of higher length. A reason it can be desirable to better fit in this way on the total amount of mobility measured in mileage is due to the trip assignment phase of the four step model. OD pairs further apart have longer routes in the road network which naturally then consist of more links compared to routes of shorter distanced OD pairs. The resulting link flows of the trip assignment phase is compared based on a subset of links that have observed traffic counts attached to them. Since the trip numbers of the higher distance bins impact more links, and so likely more links with counts attached, a better fit in the trip distribution phase likely translates to a better general fit in the trip assignment phase.

Another alternative objective function is:

$$F(\boldsymbol{\beta}) = \sum_{m,u} (1 - r_{mu}^2)$$

with r_{mu} being the Pearson correlation coefficient between the (absolute) observed and modelled trip length distributions. This objective function was used for the original calibration approach before this research. The correlation coefficient is invariant to scaling of the series and therefore automatically compares relative distributions. However we have chosen to use the objective function of equation (15) as it is easier to find the derivative of it as well as the measure itself is easier to explain.

4.2 Bilevel optimization problem

This subsection gives an mathematical optimization problem formulation for the problem of calibrating a trip distribution model of a trip purpose. The optimization problem is of a special kind. Namely it is a bilevel optimization problem where the problem of maximizing entropy or solving the gravity model is embedded within the calibration of the gravity model parameters $\boldsymbol{\beta}$. We refer to solving the gravity model as the inner optimization problem and the problem of selecting the optimal $\boldsymbol{\beta}$ during the calibration as the outer optimization problem.
The bilevel optimization problem is then formulated as:

$$\min_{\boldsymbol{\beta}<\mathbf{0},(\boldsymbol{O},\boldsymbol{D},\boldsymbol{\alpha})\geq\mathbf{0}} F(\boldsymbol{\beta},\boldsymbol{O},\boldsymbol{D},\boldsymbol{\alpha}) \quad \text{s.t. :} \tag{16}$$

$$(\boldsymbol{O},\boldsymbol{D},\boldsymbol{\alpha}) \in \operatorname*{arg\,max}_{(\tilde{\boldsymbol{O}},\tilde{\boldsymbol{D}},\tilde{\boldsymbol{\alpha}})\geq\mathbf{0}} \{entropy(\boldsymbol{T}) \mid \boldsymbol{T} = \boldsymbol{T}(\boldsymbol{\beta},\tilde{\boldsymbol{O}},\tilde{\boldsymbol{D}},\tilde{\boldsymbol{\alpha}}) \text{ satisfies the relevant trip end and modal split constraints}\}$$

The solution method for the inner optimization problem is the triproportional fitting procedure of algorithm 3. Note that the inner optimization problem actually consists of solving multiple gravity models for each sub purpose in series. Assuming convergence of the IPF procedure(s) we can regard the set of balancing factors as a function of the choice of the $\boldsymbol{\beta}$ parameter set. To emphasize this we write $\boldsymbol{O} = \boldsymbol{O(\beta)}$, $\boldsymbol{D} = \boldsymbol{D(\beta)}$ and $\boldsymbol{\alpha} = \boldsymbol{\alpha(\beta)}$. The objective value is calculated from the trip distribution which is a function of the balancing factors and the parameter set. Therefore we can write $F(\boldsymbol{\beta}, \boldsymbol{O(\beta)}, \boldsymbol{D(\beta)}, \boldsymbol{\alpha(\beta)}) = F(\boldsymbol{\beta})$ i.e. the objective function can essentially be regarded as a function of the parameter set $\boldsymbol{\beta}$ only. The flowchart in figure 12 helps to illustrate this. However, due to the iterative nature of the fitting procedure it is especially difficult to give some sort of closed-form expressions for $F(\boldsymbol{\beta})$, $\boldsymbol{O(\beta)}$, $\boldsymbol{D(\beta)}$ or $\boldsymbol{\alpha(\beta)}$. Note that the outer optimization problem of calibration is constrained as $\beta_{mu} < 0$ is required.

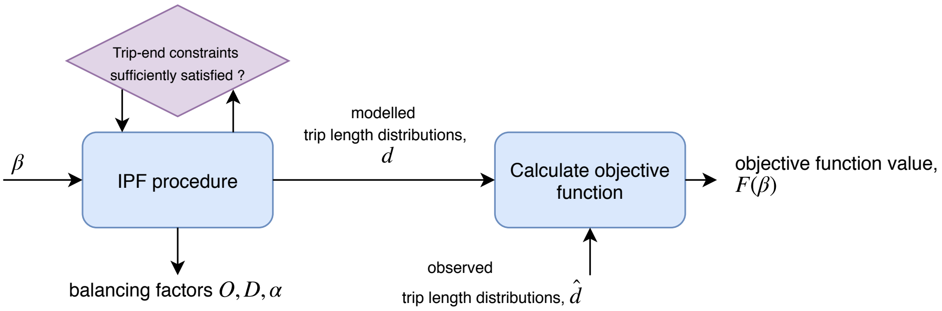

Trip-end constraints sufficiently satisfied ?
β
IPF procedure
modelled trip length distributions, d
Calculate objective function
objective function value, $F(\beta)$
balancing factors O, D, α
observed trip length distributions, $\hat{d}$

Figure 12: Flowchart of the bilevel optimization problem

4.3 Gravity model convergence criterion

Assuming the IPF-procedures converges, its solution is still inexact, meaning generally there will be some residual in the trip end constraints after any finite number of IPF iterations. Therefore a gravity model convergence criterion (the diamond shaped box in figure 12) needs to be established. The choice of a threshold used in the convergence criterion will have an impact on the bilevel optimization problem as these determine the number of IPF iterations required which naturally impacts the balancing factors and objective function. It does not make much sense to base the convergence criterion on absolute residual errors as different strategic traffic models give rise to different orders of absolute trip numbers. Therefore the convergence criterion we will be using is based on relative residuals. For example in the case of the gravity model for the 'Home → Work' sub purpose the maximum relative residuals of production and attraction constraints respectively R^{rel}_{prod} and R^{rel}_{attr} as defined by:

$$R^{rel}_{prod} = \max_{i,u} \frac{(\sum_{j,m} t_{ijmu} - P_{iu})}{P_{iu}} \tag{17}$$

$$R^{rel}_{attr} = \max_{j} \frac{(\sum_{i,m,u} t_{ijmu} - A_j)}{A_j}$$

The convergence criterion is then to enforce that both of these maximum relative residuals should be smaller than a preset percentage. Note that the modal split constraints do not need to be included in the convergence criterion as they are satisfied exactly as the IPF procedure terminates on a modal split balancing step, see algorithm 3. Indeed a different order could be chosen putting e.g. the production and attraction balancing steps last which would imply the production and attraction constraints are deemed more important than the modal split constraints. However it was observed in this research that the order favoring modal split constraints was only behind at most 1-2 iterations in terms of convergence in R^{rel}_{prod} and R^{rel}_{attr}, suggesting the order we choose only matters slightly.

4.4 Solution method performance criteria

For a given optimization problem it is often the case that there are numerous potential approaches and methods towards solving it. Therefore it is useful to identify and formulate performance criteria by which the larger set of options available can be narrowed down and the selected few methods can be tested on. Below we explain the five performance criteria that are deemed most important.

i): Reliability in finding a quality solution:

By a quality solution we mean first and foremost its objective function value. Ideally the method should converge to a global minimum. However the task of reliably finding a global minimum to an optimization problem can prove very difficult. The optimization problem formulated in this chapter is generally not convex. Therefore it can be the case that the solution space has multiple local minima of which some of them can be suboptimal. So given that a method finds a local minimum it is not necessarily a globally optimal solution. Ensuring the solution found cannot be further improved at least locally i.e. finding a local minimum seems like a reasonable aim given the difficulty of global optimization. Despite this difficulty we will at least pay some attention to exploring global optimization techniques in the next chapter. It can also be meaningful to judge the quality of a solution in terms other than its objective function value. Two distinct methods could find solutions with a seemingly significant difference in objective values. Plots of the solutions modelled trip length distribution versus the observed trip length distributions with confidence bounds as defined in section 3.3.2 could reveal the visual difference is actually meaningless to an end user (consultant) at Goudappel Coffeng.

ii): Number of solution method parameters:

The number of solution method parameters should be minimal. A minimal number of solution method parameters makes use of the method by the end user easier as it requires less knowledge on how to pick the right values. Here by solution method parameters we do not mean empirical parameters that could perhaps still be estimated from data, but heuristic parameters. The act of choosing the right values for these heuristic parameters otherwise can itself become a tedious and demanding calibration process. This calibration of solution parameters may be required as solution method parameters might not be transferable between different projects. Since a right choice of parameters might for example depend on the strategic traffic model or the objective function used.

iii): Robustness under initial solution guess:

One solution method parameter that most solution methods for optimization problems have in common is that of an initial solution guess. It is desirable to have a solution method that finds a quality solution (in terms of (i)) independent of the initial solution guess.

iv): Running time:

The time it takes for the method to converge to its final solution. With the hillclimbing method of the old calibration approach the largest scale strategic traffic model calibrated within Goudappel Coffeng takes around three days to calibrate all purposes. As the new calibration that we will be using significantly reduces the required number of IPF iterations it is reasonable to require the new solution method to stay within this time window for this reference model. The running time is, except from staying within this reasonable time limit, deemed less important than criteria (i) to (iii).

v): Difficulty of implementation:

Here important aspects are the work it takes to implement the method in general but also specifically within Omnitrans, as well as whether an implementation of the method would be flexible to changes in the calibration environment such as changes in the trip distribution model (e.g. additional balancing steps) or the objective function.

5 Potential solution methods

This chapter describes potential candidate methods for solving the optimization problem of (16). First in section 5.1 the original hillclimbing algorithm is described which was used in the old calibration approach. This is seen as the reference algorithm to which selected potential solution methods will be compared and potentially tested against on the performance criteria of section 4.4. Section 5.2 describes the general gradient based update rule and two methods to approximate the gradient used in this rule. Section 5.3 describes an alternative gradient based method, the so called SPSA method, which approximates the gradient stochastically. Section 5.4 discusses two potential techniques for global optimization. Finally section 5.5 compares the discussed potential solution methods after which an alternative candidate method to hillclimbing is selected to be further explored and tested in the next chapters.

5.1 Hillclimbing

The hillclimbing algorithm we now describe is largely the same as the hillclimbing algorithm used for the solution method of the original calibration approach by Fransen[7]. The variant we describe here is adjusted for the new calibration approach in which the $\boldsymbol{\alpha}$ set are balancing factors. It is a trimmed down version in which only the $\boldsymbol{\beta}$ parameter set is updated. The rule used has not changed but we describe it shortly for the sake of completion and since it is one of the solution methods that was implemented and tested.

The hillclimbing algorithm iteratively updates the $\boldsymbol{\beta}$ parameter set starting from an initial set $\boldsymbol{\beta^0}$. In what follows n denotes the current iteration number. The algorithm stops doing iterations when either a predefined maximum number of iterations is reached or if either the current objective value $F(\boldsymbol{\beta^n})$ or change in objective value $F(\boldsymbol{\beta^n}) - F(\boldsymbol{\beta^{n-1}})$ is smaller than a preset threshold value. For each iteration n the algorithm keeps track of the individual objective values $f^{mu}(\boldsymbol{\beta^n})$ per mode-user class pair (m, u).

Updating the β set

In addition to tracking individual objective values, each mode-user class pair (m, u) has a variable $sign_{mu}$ which always equal either 1 or -1, each of which are initially set at either of these values. The $sign_{mu}$ variable determines whether the next update increases or decreases β_{mu}. The algorithm also keeps track of a step size variables $step_{mu}$ which then determine the absolute value by which β_{mu} is changed in the next update, initially preset to some maximum step size. The final parameter that requires a value is the step size reduction factor $\gamma > 1$. Now the hillclimbing algorithm we used can be described by algorithm 4.

