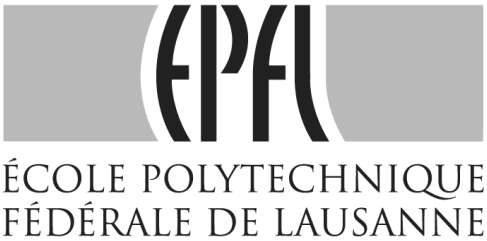
Cadyts – a free calibration tool for dynamic traffic simulations

Gunnar Flötteröd

STRC 2009 **September 2009**



Cadyts – a free calibration tool for dynamic traffic simulations

Gunnar Flötteröd
Transport and Mobility Laboratory
Ecole Polytechnique Fédérale de Lausanne
CH-1015 Lausanne
phone: +41-21-693.24.29
fax: +41-21-693.80.60
gunnar.floetteroed@epfl.ch

September 2009

Abstract

This article reports on the realization and on first applications of the Cadyts ("Calibration of dynamic traffic simulations") calibration tool. The presented first version of Cadyts calibrates disaggregate demand models of dynamic traffic assignment simulators from traffic counts. The tool is broadly applicable in that it (i) makes only very mild assumptions about the calibrated simulator's workings and (ii) allows for various modes of technical interaction with the simulation software. The article provides a both conceptual and technical overview of the tool and exemplary demonstrates its applicability to two different traffic microsimulators.

Keywords

calibration, dynamic traffic assignment, traffic counts

1 Introduction

Iterated microsimulations have become a prominent solution procedure for the dynamic traffic assignment (DTA) problem, and a wide variety of free, e.g., MATSim (accessed 2009); SUMO (accessed 2009), not-entirely-free, e.g., DRACULA (accessed 2009); DynaMIT (accessed 2009), and commercial, e.g., TSS Transport Simulation Systems (accessed 2009); INRO (accessed 2009); Quadstone Paramics Ltd. (accessed 2009), simulation software packages has become available in the last decades. Arguably, this success is to a large extent due to the intuitive workings of microsimulations when compared to mathematically more involved analytical DTA solution approaches (Peeta and Ziliaskopoulos, 2001).

However, this advantage comes with drawbacks: (i) Transport microsimulation also requires behavioral modeling at the individual level, which is a data hungry and methodologically challenging problem (Bowman and Ben-Akiva, 1998; Vovsha *et al.*, 2004), and (ii) intuitive analogies alone are insufficient to explain the dynamics of iterated DTA microsimulations (Cascetta, 1989; Nagel *et al.*, 1998). These difficulties have limited the development of mathematically consistent tools for the calibration of DTA demand microsimulations from traffic counts to approaches that at some point in the process aggregate the demand into macroscopic quantities, e.g., Ashok (1996); Antoniou (2004); Zhou (2004), which on the one hand improves the mathematical tractability but on the other hand discards much of the disaggregate information available in the simulation.

This article reports on the development of the freely available Cadyts ("Calibration of dynamic traffic simulations") calibration tool, which aims to overcome these difficulties (Flötteröd, 2008). Cadyts is compatible with a broad class of DTA microsimulators. It calibrates the disaggregate demand in the simulation from readily available sensor data such as traffic counts. While the focus of this text is on the calibration software and its applicability, the presentation also provides a basic theoretical development of the method. Further material, including a software manual, the sources of the calibration tool, and executable code can be found on the Cadyts web site (Cadyts, accessed 2009).

The remainder of this article is organized as follows. Section 2 outlines the scope of the calibration. Section 3 provides some background on the underlying methodology. Section 4 gives two examples of how the calibration can be linked to an existing DTA microsimulator, either through function calls or file exchanges. Finally, Section 5 concludes the article and gives an outlook on future developments.

2 Scope of the calibration

Informally, the DTA problem is to attain consistency between a dynamic model of travel demand and a dynamic model of network supply (traffic flow dynamics). In a fully disaggregate DTA microsimulator, every traveler is modeled as an individual agent. The travel intentions of an agent are represented by its plan, which typically comprises a sequence of trips that connect intermediate stops during which activities are conducted, including all associated timing information. This terminology comprises trip-based microsimulations when considering every single trip as an independent plan of an independent agent.

The Cadyts tool is designed to interact with a stochastic and iterative DTA microsimulator. Stochastic means that at least the agent behavior (i.e., the plan choice) is non-deterministic. Iterative means that the simulator runs according to the following logic:

1. Initialization.
2. Iterations: Repeat the following until stationary conditions are reached.
   (a) Demand simulation: All agents select new plans based on the network conditions of previous iterations.
   (b) Supply simulation: The plans of all agents are simultaneously executed in the network.

This logic is equally applicable to simulate an equilibrium-based planning model and a telematics model where drivers are spontaneous and imperfectly informed. From a simulation point of view, the only difference between these models is that an equilibrium demand simulator typically utilizes all information from the most recent supply simulations, whereas a telematics demand simulator generates every elementary decision of a plan only based on such information that could have actually been gathered up to the according point in simulated time (Bottom, 2000). In either case, the purpose of the iterations is to obtain consistency between the demand and the supply.

