Oper Res Int J (2017) 17:145–164
DOI 10.1007/s12351-015-0219-4

ORIGINAL PAPER

Macroscopic traffic flow model calibration using different optimization algorithms

Anastasia Spiliopoulou[1] · Ioannis Papamichail[1] · Markos Papageorgiou[1] · Yannis Tyrinopoulos[2] · John Chrysoulakis[2]

Received: 2 February 2015 / Revised: 26 August 2015 / Accepted: 13 November 2015 /
Published online: 8 December 2015

Abstract This study tests and compares different optimization algorithms employed for the calibration of a macroscopic traffic flow model. In particular, the deterministic Nelder–Mead algorithm, a stochastic genetic algorithm and the stochastic cross-entropy method are utilized to estimate the parameter values of the METANET model for a particular freeway site, using real traffic data. The resulting models are validated using various traffic data sets and the optimization algorithms are evaluated and compared with respect to the accuracy of the produced validated models as well as the convergence speed and the required computation time. The validation results showed that all utilized optimization algorithms were able to converge to robust model parameter sets, albeit achieving different performances considering the convergence speed and the required computation time.

Keywords Macroscopic traffic flow models · Model calibration · Comparison of optimization algorithms

1 Introduction

During the last decades, several mathematical models for the road traffic flow have been proposed. Depending on the level of detail they use, the models are classified as macroscopic or microscopic (see Hoogendoorn and Bovy (2001) for an overview on traffic flow models). Traffic flow models may be employed for the planning of new, upgraded or modified road infrastructures; as well as for the development and

✉ Anastasia Spiliopoulou
natasa@dssl.tuc.gr

[1] Dynamic Systems and Simulation Laboratory, School of Production Engineering and Management, Technical University of Crete, 73100 Chania, Greece

[2] Department of Civil and Topography/Geoinformatics Engineering, Technological Educational Institute of Athens, 12210 Egaleo, Athens, Greece

testing of traffic flow estimation algorithms, traffic control strategies and other operational tools (Kotsialos and Papageorgiou 2000). The models include a number of physical or non-physical parameters, with unknown values, which should be appropriately specified, in case of real applications, so as to reproduce the network and traffic flow characteristics with the highest possible accuracy. The macroscopic traffic flow models include a lower number of parameters compared to microscopic models; also, they have an analytical form, which allows their usage for various significant traffic engineering tasks (estimation, control strategy design) beyond simulation. Before employing a traffic flow model in practice, it is important to first calibrate it against real traffic data. The calibration procedure aims to appropriately specify the model parameter values, so that the representation of the network and traffic flow characteristics is as accurate as the model structure allows. The most common approach is to minimize the discrepancy between the model's estimation and the real traffic data, by use of appropriate optimization tools (see Kontorinaki et al. (2014) and Treiber and Kesting (2013) for related overviews).

Within the, quite limited, literature on macroscopic traffic flow model calibration, various methods have been employed to solve the parameter estimation problem. In particular, Grewal and Payne (1976) utilized the least-squares method and an extension of the Kalman Filter; Michalopoulos et al. (1993) and Helbing (1996) have applied some trial-and-error methods; Cremer and Papageorgiou (1981), Cremer and May (1986), Papageorgiou et al. (1989, 1990), Sanwal et al. (1996), Kotsialos et al. (2002) and Monamy et al. (2012) have used the deterministic Complex algorithm of Box (1965); Ngoduy et al. (2004) and Spiliopoulou et al. (2014) have employed the deterministic Nelder–Mead algorithm (Nelder and Mead 1965), which is, actually, similar to the above mentioned Complex algorithm; Poole and Kotsialos (2012) utilized stochastic genetic algorithms; and Ngoduy and Maher (2012) the stochastic cross-entropy method (de Boer et al. 2005). Surprisingly, there is no work addressing the suitability and effectiveness of the employed optimization algorithms for this specific parameter estimation problem. Thus, the goal of this study is to test and compare various optimization algorithms, both stochastic and deterministic, for the calibration of a macroscopic traffic flow model, namely the METANET model (Messmer and Papageorgiou 1990; Papageorgiou et al. 2010), using real traffic data from a freeway stretch. In particular, the optimization algorithms considered are the deterministic Nelder–Mead algorithm, the stochastic genetic algorithms and the stochastic cross-entropy method. These algorithms are compared in terms of optimum cost function value, computation time and accuracy of the produced validated model.

The paper is organized as follows: Sect. 2 presents the macroscopic traffic flow model and the unknown model parameters to be estimated. Section 3 describes the model calibration procedure which is formulated as a least-squares minimization problem; and Sect. 4 presents the considered optimization algorithms that are employed to solve the parameter estimation problem. Section 5 describes the considered freeway stretch and the traffic data used in the current investigations; followed by Sect. 6 which includes the calibration results and the comparison of the employed optimization algorithms. Finally, Sect. 7 summarizes the main conclusions and remarks of the study.

2 Macroscopic traffic flow model

The macroscopic traffic flow model METANET (Messmer and Papageorgiou 1990; Papageorgiou et al. 2010) will be calibrated for a particular freeway site. Within METANET, the freeway is divided into homogeneous, consecutively numbered sections i, with respective lengths L_i and number of lanes λ_i. Each section may have an on-ramp and off-ramp near its upstream boundary as shown in Fig. 1. The time is also discretized into uniform intervals of duration T. For each discrete time $k = 0, 1, \ldots, K$, the model calculates at each section i, the density $\rho_i(k)$ (in veh/km/lane), the flow $q_i(k)$ (in veh/h) and the mean speed $v_i(k)$ (in km/h) according to the following equations:

$$\rho_i(k+1) = \rho_i(k) + \frac{T}{L_i\lambda_i}[q_{i-1}(k) - q_i(k) + r_i(k) - s_i(k)]. \tag{1}$$

This is a conservation-of-vehicles equation, where $r_i(k)$ is the traffic flow entering the freeway from an on-ramp and $s_i(k)$ is the traffic flow exiting the freeway from an off-ramp, and equals to $s_i(k) = \beta_i(k)q_{i-1}(k)$ where $\beta_i(k)$ is the splitting ratio. Furthermore we have

$$q_i(k) = v_i(k)\rho_i(k)\lambda_i \tag{2}$$

$$\begin{aligned} v_i(k+1) = v_i(k) &+ \frac{T}{L_i}v_i(k)[v_{i-1}(k) - v_i(k)] + \frac{T}{\tau}[V^e(\rho_i(k)) - v_i(k)] \\ &- \frac{\nu T[\rho_{i+1}(k) - \rho_i(k)]}{\tau L_i[\rho_i(k) + \kappa]} \end{aligned} \tag{3}$$

where τ (a time constant), ν (an anticipation constant) and κ are model parameters. The function $V^e(\rho_i(k))$ corresponds to the fundamental diagram and is calculated as follows:

$$V^e(\rho_i(k)) = v_{f,i}\exp\left[-\frac{1}{\alpha_i}\left(\frac{\rho_i(k)}{\rho_{cr,i}}\right)^{\alpha_i}\right] \tag{4}$$

where $v_{f,i}$ is the free speed, $\rho_{cr,i}$ is the critical density (for which the flow at section i is maximized) and α_i is a further model parameter for section i. Moreover, if the mean speed calculated by the model is lower than a minimum value $v_{\min}$, then it is