Algorithm 4 Adjusted hillclimbing algorithm

```
for each mode-user class pair $(m, u)$ do
    $sign_{mu} \leftarrow initialsign_{mu}$
    $step_{mu} \leftarrow maxstep$
end for
Run gravity model and calculate individual objective values $f^{mu}(\boldsymbol{\beta^0})$
for each mode-user class pair $(m, u)$ do
    $\beta^1_{mu} \leftarrow P_{<0}(\beta^0_{mu} + sign_{mu} \cdot step_{mu})$
end for
Run gravity model and calculate individual objective values $f^{mu}(\boldsymbol{\beta^1})$
$n \leftarrow 1$
while $n \leq maxiterations$ & $\mathrm{F}(\boldsymbol{\beta^n}) \geq threshold_{obj}$ & $\mathrm{F}(\boldsymbol{\beta^n}) - F(\boldsymbol{\beta^{n-1}}) \geq threshold_{change}$ do
    for each mode-user class pair $(m, u)$ do
        if $f^{mu}(\boldsymbol{\beta^n}) \geq f^{mu}(\boldsymbol{\beta^{n-1}})$ then
            $sign_{mu} \leftarrow -sign_{mu}$
            $step_{mu} \leftarrow \frac{1}{\gamma} \cdot step_{mu}$
        end if
        $\beta^{n+1}_{mu} \leftarrow P_{<0}(\beta^n_{mu} + sign_{mu} \cdot step_{mu})$
    end for
    Run gravity model and calculate individual objective values $f^{mu}(\boldsymbol{\beta^{n+1}})$
    $n \leftarrow n + 1$
end while
```

In algorithm 4 the function $P_{<0}$ projects for a solution $\boldsymbol{\beta}^k$ each nonnegative β^k_m into a feasible negative coordinate by:

$$(P_{<0}(\boldsymbol{\beta}))_i = \begin{cases} \beta_i, & \beta_i \leq \beta_{max} \\ \beta_{max}, & \beta_i > \beta_{max} \end{cases}$$

Here β_{max} should be chosen as a negative number close but not equal to zero, as zero itself is not allowed.

The intuition behind hillclimbing is that it checks each iteration whether the last movement of β_{mu} (increasing or decreasing) has been profitable in terms of its individual objective value $f^{mu}(\boldsymbol{\beta^n})$. Each time the last movement of a parameter was not profitable it reverses its direction and it reduces its step size geometrically by applying $\frac{1}{\gamma}$ as a reduction factor.

One special case in the hillclimbing algorithm which algorithm 4 does not include, is the following: If the initial movement for a parameter β_{mu} was not profitable it is then set to the value it would be at now given the initial sign was reversed. In addition its sign is negated while the step size is not changed from the maximum initial step size. The reason for this step is simply to quickly correct a perhaps badly chosen initial sign value.

Before this research a heuristic rule of setting the initial $sign_{mu}$ values was used by basing them on the modelled trip length distributions of the initial $\boldsymbol{\beta}^0$ guess. The observed and modelled proportion of short trips were compared. Here short bins are defined as the trips of the first n_{short} bins. This rule was omitted in this research as it did not seem to make a difference in earlier application and it requires a choice for n_{short}, another method parameter, which is not entirely obvious. Also it seems like the special case rule mentioned above should correct already for a wrong starting course. This insensitivity to the initial sign values was also observed in earlier application. The initial step size *maxstep* was observed to be a more sensitive parameter. The step size reduction factor γ was set to the golden ratio $\frac{1+\sqrt{5}}{2}$ in this thesis, as was the case before this research. This seems like a reasonable choice, as it is often found in optimization such as in golden-section search.

An advantage of the hillclimbing method is that the computational time it takes to do one hillclimbing iteration is practically speaking equal to the time it takes to evaluate one objective function. So in a sense the time hillclimbing spends outside of exploring the solution space is negligible (even for a model of small size the time spent on running the IPF procedures is much larger than the time spent on the remaining lines in algorithm 4).

A disadvantage of the method is that as there is no proof of convergence of the iterates to a local minimum. In fact in practice it was frequently observed for the old calibration approach that the objective value increases sometimes multiple iterations before improving. The method seems heuristic in nature. It changes the $\boldsymbol{\beta_{mu}}$ by looking at changes in individual objective values $f^{mu}(\boldsymbol{\beta})$ as if only the change in an individual parameter β_{mu} effects its individual objective value. So hillclimbing does not take into account the cross correlation that might exist between a change in parameters and the resulting change in total objective value.

5.2 Gradient based methods

The steepest descent method aims to find a local minimum by updating the parameters in the direction of steepest descent, which is the negative of the gradient of the objective function $F(\boldsymbol{\beta}) : \mathbb{R}^n \to \mathbb{R}$. The gradient of F with respect to $\boldsymbol{\beta}$ is a vector containing the partial derivatives of F with respect to the β_{mu}'s:

$$\nabla F(\boldsymbol{\beta}) = \left(\frac{\partial F(\boldsymbol{\beta})}{\partial \beta_{mu}}, \ \dots \ , \quad \forall m, u \right)^T$$

The gradient descent method then is characterized by the following update rule:

$$\boldsymbol{\beta_{k+1}} = \boldsymbol{\beta_k} - a_k \nabla F(\boldsymbol{\beta_k})$$

Where $a_k > 0$ is the step size taken in negative gradient direction for which a suitable value has to be chosen. A step size that is too small would mean a slow convergence towards the local minimum while a constant step size that is too large can potentially lead to constantly overshooting the local optimum. The a_k does not have to constant but could for example become smaller over the iterations e.g.:

$$a = a_k = \frac{a_0}{k}$$

Another possibility to determine the step size is to use a line search algorithm. A line search algorithm tries for a calculated gradient different solutions by taking different step sizes along the negative gradient search direction until sufficient improvement is found. However depending on the type of line search this can make a descent iteration a lot more costly in terms of computational time. Next we describe two approaches to calculating the gradient $\nabla F(\boldsymbol{\beta})$. We now discuss two seemingly suitable methods to calculate an approximation to the gradient. We refer to Martins[10] for a more through discussion on these and other methods to calculate gradients.

5.2.1 Approximating the gradient: finite differences

Here for a more convenient notation we represent the parameter set now as a vector $\boldsymbol{\beta} \in \mathbb{R}^n_{<0}$ with its components assumed to be n in number individual parameters denoting $\beta_i, i = 1, \dots, n$. The method of finite differences is a popular and easily implemented method for approximating the gradient of a function. One option is to use the so called central difference approximation:

$$\frac{dF(\boldsymbol{\beta})}{d\beta_i} \approx \frac{F(\boldsymbol{\beta} + \Delta \boldsymbol{e_i}) - F(\boldsymbol{\beta} - \Delta \boldsymbol{e_i})}{2\Delta}, \quad \forall i = 1, \ \dots \ , n \tag{18}$$

For $\Delta > 0$ sufficiently small. Here $\boldsymbol{e_i}$ denotes the i_{th} unit vector in $\mathbb{R}^n$. Using forward differences gives a cheaper ($n+1$ objective function evaluations compared to $2n$) but less precise method to approximate the gradient:

$$\frac{dF(\boldsymbol{\beta})}{d\beta_i} \approx \frac{F(\boldsymbol{\beta} + \Delta \boldsymbol{e_i}) - F(\boldsymbol{\beta})}{\Delta}, \quad \forall i = 1, \; \dots \; , n \tag{19}$$

For a sufficiently smooth function the forward difference approximation error is $O(\Delta)$. The central difference is more precise with an approximation error of $O(\Delta^2)$. Both central and forward methods require a suitable finite difference value Δ to be chosen. The difference interval must be chosen small enough to give a good approximation to the derivative, however on the other hand a value that is too small would give a bad approximation due to rounding errors (as a computer has finite numerical precision).

5.2.2 Calculating the gradient analytically

The following technique of calculating an analytical gradient of the objective function was introduced to the author by his supervisor Dickinson[6]. For the sake of clarity we describe the technique here for a trip distribution model consisting of a single gravity model (sub purpose) with no distinction in user classes.
As was already shown in section 4.2, we can regard the balancing factors as a function of $\boldsymbol{\beta}$ i.e. we can write $\boldsymbol{O}(\boldsymbol{\beta})$, $\boldsymbol{D}(\boldsymbol{\beta})$ and $\boldsymbol{\alpha}(\boldsymbol{\beta})$. We too saw that the objective function can be regarded as a function of $\boldsymbol{\beta}$ and the balancing factors:

$$F(\boldsymbol{\beta}, \boldsymbol{O}(\boldsymbol{\beta}), \boldsymbol{D}(\boldsymbol{\beta}), \boldsymbol{\alpha}(\boldsymbol{\beta}))$$

By applying the multivariable chain rule, the total derivative of the objective function F with respect to β_m can be expressed as:

$$\frac{dF}{d\beta_m} = \frac{\partial F}{\partial \beta_m} + \sum_i \frac{\partial F}{\partial O_i}\frac{dO_i}{d\beta_m} + \sum_j \frac{\partial F}{\partial D_j}\frac{dD_j}{d\beta_m} + \sum_{\tilde{m}} \frac{\partial F}{\partial \alpha_{\tilde{m}}}\frac{d\alpha_{\tilde{m}}}{d\beta_m} \tag{20}$$

Here and in what follows the partial derivatives should be interpreted as being evaluated in the balancing factors and parameter set corresponding to the current iteration. In (20) the partial derivatives of the objective function F with respect to β_m i.e. $\frac{\partial F}{\partial \beta_m}$ as well as the partial derivatives of the objective function F with respect to the balancing factors $\frac{\partial F}{\partial O_i}$, $\frac{\partial F}{\partial D_j}$ and $\frac{\partial F}{\partial \alpha_{\tilde{m}}}$ can be computed directly. In what follows we show how the remaining factors in (20) which are the derivatives of the balancing factors with respect to β_{mu} can be found by solving a linear system.

First we define residual functions for each of the trip-end constraints g_i^{prod}, g_j^{attr} and $g_{\tilde{m}}^{MS}$ for respectively the production, attraction and modal split constraints by:

$$g_i^{prod} = \sum_{j,m} t_{ijm} - P_i \quad \forall i$$

$$g_j^{attr} = \sum_{i,m} t_{ijm} - A_j \quad \forall j$$

$$g_{\tilde{m}}^{MS} = MS_{\tilde{m}} - \widehat{MS}_{\tilde{m}} \quad \forall \tilde{m}$$

Note that these residual functions are similar to the objective function F functions of the calibration parameters as well as the balancing factors. Given the IPF procedure has converged sufficiently the residuals are approximately zero, i.e. the following system of equations holds approximately:

$$g_i^{prod}(\boldsymbol{\beta}, \boldsymbol{O}(\boldsymbol{\beta}), \boldsymbol{D}(\boldsymbol{\beta}), \boldsymbol{\alpha}(\boldsymbol{\beta})) = 0, \quad \forall i$$

$$g_j^{attr}(\boldsymbol{\beta}, \boldsymbol{O}(\boldsymbol{\beta}), \boldsymbol{D}(\boldsymbol{\beta}), \boldsymbol{\alpha}(\boldsymbol{\beta})) = 0, \quad \forall j$$

$$g_{\tilde{m}}^{MS}(\boldsymbol{\beta}, \boldsymbol{O}(\boldsymbol{\beta}), \boldsymbol{D}(\boldsymbol{\beta}), \boldsymbol{\alpha}(\boldsymbol{\beta})) = 0, \quad \forall \tilde{m}$$

Differentiating each of these equations with respect to β_m by again using the multivariable chain rule we obtain:

$$\frac{dg_i^{prod}}{d\beta_m} = \frac{\partial g_i^{prod}}{\partial \beta_m} + \sum_i \frac{\partial g_i^{prod}}{\partial O_i}\frac{dO_i}{d\beta_m} + \sum_j \frac{\partial g_i^{prod}}{\partial D_j}\frac{dD_j}{d\beta_m} + \sum_{\tilde{m}} \frac{\partial g_i^{prod}}{\partial \alpha_{\tilde{m}}}\frac{d\alpha_{\tilde{m}}}{d\beta_m} = 0, \quad \forall i \tag{21}$$

$$\frac{dg_j^{attr}}{d\beta_m} = \frac{\partial g_j^{attr}}{\partial \beta_m} + \sum_i \frac{\partial g_j^{attr}}{\partial O_i}\frac{dO_i}{d\beta_m} + \sum_j \frac{\partial g_j^{attr}}{\partial D_j}\frac{dD_j}{d\beta_m} + \sum_{\tilde{m}} \frac{\partial g_j^{attr}}{\partial \alpha_{\tilde{m}}}\frac{d\alpha_{\tilde{m}}}{d\beta_m} = 0, \quad \forall j$$

$$\frac{dg_{\tilde{m}}^{MS}}{d\beta_m} = \frac{\partial g_{\tilde{m}}^{MS}}{\partial \beta_m} + \sum_i \frac{\partial g_{\tilde{m}}^{MS}}{\partial O_i}\frac{dO_i}{d\beta_m} + \sum_j \frac{\partial g_{\tilde{m}}^{MS}}{\partial D_j}\frac{dD_j}{d\beta_m} + \sum_{\tilde{m}} \frac{\partial g_{\tilde{m}}^{MS}}{\partial \alpha_{\tilde{m}}}\frac{d\alpha_{\tilde{m}}}{d\beta_m} = 0, \quad \forall \tilde{m}$$