The calibration adjusts the plan choice probabilities of all agents such that they result in simulated network conditions that are consistent with the traffic counts. Note that the choice of a plan includes the choice of a complete set of departure times for all trips contained in the plan. In order to technically apply the calibration tool to a microsimulator, the following additional operations are needed:

1. Initialization. When the calibration is started, it needs to be provided with all available traffic counts and some further parameters.
2. Iterations. The calibration is run jointly with the simulation until (calibrated) stationary conditions are reached.

(a) Demand simulation: The calibration needs an access point in the simulation in order to affect the plan choice. There are various ways to realize this, depending on the concrete simulator.
(b) Supply simulation: The calibration needs to observe the simulated network conditions in order to evaluate their deviation from the traffic counts.

How these tasks are realized in detail depends on the simulation system at hand. Section 4 gives two examples.

3 Some background

First, a mathematical formulation of the calibration problem is given in Section 3.1. Second, some implications and the current implementation of this formulation are outlined in Section 3.2.

3.1 What problem does Cadyts solve?

To begin with, the familiar problem of estimating path flows (i.e., trips) between a set of N origin/destination (OD) pairs from traffic counts is considered (Bell *et al.*, 1996, 1997; Nie and Lee, 2002; Nie *et al.*, 2005). The largest possible number of trips between OD pair n is denoted by d_n, the symbol C_n represents the set of available routes that connect OD pair n, and d_{ni} is the number of trips on route $i \in C_n$, where $d_n = \sum_{i \in C_n} d_{ni}$. Variations in the total demand levels can be enabled by adding one fictitious route to every OD pair that bypasses the physical network.

The probability that a traveler in OD relation n chooses path i is denoted by $P_n(i|\mathbf{d})$ where $\mathbf{d} = (d_{ni})$. This probability is in general a function of all demand levels $\mathbf{d}$ because in equilibrium conditions the route choice of a traveler depends on the network conditions, which in turn depend on the route choice of all travelers in the system. Mathematically, the problem of finding path demand levels that are self-consistent in this regards can be expressed as the problem of maximizing the prior entropy

$$W(\mathbf{d}) = \prod_{n=1}^{N} d_n! \frac{\prod_{i \in C_n} (P_n(i|\mathbf{d}))^{d_{ni}}}{\prod_{i \in C_n} d_{ni}!}, \tag{1}$$

which represents the probability that, for a given route choice model $P_n(i|\mathbf{d})$, a particular demand pattern $\mathbf{d}$ occurs in the system. Appendix A shows that the demand levels that maximize $W(\mathbf{d})$ solve the route assignment problem $d_{ni} = P_n(i|\mathbf{d})d_n$ for all n and $i \in C_n$.

Given a set $\mathbf{y}$ of traffic counts that is observed on some or all links of the network, the calibration adjusts all path flows in a way such that the counts are reproduced to a reasonable degree. For this purpose, the posterior entropy

$$W(\mathbf{d}|\mathbf{y}) \propto p(\mathbf{y}|\mathbf{d})W(\mathbf{d}) \tag{2}$$

is maximized, where the likelihood $p(\mathbf{y}|\mathbf{d})$ is the probability of observing the measurements $\mathbf{y}$ for given demand levels $\mathbf{d}$. The posterior entropy models the probability distribution of a certain demand pattern $\mathbf{d}$ given both the route choice model $P_n(i|\mathbf{d})$ and the measurements $\mathbf{y}$. Appendix B shows that $W(\mathbf{d}|\mathbf{y})$ is maximized by the posterior route choice probabilities

$$P_n(i|\mathbf{d},\mathbf{y}) = \frac{\exp(\Lambda_{ni}+\Gamma_{ni})P_n(i|\mathbf{d})}{\sum_{j\in C_n}\exp(\Lambda_{nj}+\Gamma_{nj})P_n(j|\mathbf{d})} \tag{3}$$

where

$$\Lambda_{ni} = \frac{\partial \ln p(\mathbf{y}|\mathbf{d})}{\partial d_{ni}} \tag{4}$$

$$\Gamma_{ni} = \sum_{m=1}^{N}\sum_{j\in C_m}\frac{d_{mj}}{P_m(j|\mathbf{d})}\frac{\partial P_m(j|\mathbf{d})}{\partial d_{ni}}. \tag{5}$$

That is, a demand calibration in the maximum posterior entropy sense requires to scale the choice probability of every route i of every OD pair n by $\exp(\Lambda_{ni}+\Gamma_{ni})$ (and to re-normalize). Λ_{ni} captures the effect of the demand d_{ni} on the log-likelihood, i.e., on the measurement reproduction. Γ_{ni} essentially describes how a change in d_{ni} affects all demand levels $\mathbf{d}$ (through the network conditions).

Formally, this estimator can be immediately applied to calibrate the plan choice of a disaggregate agent population in a fully dynamic setting by associating the index $n = 1 \ldots N$ with the agents and C_n with the plan choice set of agent n. That is, the OD pairs are now replaced by agents and the routes are replaced by plans.