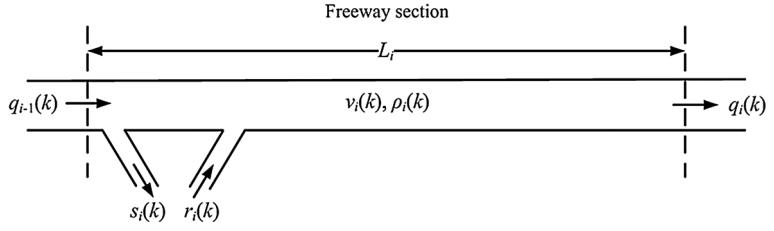

Fig. 1 Freeway discretization within METANET

set equal to $v_{\min}$. In Papageorgiou et al. (1989), two additional terms were proposed for more accurate modeling of merging and lane drop phenomena. In particular, the impact on mainstream speed due to an on-ramp merging flow is considered by adding the term $-\delta \mathrm{T} r_i(k) v_i(k) / L_i \lambda_i (\rho_i(k) + \kappa)$ into (3) for the merging section, where δ is a model parameter. This term is not used if there is a lane gain downstream of the on-ramp, i.e., if there is a dedicated lane for entering vehicles. Moreover, in order to take into account the impact on speed due to intensive lane-changing at lane-drop areas, the term $-\varphi \mathrm{T} \Delta\lambda \rho_i(k) v_i(k)^2 \Big/ L_i \lambda_i \rho_{cr,i}$, is added to (3) for the section immediately upstream of the lane drop, where φ is a model parameter and $\Delta\lambda$ is the number of dropped lanes.

At bifurcation locations (e.g. off-ramps), a downstream density $\rho_{i+1}(k)$ is needed in (3) for the section i entering the bifurcation; this density reflects the upstream influence of the downstream traffic conditions. However, as we have at least two downstream sections at bifurcations, the following formula was proposed (Messmer and Papageorgiou 1990):

$$\rho_{i+1}(k) = \sum_{\mu \in O_i} \rho_\mu^2(k) \Big/ \sum_{\mu \in O_i} \rho_\mu(k) \tag{5}$$

where $\rho_{i+1}(k)$ is the virtual density downstream of section i, which is used in (3) of section i; and $\rho_\mu(k)$ is the density of each section downstream of section i, O_i being the set of exiting sections. The quadratic average used in (5) accounts for the fact that congestion may spill back to a section i from any of its downstream sections (e.g., in case of spillback from a saturated off-ramp), even if the rest downstream sections are not congested. Notice that (5) does not include any parameter to be calibrated.

As presented above, the model includes a number of parameters, whose values may differ for different freeway sites and depend on factors such as network geometry, driver behavior, percentage of trucks, weather conditions etc. Thus, the reliability and accuracy of the model depends on the appropriate specification of its parameter values; and hence a calibration exercise is required before using the model in a potential real application.

3 Model calibration procedure

The model parameter calibration (or parameter estimation) procedure aims at enabling a macroscopic traffic flow model to represent traffic conditions of a freeway network with the highest achievable accuracy. The estimation of the unknown model parameters is not a trivial task, since the system equations are highly nonlinear in both the parameters and the state variables. Consider that a macroscopic discrete-time state-space model is described by the following state equation,

$$\begin{aligned} &\mathbf{x}(k+1) = f[\mathbf{x}(k), \mathbf{d}(k), \mathbf{p}] \\ &\mathbf{x}(0) = \mathbf{x}_0 \end{aligned} \qquad k = 0, 1, \ldots, K-1 \tag{6}$$

where $\mathbf{x}$ stands for the state vector, $\mathbf{d}$ is the disturbance (external variable) vector and $\mathbf{p}$ is the parameter vector. The METANET model can readily assume the state-space form of (6) for any freeway network. In particular, the state vector $\mathbf{x}$ includes the section densities and mean speeds, the external variable vector $\mathbf{d}$ consists of the origin speeds and inflows, the turning rates at bifurcations, and the downstream densities; and $\mathbf{p}$ includes the unknown model parameters that need to be specified.

If the initial state $\mathbf{x}_0$ is given and the external vector $\mathbf{d}(k)$ is known over a time horizon $k = 0, \ldots, K-1$, then the parameter estimation problem can be formulated as a nonlinear least-squares output error problem which aims at the minimization of the discrepancy between the model calculations and the real traffic data by use of the following cost function,

$$J(\mathbf{p}) = \sqrt{\frac{1}{K}\sum_{k=0}^{K-1}[\mathbf{y}(k) - \mathbf{y}^{\mathbf{m}}(k)]^2} \tag{7}$$

subject to (6); where $\mathbf{y}(k) = \mathbf{g}[\mathbf{x}(k)]$ is the measurable model output vector (typically consisting of flows and mean speeds at various network locations) and $\mathbf{y}^{\mathbf{m}}(k)$ includes the real measured traffic data (consisting of flows and speeds at the corresponding network locations). The model parameter values are selected from a closed admissible region of the parameter space, which may be defined on the basis

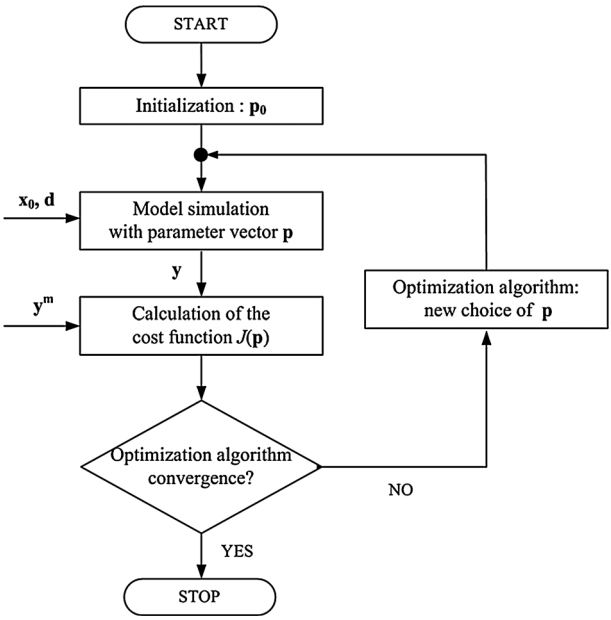

Fig. 2 Model calibration procedure

of physical considerations and previous experience. The determination of the optimal parameter set must be performed by means of a suitable nonlinear programming routine, whereby for each choice of a new parameter vector $\mathbf{p}$, the value of the performance index (PI) (7) may be computed by a calibration run of the model equations as shown in Fig. 2.