Defining vectors $\boldsymbol{x}$ and $\boldsymbol{b}$ by:

$$\boldsymbol{x} = \left(\frac{dO_i}{d\beta_m}, \ \dots, \frac{dD_j}{d\beta_m}, \ \dots, \frac{d\alpha_{\tilde{m}}}{d\beta_m}, \ \dots \right)^T$$

$$\boldsymbol{b} = -\left(\frac{\partial g_i^{prod}}{\partial \beta_m}, \ \dots, \frac{\partial g_j^{attr}}{\partial \beta_m}, \ \dots, \frac{\partial g_{\tilde{m}}^{MS}}{\partial \beta_m}, \ \dots \right)^T$$

Then the equations in (21) are equivalent to the linear system $\mathbf{A}\boldsymbol{x} = \boldsymbol{b}$. The matrix $\mathbf{A}$ then is the Jacobian of the vector valued function $\boldsymbol{g}$ with respect to the balancing factors $(\boldsymbol{O}, \boldsymbol{D}, \boldsymbol{\alpha})$. Note that since the number of trip-end constraints is equal to the number of balancing factors, $\mathbf{A}$ is a square matrix. The matrix $\mathbf{A}$ and vector $\boldsymbol{b}$ can be computed directly. The solution to the linear system $\boldsymbol{x}$ gives us the desired derivatives of the balancing factors with respect to β_m which can then be plugged into (20) to compute the total derivative $\frac{dF}{d\beta_m}$.

By repeating this technique for each parameter β_m the objective function gradient $\nabla F(\boldsymbol{\beta})$ can be calculated. However like the finite difference method this analytical approach also at best approximates the actual gradient. This is due to the fact that practically speaking the IPF procedure has inexact convergence and so the residual functions only approximate zero. Therefore it seems helpful to use the gradients calculated by the finite difference methods as a reference when testing this analytical gradient method. The running time of these gradient calculation methods is also an interesting aspect to be investigated. Should both methods give similar gradients the method that calculates it in a shorter time period will be preferred. The ranking of the methods in terms of running time might vary depending on the size of the trip distribution model (in terms of the number of zones).

5.2.3 Adjoint gradient method

This section discusses the adjoint gradient method which is a more efficient variant of the analytical gradient method of section 5.2.2. Again for sake of clarity we assume the trip distribution model consists of a single sub purpose and we omit user classes. Now denote by $\boldsymbol{c}$ the vector of partials of the objective function with respect to the balancing factors i.e.:

$$\boldsymbol{c}^T = \left(\frac{\partial F}{\partial O_i}, \ \dots\ , \frac{\partial F}{\partial D_j}, \ \dots\ , \frac{\partial F}{\partial \alpha_{\tilde{m}}}, \ \dots \right)^T$$

Also let $d(m) = \frac{\partial F}{\partial \beta_m}$ and let $\boldsymbol{A}$, $\boldsymbol{b}(m)$ and $\boldsymbol{x}$ be defined again as in section 5.2.2. Then the method described there for calculating a parameter sensitivity $\frac{dF}{d\beta_m}$ can be summarized by:

$$\frac{dF}{d\beta_m} = d(m) + \boldsymbol{c}^T \boldsymbol{x}(m)$$

Where each $\boldsymbol{x}(m)$ solves the linear system:

$$\boldsymbol{A}\boldsymbol{x}(m) = \boldsymbol{b}(m)$$

Or put in one equation:

$$\frac{dF}{d\beta_m} = d(m) + \boldsymbol{c}^T \boldsymbol{A}^{-1} \boldsymbol{b}(m) \tag{22}$$

In particular calculating the full gradient by this method requires solving the system $\boldsymbol{A}\boldsymbol{x}(m) = \boldsymbol{b}(m)$ for each mode m. In our case it is more efficient to use the so called adjoint method to calculate the gradient. The adjoint gradient method is also described in Martins[10]. Instead of solving multiple linear systems it is also sufficient for calculating the gradient to solve just one linear system: letting $\boldsymbol{y} = \boldsymbol{c}^T \boldsymbol{A}^{-1}$ equation (22) is equivalent to:

$$\frac{dF}{d\beta_m} = d(m) + \boldsymbol{y}^T \boldsymbol{b}(m) \tag{23}$$

Then to compute $\boldsymbol{y}$ requires solving the linear system with adjoint matrix $\boldsymbol{A}^T$ and $\boldsymbol{c}^T$ as a right-hand side vector, so $\boldsymbol{y}$ solves the linear system:

$$\boldsymbol{A}^T \boldsymbol{y}^T = \boldsymbol{c}^T$$

5.3 Simultaneous perturbation stochastic approximation (SPSA)

Here we give a short overview of the method of Simultaneous perturbation stochastic approximation, abbreviated SPSA, in which the descent vectors stochastically approximate the gradient. For a more in depth presentation of the method we refer to Spall[13].

The update rule of SPSA is similar to the gradient descent method:

$$\boldsymbol{\beta_{k+1}} = \boldsymbol{\beta_k} - a_k \widetilde{\nabla} F(\boldsymbol{\beta_k})$$

The vector $\widetilde{\nabla} F(\boldsymbol{\beta})$ approximates the gradient and is obtained from the average of a set of m vectors $\boldsymbol{g}^l(\boldsymbol{\beta}, \boldsymbol{\Delta}^l)$:

$$\widetilde{\nabla} F(\boldsymbol{\beta}) = \frac{\sum_{l=1}^{m} \boldsymbol{g}^l(\boldsymbol{\beta}, \boldsymbol{\Delta}^l)}{m} \tag{24}$$

These vectors $\boldsymbol{g}^l(\boldsymbol{\beta}, \boldsymbol{\Delta}^l)$, $l = 1, \ \dots, m$ are calculated component wise by:

$$g_i^l(\boldsymbol{\beta}, \Delta^l) = \frac{F(\boldsymbol{\beta} + c_k \boldsymbol{\Delta}^l) - F(\boldsymbol{\beta} - c_k \boldsymbol{\Delta}^l)}{2 c_k \boldsymbol{\Delta}_i^l}, \quad i = 1, \ \dots\ , n \tag{25}$$

For each l, $\boldsymbol{\Delta}^l$ is a so called perturbation vector each consisting of components Δ_i^l that are drawn from a Bernoulli ± distribution:

$$\Delta_i^l = \begin{cases} 1, & \text{with probability } \frac{1}{2} \\ -1, & \text{with probability } \frac{1}{2} \end{cases} \quad \forall i = 1, \ \dots \ , n$$

Each $\boldsymbol{\Delta}^l$ perturbs $\boldsymbol{\beta}$ once along its direction and once along its opposite direction in (25). The magnitude of each perturbation is c_k in iteration k. The magnitudes are of the form:

$$c_k = \frac{C}{k^\gamma}$$

The step sizes a_k are of the form:

$$a_k = \frac{a}{(A+k)^\alpha}$$

Note that so far we require a specification of the all positive constants C, γ, a, A, α and m mentioned so far. The following conditions are necessary by Spall:

$$\alpha - 2\gamma > 3\gamma - \frac{\alpha}{2} \geq 0$$

The advantage that comes with the SPSA method is that it needs less function evaluations compared to the finite difference methods to get an approximation to the gradient that is at least as good. To clarify this: note that for $m = 1$ each iteration we would according to (24) need only two function evaluations compared to $2n+1$ and $n+1$ respectively for the central and forward finite difference methods equations (18) and (19). That SPSA gives at least as good of an approximation to the gradient as using central difference approximations we get by the following lemma with proof given by Spall[13]:

For F three times continuously differentiable in some neighborhood of $\boldsymbol{\beta}$, we have for the expectation of $\boldsymbol{g}^l(\boldsymbol{\beta}, \boldsymbol{\Delta}^l)$ for $c \to 0$:

$$E[\boldsymbol{g}^l(\boldsymbol{\beta}, \boldsymbol{\Delta}^l) - \nabla F(\boldsymbol{\beta})] = O(c^2)$$

5.4 Global optimization techniques

The solution methods described so far are at best local minimizers. In this subsection we shortly describe two optimization techniques that can be used to approximate the global optimum. The first one is built from a given local search method. The second technique is called simulated annealing.

5.4.1 Multi-start methods

Given an algorithm that can find a local minimum one can generate different starting parameter vectors $\boldsymbol{\beta_0^j}$, on each of which a local search is performed to obtain different local minima. If we indefinitely do local searches from new starting $\boldsymbol{\beta_0}$, we approximate the global minimum. Techniques of this kind are called multi-start methods. Palomares et. al[8] mentions multi-start methods one of which partitions the search space into boxes within each a initial guess is randomly selected and from which then local search is started.

5.4.2 Simulated annealing

Simulated annealing is a probabilistic technique for approximating the global minimum of a function. The method of simulated annealing was first introduced in combinatorial optimization but algorithms have been derived from simulated annealing for optimizing functions of continuous variables. For an example of such an algorithm and a more comprehensive discussion on simulated annealing in the context of continuous optimization we refer to Corana et al[4]. Here we briefly discuss the essential idea in simulated annealing which is that simulated annealing will accept a solution with a worse objective value with a certain positive probability allowing for a more explorative search for the global optimum in the solution space.

In simulated annealing a new trial solution $\boldsymbol{\beta}^{'}$ is constructed by perturbing the current solution $\boldsymbol{\beta}_k$ at iteration k in some random fashion. If the trial is an improvement i.e. $F(\boldsymbol{\beta}^{'}) < F(\boldsymbol{\beta}_k)$ we will immediately accept it as our new solution: $\boldsymbol{\beta}_{k+1} = \boldsymbol{\beta}^{'}$. In case the trial is worse we will accept it with a probability P which decreases as the worsening gets larger or the temperature at iteration k denoted T^k decreases:

$$P(\boldsymbol{\beta}_k, \boldsymbol{\beta}^{'}, T^k) = e^{\frac{-(F(\boldsymbol{\beta}^{'})-F(\boldsymbol{\beta}_k))}{T^k}}$$

The algorithm starts at some high temperature T_0 and during the algorithm it is cooled down so that over time the probability to accept a worsening solution decreases.

5.5 Comparison of potential solution methods

We decided to select both the finite differences and analytical gradient descent methods from the potential solution methods described so far as a method to be actually implemented and tested against the hillclimbing method which serves as a reference. In terms of the performance criteria gradient descent based methods seem to complement hillclimbing: gradient descent seems slower computationally but trading this for being potentially more reliable in improving per iteration (reliability of finding a quality solution) compared to hillclimbing.

The main reason Stochastic simultaneous perturbation approximation (SPSA) and Simulated annealing are not further investigated is because they have a lot of solution method parameters and seem much more complicated and heuristic compared to gradient descent. Another reason we did not choose SPSA is that we do not have a lot parameters to calibrate (only 6) and SPSA usually is applied in case one wants to calibrate many parameters as then SPSA becomes especially advantageous relative to a finite differences approach. Overall gradient descent seems to be a more logical choice to go with as a first alternative method to hillclimbing.

Table 2 summarizes the preliminary assessment of the proposed solution methods in terms of the performance criteria established in section 4.4. Note that columns are missing for both the *Reliability of finding a quality solution* and *Robustness under initial solution guess* criteria because it is difficult to assess the methods on these at this stage. The selected methods, so both hillclimbing and gradient descent, will need to be implemented and tested on actual models before coming back to these criteria. The *Minimizer type* column in table 2 did not appear as a criterium in section 4.4, but ties in with the *Reliability of finding a quality solution* criterium since it aims to classify what type of solution each algorithm at best finds.