Consistency with the large population assumption of the entropy maximization approach is maintained by considering now a large number of R iterations in the simulation. This implies that $d_n = R$ becomes the total number of plan choices made by agent n during R iterations and d_{ni} becomes the number of times agent n chooses plan $i \in C_n$. That is, $\mathbf{d}$ represents as from now the accumulated demand levels over R iterations. While this approach is intuitively fairly straightforward, it requires some further considerations, which are outlined in the following section.

3.2 How does Cadyts solve the problem?

The previous section outlines the conceptual workings of the calibration as an intuitive generalization of a mathematical specification. This section discusses the major conceptual and algorithmic implications of this approach.

The idea of re-establishing the large population property by observing the same agent during many iterations of the simulation implies two assumptions: (i) The demand model $P_n(i|\mathbf{d})$ represents a stable average plan choice distribution of agent n over these iterations, and (ii) this choice distribution is a function only of the average demand levels. Assumption (i) is consistent with the inertia of most (if not all) DTA simulators presented in the literature, which mirrors the inertia of actual travelers' decision making. Assumption (ii) implies an approximation: A real traveler might very well evaluate distributional information about the network conditions when selecting a plan. Average demand levels constitute only an imperfect proxy for this information. Note, however, that these are only internal assumptions of the calibration and that the simulation is not required to fully comply with them for a practical application.

An iterated microsimulation typically maintains some variability in the network conditions even when the transients of the iterations have ceased. This variability results from the stochasticity of both the supply and the demand simulator. For reasons of numerical stability, the Λ coefficients in (4) are calculated as average values in calibrated conditions. This constitutes a fixed point problem in that the Λ coefficients depend on the demand levels, which in turn are affected by the calibration and hence by the Λ coefficients. Technically, this is realized by running a recursive regression concurrently with the simulation that tracks a linear model of the log-likelihood given the demand levels (Flötteröd and Bierlaire, 2009, accepted for presentation). The coefficients of this model serve as approximations of the derivatives in (4). Since the linear model is updated in every single iteration, it implicitly captures all correlations in the network conditions that result from variability in the demand. This is relevant because otherwise this correlation would have to be additionally modeled in the likelihood, which is originally specified in (2) to depend on a single demand realization $\mathbf{d}$ only.

The Γ coefficients in (5) require to calculate the sensitivities of all plan choice probabilities with respect to all demand levels, where the coupling of these quantities is given through the simulated network conditions. These sensitivities are hard to obtain for generic demand and supply simulators, and therefore these coefficients are currently set to zero in the calibration. Essentially, this simplification is as good (or as bad) as a "proportional assignment" in OD matrix estimation, where fixed (i.e., insensitive) route choice behavior is assumed in every single iteration of the calibration.

Once the Λ and Γ coefficients are available, the modified plan choice distribution (3) can be enforced in various ways, depending on the simulation. If the simulator provides access to the

uncalibrated choice probabilities, these probabilities can be explicitly adjusted before a choice is made. If the demand model is utility-driven, the utilities of the plans can be modified such that the desired posterior choice distribution results. If the demand simulation operates completely as a black box in that only realized choices i of any agent n can be observed, rejection sampling is applicable with an acceptance probability that is proportional to $\exp(\Gamma_{in} + \Lambda_{in})$ (Ross, 2006).

4 Applications

This section presents two applications of Cadyts. First, some technical preliminaries are given in Subsection 4.1. Second, Subsection 4.2 describes how the calibration is linked through function calls with the MATSim microsimulator and presents an exemplary result from a large real-world case study. Third, Subsection 4.3 describes an application of the calibration in conjunction with the SUMO microsimulator that relies on a file-based communication between the programs. For the latter application, only some very preliminary results can be given.

4.1 Technical decoupling of calibration and simulation

Cadyts is implemented in Java (Sun Microsystems, accessed 2009). Java structures software into packages, which essentially are name spaces that coincide with directories of the file system. In order to customize the calibration for a particular simulation, some Java programming is likely to become necessary, e.g., to account for the simulation-specific data and file formats. However, the amount of programming is minimized through the provision of various default classes for different interaction modes.

A major issue with the maintenance of software that is used by different groups is to ensure stable interfaces to the users while enabling sufficient internal flexibility for future developments. Cadyts deals with this issue by assuming that a separate interface package is set up for every linked simulator. This package (i) accesses and possibly extends the pre-fabricated facilities of the basic calibration code and (ii) constitutes an exclusive connection point between calibration and simulation. Given that this package is added to the Cadyts software repository, all internal modifications ("refactorings", Fowler (1999)) of the calibration code can be applied consistently to the interface package, while the logic according to which this package connects to the simulation remains unchanged. In the language of design patterns, the interface package implements a "facade" (Gamma *et al.*, 1994).

4.2 Calibration of MATSim

MATSim ("Multi-Agent Transport Simulation Toolkit") is a truly disaggregate DTA microsimulator in that it entirely discards the fairly typical OD matrix-based demand representation and instead tracks the trip sequences of individual agents throughout the the entire modeling process (MATSim, accessed 2009; Raney and Nagel, 2006). This feature allows to model how the network conditions affect not only route choice but, at least in principle, arbitrary choice dimensions. The current implementation of MATSim equilibrates both route and departure time choice based on an all-day utility function that accounts for the cost of travel and the benefits of performing activities (Charypar and Nagel, 2005). A simple queuing model is implemented in the supply simulation (Cetin *et al.*, 2003).