The nonlinear, non-convex least-squares optimization problem of parameter calibration is known to have multiple local minima (see Ngoduy and Maher (2012) for an illustration), and hence gradient-based optimization algorithms are not an option. In previous calibration studies, see Sect. 1 for an overview, various derivative-free optimization algorithms have been employed to solve the parameter estimation problem, without much insight though regarding their respective properties for the particular problem at hand. Within this study, various optimization algorithms are tested and compared in order to investigate which optimization methodology is more suitable for the problem of macroscopic traffic flow model calibration.

4 Optimization algorithms

Three derivative-free optimization algorithms are employed to solve the parameter estimation problem; in particular, the deterministic Nelder–Mead algorithm, a stochastic genetic algorithm and the stochastic cross-entropy method. In the following, the selected algorithms are shortly described along with their potential advantages and weak points.

4.1 Nelder–Mead algorithm

The Nelder–Mead method (Nelder and Mead 1965; Lagarias et al. 1998) is one of the best known algorithms for multidimensional unconstrained optimization. The method does not require any derivative information, which makes it suitable for problems with non-linear, discontinuous or stochastic cost function.

The method uses a simplex, i.e. a n-dimensional geometrical shape with $n + 1$ vertices. Every vertex $\mathbf{p}_i$, where $i = 1, \ldots, n + 1$, corresponds to a potential solution which in turn corresponds to a cost function value, $J(\mathbf{p}_i)$. The algorithm starts with an initial working simplex and then performs a sequence of transformations aiming at reducing the cost function value at its vertices. In particular, at each iteration the algorithm orders the simplex vertices with respect to the corresponding cost function values e.g. $J(\mathbf{p}_1) \leq J(\mathbf{p}_2) \leq \cdots \leq J(\mathbf{p}_{n+1})$ and calculates the centroid $\mathbf{p}_c$ of all vertices excluding the worst vertex $\mathbf{p}_{n+1}$. Then, it computes the new working simplex from the current one as follows. First, an attempt is made to replace only the worst vertex $\mathbf{p}_{n+1}$ with a better point by using *reflection*, *expansion* or *contraction*. If this succeeds, the accepted point becomes the new vertex of the working simplex. Otherwise, the algorithm *shrinks* the simplex towards the best vertex $\mathbf{p}_1$. In this case, n new vertices are computed. Simplex transformations are controlled by four parameters: ζ for *reflection*, χ for *contraction*, γ for *expansion* and

σ for *shrinkage*. Note that there is a low need for fine-tuning the algorithm parameters since the parameter values proposed in the original papers seem to work good in a broad number of applications. The above procedure continues until the working simplex becomes sufficiently small or when the function values $J(\mathbf{p}_i)$ are close enough to each other.

In contrast to other direct search methods which call, at each iteration, for multiple cost function evaluations, Nelder–Mead typically requires only one or two function evaluations, except when performing the shrinkage transformation which is, actually, quite rare in practice. As a result, the method typically gives significant ameliorations of the cost function value quite fast. On the other hand, in some cases the method may perform a large number of iterations without significant improvement of the cost function value. To cope with this problem, restarting the algorithm several times, with reasonably small number of allowed iterations per each run, may prove helpful. Generally, the evolution of the working simplex and the produced best solution are dependent on the initial working simplex, since the algorithm searches for new points using the vertices of the working simplex, and this may lead to different paths for different initial simplexes. Such different evolution paths may or may not lead to the same final best solution. To face this fact, multiple algorithm runs may be carried out using different initial vertices for the working simplex and checking the corresponding obtained solutions.

4.2 Genetic algorithm (GA)

A genetic algorithm (Goldberg 1989; Holland 1992) is a heuristic search method which belongs to the larger class of evolutionary algorithms. GA mimics the process of biological evolution and uses techniques inspired by natural selection, mutation and crossover. It is suitable for a variety of optimization problems, in which the objective function is discontinuous, non-differentiable, stochastic, or highly nonlinear.

The method uses a population of candidate solutions to an optimization problem and evolves it towards better solutions. The evolution starts from an initial *population* of randomly generated *individuals* (solutions) which are evaluated through their respective cost function values (fitness). At each iteration, called *generation*, the algorithm selects individuals (*parents*) from the current generation and uses them to produce the individuals (*offspring*) for the next population. To do so, the GA uses three main types of rules:

- *Selection rules* select individuals (parents), with probabilities proportional to their fitness; the selected parents contribute to the population of the next generation. Some of the individuals in the current population, which have best fitness, are chosen as *elite*. These elite individuals are passed directly and unchanged to the next population.
- *Crossover rules* combine (random) couples of parents to form offspring for the next generation, thus exchanging information between two candidate solutions.
- *Mutation rules* apply random changes to individual parents, which may introduce new features (i.e. new parameter space regions) to the population.

Through the stochastic operations of *selection*, *crossover* and *mutation,* the population "evolves", over successive generations, towards potentially better solutions, and the algorithm stops when one of the stopping criteria is met, e.g. when no significant improvement in the cost function values is achieved over successive iterations (generations), or when the maximum allowed number of iterations is reached.

The main advantage of GA is its flexibility to search complex solution spaces; thanks to its stochastic operations, it is less likely to restrict the search to a bad local minimum area, in contrast to point-to-point movement optimization techniques. On the other hand, each iteration requires many cost function evaluations, which increases substantially the computational cost, especially for problems with computationally expensive cost function or problems which require large population size. It is worth noting that, since the evaluation of the cost function for each individual is independent of all others, the parallelization of GA is an option. Finally, it is important to tune the algorithm's parameters, i.e. the population size, the elite rate, the crossover probability and the mutation rate in order to find appropriate and efficient settings for the specific problem being examined.

4.3 Cross-entropy method (CE method)

The cross-entropy method (Rubinstein and Kroese 2004; de Boer et al. 2005) is a general Monte-Carlo approach to combinatorial and continuous multi-extremal optimization and importance sampling problems. The method originates from the field of rare event simulation, where very small probabilities need to be accurately estimated.

The algorithm starts from an initial population of potential solutions generated using a continuous, usually uniform, distribution f_0. At each iteration t, the solutions are evaluated through the cost function and sorted into ascending order; and the best b % solutions comprise the elite sample. The probability density function, $\hat{g}_t$, of this elite sample is estimated, e.g. using a Kernel density estimator as proposed by Ngoduy and Maher (2012), and the probability distribution of the population is updated using the equation:

$$\hat{f}_{t+1} = \hat{f}_t(1-\varepsilon) + \hat{g}_t\varepsilon \tag{8}$$

where ε is a smoothing parameter, typically in the range [0.7, 0.9]. The updated density equation $\hat{f}_{t+1}$ is used in the next iteration to generate the new random sample of solutions. The algorithm continues leading, over iterations, to increasingly more spiked shapes of the population probability distribution; and it stops when one of the stopping criteria is met, e.g. when the shape of the probability density function becomes very spiked (i.e. concentrated around the optimal value) or when the maximum allowed number of iterations is reached.