	Performance criteria			
Solution method	Minimizer type	# solution method parameters	Running time (expressed in # of IPF runs per iteration)	Difficulty of implementation
Hillclimbing:	Local minimizer	Moderate	1	Easy
gradient descent (finite differences):	Local minimizer	Few	$2n + 1$ (central) or $n + 1$ (forward)	Easy
gradient descent (analytical):	Local minimizer	Few	dependent on # of zones	Hard
SPSA:	Stochastic local minimizer	Many	2*(# of perturbations (m))	Easy
Simulated annealing:	Stochastic global minimizer	Many	1	Easy

Table 2: Preliminary assessment of the potential solution methods on some of the performance criteria of section 4.4

6 BFGS method

This chapter describes a BFGS method that we implemented and tested for the calibration problem. Section 6.1 gives a short discussion of line search methods. Then in section 6.2 the particular Quasi Newton method that was implemented for the calibration problem, which is a BFGS method, is described.

6.1 Line search approach

In optimization, line search is a basic iterative approaches to find a local minimum of an objective function. The line search approach first finds a descent direction along which the objective function will be reduced and then computes a step size determining how far the solution should move along that direction to find the next iterate. The descent direction can be computed by various methods, such as the method of steepest descent, Newton's method and the Quasi-Newton method. In section 5.2 we mentioned the steepest descent method which uses only gradients to update the solution. Newton's method also uses second order derivatives of the objective function in the form of the Hessian matrix $\boldsymbol{H}$. The Hessian matrix $\boldsymbol{H}^k$ for the k_{th} iterate $\boldsymbol{\beta}^k$ is defined in terms of the second order derivatives by $H_{ij} = \frac{\partial^2 F(\boldsymbol{\beta}^k)}{\partial \beta_i \partial \beta_j}$. The update rule for Newton's method then is given by:

$$\boldsymbol{\beta_{k+1}} = \boldsymbol{\beta_k} - step_k \cdot [\boldsymbol{H}^k]^{-1} \nabla F(\boldsymbol{\beta_k}) \tag{26}$$

Newton's method usually leads to a much faster convergence to a local minimum (quadratic instead of a linear rate of convergence) but requires calculation of the Hessian and solving a linear system which can be expensive. In our case calculating the Hessian by a finite difference approach would likely require too many function evaluations, however solving the linear system part would be effortless due to low dimensionality.

Quasi-Newton methods use the same update rule as (26). However instead of calculating the actual Hessian the matrix $\boldsymbol{H}$ in the update rule is an approximation to the Hessian that is updated from the sampled gradients and function values throughout the iterations. The Quasi-Newton methods are in between steepest descent and Newton's method in terms of convergence rate with a superlinear convergence rate (in case of convergence). The extra time required for Quasi-Newton compared to steepest descent is mainly in solving a linear system in (26) and is negligible again due to low dimensionality of the calibration problem. The promise of better convergence properties with no added running time increase motivated us to also implement a Quasi-Newton line method. Unlike in the case of convex problems, proofs of convergence for Quasi-Newton methods are not known for general nonlinear problems. However Quasi-Newton methods are still a popular choice for nonlinear optimization problems.

6.2 BFGS method implementation details

Algorithm 5 Projected bound-constrained BFGS with damped update.

```
Solution method parameters:
Relative convergence treshold: $\epsilon \in (0,1)$,
Step size reduction factor: $\mu \in (0,1)$,
Sufficient decrease parameter: $c_1 \in (0,1)$,
Maximum number of iterations: $max_{it} \in \mathbb{N}$,
Maximum value of $\beta$: $\beta_{max} < 0$
$F^0 \leftarrow F(\boldsymbol{\beta}^0)$
$\boldsymbol{g}^0 \leftarrow \nabla F(\boldsymbol{\beta}^0)$
$\epsilon_0 \leftarrow \epsilon \cdot \|\boldsymbol{g}^0\|_2$
$\boldsymbol{H}^0 \leftarrow \|\boldsymbol{g}^0\|_2 \cdot \boldsymbol{I}$
$Converged \leftarrow FALSE$
$k \leftarrow 0$

while $Converged = FALSE$ do
    $\boldsymbol{S}^k \leftarrow \boldsymbol{H}_k^{-1}$
    Calculate the first index set $I_1^k$ by 29
    Calculate $\bar{\boldsymbol{S}}^k$ by 30
    Calculate the second index set $I_2^k$ by 31
    $I^k = I_1^k \cup I_2^k$
    Calculate $\widehat{\boldsymbol{S}}^k$ by 32

    $\boldsymbol{p}^k \leftarrow -\widehat{\boldsymbol{S}}^k \cdot \boldsymbol{g}^k$
    $step \leftarrow 1$
    $\boldsymbol{\beta}^{k+1} \leftarrow P_{<0}(\boldsymbol{\beta}^k + step \cdot \boldsymbol{p}^k)$
    $F^{k+1} \leftarrow F(\boldsymbol{\beta}^{k+1})$
    while $F^{k+1} > F^k - c_1 \cdot step \cdot \boldsymbol{p}_k^T \boldsymbol{g}^k$ do
        $step \leftarrow \mu \cdot step$
        $\boldsymbol{\beta}^{k+1} \leftarrow P_{<0}(\boldsymbol{\beta}^k + step \cdot \boldsymbol{p}^k)$
        $F^{k+1} \leftarrow F(\boldsymbol{\beta}^{k+1})$
    end while
    $\boldsymbol{g}^{k+1} \leftarrow \nabla F(\boldsymbol{\beta}^{k+1})$
    $\boldsymbol{s} \leftarrow \boldsymbol{\beta}^{k+1} - \boldsymbol{\beta}^k$
    $\boldsymbol{y} \leftarrow \boldsymbol{g}^{k+1} - \boldsymbol{g}^k$
    if $\boldsymbol{s}^T\boldsymbol{y} \geq 0.2 \cdot \boldsymbol{s}^T\boldsymbol{H}^k\boldsymbol{s}$ then $\theta \leftarrow 1$ else $\theta \leftarrow 0.8 \cdot \frac{\boldsymbol{s}^T\boldsymbol{H}^k\boldsymbol{s}}{\boldsymbol{s}^T\boldsymbol{H}^k\boldsymbol{s} - \boldsymbol{s}^T\boldsymbol{y}}$
    $\boldsymbol{y} \leftarrow \theta \cdot \boldsymbol{y} + (1-\theta) \cdot \boldsymbol{H}^k\boldsymbol{s}$
    $\boldsymbol{H}^{k+1} \leftarrow \boldsymbol{H}^k + \frac{\boldsymbol{y}\boldsymbol{y}^T}{\boldsymbol{y}^T\boldsymbol{s}} - \frac{\boldsymbol{H}^k\boldsymbol{s}\boldsymbol{s}^T\boldsymbol{H}^k}{\boldsymbol{s}^T\boldsymbol{H}^k\boldsymbol{s}}$
    if $\|\widehat{\boldsymbol{g}}^{k+1}\|_2 < \epsilon_0$   OR   $k > max_{it}$ then          (with $\widehat{\boldsymbol{g}}^{k+1}$ defined by (33))
        $Converged \leftarrow TRUE$
    end if
    $k \leftarrow k + 1$
end while
```

The method described in algorithm 5 for the most part is the basic BFGS (named after its inventors Broyden, Fletcher, Goldfarb and Shanno) Quasi-Newton method which is described in the classic optimization book of

Nocedal and Wright[16]. It is characterized by the Hessian approximation update rule given by:

$$\boldsymbol{H}^{k+1} \leftarrow \boldsymbol{H}^k + \frac{\boldsymbol{y}\boldsymbol{y}^T}{\boldsymbol{y}^T\boldsymbol{s}} - \frac{\boldsymbol{H}^k\boldsymbol{s}\boldsymbol{s}^T\boldsymbol{H}^k}{\boldsymbol{s}^T\boldsymbol{H}^k\boldsymbol{s}} \tag{27}$$

Other Quasi-Newton methods exist and are characterized by different update rules. We have chosen the BFGS update rule as it is the most popular one and is believed to be the most efficient among the existing Quasi-Newton methods.

6.2.1 Initial Hessian approximation choice

All Quasi-Newton methods require an initial Hessian approximation $\boldsymbol{H}^0$. The most simple choice is the identity matrix $\boldsymbol{I}$ which meets the only requirement of being positive definite. In this case the first iteration of BFGS would be a steepest descent iteration, as the search direction $\boldsymbol{p}^0$ would equal the negative initial gradient $-\boldsymbol{g}^0$, by line 8 of algorithm 5. Nocedal and Wright[16] presents two other suggestions that scale the identity matrix by a positive constant. The first option scales the identity matrix by $\frac{\|\boldsymbol{g}^0\|_2}{\sigma}$. Here σ is a solution parameter and has the effect of scaling the norm of the initial step to σ. The other option uses the scaling factor $\frac{\boldsymbol{y}^T\boldsymbol{s}}{\boldsymbol{y}^T\boldsymbol{y}}$ after the first step is computed but before the BFGS update. This option has, compared with the first option, a more sound mathematical motivation as it tries to make the size of $\boldsymbol{H}^0$ equal to the true Hessian in a sense explained by Nocedal and Wright. However we tested both scaling options for small synthetic models and the first option with σ set to 1 was observed to be slightly favourable in terms of the time till convergence, and was therefore used in the final implementation.

6.2.2 Choice of step size rule

Given we have computed a search direction $\boldsymbol{p}^k$ an appropriate step size to take along this take direction has to be computed. The Armijo step size rule makes sure the Armijo condition is satisfied. The Armijo condition is also called the sufficient decrease condition and requires:

$$F^{k+1} \leq F^k + c_1 \cdot step \cdot \boldsymbol{p}_k^T\boldsymbol{g}^k \tag{28}$$

Here $c_1 \in (0, 1)$ is a solution parameter. The Armijo rule is also called backtracking line search. From an initial step size it iteratively reduces the step size by a reduction factor $\mu \in (0, 1)$ until the Armijo condition is enforced. A search direction $\boldsymbol{p}^k$ is defined to be a descent direction if there exists a step size $\lambda > 0$ such that the current solution is improved with this step size along the direction i.e. $F(\boldsymbol{\beta}^k + \lambda \cdot \boldsymbol{p}^k) < F(\boldsymbol{\beta}^k)$. Given that the search direction $\boldsymbol{p}^k$ is a descent direction clearly the backtracking procedure will terminate at some step size satisfying the sufficient decrease condition as the step size *step* in (28) will decrease and make the condition easier to be satisfied. For $\boldsymbol{p}^k$ in the BFGS method to be a descent direction requires that the Hessian approximation $\boldsymbol{H}^k$ remains positive definite i.e. the update formula (27) ensures positive definiteness. In case the so called curvature condition holds, which is $\boldsymbol{s}^T\boldsymbol{y} > 0$, positive definiteness is implied. This curvature condition holds for a strictly convex objective function automatically, however our problem is nonconvex. Therefore the curvature condition needs to be enforced by the step size algorithm. A popular choice is to enforce the so called Wolfe conditions which consists of both the sufficient decrease condition in (28) and the following curvature condition, which implies that $\boldsymbol{s}^T\boldsymbol{y} > 0$:

$$\nabla F(\boldsymbol{\beta}_k + step_k \cdot \boldsymbol{p}_k)^T\boldsymbol{p}_k \geq c_2 \cdot \boldsymbol{g}_k^T\boldsymbol{p}_k$$

Where c_2 satisfying $0 < c_1 < c_2 < 1$ is a solution parameter. A potential disadvantage of a line search enforcing this condition is that multiple gradients $\nabla F(\boldsymbol{\beta}_k + step_k \cdot \boldsymbol{p}_k)$ might be required to be calculated for the different step size candidates during one iteration. Because the Armijo rule with its 1 gradient calculation per line search seemed cheap computationally we implemented the Armijo step size rule. Another reason is that the Armijo step size rule was also used in Kim et al.[9] which we used as a reference for dealing with the negativity constraints. However a drawback is that the Armijo rule does not enforce the curvature condition so additional measures have to be taken.

Damped Hessian update rule

The first option discussed by Nocedal and Wright[16] to prevent the Hessian approximation from becoming non positive definite is to skip the update in case $\boldsymbol{s}^T\boldsymbol{y} \leq 0$ is observed i.e. set $\boldsymbol{H}^{k+1} = \boldsymbol{H}^k$. However skipping has the risk of not including important curvature information. Therefore they suggest a more effective alternative which is to use a damped update. The damped update is done via the normal BFGS formula (27) but ensures positive definiteness of $\boldsymbol{H}^{k+1}$ by modifying the definition of $\boldsymbol{y}$ in case $\boldsymbol{s^T y}$ is not sufficiently large. The exact implementation of the damping rule is described in lines 26 and 27 of algorithm 5.