Since both Cadyts and MATSim are implemented in Java, they can be linked through function calls. For this purpose, a (pre-fabricated and not MATSim-specific) rejection sampling facility of Cadyts is utilized: Whenever an agent chooses a plan, it proposes this plan to the calibration, which either accepts or rejects the plan. If the plan is rejected, the agent draws again from its plan choice distribution. Eventually, an accept occurs, which constitutes a draw from the calibrated plan choice distribution. Essentially, the following three functions are called by MATSim:

`void addMeasurement(L link, int start_s, int end_s, double value, double stddev, Measurement.TYPE type)`

This function is called once for every measurement before the simulation starts. It registers a measurement of a certain `type` (currently, there are only traffic counts), which has been observed on a certain `link` between `start_s` and `end_s` seconds of the day. The measured `value` and its standard deviation `stddev` are also provided. The generic network link type `L` is in internally substituted by the MATSim link type.

`boolean getSampler(Object agent).isAccepted(Plan<L> plan)`

Whenever an `agent` chooses a `plan`, it asks the calibration through this function if the plan is accepted or if another plan needs to be generated. The calibration guarantees that an accept occurs after a pre-specified maximum number of rejections while at the same time making a best effort to comply as far as possible with the calibrated choice distribution (3). In every iteration, every agent makes as many function calls of this type as it is necessary to obtain an accepted plan.

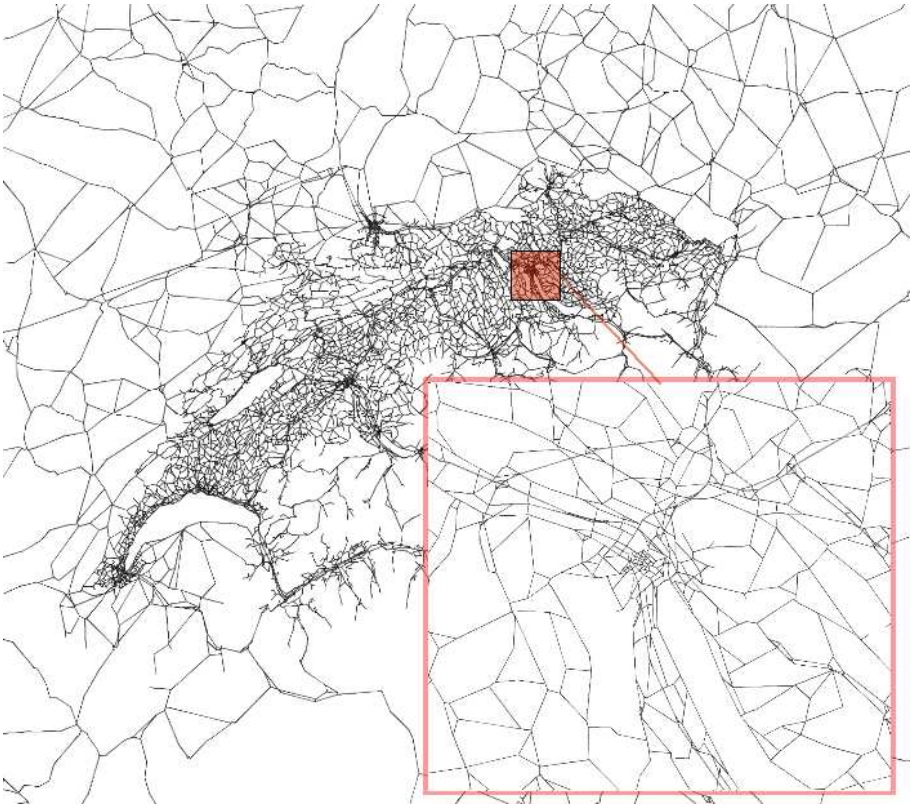


Figure 1: Zurich network
The analysis zone of the MATSim test case comprises the major road network of the city of Zurich, which is enlarged (Grether *et al.*, 2008).

`void afterNetworkLoading(SimResults<L> simResults)`

This function is called once after each network loading. It passes a container object to the calibration that provides information about the results of the most recent network loading, in particular about the simulated flows at the measurement locations. Although a pre-fabricated implementation of the `SimResults<L>` interface is available, MATSim uses a proprietary implementation for greater efficiency.

MATSim has by now been successfully calibrated in one large real-world scenario, where both route and departure time choice are concurrently adjusted from time-dependent traffic counts at 159 sensor locations for a population of 187 484 agents on a 60 492 link network (Flötteröd *et al.*, 2009, accepted for presentation). For illustration, Figure 1 shows the network of this application, and Figure 2 exemplifies how the log-likelihood of the sensor data changes over the iterations. The calibrated simulation stabilizes after a few hundred iterations, which is in the same order of magnitude as a plain simulation that does not account for the sensor data. Also, the duration of a single iteration is less than doubled by the calibration's computational overhead that mainly results from the rejection sampling.