As with the previous algorithms, the CE method does not require any derivative information, thus it may be applied to problems where the objective function is discontinuous, non-differentiable or highly nonlinear. In contrast to other stochastic

methods, the selection of the potential solutions is not a completely random process, since the utilized distribution is affected by the best solutions of each iteration. The main disadvantage of the method is that it requires as many cost function evaluation as the size of the population, resulting in large computational cost and slow convergence. Again, it is important to tune the algorithm's parameters, i.e. the population size, the elite rate b and the smoothing parameter ε in order to find appropriate and efficient settings for the specific problem being examined.

5 Test site and traffic data

The test network considered in this study is a part of Attiki Odos freeway (34th to 28th km, direction Airport to Elefsina) in Athens, Greece. Figure 3 represents the examined freeway stretch in terms of nodes and links. Each node (N0–N8) illustrates a bifurcation point or a junction or any location marking a change of the network geometry; whereas the homogeneous road stretches between these locations are represented by links (L1–L8). Each network link is subdivided in sections of equal length; see for example link L1 which is divided in 3 sections, with the vertical short lines denoting the section borders. Using this representation, the network sections are well-defined, and the model equations presented in Sect. 2 are directly applied to these sections. Figure 3 displays the length, number of sections and number of lanes for each link, the exact location of the on-ramps and off-ramps, as well as the location of the available detector stations which are depicted by bullets.

The real traffic data used in this study were provided by ATTIKES DIADROMES S.A., which is the freeway operating company. In particular, the provided traffic data include flow and speed measurements at the corresponding detector station locations, with a time resolution of 20 s, for several days within June 2009. The traffic data analysis showed that, within this particular freeway stretch, recurrent traffic congestion is formed during the morning peak hours.

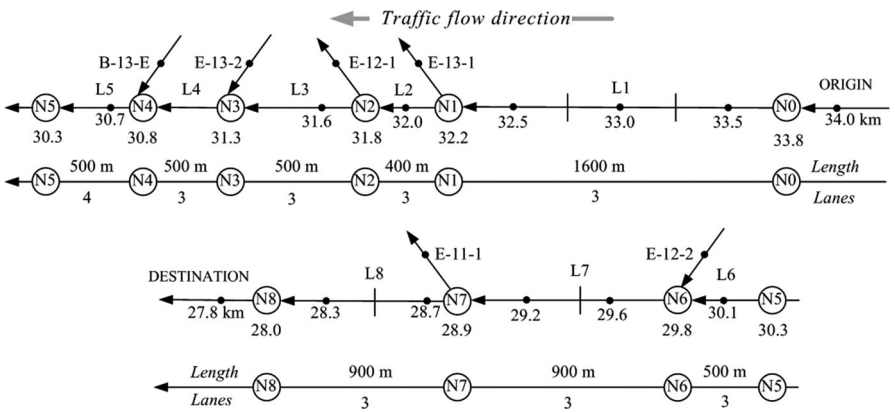

Fig. 3 Representation of the considered freeway stretch within METANET

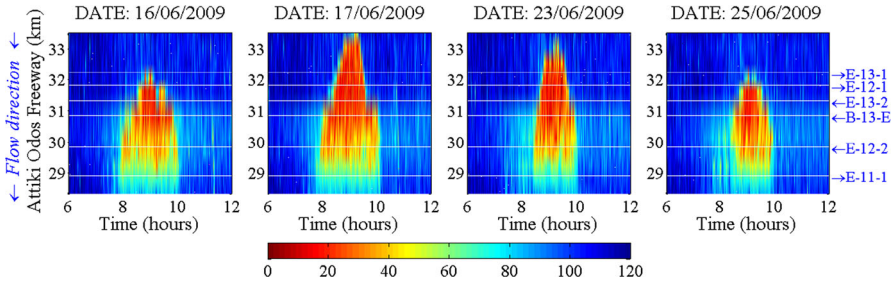

Fig. 4 Time-space diagram of measured speed at the considered freeway stretch for four different days

Figure 4 illustrates the space–time diagram of real speed measurements for 4 different days: 16/06/2009, 17/06/2009, 23/06/2009 and 25/06/2009.

It is observed that congestion is created during 8–10 a.m.; the congestion originates at the 29th km of the freeway stretch and spills back several kilometers upstream, up to the 32nd km, and on some days up to the 33rd km. Figure 3 shows that the congestion creation area is actually a diverge area, with the off-ramp E-11-1 receiving high exit flow during the morning peak hours, according to real traffic data. The high exit flow rate, in combination with the limited capacity of the off-ramp, lead to the creation of congestion, which propagates upstream for several kilometers on the freeway mainstream. The test network and traffic data presented above are used to calibrate and validate the METANET traffic flow model. It should be noted that the main criterion for selecting these 4 days was that, during the morning hours 6–12 a.m., no incident and no detector failure occurred at the examined freeway stretch, which can, of course, not be reproduced by any traffic flow model.

6 Calibration results and comparison of the employed algorithms

The calibration procedure, as described in Sect. 3, was applied to METANET model using real traffic data from 16/06/2009 and a simulation step equal to $T = 5$ s. The model parameter vector consists of the free flow speed v_f, the critical density ρ_{cr} and the parameters α, τ, ν, δ and φ which are common for all the freeway sections. Thus, one single fundamental diagram is considered for all freeway sections. Moreover, the model includes two extra parameters which are known from previous validation exercises to be of minor importance and are, therefore, given constant values, in order to reduce the dimension of the parameter vector. In particular, κ is set equal to 10 veh/km/lane and $v_{\min}$ is set to 7 km/h. Furthermore, the utilized Performance Index (PI) (see (7)) includes the model estimation of speed and the real speed measurements at the corresponding detector station locations (see Fig. 3). This is because traffic densities are difficult to measure directly; on the other hand, experience from previous validation procedures has shown that the calculation of reasonably accurate flows is not a major problem for a macroscopic traffic flow model, since the conservation equation guarantees that whatever flows in will

eventually flow out; in contrast, it is much more challenging to correctly model the time evolution of the mean speeds in each section.

All simulations were performed using a desktop computer with 2.4 GHz CPU and 2.0 GB of RAM. The calibration procedure, including the traffic flow model and the optimization algorithms, has been programmed in MATLAB (R2010a). In the following sections, the optimization results of each utilized algorithm are presented first, followed by the evaluation of the algorithms' performance using various criteria. It should be noted that, for each utilized algorithm, various initial calibration tests were carried out using different values for the algorithms' parameters. These tests helped to fine-tune the algorithms parameters for the present problem, and these parameters were used in the investigations presented in the following sections.