6.2.3 Negativity constraints

The calibration problem we are trying to solve is a nonlinear bound-constrained optimization problem. The bound-constraints here are the negativity constraints on the β's. To deal with the bound-constraints we have used the projected quasi-Newton approach discussed in Kim et al.[9]. We now describe the implementation details related to the bound-constraints of algorithm 5.

Before each line search, the variables are partitioned into fixed and free variables. These fixed variables are left unchanged in the optimization, while only the free ones are updated and are considered in the (reduced) Hessian approximation for scaling the gradient. The fixed variables are a subset of the active variables which are variables at which the corresponding bound-constraint hold with equality, in our case: the variable β_i^k is active at iteration k if $\beta_i^k = \beta_{max}$. For an active variable to be fixed requires either the corresponding gradient entry to be negative or a scaled gradient based on second order information to be negative. In particular, we compute two sets I_1^k and I_2^k at iteration k. The first index set, also called the binding set, is calculated as:

$$I_1^k = \{i \mid \beta_i^k = \beta_{max} \text{ and } \boldsymbol{g}_i^k < 0\} \tag{29}$$

Then the matrix $\bar{\boldsymbol{S}}^k$ is obtained from $\boldsymbol{S}^k$ (which is the inverse of the current iterations Hessian approximation i.e. $\boldsymbol{S}_k = \boldsymbol{H}_k^{-1}$, see line 8) by vanishing all rows and columns corresponding with the first index set or binding set I_1^k:

$$\bar{\boldsymbol{S}^k}_{ij} = \begin{cases} \boldsymbol{S}_{ij}^k, & i,j \notin I_1^k \\ 0, & else \end{cases} \tag{30}$$

The second index set I_2^k consists of variables for which the corresponding entry of the gradient scaled by $\bar{\boldsymbol{S}}^k$ is negative:

$$I_2^k = \{i \mid \beta_i^k = \beta_{max} \text{ and } (\bar{\boldsymbol{S}}^k \boldsymbol{g}^k)_i < 0\} \tag{31}$$

Then the matrix $\widehat{\boldsymbol{S}^k}$ is obtained from $\boldsymbol{S}^k$ by vanishing all rows and columns of indices in the fixed set $I^k = I_1^k \cup I_2^k$:

$$\widehat{\boldsymbol{S}^k}_{ij} = \begin{cases} \boldsymbol{S}_{ij}^k, & i,j \notin I^k \\ 0, & else \end{cases} \tag{32}$$

The search direction is obtained by scaling the (negative) gradient by this matrix $\widehat{\boldsymbol{S}^k}$ in line 14 of algorithm 5.

Optimization convergence criterion

The final detail in which the projected BFGS method is different from the basic non projected framework is in its convergence criterion. The convergence criterion considers a reduced gradient $\widehat{\boldsymbol{g}}^k$ in line 29 of algorithm 5 so that only the free parameters are counted in calculating the gradients norm:

$$\widehat{\boldsymbol{g}}_i^k = \begin{cases} \boldsymbol{g}_i^k, & i \notin I^k \\ 0, & else \end{cases} \tag{33}$$

The convergence criterion we used considers the relative improvement from the initial gradient norm. The threshold $\epsilon \in (0, 1)$ parameter sets the fraction of the initial gradient norm below which the gradient should vanish. The relative criterion gives an uniform threshold for different choices of weighting in the objective function.

7 Results

In this chapter we describe the results of the tests that were performed. The trip distribution models as described in chapter 2 and 3, the calibration methods of hillclimbing, steepest descent described in chapter 5 and the BFGS method described in chapter 6, including the gradient calculation routines of finite differences and the adjoint method were all implemented in MATLAB. Test results pertaining to the calculation of the gradient are discussed in section 7.1. In section 7.2 results of the calibration using the different methods on a medium scale and large scale strategic traffic model are presented.

7.1 Calculating the gradient

We calculated gradients using both the finite difference approach of section 5.2.1 and the adjoint analytical approach of section 5.2.3. It is interesting to consider how accurate both methods are and whether they compute approximately the same gradient. Also it is interesting to look at the methods' running times. We explore these aspects in section 7.1.1 which is on the accuracy of the methods and section 7.1.2 which on the computational effort of the methods.

7.1.1 Accuracy

We compared the accuracy of the finite difference approach and analytical approach to calculating gradients on the BBMA Work trip distribution model. The number of iterations that is done inside the gravity model influences the gradient that is computed for both approaches. For the finite differences approach choosing a lower number of iterations will alter the trip distribution obtained and so the objective values computed. This impacts the gradient calculated. For the analytical approach the trip-end residuals are assumed to be zero (to obtain the linear system in (21)). This brings about an inaccuracy in the calculation of the gradient too. It seems reasonable that given the number of iterations is high enough both gradient approaches should converge to the same gradient vector i.e. the relative norm between them should become small. The difference between forward and central differences was observed to be negligible. Therefore we chose to use the forward differences approach requiring $n + 1$ objective function evaluations instead of central difference's $2n$. For the perturbation size we set $\Delta = 10^{-8}$ in 19. The results are shown in Figure 13. Table 3 shows the data points for the lower number of iterations of figure 13. After 25 iterations the relative error between the two gradients is 1%.

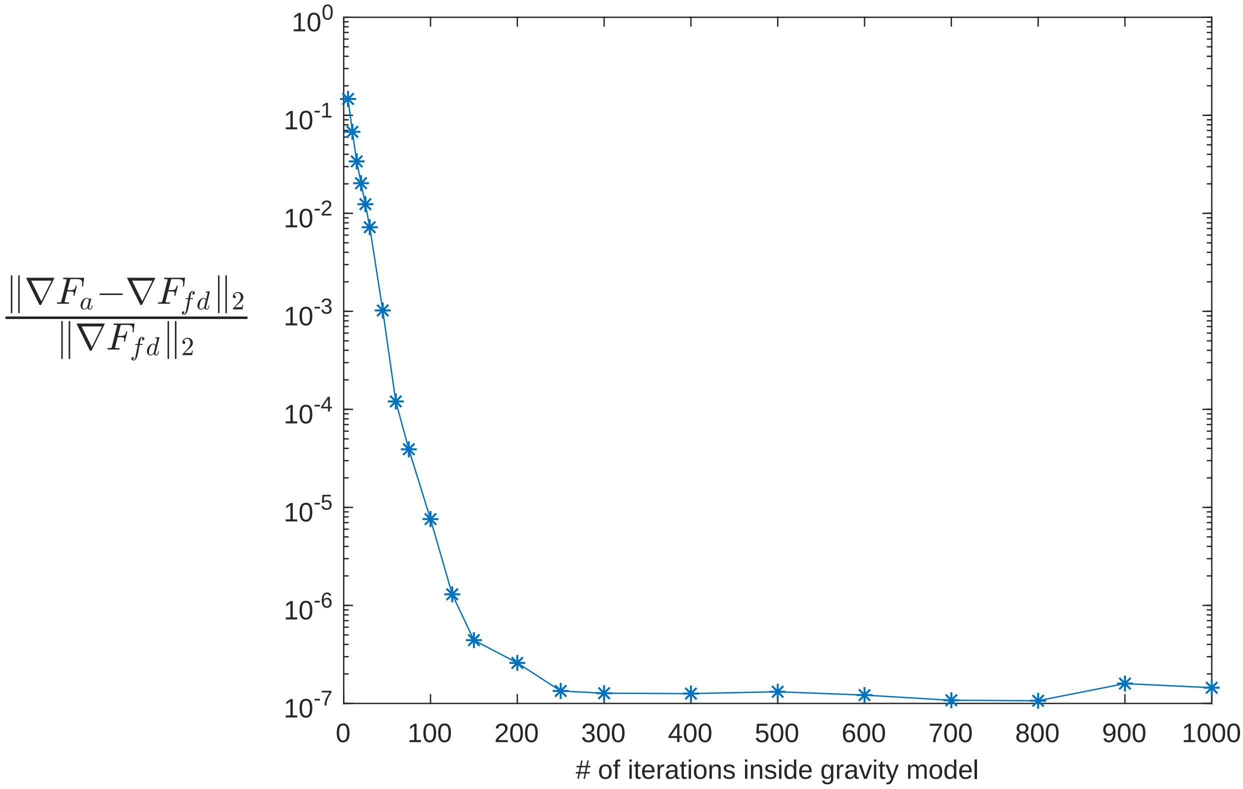


Figure 13

# of iterations gravity model:	5	10	15	20	25	30	45
Relative norm:	0.15	0.07	0.03	0.02	0.01	0.007	0.001

Table 3: Relative norm for low number of iterations in figure 13.

Singular Jacobians

It was observed occasionally that the linear the system of (21) was singular. This is equivalent to the corresponding Jacobian matrix $\boldsymbol{A}$ not having an inverse. In case a matrix is singular either the corresponding system admits zero solutions or infinitely many. This first option was observed to be the case. A possible explanation is that the coefficients and equations that are involved in the analytical gradient calculation are inconsistent with the coefficients and equations of the gravity model due to an error or inaccuracy in the MATLAB implementation. Because of this difficulty we decided to compute gradients by the forward finite difference approach instead of using the analytical method in the steepest descent and BFGS method. We showed how both gradient routines compute the same gradient. So using finite differences should not make a difference in the test results other than different running times.

7.1.2 Computational effort

We observed for the two methods for different model sizes n how much time it takes to compute the gradient. For each number of zones n a trip distribution model with n zones was obtained by resizing and interpolating the cost matrices, production- and attraction vectors of the BBMA Work purpose model of 1425 zones for the various n. Specifically we used MATLAB's *imresize* function to rescale the matrices and vectors and used 'bilinear interpolation' as the method parameter.

The gravity convergence criterion used in this test but also in what follows for the rest of this thesis is to require R^{rel}_{prod} and R^{rel}_{attr} in (17) to be both less than 5%. Let $t_{fd}(n)$ and $t_{al}(n)$ be the time needed to compute a gradient with the finite difference approach and the analytical approach respectively. Then we have for the approach of finite differences (assuming 6 parameters β_{mu}, which is the case for our models of interest in this thesis) that:

$$t_{fd}(n) = 6 \cdot t_{gravityrun}(n) = 6 \cdot \#_{it}(n) \cdot t_{it}(n)$$

With $t_{gravityrun}(n)$ the observed running time to calculate one gravity run, which is simply the number of gravity model iterations $\#_{it}(n)$ multiplied by the time to compute one iteration $t_{it}(n)$. The convergence of the gravity model i.e. $\#_{it}(n)$ was observed to vary quite significantly with n. Here we wanted to study the relationship between model size n and the running times of both methods. To get this relationship more clearly without the gravity model convergence impact we normalized the iteration numbers $\#_{it}(n)$ to the same number 15. The results after this normalization step are shown in figure 14. The figure shows the factor by which the analytical gradient approach is faster than the finite difference approach i.e. $\frac{t_{fd}(n)}{t_{al}(n)}$. The constant red line of factor 1 is drawn to indicate the turnover point (which is at the intersection of the blue and red line) at which the analytical method becomes more advantageous than the finite differences approach. The analytical approach is significantly faster for models of size smaller than 3300 zones. If one wants to compare the competitiveness of the methods based on these results in case convergence of the gravity model requires a different number of iterations say n_x, one can simply multiply the factors by $\frac{n_x}{15}$.

Running time analysis: big O notation

The results on running time presented in figure 14 are not unexpected from an Big O notation analysis standpoint. The finite difference approach boils down to running the IPF procedure multiple times which is essentially repeated scaling of all rows and columns. Therefore the finite difference approach has a running time of $O(n^2)$. The analytical gradient approach at least requires us to solve a linear system of size $3n + 6$. In general solving a linear system of such a size takes $O(n^3)$. This confirms again that the finite differences approach should be faster than the analytical approach for large n.