The experiment starts from equilibrated network conditions such that all improvements in the

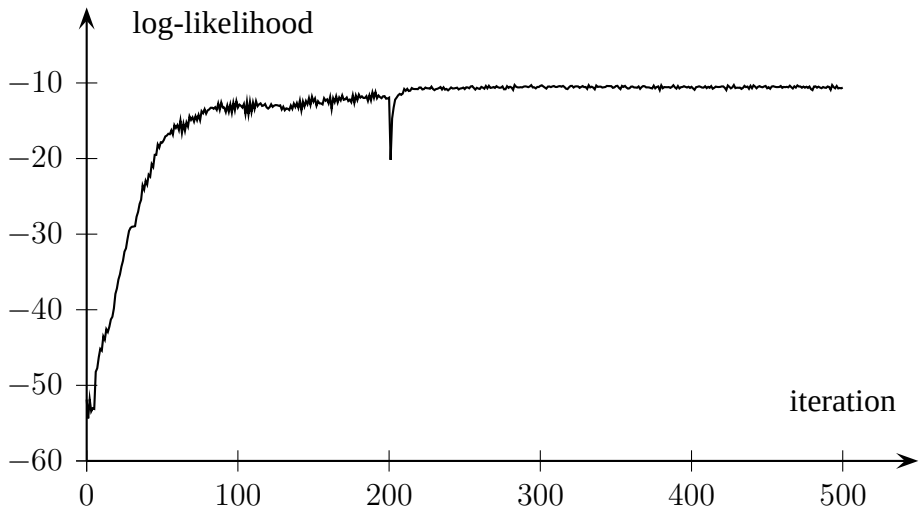


Figure 2: MATSim results
Evolution of the log-likelihood (scaled by the number of measurements) over the iterations. The calibration starts from equilibrated network conditions such that all improvements in the log-likelihood can be assigned to the calibration. The jump-like stabilization after iteration 200 results from a switch in MATSim's replanning logic, which excludes certain choice dimensions in order to improve the simulation's convergence.

log-likelihood can be assigned to the calibration. The initial log-likelihood (normalized by the number of measurements) is -51.9 and its final value fluctuates between -11 and -10. It would become zero if all measurements were perfectly reproduced. This shows that Cadyts improves the measurement fit substantially, but it does not say anything about the extrapolation quality beyond the sensor locations or about how the calibration can be used to identify the parameters of the plan choice model. These items are discussed in Flötteröd *et al.* (2009, accepted for presentation).

4.3 Calibration of SUMO

SUMO ("Simulation of Urban Mobility") is a trip-based DTA microsimulator (SUMO, accessed 2009). It takes time-dependent OD matrices as inputs and disaggregates them into individual vehicles before evaluating the network performance through a detailed traffic flow microsimulator. The iterative feedback loop only adjusts route choice. Both the demand simulator and the supply simulator of SUMO are implemented in C++. The iterative simulation is enabled through a Python script that alternately call each model component. The data exchange among the modules is realized through files.

Cadyts provides extensive facilities for the file-based interaction with a simulation. These classes are configured with implementations of some interfaces for the reading and writing of the SUMO file formats. An executable jar file is generated from this code, and the follow-

ing three calls to the jar file are implemented in SUMO's Python script (some command line parameters are left out for clarity):

```
java -jar SumoController.jar INIT -measfile meas.xml
```

Everything before the `INIT` keyword is just a call to the pre-compiled java program. `INIT` indicates that this call initializes the calibration. The `-measfile` keyword is followed by the file in which the traffic counts are stored. This call is made once when the simulation is started.

```
java -jar SumoController.jar CHOICE -choicesetfile
choicesets.xml -choicefile choices.xml
```

The `CHOICE` keyword indicates that the calibration is expected to generate calibrated choices for every trip maker in the simulation and to write these choices in the file preceded by the `-choicefile` keyword. For this, the calibration is provided with both the choice sets and the prior choice probabilities of all trip makers in the file following the `-choicesetfile` keyword. This call is made once in every iteration of the simulation.

```
java -jar SumoController.jar UPDATE -netfile flows.xml
```

The `UPDATE` keyword tells the calibration that new simulated network conditions from the most recent run of the traffic flow simulation are available in the file following the `-netfile` keyword. This call is made once in every iteration of the simulation.

Being limited to route choice based on OD matrices, SUMO can also be calibrated only in these regards. The basic application of a mere route choice adjustment given fixed total demand levels has by now successfully been tested. The joint calibration of OD flows and route choice is currently being implemented, where a fixed maximum demand level per OD pair is assumed and every trip-makers is provided with one additional routing alternative that (i) represents the decision of not making a trip at all and (ii) has a prior choice probability that guarantees that the number of a priori made trips is consistent with the uncalibrated OD matrix.