6.1 Nelder–Mead algorithm

The Nelder–Mead algorithm was employed using the following parameters: $\zeta = 1$, $\chi = 2$, $\gamma = -0.5$ and $\sigma = 0.5$ (see Sect. 4.1 for a description of the algorithm's parameters). Moreover, the utilized termination criteria are the cost function convergence or the working simplex convergence, with tolerance equal to 0.1; and the maximum allowed number of iterations which was set equal to 1000.

Figure 5 presents the convergence of the algorithm, i.e. the evolution of the best PI value in the simplex, for ten calibration runs, starting from the same initial point. In particular, at each calibration run, the algorithm received the very same initial vertex and constructed the initial working simplex by generating the required vertices randomly around the given initial point. As indicated at Sect. 4.1, the number of vertices of the working simplex is set to $n + 1$, where n is the number of parameters under calibration. It should also be noted that it is preferable to use a physically reasonable initial vertex in order to speed up the algorithm convergence. Figure 5 shows that, although the algorithm starts from high PI values, it achieves a significant improvement already in the first iterations. Table 1 presents the

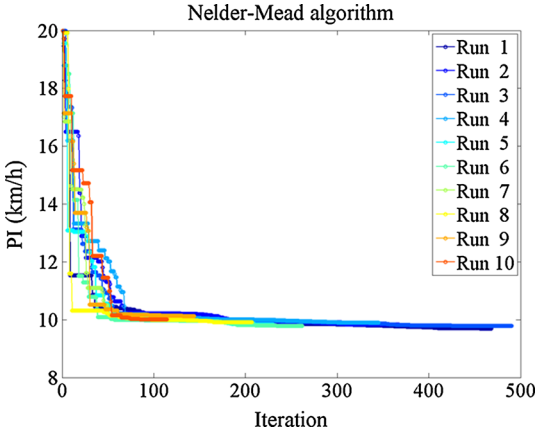

Fig. 5 Nelder–Mead algorithm: best performance index value over iterations during the model calibration procedure for ten calibration runs

Table 1 Nelder–Mead algorithm: performance criteria

Calibration run	PI (km/h)	Iterations	Cost function evaluations	Computation time (min)
1	9.7	467	712	0.9
2	9.8	366	573	0.7
3	9.8	489	749	0.9
4	9.9	344	561	0.7
5	9.8	229	372	0.4
6	9.8	261	423	0.5
7	10.0	137	240	0.3
8	9.9	206	337	0.4
9	10.1	143	238	0.3
10	10.0	113	201	0.2
Average	9.9	275.5	440.6	0.5

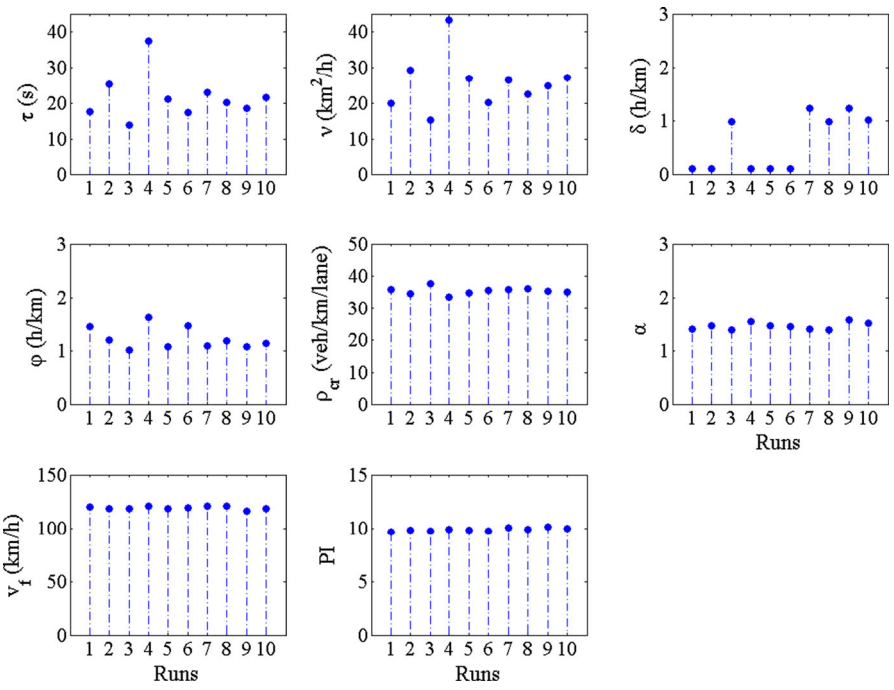

Fig. 6 Nelder–Mead algorithm: estimated optimal parameter values for each calibration run

performance of the algorithm in terms of various criteria. It may be seen that the algorithm converges to low and very similar PI values, equal to 9.9 km/h on average. Moreover, the algorithm requires 113–489 iterations (275.5 iterations on average) to converge which correspond to less than 1 min (0.5 min on average) computation time.

Figure 6 displays the optimal values estimated by the algorithm at the end of each calibration run for all model parameters and also the corresponding achieved PI value. It may be seen that the deviation of the estimated parameter values across the different runs is relatively small. This is especially true for the parameters of the fundamental diagram (FD), i.e. the free flow speed v_f the critical density ρ_{cr} and the parameter a (see also Eq. (4)). Moreover, analogy strong correlation is observed in the estimated optimal values of the parameters τ and ν. Finally, the stronger variation observed in the rest of the parameters is deemed to reflect their limited significance and low sensitivity of the PI with respect to these parameters. Table 4 includes the estimated parameter values for one calibration run with PI value closest to the average PI value of the 10 runs. This parameter set will be validated and compared with the optimal sets estimated by the other two algorithms.

6.2 Genetic algorithm

The employed genetic algorithm is included in the Matlab Global Optimization Toolbox. The population size was set equal to 500, the elite rate equal to 0.01, the crossover rate equal to 0.8 and the mutation rate equal to 0.1. The termination criteria utilized are again the cost function convergence which was set equal to 0.001 and the maximum allowed number of iterations (generations) which was set equal to 1000.