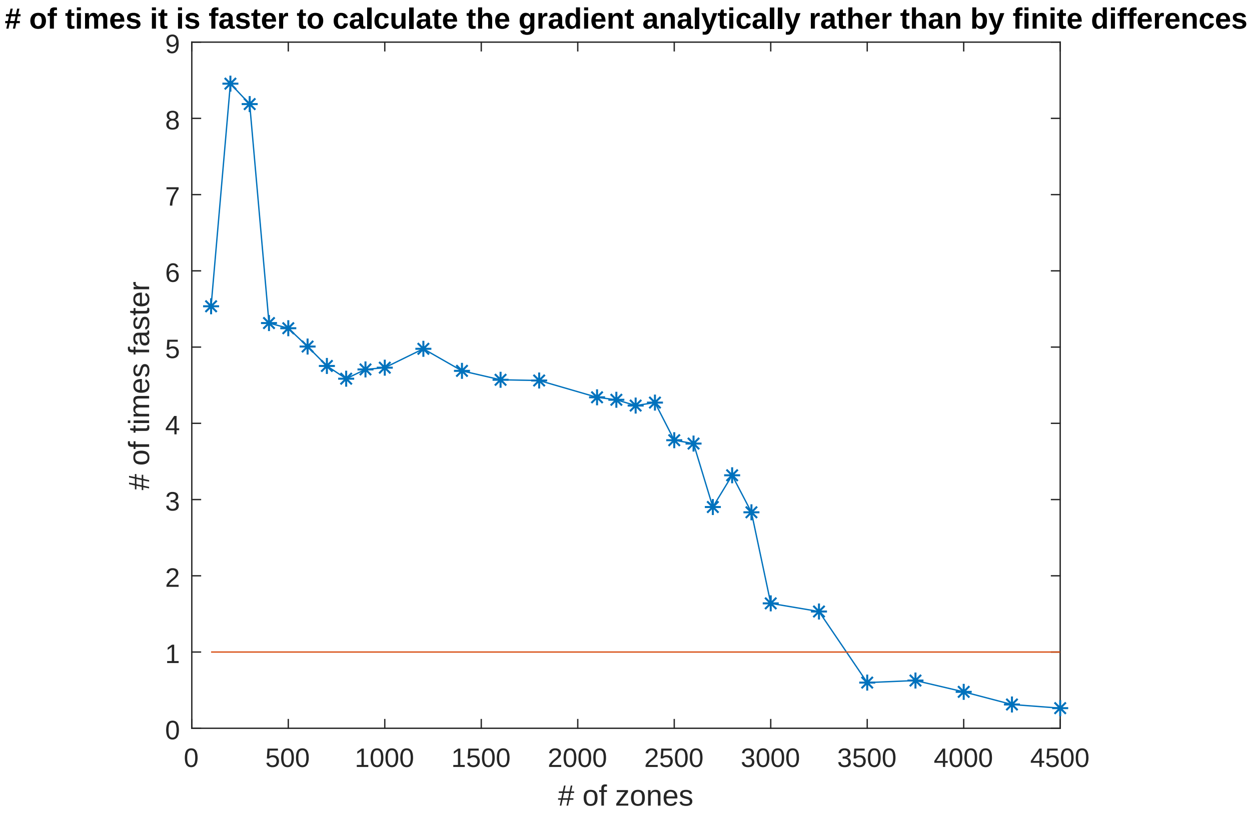


Figure 14: Observed factor by which calculating the gradient analytically is faster than computing it by finite differences for varying number of zones (assuming gravity model convergence in 15 iterations per sub purpose)

7.2 Calibration results

Calibration runs were done using the hillclimbing algorithm, a steepest descent and the BFGS method of chapter 6 on two models: the medium scale BBMA strategic traffic model and the large scale MRDH strategic traffic model. The main goal of these tests was to investigate the performance of the hillclimbing algorithm and the BFGS method in terms of the (i) *reliability in finding a quality solution*, (iii) *robustness under initial solution guess* and (iv): *Running time* criteria, as discussed in section 4.4. For the *reliability in finding a quality solution* criterium we have the goal to answer the following questions: What is the nature of the solution each of the methods converge to? Is it a global minimum, local minimum or perhaps not even a local minimum and thus suboptimal. An important question that ties in with the *robustness under initial solution guess* is whether or not there exist multiple local minima in the solution space. Finally the running times of both methods are of interest.

7.2.1 Calibration of the medium scale BBMA strategic traffic model

The first model we calibrated is the medium scale BBMA model of 1425 zones. The trip distribution model for the BBMA model is similar in structure to the MRDH model which is discussed in the next section. Both have a stratification into purposes and sub purposes as summarized in table 1. The advantage of testing on the BBMA model is that we are able to test more initial solution guesses in the same time period due to its smaller size (1425 versus MRDH model's 7786 zones) the running time of both our methods is much faster. This is useful in obtaining data to answer the questions surrounding the *reliability in finding a quality solution* and *robustness under initial solution guess* criteria.

As no count numbers were available for the BBMA model, no confidence intervals could be constructed, therefore the calibration objective function was taken to be (15), so neither normalization by bin width or confidence interval width was applied here.

For each purpose 16 different starting solutions were randomly selected, where $\beta^0_{mu} \in [-1, 0]$ for each of the 6 mode-user class combinations (m, u). We believe that the initial guesses can be constrained to this cube after observing many bad objective values corresponding to solutions outside the cube in an experiment preliminary to this test. Both the hillclimbing and a steepest descent algorithm were, for each purpose, started from the same set of 16 $\boldsymbol{\beta}$'s.

Solution method setup

For the hillclimbing algorithm, the initial step size *maxstep*, was set to 0.5, as this value was used for the BBMA model calibration before this research. The initial signs $sign_{mu}$ were all set to 1 and the step size reduction factor γ was set to the golden ratio $\frac{1+\sqrt{5}}{2}$. For the hillclimbing convergence criterium we simply iterated for $maxiterations = 40$ iterations. Each run this number was observed to be more than necessary for convergence.

We mentioned that we used a steepest descent algorithm instead of the BFGS method for the BBMA model. The reason for this is that initially we started by implementing the steepest descent for the sake of it being the most simple and straightforward type of line search algorithm. The BFGS method was implemented only in a later phase in hopes of improving the convergence rate. However the results of the BBMA tests employing steepest descent were sufficient to answer the earlier mentioned questions about whether there are multiple local minima and whether the hillclimbing algorithm finds a local minimum.

The details of this steepest descent method then are as follows: we used a simple backtracking line search with the initial step size set to 0.25 and the step size reduction factor set to $\frac{1}{2}$. The sufficient decrease parameter (see (28)) was chosen to be $c_1 = 10^{-4}$, which is often used in backtracking line search in practice[16]. For the convergence criterium the maximum number of iterations was set to $max_{it} = 40$ iterations, which each time proved to be more than necessary to reduce the gradients norm below 0.01, the absolute convergence treshold used.

Results on the BBMA model

For each calibration run we observed one of two outcomes: either hillclimbing converges to the same solution as steepest descent or it finds a worse solution in terms of objective function value. The frequencies of these outcomes are summarized in table 4. For each purpose it was observed that for each of the 16 different initial solutions steepest descent converged to the same solution. Figure 15 and 16 respectively compare the convergence in objective function value of the methods in case hillclimbing converges to the same solution as steepest descent and in case it does not.

In our discussion of the *reliability in finding a quality solution* criterium in section 4.4 we mentioned that plots of observed versus modelled trip length distributions are often more meaningful to a consultant than an objective function value. Figure 17 and figure 18 compare the modelled trip length distributions of the hillclimbing and steepest descent solution with the observed trip length distribution for respectively a run for the Education purpose and Work purpose in both of which hillclimbing ended in a suboptimal solution.

Purpose	# of times hillclimbing	# of times hillclimbing	# of different	hillclimbing obj	steepest descent
	finds a worse solution	finds the same solution	starting solutions	value avg.:	obj value avg.
Work:	1	15	16	0.124	0.082
Business:	0	16	16	0.115	0.115
Education:	3	13	16	0.118	0.084
Stores:	1	15	16	0.136	0.121
Other:	0	16	16	0.040	0.040
Total:	5 (6.25%)	75 (93.75) %	80 (100 %)		

Table 4: Summary of results comparing hillclimbing and steepest descent for the BBMA strategic traffic model.

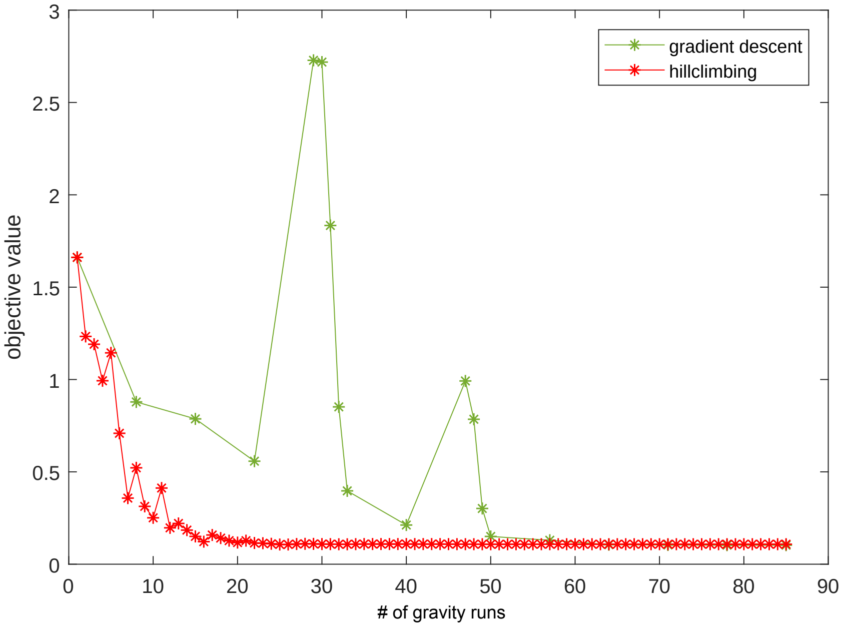


Figure 15: Typical objective function convergence for a good initial solution for hillclimbing, Purpose: Work

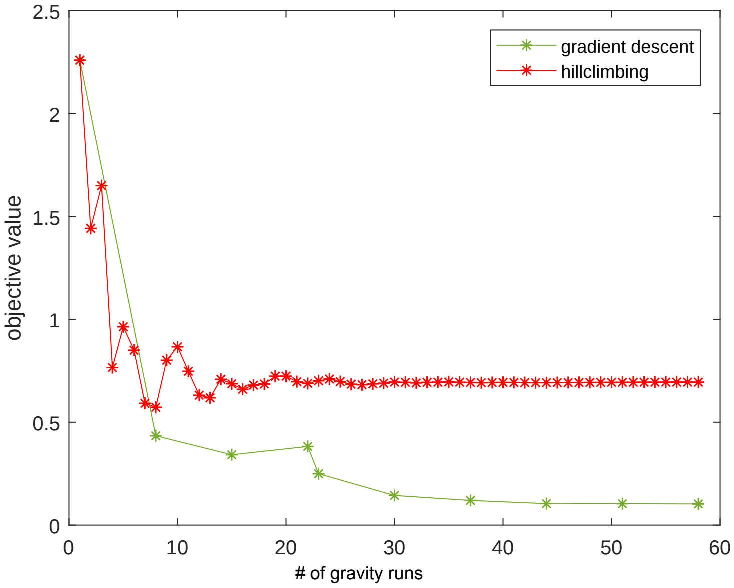


Figure 16: Typical objective function convergence for a bad initial solution for hillclimbing, Purpose: Work