The calibration of SUMO is a recent venture, and only very preliminary results are available. Figure 3 shows the simple test network that is used to validate the technical correctness of the interactions between SUMO and Cadyts. Vehicles enter the network at D1 or D2, choose one of the 4 possible routes through the network, and leave it at D3. Figure 4 gives some exemplary log-likelihood trajectories that result when a single flow sensor is located on link L9 and the standard deviation of the according measurement is varied. As the standard deviation decreases, the measurement fit improves. If nothing else, this shows that simulation

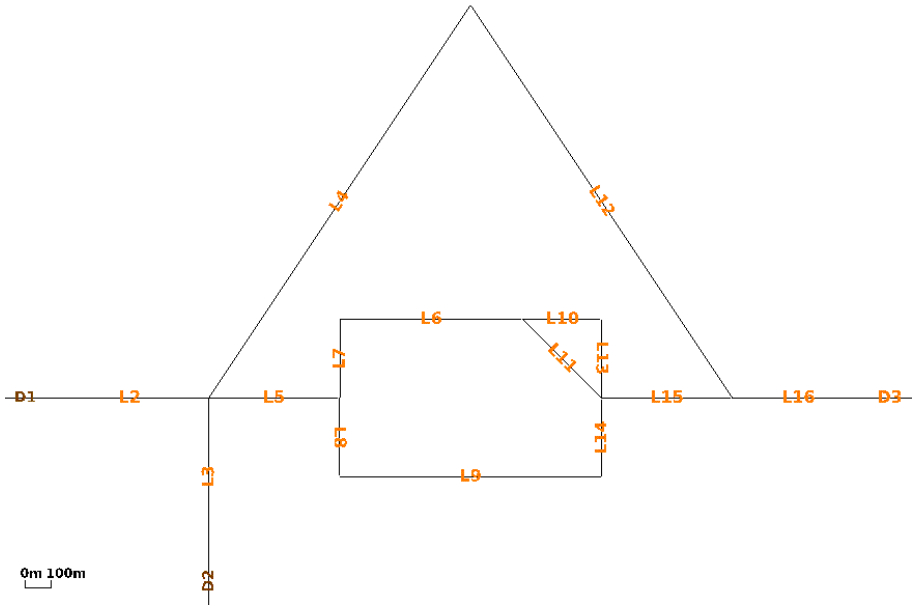


Figure 3: SUMO network
A synthetic network for testing purposes. Vehicles enter the system at D1 and D2, and they leave at D3.

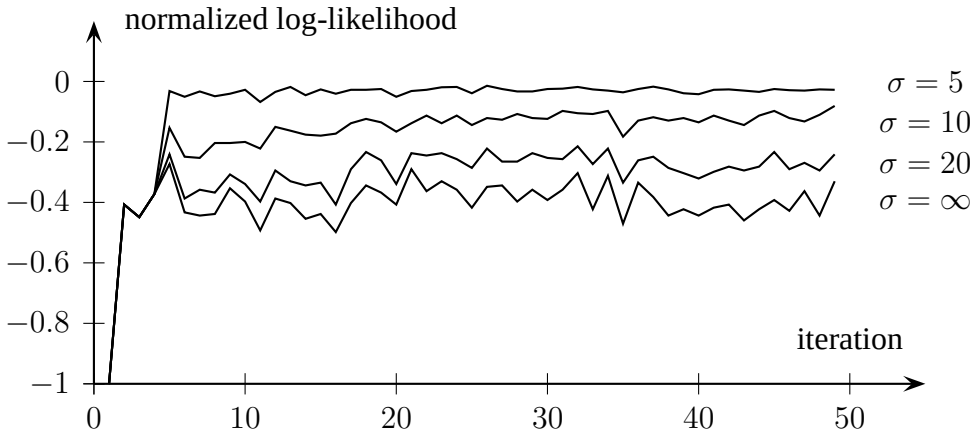


Figure 4: SUMO results
Some exemplary log-likelihood curves for a single sensor on link L9 of the network shown in Figure 3. The higher the belief in the measurement (lower standard deviation σ, in vehicles per hour), the better its reproduction: A zero log-likelihood indicates a perfect reproduction of the measurement. For better comparability, all curves are scaled to begin at the value minus one.

and calibration interact in a meaningful way. Computationally, the overhead introduced by the calibration is very low because SUMO communicates its route choice probabilities such that no rejection sampling is necessary.

5 Summary and outlook

This article demonstrates that it is possible to calibrate the demand of a DTA microsimulation from traffic counts without resorting to the usual aggregations in terms of OD matrices or path flows. The theoretical underpinnings of the proposed calibration approach are outlined, and its implementation in the freely available software package Cadyts is described. The system's flexibility is demonstrated through exemplary applications to two different DTA microsimulators.

Future work will cover both methodological and technical aspects of the calibration. Methodologically, a major and yet unresolved challenge is the mathematically consistent incorporation of the equilibrium-related interactions of different agents' plan choices (captured through the Γ coefficients in (5), which are set to zero in the current implementation). Beyond this, some means to calibrate a supply simulator jointly with the demand simulator would certainly improve the overall calibration quality. Technically, there is a vast number of thinkable add-ons that would improve the convenience of using the tool. Finally, new challenges are likely to be identified through new applications of Cadyts to further simulation systems.

6 Acknowledgments

For the calibration of MATSim, Yu Chen and Marcel Rieser did the programming on the simulation side. Yu also generated the presented results. For the calibration of SUMO, Michael Behrisch and Yun-Pang Wang did the programming on the simulation side. Yun-Pang also helped with the generation of the presented results.