Figure 7 presents the convergence of the genetic algorithm, i.e. the evolution of the best PI value, for ten calibration runs, starting from random (within reasonable bounds) initial populations. It is observed that the algorithm achieves low PI values already in few iterations thanks to the fact that it searches, during each iteration, at a large number of points within the solution space. Table 2 includes the performance of the algorithm in terms of various criteria. It may be seen that the algorithm converges to low and very similar PI values, equal to 10.2 km/h on average. The number of iterations to converge is much lower compared to the Nelder–Mead algorithm; however, due to the fact that at each iteration a high number of cost

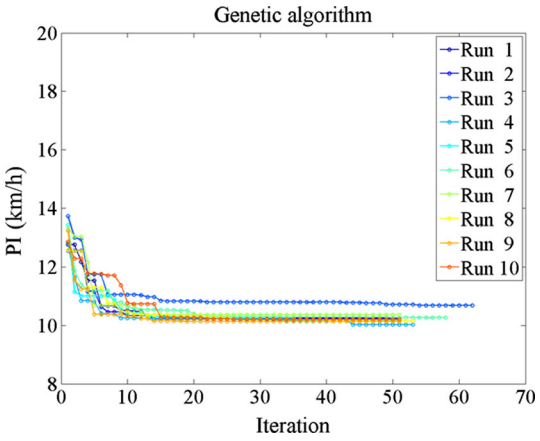

Fig. 7 Genetic algorithm: best performance index value over iterations during the model calibration procedure for ten calibration runs

Table 2 Genetic algorithm: performance criteria

Calibration run	PI (km/h)	Iterations	Cost function evaluations	Computation time (min)
1	10.2	51	25,500	123.5
2	10.2	51	25,500	124.0
3	10.7	62	31,000	150.0
4	10.0	53	26,500	129.1
5	10.2	51	25,500	124.3
6	10.3	58	29,000	140.8
7	10.4	51	25,500	124.4
8	10.2	53	26,500	127.3
9	10.1	51	25,500	122.9
10	10.2	51	25,500	122.8
Average	10.2	53.2	26,600	128.9

function evaluations is required, this results in considerably higher computation time, i.e. about 129 min on average, while the Nelder–Mead algorithm needed less than 1 min to converge.

Figure 8 displays the optimal values estimated by the algorithm at the end of each calibration run for all model parameters and also the corresponding achieved PI value. It is observed, also here, that the deviation of the estimated parameter

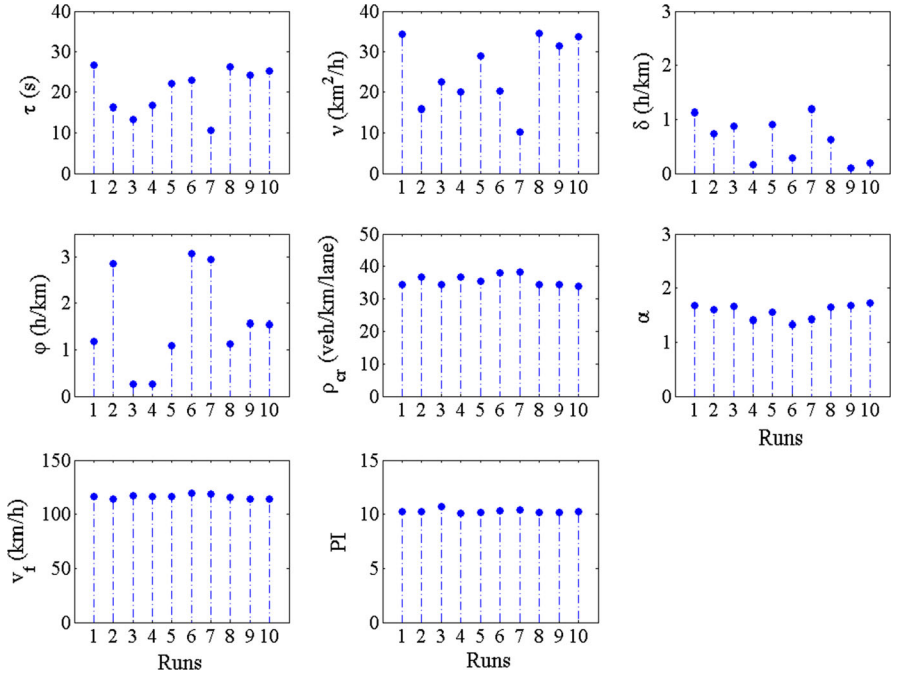

Fig. 8 Genetic algorithm: estimated optimal parameter values for each calibration run

values is small, especially for the parameters involved in the FD. Finally, Table 4 includes the estimated parameter values for one calibration run with PI value closest to the average PI value of the 10 runs.

6.3 Cross-entropy method (CE method)

The cross-entropy method was applied using population size equal to 500, elite rate 0.05 and smoothing parameter ε equal to 0.8. The utilized termination criteria are the bandwidth of the kernel estimation function, which was set equal to 0.1 and the maximum allowed number of iterations which was set to 1000.

Figure 9 presents the convergence of the cross-entropy method, i.e. the evolution of the best PI value in the population, for ten calibration runs, starting from random (within reasonable bounds) initial populations. It is observed that, similarly to the genetic algorithm, this method achieves low PI values even already at the first iterations thanks to the large number of potential solutions considered within the search space. Table 3 includes the performance of the algorithm in terms of various criteria. It may be seen that also this algorithm converges to low and similar PI values, equal to 10.0 km/h on average, but, as with the genetic algorithm, it requires a high number of cost function evaluations which corresponds to high computation time compared to the Nelder–Mead algorithm. Finally, comparing the genetic algorithm with the CE method, it may be seen that both algorithms achieve similar PI values and also require similar computation time, with the CE method being some 8 min, on average, faster than the genetic algorithm.

Figure 10 displays the optimal values estimated by the algorithm at the end of each run for all model parameters and also the achieved PI value. It is observed, also here, that the deviation of the estimated parameter values is small, especially for the parameters involved in the FD. Finally, Table 4 includes the estimated parameter values for one run with PI value closest to the average PI value of the 10 calibration runs.

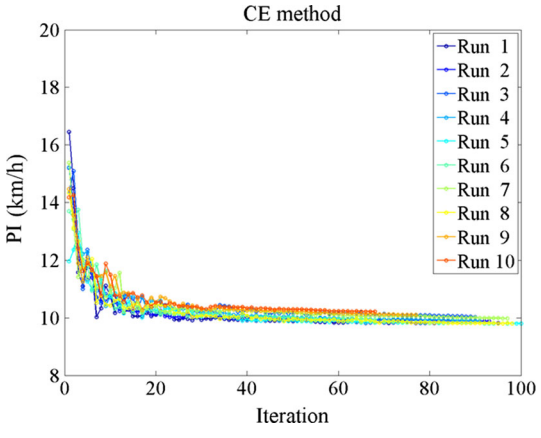

Fig. 9 Cross-entropy method: best performance index value over iterations during the model calibration procedure for ten calibration runs

Table 3 Cross-entropy method: performance criteria

Calibration run	PI (km/h)	Iterations	Cost function evaluations	Computation time (min)
1	9.8	95	47,500	130.2
2	10.0	93	46,500	127.9
3	10.1	90	45,000	124.6
4	10.0	91	45,500	125.2
5	9.8	100	50,000	137.7
6	10.1	68	34,000	94.1
7	10.0	97	48,500	135.0
8	9.8	98	49,000	135.3
9	10.1	77	38,500	106.6
10	10.2	68	34,000	95.0
Average	10.0	87.7	43,850	121.1

