An Aggregate Approach for the Calibration of Time-Dependent Demand in Dynamic Traffic Assignment Models using SPSA Algorithm

Bojan Kostić[1], Guido Gentile[1], Constantinos Antoniou[2]

[1]Sapienza University of Rome
Department of Civil, Constructional and Environmental Engineering (DICEA)
Transport Division
Via Eudossiana 18, 00184 Rome, Italy
{bojan.kostic; guido.gentile}@uniroma1.it

[2]National Technical University of Athens (NTUA)
School of Rural and Surveying Engineering
Laboratory of Transportation Engineering
Iroon Polytechniou 9, Zografou campus, GR-15780 Athens, Greece
antoniou@central.ntua.gr

Abstract – Demand calibration in Dynamic Traffic Assignment is often regarded as an optimization problem. The most widely algorithm used in the literature is SPSA, using single O-D pairs as calibration variables. This approach is very difficult to apply to large-scale networks due to the large number of variables. In this paper we present a novel approach that can lead to superior demand calibration results. It consists of the aggregation of single demand components based on a correlation object. An analytical model was created to represent real-world dynamics and is used for tests. We tested different scenarios regarding different levels of measurement errors. The results show that the calibration using SPSA algorithm with total origin demand and total destination demand can converge to the solution, whereas using standard approach, i.e. O-D pair calibration, breaks. This approach has a number of advantages which are analysed and discussed.
Keywords: demand calibration, Simultaneous Perturbation Stochastic Approximation (SPSA), Dynamic Traffic Assignment (DTA), demand aggregation

1. Introduction

Dynamic Traffic Assignment (DTA) has been increasingly used as more robust models are becoming available. In order to use a DTA model in real time, it first has to be calibrated off-line. Demand is an essential input to a DTA model that needs to be calibrated. Recent literature has been going in the direction of formulating demand calibration problem as an optimization problem and using stochastic optimization algorithms to solve it. Simultaneous Perturbation Stochastic Approximation (SPSA) algorithm (Spall, 1992; Spall, 1998) has been used in the literature for demand calibration in DTA models. It is a derivative-free optimization algorithm that performs stochastic search based on gradient approximation. Most of the case studies proving the effectiveness of the original SPSA are conducted on small-scale networks. However, when applied to large-scale networks SPSA does not converge as demonstrated by several authors (see for example Lu et al., 2015). Therefore there is a need to improve its performance when a lot of variables are involved. Some authors provide techniques to deal with this problem by suggesting improved variants of the standard SPSA algorithm, such as W-SPSA (Lu et al., 2015) which is an improved SPSA algorithm that treats the spatial and temporal correlations in a network to reduce the noise in the objective function. Other solutions include c-SPSA algorithm (Tympakianaki et al., 2015) which uses clusters to group demand components and calibrate them separately. There are also directions that utilize polynomial interpolation (Cipriani et al., 2013) to enhance the performance of the SPSA algorithm.

Following the research direction of improving the convergence properties of SPSA algorithm in dynamic demand calibration, we present an aggregate approach to demand perturbation schemes that makes SPSA algorithm applicable to large-scale networks. In this approach we do not use single origin-destination (O-D) components, but rather aggregate demand components according to their correlation object. Therefore we test the aggregation regarding total origin demand, total destination demand and/or total temporal profile demand. This way we significantly reduce the problem dimension thus allowing for SPSA to utilize its proven benefits in convergence in low-dimensional spaces. This solution does not only simplify the problem but also has a number of other benefits.

The paper is organized as follows. Chapter 2 introduces the formulation of the calibration problem and describes SPSA as the solution algorithm. Chapter 3 describes the creation of the synthetic model used in simulations. Chapter 4 describes demand perturbation possibilities using various levels of aggregation of demand components as correlation objects. Chapter 5 provides experimental results and analysis. In the end, Chapter 6 gives concluding remarks.