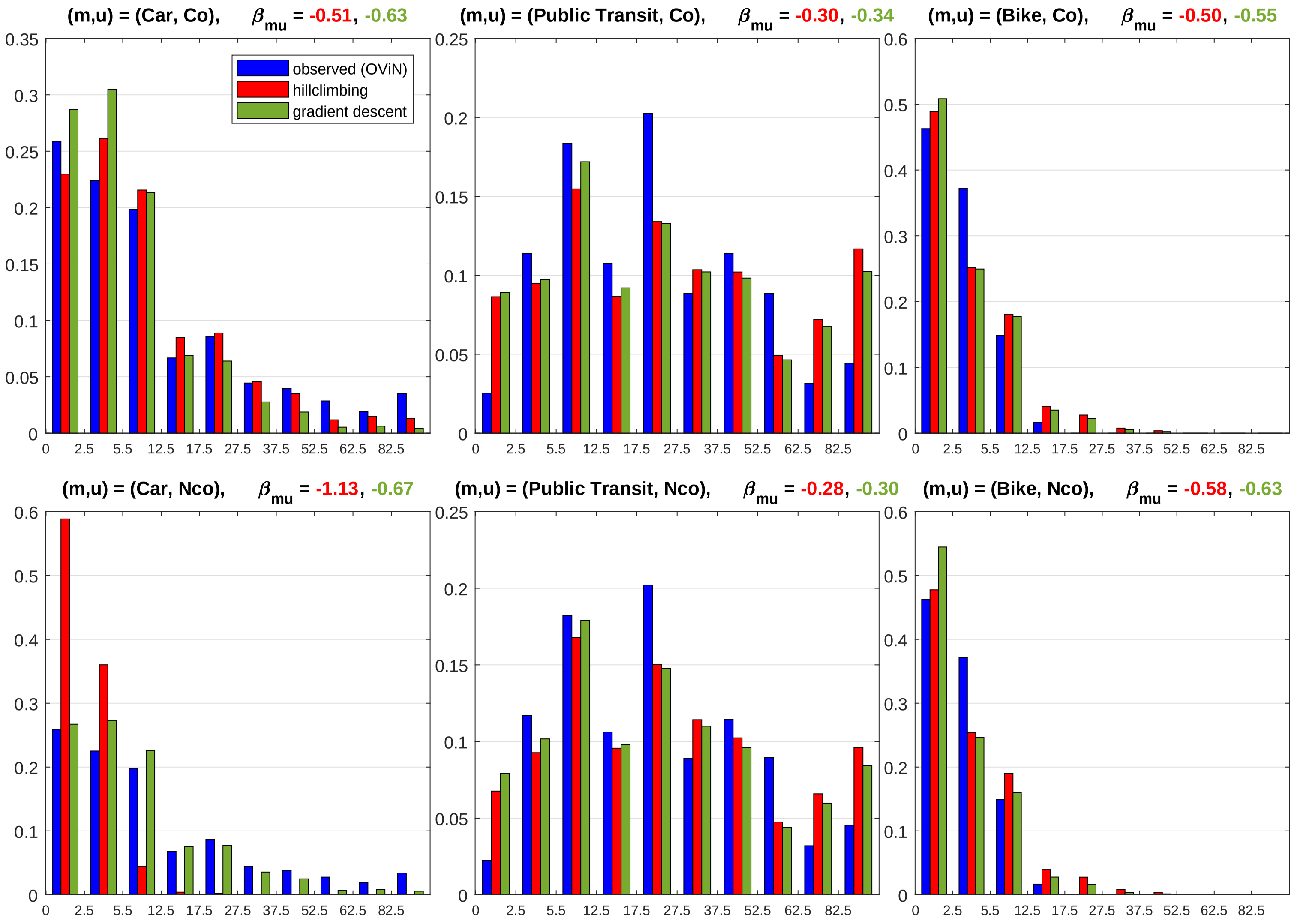


Figure 17: Bad solution for hillclimbing for the Education purpose of the BBMA model

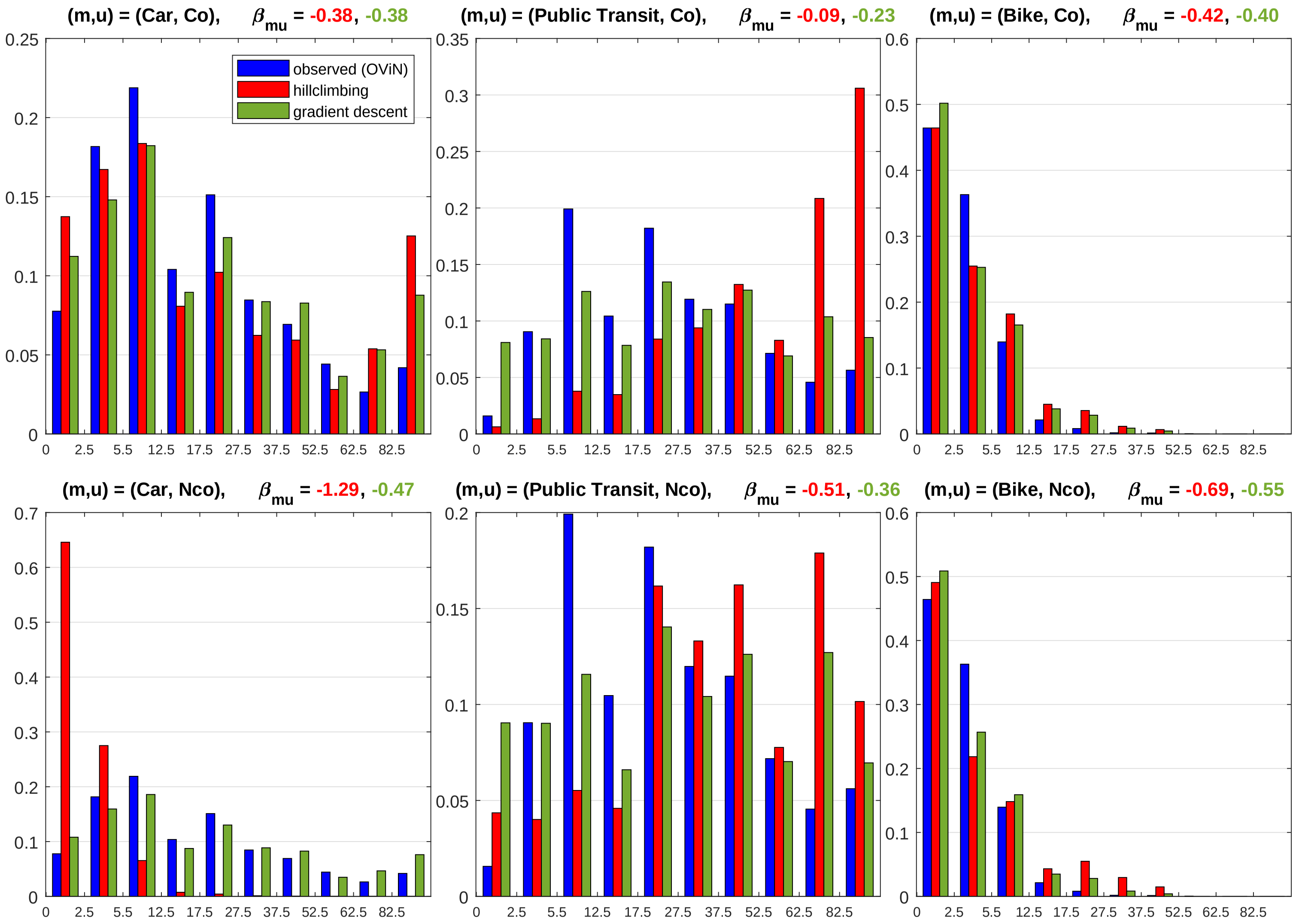


Figure 18: Bad solution for hillclimbing for the Work purpose of the BBMA model

7.2.2 Calibration of the large scale MRDH strategic traffic model

Robustness under initial starting solution

The first test comparing hillclimbing and the BFGS method for the MRDH model was done for the Other purpose. Initially we intentionally focused on this single purpose and did multiple runs from different initial solutions as the calibration running times for the MRDH model compared to the BBMA are significantly longer (a single BFGS calibration run can take up to twenty hours compared to 25 minutes for BBMA). For the Other purpose 4 runs were done with different starting solutions randomly drawn from the $[-1, 0]^6$ cube in which we expect the minimum to be in, given now our experience with the BBMA model.

The objective function of the calibration is again the unweighted objective function of (15) with no binwidth normalization or incorporation of confidence intervals.

Solution method setup

For the hillclimbing algorithm, the initial step size *maxstep*, was finally set to 0.1, as it was observed to be somewhat more favourable than some larger step sizes of 0.25, 0.5 and 1 that were also tried. The initial signs $sign_{mu}$ were all set to 1 and the step size reduction factor γ was set to the golden ratio $\frac{1+\sqrt{5}}{2}$ as was the case for the BBMA model. Again iterating for $maxiterations = 40$ iterations proved to be sufficient for practical convergence.

The parameter values used for the BFGS method i.e. algorithm 5 are as follows: the step size reduction factor was set to $\mu = \frac{1}{2}$ and the sufficient decrease parameter was set to $c_1 = 10^{-4}$. For the convergence criterium the maximum number of iterations was set to $max_{it} = 40$ iterations, however convergence each time was achieved in the relative gradient norm criterion with relative convergence threshold value $\epsilon = 10^{-4}$ before this number of iterations.

Results Other purpose MRDH model

It was observed for each starting solution that BFGS found a solution with smaller objective value than hillclimbing. The converged hillclimbing solutions also were dependent on the initial solution guess, while for BFGS method independence was observed i.e. each run converged to the same solution. A visual comparison of the solution in terms of the dashboard a consultant would judge the solution on for one of these runs, is shown in figure 19.

Change of bin division

The solution obtained from the BFGS method of the Other purpose was still found to be unsatisfying. The modelled trip length distributions did not approach the observed trip length distributions sufficiently. Therefore it was decided for the final tests on the MRDH model to use the BBMA bin division, instead of the MRDH bin division. The BBMA bin division differs in that it is more aggregate for the smallest trip lengths. The fact we changed the bin division does not necessarily reflect poorly on the BFGS method itself. Although it is possible that the solution obtained is a local minimum that is significantly worse than the global solution. This seems unlikely as the BBMA and MRDH test results suggest the BFGS method converges to the same local minimum independent from its starting solution, therefore suggesting it converges to the global minimum. The measure of using the BBMA bin division can be thought of as changing the model's data.

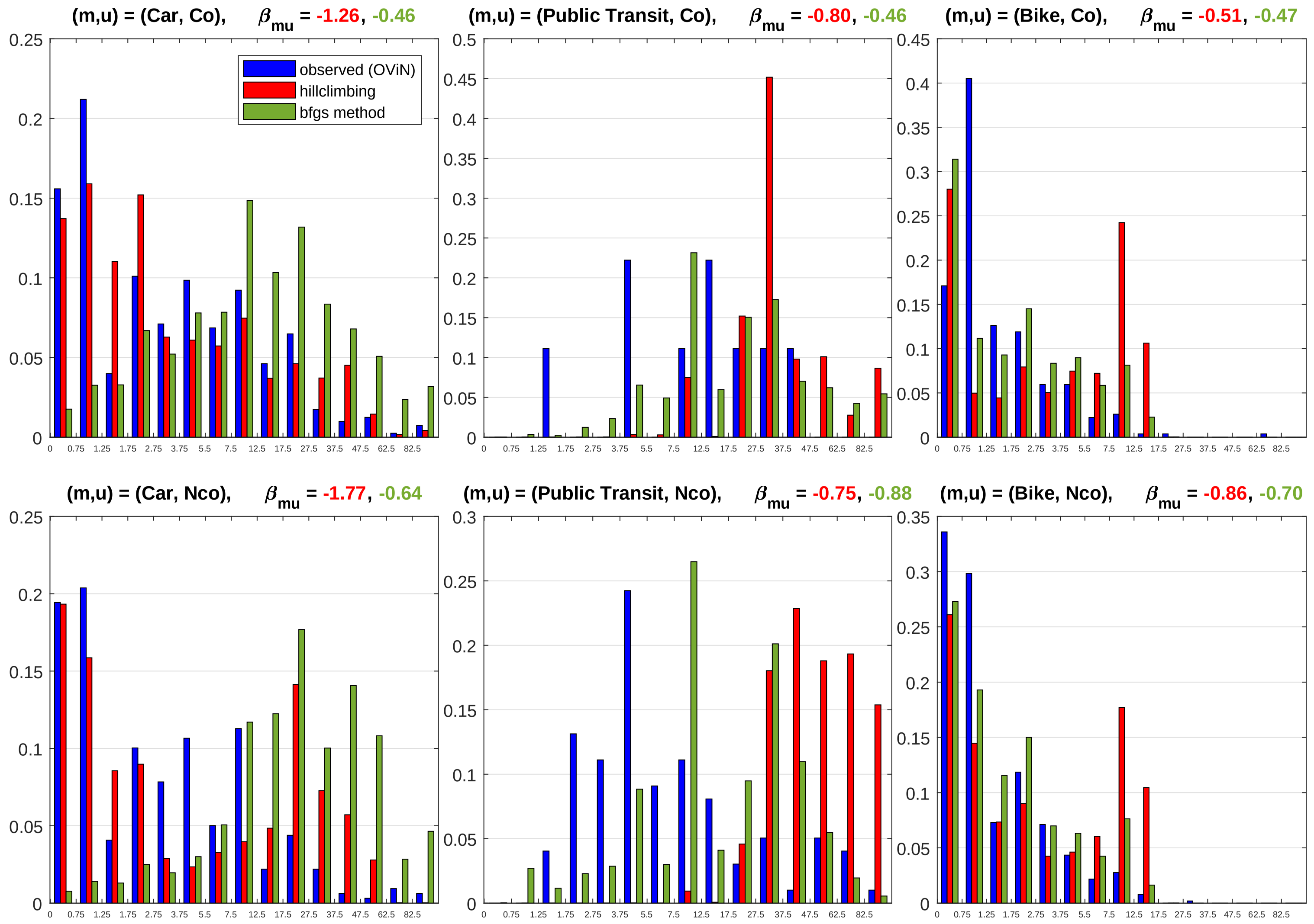


Figure 19: Bad solution for hillclimbing for the Other purpose of the MRDH model

Final test on the MRDH model: calibrating the remaining purposes

In the final test on the MRDH model we calibrated also the Work, Shops and Business purposes. from the starting solution $\boldsymbol{\beta}_{mu} = -0.5$ for each of the 6 mode-user class combinations (m, u) using hillclimbing and the BFGS method. As the running time for calibrating the MRDH model is so much higher than for the BBMA model, it was unpractical to test multiple initial solution for each purpose. This initial guess of $\boldsymbol{\beta}_{mu} = -0.5$ was picked for each purpose as it is the midpoint of the $[-1, 0]^6$ cube in which we expect the minimum to be in.