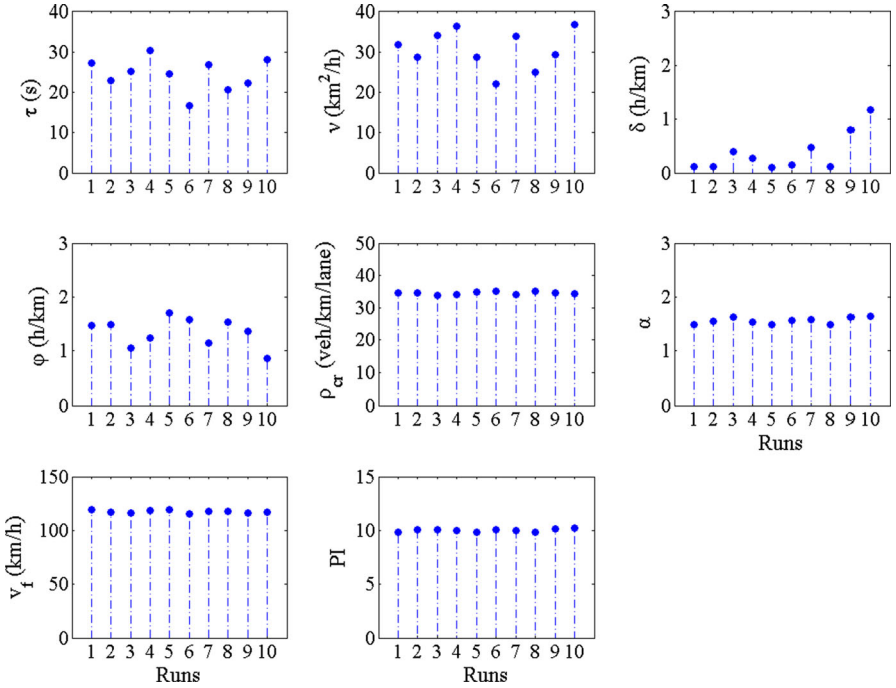

Fig. 10 Cross-entropy method: estimated optimal parameter values for each calibration run

6.4 Validation of the produced models

Apart from the performance of the optimization methods, the decision on the optimization algorithm to be employed for the calibration of a macroscopic traffic flow model should also consider the quality of the produced solutions (models). As

Table 4 Optimal model parameter sets estimated by use of different optimization algorithms

Optimization algorithm	Model parameters						
	τ (s)	ν (km^2/h)	δ (h/km)	φ (h/km)	ρ_{cr} (veh/km/lane)	α	v_f (km/h)
NM algorithm	20.2	22.5	0.9	1.2	36.3	1.4	120.9
GA	26.7	34.2	1.1	1.2	34.3	1.7	116.4
CE method	30.4	36.3	0.3	1.2	34.1	1.5	118.7

presented above, the calibration of METANET model for the particular freeway site, using three different optimization algorithms, may result in different model parameter sets, with similar but not the same parameter values, see for example Table 4. In particular, Table 4 indicates the very close proximity of, particularly, the optimal parameter values which are involved in the fundamental diagram (FD). Moreover, it is known from previous model validation work (e.g. Papageorgiou et al. 1990) that the calibration PI features low sensitivity around the optimum if the parameters ν and τ are changing values simultaneously. This is confirmed with the results of Table 4, where the ratio ν/τ may be calculated to be 1.1, 1.3, 1.2 for the three respective optimization methods, despite the stronger deviation of the underlying absolute parameter values.

The resulting traffic flow models should reflect reliably the traffic characteristics of the considered network, thus they should be able to reproduce its typical traffic conditions. In order to test the accuracy and robustness of the produced models, the models are validated, i.e. are applied using different traffic data sets (from the same freeway stretch) than the ones used for their calibration. To this end, the models included in Table 4, were applied using traffic data from 17/06/2009, 23/06/2009 and 25/06/2009. Table 5 presents the validation results in terms of PI values for all three models and all utilized traffic data sets. Moreover, Fig. 11 presents the space–time diagrams of the real measured speeds and the models' estimation of speed for all considered dates. It is observed that all models are able to reproduce the traffic conditions of other days with sufficient accuracy, achieving low PI values for both the calibration and the validation dates.

Table 5 Performance index value for each optimal parameter set for four different dates

Optimization algorithm	Validation results (PI)				
	16/06/2009	17/06/2009	23/06/2009	25/06/2009	Average
NM algorithm	9.9	9.2	11.2	9.6	10.0
GA	10.2	10.63	12.8	8.4	10.5
CE method	10.0	10.1	12.1	8.6	10.2

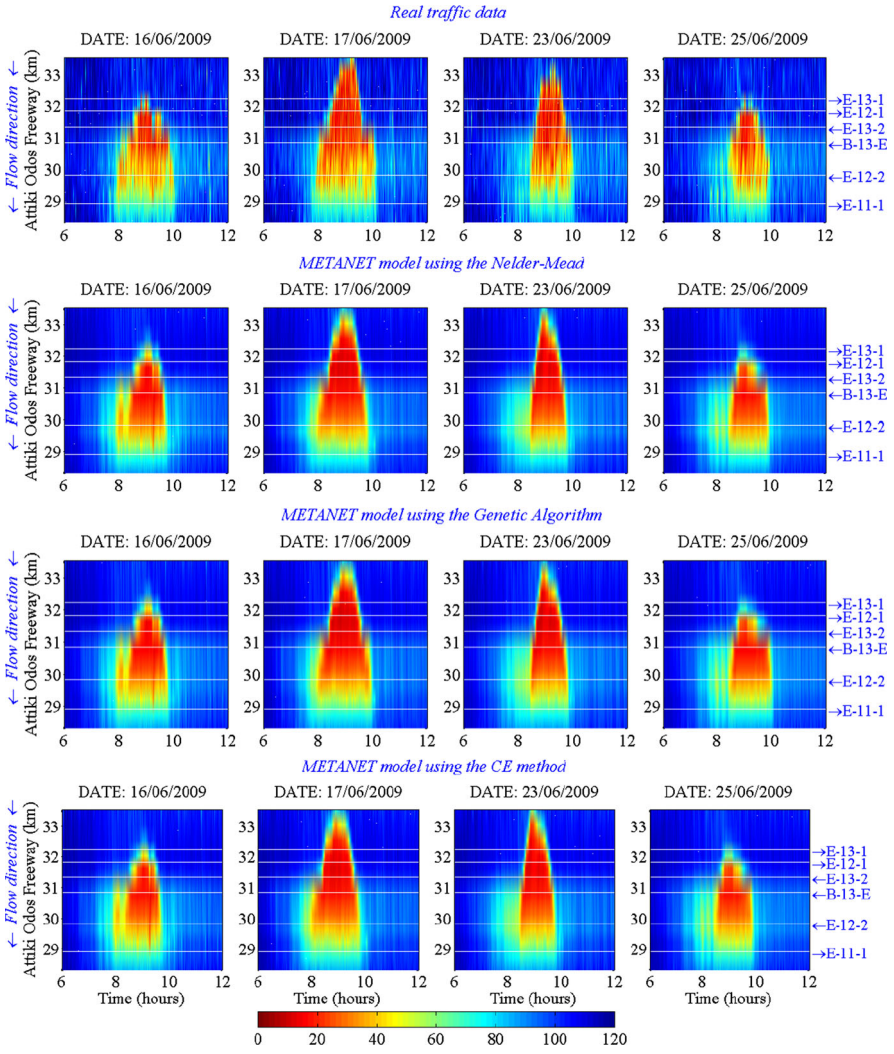

Fig. 11 Space-time diagrams of real measured speed and the models' estimation of speed for 16/06/2009, 17/06/2009, 23/06/2009 and 25/06/2009