2. Problem formulation and solution algorithm

We look at the demand calibration problem as an optimization problem. On the traffic data side we assume the presents of flow measurements only. In general, the optimization problem can be formulated as follows:

$$\mathbf{d}^* = \arg\min_{\mathbf{d}\geq 0}\left[w_d \cdot \mathbf{r}_d \cdot f_d\left(\tilde{\mathbf{d}}, \hat{\mathbf{d}}\right) + w_q \cdot \mathbf{r}_q \cdot f_q\left(\mathbf{f}, \mathbf{y}\right)\right] \tag{1}$$

where $\mathbf{d}^*$ is the solution demand vector we are searching for, w_d is a weight of demand distance part in the objective function, $\mathbf{r}_d$ is a vector containing accuracy information for each demand component, f_d represents a distance function between assigned demand $\tilde{\mathbf{d}}$ and historic demand $\hat{\mathbf{d}}$. On the other side, analogously, w_q represents a weight of distance from observed traffic data, in our case flows, in the objective function; $\mathbf{r}_q$ is a vector containing the accuracy information for each flow measurement; f_q is a distance function between simulated and observed flows with $\mathbf{f}$ being a vector of estimated link flows and $\mathbf{y}$ a vector of observed link flows. In the tests we do not use the demand distance part of the objective function hence the assigned weights are $w_d = 0$ and $w_q = 1$. We try to reduce only the distance between simulated and observed flows and the optimization problem therefore becomes:

$$\mathbf{d}^* = \arg\min_{\mathbf{d}\geq 0}\left[\mathbf{r}_q \cdot f_q\left(\mathbf{f}, \mathbf{y}\right)\right] \tag{2}$$

We are using standard 1st order SPSA algorithm. It is known to be very efficient as it does only two function evaluations per iteration. These two function evaluation are sufficient to approximate the gradient and calculate new point in a given direction. Gradient approximation is calculated as follows:

$$\hat{\mathbf{g}}_k(\hat{\boldsymbol{\theta}}_k) = \frac{y(\hat{\boldsymbol{\theta}}_k + c_k\boldsymbol{\Delta}_k) - y(\hat{\boldsymbol{\theta}}_k - c_k\boldsymbol{\Delta}_k)}{2c_k} \begin{bmatrix} \Delta_{k[1]}^{-1} \\ \Delta_{k[2]}^{-1} \\ \vdots \\ \Delta_{k[p]}^{-1} \end{bmatrix} \tag{3}$$

where $\hat{\boldsymbol{\theta}}_k$ is a vector with values of calibration variables at iteration k; $\boldsymbol{\Delta}_k$ is a Bernoulli-distributed ±1 random variable at iteration k; c_k represents a gain sequence responsible for sampling two points for function evaluations used for gradient approximation and is given by:

$$c_k = \frac{c}{k^{\gamma}} \tag{4}$$

where c and γ are algorithm coefficients. New point is then obtained using the step size and previously computed gradient:

$$\hat{\boldsymbol{\theta}}_{k+1} = \hat{\boldsymbol{\theta}}_k - a_k \hat{\mathbf{g}}_k(\hat{\boldsymbol{\theta}}_k) \quad (5)$$

where a_k is a gain sequence responsible for the step size, i.e. the advancement in the direction of the gradient and is given by:

$$a_k = \frac{a}{(A+k)^{\alpha}} \quad (6)$$

where A and α are algorithm coefficients.

The aggregation that we introduce is related to the vector $\hat{\boldsymbol{\theta}}_k$, i.e. its elements. Therefore the novel approach we test cannot be seen in the solution algorithm description and is presented in Chapter 4.