We used a weighted objective function with weights based on the confidence interval widths as defined in 3.3.2: The original trip distribution numbers d_{muk} and $\widehat{d}_{muk}$ are first divided by their binwidth and subsequently normalized so that the resulting distributions sum to 1 for each mode-user class combinations (m, u). Denote by d_{muk}^{rel} and $\widehat{d}_{muk}^{rel}$ these modelled and observed resulting distributions respectively. Then with the relative error r_k calculated by (13) the objective function is given by:

$$F(\boldsymbol{\beta}) = \sum_{m,u,k} \left(\frac{d_{muk}^{rel} - \widehat{d}_{muk}^{rel}}{r_k \cdot \widehat{d}_{muk}^{rel}} \right)^2$$

For the construction of the confidence intervals we used survey count data from OViN (dutch: 'Onderzoek Verplaatsing in Nederland')[1]. Runs for the Education purpose were not done in this final test, as OViN counts were missing for it. For both hillclimbing and the BFGS method the same parameter values and general solution method setup were used as in the earlier test runs on the Other purpose of MRDH.

Results of the final test on the MRDH model

It was observed that hillclimbing converged to the same solution as the BFGS method for all purposes except again for the Other purpose for which hillclimbing converged to a suboptimal solution. Figure 20 compares the solutions of the methods for this test run. Table 5 shows the running time of BFGS for each of the purposes calibrated. Also the number of line search steps[5], the total number of extra gravity runs required i.e. the total number of times the step size is reduced in the line search steps and the total number of gravity runs done. Assuming the uncalibrated Education purpose would take 10 hours which is the mean of the running times of the calibrated purposes. The total time of calibrating the MRDH strategic traffic model using the BFGS method would take approximately 60 hours. Therefore the requirement of calibrating the MRDH model in 3 days discussed in the performance criteria section 4.4 (criterium (iv): *Running time*) is well met. Hillclimbing however has a much shorter running time, approximately $\frac{1}{6}$ of BFGS method's running time. The test results overall suggest that the BFGS method although much slower in convergence is more reliable in terms of finding a quality solution. We also checked for this test and the BBMA tests whether the bad solutions of hillclimbing are not just suboptimal local minima by calculating the gradient norm. These norms were observed to be relatively large excluding this possibility.

[5] By a line search step we mean the computation of the gradient followed by computing a step size that satisfies the sufficient decrease condition.

Purpose	time (hours)	# of line-	# of extra gravity	total # of
		search iterations	runs in line search	gravity runs
Work:	6.8	15	10	115
Business:	16	20	2	142
Stores:	20.9	31	2	219
Other:	7.3	17	1	120

Table 5: Running time of BFGS method and number of line search steps and gravity runs needed for the four purposes of the MRDH model that were calibrated.

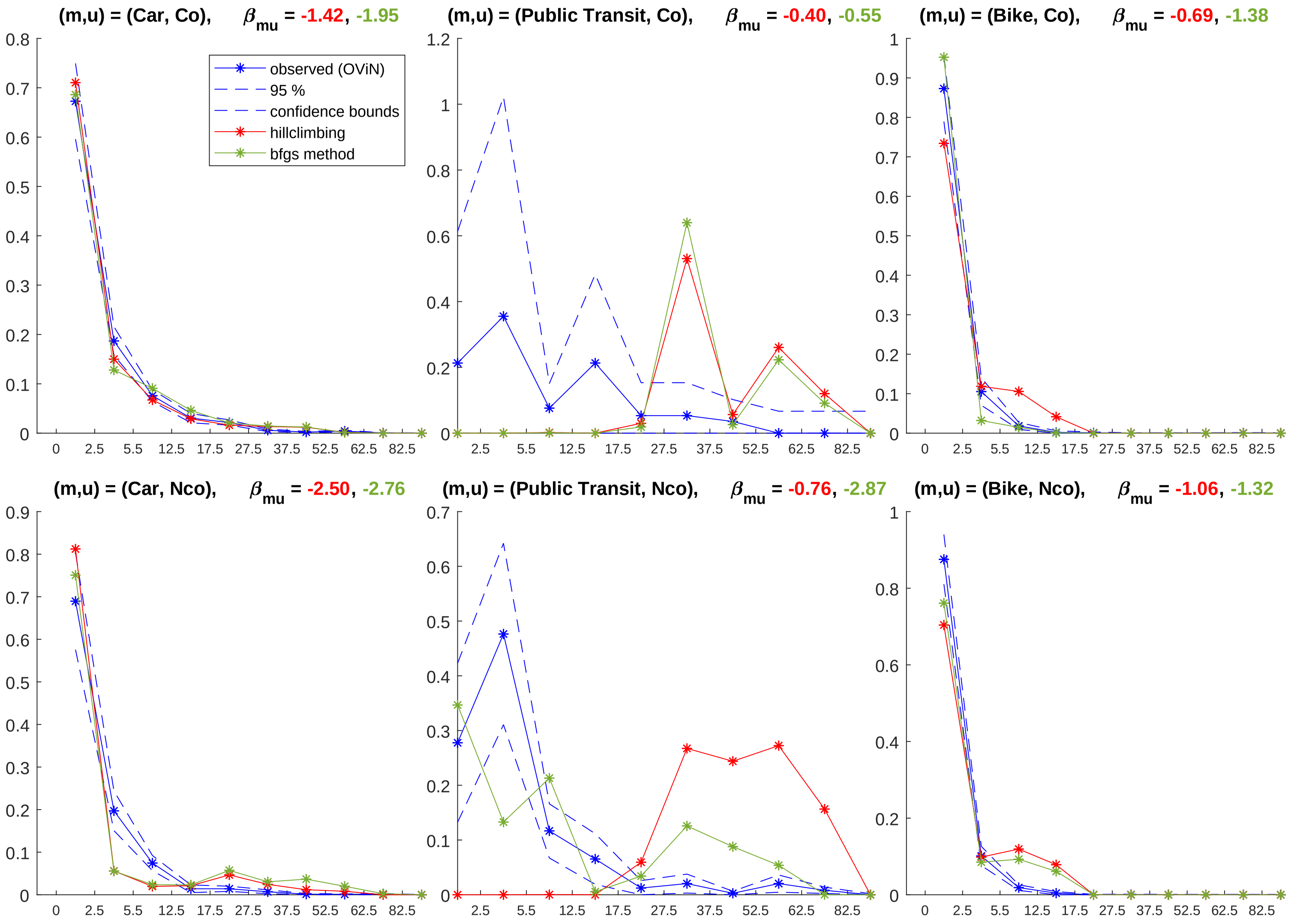


Figure 20: Comparing the solutions for the Other purpose of the MRDH model

8 Conclusions and recommendations

The first recommendation following from this thesis is to use a triply constrained gravity model in the calibration. Advantages are that the $\boldsymbol{\alpha}$ parameters become a function of the $\boldsymbol{\beta}$ parameters choice simplifying the calibration process and reducing its running time significantly by about 80% independent of the actual $\boldsymbol{\beta}$ calibration algorithm used. Also the modal split constraints are met exactly and the original biproportional gravity model structure is still retained.

The results from the calibration test runs done on the BBMA and MRDH strategic traffic model show that the BFGS method is more reliable in finding a quality solution than the hillclimbing algorithm that is currently used. The method is approximately 6 times slower approximately canceling the running time reduction from the new triproportional fitting approach. Though slow it would already fit the time budget of 3 days for calibrating the entire largest strategic traffic model of MRDH. On the other hand only in about 5% of the time hillclimbing finds a worse solution than the BFGS method. In the other cases hillclimbing works just fine and would be prefered. It would therefore be interesting in further research to investigate whether the methods and their benefits, speed and reliability, can be combined. Hillclimbing could be used initially and hopefully converge to the optimal solution quickly. Whether the found solution is optimal can be checked by calculating the gradient in the converged solution. Then a switch can be made to the BFGS method in case the solution is not locally optimal.

Another area of study for further research could be with the analytical gradient method. It was observed to be significantly faster than using finite differences up to model sizes of 3300 zones. It might be possible to speed up the analytical method even further by considering alternative methods to solve the linear system. Right now it is however not ready to be used as an alternative to finite differences as occasionally the matrix of the linear system becomes singular.

References

[1] https://www.cbs.nl/nl-nl/onze-diensten/methoden/onderzoeksomschrijvingen/korte-onderzoeksbeschrijvingen/onderzoek-verplaatsingen-in-nederland--ovin--.

[2] Mounir Mahmoud Moghazy Abdel-Aal. "Calibrating a trip distribution gravity model stratified by the trip purposes for the city of Alexandria". In: *Alexandria Engineering Journal* 53.3 (2014), pp. 677–689.

[3] J. Brethouwer. *The multi-constrained gravity model, And how to solve it using multi-proportional fitting.* Report of internship at DAT.Mobility. 2018.

[4] Angelo Corana et al. "Minimizing multimodal functions of continuous variables with the simulated annealing algorithm". In: *ACM Transactions on Mathematical Software (TOMS)* 13.3 (1987), pp. 262–280.

[5] J. de D. Ortúzar and L.G. Willumsen. *Modelling transport.* John Wiley & Sons, 1990.

[6] P.J.C Dickinson. Personal communication. April, 2018.

[7] R. Fransen. *Automated parameter calibration for a gravity model.* Report of internship at DAT.Mobility. 2015.

[8] Ubaldo M García-Palomares, Francisco J González-Castano, and Juan C Burguillo-Rial. "A combined global & local search (CGLS) approach to global optimization". In: *Journal of Global Optimization* 34.3 (2006), pp. 409–426.

[9] Dongmin Kim, Suvrit Sra, and Inderjit S Dhillon. "Tackling box-constrained optimization via a new projected quasi-newton approach". In: *SIAM Journal on Scientific Computing* 32.6 (2010), pp. 3548–3563.

[10] R. R. A. Martins. *Sensitivity Analysis AA222 - Multidisciplinary Design Optimization.* aero-comlab.stanford.edu/jmartins/aa222/aa222sa.pdf. Accessed: 07-09-2018.

[11] Friedrich Pukelsheim and Bruno Simeone. *On the iterative proportional fitting procedure: Structure of accumulation points and L1-error analysis.* https://opus.bibliothek.uni-augsburg.de/opus4/frontdoor/deliver/index/docId/1229/file/mpreprint_09_005.pdf. Accessed: 07-09-2018. 2009.

[12] Uriel G Rothblum and Hans Schneider. "Scalings of matrices which have prespecified row sums and column sums via optimization". In: *Linear Algebra and its Applications* 114 (1989), pp. 737–764.

[13] James C Spall. "Multivariate stochastic approximation using a simultaneous perturbation gradient approximation". In: *IEEE transactions on automatic control* 37.3 (1992), pp. 332–341.

[14] Sean Wallis. "Binomial confidence intervals and contingency tests: mathematical fundamentals and the evaluation of alternative methods". In: *Journal of Quantitative Linguistics* 20.3 (2013), pp. 178–208.

[15] Alan Geoffrey Wilson. "The use of entropy maximising models, in the theory of trip distribution, mode split and route split". In: *Journal of Transport Economics and Policy* (1969), pp. 108–126.

[16] Stephen Wright and Jorge Nocedal. *Numerical optimization.* Vol. 35. 67-68. Springer, 1999.