Considering the above, all three optimization algorithms are able to estimate a robust model parameter set. Nevertheless, while employing an optimization algorithm, the required computation time for calibration should be taken into account. Note that multiple calibration runs may have to be carried out in order to decide on the number of the utilized model parameters and also in order to tune the algorithms' parameters for the particular problem. Finally, it should further be noted that, in this study, a low number of model parameters were considered, while in other problems with higher number of parameters the performance of the presented algorithms may differ.

7 Conclusions

Within this study, three different optimization algorithms were employed to solve the parameter estimation problem for a macroscopic traffic flow model. In particular, the deterministic Nelder–Mead algorithm, the stochastic genetic algorithm and the stochastic cross-entropy method were utilized in order to calibrate the METANET model for a particular freeway stretch using real traffic data. The optimization results showed that all three algorithms were able to converge to robust model parameter sets, albeit achieving different performances considering the convergence speed and the required computation time.

Acknowledgments This research was co-financed by the European Union (European Social Fund—ESF) and by national funds through the Operational Program "Education and Lifelong Learning" of the National Strategic Reference Framework (NSRF)—Research Funded Project: ARCHIMEDES III, Investing in Society's Knowledge through the European Social Fund. The authors would like to thank ATTIKES DIADROMES S.A. for providing the utilised traffic data from Attiki Odos freeway in Athens, Greece and also Prof. Mike Maher and Dr. Dong Ngoduy for providing the source code for the kernel-based cross-entropy method.

References

Box MJ (1965) A new method of constrained optimization and a comparison with other methods. Comput J 8:42–52

Cremer M, May AD (1986) An extended traffic flow model for inner urban freeways. In: Proceedings of the 5th IFAC/IFIP/IFORS international conference on control in transportation systems, pp 383–388

Cremer M, Papageorgiou M (1981) Parameter identification for a traffic flow model. Automatica 17(6):837–843

De Boer P-T, Kroese DP, Mannor S, Rubinstein RY (2005) A tutorial on the cross-entropy method. Ann Oper Res 134:19–67

Goldberg D (1989) Genetic algorithms in search, optimization and machine learning. Addison-Wesley Professional, Reading

Grewal MS, Payne HJ (1976) Identification of parameters in a freeway traffic model. IEEE Trans Syst Man Cybern 6(3):176–185

Helbing D (1996) Derivation and empirical validation of a refined traffic flow model. Phys. A 233(1):253–282

Holland J (1992) Adaptation in natural and artificial systems. MIT Press, Cambridge

Hoogendoorn SP, Bovy PH (2001) State-of-the-art of vehicular traffic flow modelling. Proc Inst Mech Eng I J Syst Control Eng 215(4):283–303

Kontorinaki M, Spiliopoulou A, Papamichail I, Papageorgiou M, Tyrinopoulos Y, Chrysoulakis J (2014) Overview of nonlinear programming methods suitable for calibration of traffic flow models. Oper Res Int J. doi:10.1007/s12351-014-0146-9

Kotsialos A, Papageorgiou M (2000) The importance of traffic flow modelling for motorway traffic control. Netw Spat Econ 1:179–203

Kotsialos A, Papageorgiou M, Diakaki C, Pavlis Y, Middelham F (2002) Traffic flow modeling of large-scale motorway networks using the macroscopic modeling tool METANET. IEEE Trans Intell Trans Syst 3:282–292

Lagarias JC, Reeds JA, Wright MH, Wright PE (1998) Convergence properties of the Nelder–Mead simplex method in low dimensions. SIAM J Optim 9(1):112–147

Messmer A, Papageorgiou M (1990) METANET: a macroscopic simulation program for motorway networks. Traffic Eng Control 31(8–9):466–470

Michalopoulos PG, Yi P, Lyrintzis AS (1993) Continuum modelling of traffic dynamics for congested freeways. Transp Res Part B 27(4):315–332

Monamy T, Haj-Salem H, Lebacque J-P (2012) A macroscopic node model related to capacity drop. In: Proceedings of EWGT2012—15th meeting of the EURO working group on transportation, vol 54, pp 1388–1396

Nelder JA, Mead R (1965) A simplex method for function minimization. Comput J 7:308–313

Ngoduy D, Maher MJ (2012) Calibration of second order traffic models using continuous cross entropy method. Transp Res Part C 24:102–121

Ngoduy D, Hoogendoorn SP, van Zuylen HJ (2004) Comparison of numerical schemes for macroscopic traffic flow models. Transp Res Rec 1876:52–61

Papageorgiou M, Blosseville JM, Hadj-Salem H (1989) Macroscopic modelling of traffic flow on the Boulevard Périphérique in Paris. Transp Res Part B 23(1):29–47

Papageorgiou M, Blosseville J-M, Hadj-Salem H (1990) Modelling and real-time control of traffic flow on the southern part of Boulevard Périphérique in Paris—Part I: modelling. Transp Res Part A 24(5):345–359

Papageorgiou M, Papamichail I, Messmer A, Wang Y (2010) Traffic simulation with METANET. In: Barceló J (ed) Fundamentals of traffic simulation. Springer, New York, pp 399–430

Poole AJ, Kotsialos A (2012) METANET Model validation using a genetic algorithm. Control Trans Syst 13(1):7–12

Rubinstein RY, Kroese DP (2004) The cross-entropy method: a unified approach to combinatorial optimization, Monte-Carlo simulation, and machine learning. Springer, New York

Sanwal KK, Petty K, Walrand J, Fawaz Y (1996) An extended macroscopic model for traffic flow. Transp Res Part B 30(1):1–9

Spiliopoulou A, Kontorinaki M, Papageorgiou M, Kopelias P (2014) Macroscopic traffic flow model validation at congested freeway off-ramp areas. Transp Res Part C 41:18–29

Treiber M, Kesting A (2013) Traffic flow dynamics. Springer, Berlin. doi:10.1007/978-3-642-32460-4