A Maximization of prior entropy

Denote by d_n the total demand level of OD pair n and by d_{ni} the demand level of path $i \in C_n$. The prior entropy of the global demand pattern $\mathbf{d} = (d_{ni})$ is

$$W(\mathbf{d}) = \prod_{n=1}^{N} \left(\sum_{i \in C_n} d_{ni} \right)! \frac{\prod_{i \in C_n} (P_n(i|\mathbf{d}))^{d_{ni}}}{\prod_{i \in C_n} d_{ni}!}. \tag{6}$$

Taking the logarithm and applying Stirling's approximation ($\ln X! \rightarrow X \ln X - X$ for large X),

$$\begin{aligned} \ln W(\mathbf{d}) &= \sum_{n=1}^{N} \left[\sum_{i \in C_n} d_{ni} \cdot \ln \sum_{i \in C_n} d_{ni} \right. \\ &\qquad \left. + \sum_{i \in C_n} d_{ni} \ln P_n(i|\mathbf{d}) - \sum_{i \in C_n} d_{ni} \ln d_{ni} \right]. \end{aligned} \tag{7}$$

The derivative of $W(\mathbf{d})$ with respect to d_{mj} (where m is an OD pair and $j \in C_m$) is

$$\frac{\partial \ln W(\mathbf{d})}{\partial d_{mj}} = \ln d_m + \ln \frac{P_m(j|\mathbf{d})}{d_{mj}} + \sum_{n=1}^{N} \sum_{i \in C_n} \frac{d_{ni}}{P_n(i|\mathbf{d})} \frac{\partial P_n(i|\mathbf{d})}{\partial d_{mj}}. \tag{8}$$

A substitution of the equilibrium flow pattern $d_{ni} = P_n(i|\mathbf{d}) d_n$ yields

$$\frac{\partial \ln W(\mathbf{d})}{\partial d_{mj}} = \ln d_m - \ln d_m + \sum_{n=1}^{N} d_n \sum_{i \in C_n} \frac{\partial P_n(i|\mathbf{d})}{\partial d_{mj}} = 0, \tag{9}$$

where the sum over all choice probability derivatives of any demand segment n is zero because the probabilities themselves must sum up one.

B Maximization of posterior entropy

Before maximizing (the logarithm of) the posterior entropy function

$$W(\mathbf{d}|\mathbf{y}) \propto p(\mathbf{y}|\mathbf{d}) W(\mathbf{d}), \tag{10}$$

the additional requirement of constant demand levels d_n per OD pair n is introduced in the Lagrangian

$$L(\mathbf{d}|\mathbf{y}) = \ln W(\mathbf{d}|\mathbf{y}) + \sum_{n=1}^{N} u_n \left(\sum_{i \in C_n} d_{ni} - d_n \right). \tag{11}$$

Using (8), the derivative of $L(\mathbf{d}|\mathbf{y})$ with respect to d_{mj} (where m is an OD pair and $j \in C_m$) becomes

$$\frac{\partial L(\mathbf{d}|\mathbf{y})}{\partial d_{mj}} = \frac{\partial \ln p(\mathbf{y}|\mathbf{d})}{\partial d_{mj}} + \ln d_m + \ln \frac{P_m(j|\mathbf{d})}{d_{mj}} + \sum_{n=1}^{N} \sum_{i \in C_n} \frac{d_{ni}}{P_n(i|\mathbf{d})} \frac{\partial P_n(i|\mathbf{d})}{\partial d_{mj}} + u_m. \tag{12}$$

Setting this to zero and solving for d_{mj} yields

$$d_{mj} = d_m \exp(u_m) \exp(\Lambda_{mj} + \Gamma_{mj}) P_m(j|\mathbf{d}) \tag{13}$$

where

$$\Lambda_{mj} = \frac{\partial \ln p(\mathbf{y}|\mathbf{d})}{\partial d_{mj}} \tag{14}$$

$$\Gamma_{mj} = \sum_{n=1}^{N} \sum_{i \in C_s} \frac{d_{ni}}{P_n(i|\mathbf{d})} \frac{\partial P_n(i|\mathbf{d})}{\partial d_{mj}}. \tag{15}$$

The Lagrange multipliers result from a substitution of (13) in $d_m = \sum_{i \in C_m} d_{mi}$ such that

$$\exp(u_m) = \frac{1}{\sum_{i \in C_m} \exp(\Lambda_{mi} + \Gamma_{mi}) P_m(i|\mathbf{d})}. \tag{16}$$

Inserting this in (13) finally results in the posterior choice probabilities

$$P_m(j|\mathbf{d}, \mathbf{y}) = \frac{d_{mj}}{d_m} = \frac{\exp(\Lambda_{mj} + \Gamma_{mj}) P_m(j|\mathbf{d})}{\sum_{i \in C_m} \exp(\Lambda_{mi} + \Gamma_{mi}) P_m(i|\mathbf{d})}. \tag{17}$$

References

Antoniou, C. (2004) On-line calibration for dynamic traffic assignment, Ph.D. Thesis, Massachusetts Institute of Technology.

Ashok, K. (1996) Estimation and prediction of time-dependent origin-destination flows, Ph.D. Thesis, Massachusetts Institute of Technology.

Bell, M., W. Lam and Y. Iida (1996) A time-dependent multi-class path flow estimator, paper presented at *Proceedings of the 13th International Symposium on Transportation and Traffic Theory*, 173–193, Lyon, France, July 1996.

Bell, M., C. Shield, F. Busch and G. Kruse (1997) A stochastic user equilibrium path flow estimator, *Transportation Research Part C*, **5** (3/4) 197–210.

Bottom, J. (2000) Consistent anticipatory route guidance, Ph.D. Thesis, Massachusetts Institute of Technology.

Bowman, J. and M. Ben-Akiva (1998) Activity based travel demand model systems, in P. Marcotte and S. Nguyen (eds.) *Equilibrium and advanced transportation modelling*, 27–46, Kluwer.

Cadyts (accessed 2009) Cadyts web site, http://transp-or2.epfl.ch/cadyts.

Cascetta, E. (1989) A stochastic process approach to the analysis of temporal dynamics in transportation networks, *Transportation Research Part B*, **23** (1) 1–17.

Cetin, N., A. Burri and K. Nagel (2003) A large-scale agent-based traffic microsimulation based on queue model, paper presented at *Proceedings of the 3rd Swiss Transport Research Conference*, Monte Verita/Ascona, March 2003.

Charypar, D. and K. Nagel (2005) Generating complete all-day activity plans with genetic algorithms, *Transportation*, **32** (4) 369–397.

DRACULA (accessed 2009) DRACULA web site, http://www.its.leeds.ac.uk/software/dracula/index.html.

DynaMIT (accessed 2009) DynaMIT web site, http://web.mit.edu/its/dynamit.html.

Flötteröd, G. (2008) Traffic state estimation with multi-agent simulations, Ph.D. Thesis, Berlin Institute of Technology, Berlin, Germany.

Flötteröd, G. and M. Bierlaire (2009, accepted for presentation) Improved estimation of travel demand from traffic counts by a new linearization of the network loading map, paper presented at *Proceedings of the European Transport Conference*, The Netherlands, October 2009, accepted for presentation.

Flötteröd, G., Y. Chen, M. Rieser and K. Nagel (2009, accepted for presentation) Behavioral calibration of a large-scale travel behavior microsimulation, paper presented at *Proceedings of 12th International Conference on Travel Behaviour Research*, Jaipur, India, December 2009, accepted for presentation.

Fowler, M. (1999) *Refactoring. Improving the Design of Existing Code*, The Addison-Wesley Object Technology Series, Addison-Wesley.

Gamma, E., R. Helm, R. Johnson and J. Vlissides (1994) *Design Patterns*, Addison–Wesley Professional Computing Series, Addison–Wesley.

Grether, D., Y. Chen, M. Rieser, U. Beuck and K. Nagel (2008) Emergent effects in multi-agent simulations of road pricing, paper presented at *Proceedings of the Annual Meeting of the European Regional Science Association ERSA*.

INRO (accessed 2009) Dynameq web site, http://www.inro.ca/en/products/dynameq/.

MATSim (accessed 2009) MATSim web site, http://www.matsim.org.

Nagel, K., M. Rickert, P. Simon and M. Pieck (1998) The dynamics of iterated transportation simulations, paper presented at *Proceedings of the 3rd Triennial Symposium on Transportation Analysis*, San Juan, Puerto Rico.

Nie, Y. and D.-H. Lee (2002) An uncoupled method for the equilibrium-based linear path flow estimator for origin-destination trip matrices, *Transportation Research Record*, **1783**, 72–79.

Nie, Y., H. Zhang and W. Recker (2005) Inferring origin-destination trip matrices with a decoupled GLS path flow estimator, *Transportation Research Part B*, **39** (6) 497–518.

Peeta, S. and A. Ziliaskopoulos (2001) Foundations of dynamic traffic assignment: the past, the present and the future, *Networks and Spatial Economics*, **1** (3/4) 233–265.

Quadstone Paramics Ltd. (accessed 2009) Paramics web site, http://www.paramics-online.com.

Raney, B. and K. Nagel (2006) An improved framework for large-scale multi-agent simulations of travel behavior, in P. Rietveld, B. Jourquin and K. Westin (eds.) *Towards better performing European Transportation Systems*, 305–347, Routledge.

Ross, S. (2006) *Simulation*, fourth edn., Elsevier.

SUMO (accessed 2009) SUMO web site, http://sourceforge.net/projects/sumo.

Sun Microsystems (accessed 2009) Java web site, http://java.sun.com.

TSS Transport Simulation Systems (accessed 2009) AIMSUN web site, http://www.aimsun.com.

Vovsha, P., M. Bradley and J. Bowman (2004) Activity-based travel forecasting models in the United States: progress since 1995 and prospects for the future, paper presented at *Proceedings of the EIRASS Conference on Progress in Activity-Based Analysis*, Maastricht, The Netherlands, May 2004.

Zhou, X. (2004) Dynamic origin-destination demand estimation and prediction for off–line and on–line dynamic traffic assignment operation, Ph.D. Thesis, University of Maryland, College Park.