3. Simulation model

A synthetic environment was established for running tests. A detailed description of the model is presented below. The model was created to represent a real-world network with all its correlations and noise. Synthetic traffic data used in calibration are represented by a vector **y** which is a vector of traffic counts (i.e. flow measurements), given by Eq. (**7**). Each element of a vector **y** is generated as a time-dependent flow measurement, such that a generic measurement i, y_i, is equal to $q_{k,j}$, where q represents a flow value, subscript k denotes detector index and subscript j is temporal profile index. The number of traffic measurements is denoted with n_m.

$$\mathbf{y}^{\mathrm{T}} = \begin{bmatrix} y_1 & y_2 & \cdots & y_{n[m]} \end{bmatrix} \quad (7)$$

Traffic counts are generated using analytical relations which purpose is to realistically represent real-world traffic counts. They are calculated as a function of several variables, such as the assignment matrix, errors related to the assignment matrix, true demand vector and measurement errors. An $n_m \times n_v$ assignment matrix is denoted with **A** and given by Eq. (**8**). The number of unknown variables to be calibrated, i.e. time-dependent demand components, is denoted with n_v.

$$\mathbf{A} = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n[v]} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n[v]} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n[m],1} & a_{n[m],2} & \cdots & a_{n[m],n[v]} \end{bmatrix} \quad (8)$$

The errors contained in the assignment matrix **A** are stored in the matrix $\mathbf{E}^{\mathrm{A}}$, which has the same $n_m \times n_v$ dimension as the assignment matrix. Each element ε^{A} from $\mathbf{E}^{\mathrm{A}}$ corresponds to an element a from **A** at the same position. The error matrix is given by Eq. (**9**).

$$\mathbf{E}^{\mathrm{A}} = \begin{bmatrix} \varepsilon^{\mathrm{A}}_{1,1} & \varepsilon^{\mathrm{A}}_{1,2} & \cdots & \varepsilon^{\mathrm{A}}_{1,n[v]} \\ \varepsilon^{\mathrm{A}}_{2,1} & \varepsilon^{\mathrm{A}}_{2,2} & \cdots & \varepsilon^{\mathrm{A}}_{2,n[v]} \\ \vdots & \vdots & \ddots & \vdots \\ \varepsilon^{\mathrm{A}}_{n[m],1} & \varepsilon^{\mathrm{A}}_{n[m],2} & \cdots & \varepsilon^{\mathrm{A}}_{n[m],n[v]} \end{bmatrix} \quad (9)$$

The demand, as the crucial component in the calibration, is represented with a vector $\boldsymbol{\delta}$. This vector is composed of time-dependent O-D matrices, as given by Eq. (**10**).

$$\boldsymbol{\delta}^{\mathrm{T}} = \begin{bmatrix} \mathbf{D}_1 & \mathbf{D}_2 & \cdots & \mathbf{D}_{n[t]} \end{bmatrix} \quad (10)$$

Elements of a vector $\boldsymbol{\delta}$ are matrices $\mathbf{D}_t$, where $t = 1, \ldots, n_t$, n_t being the number of temporal profiles, i.e. the time discretization of demand. Elements of a matrix $\mathbf{D}_t$ are dymanic O-D demand flows d, as given by Eq. (**11**). These are square matrices with a size of $n_z \times n_z$, n_z denoting the number of zones in the model.

$$\mathbf{D}_t = \begin{bmatrix} d_{1,1}^t & d_{1,2}^t & \cdots & d_{1,n[z]}^t \\ d_{2,1}^t & d_{2,2}^t & \cdots & d_{2,n[z]}^t \\ \vdots & \vdots & \ddots & \vdots \\ d_{n[z],1}^t & d_{n[z],2}^t & \cdots & d_{n[z],n[z]}^t \end{bmatrix} \tag{11}$$

In the model, demand components are represented with a vector **x**, which elements x_i are single demand components, with n_d being number of demand components, given by Eq. (**12**).

$$\mathbf{x}^{\mathrm{T}} = \begin{bmatrix} x_1 & x_2 & \cdots & x_{n[d]} \end{bmatrix} \tag{12}$$

To represent errors that come from traffic counts, measurement errors are also incorporated in the model and are represented with a vector $\boldsymbol{\varepsilon}^{\mathrm{M}}$, given by Eq. (**13**). Each element ε^{M} corresponds to a one measurement value y.

$$\left(\boldsymbol{\varepsilon}^{\mathrm{M}}\right)^{\mathrm{T}} = \begin{bmatrix} \varepsilon_1^{\mathrm{M}} & \varepsilon_2^{\mathrm{M}} & \cdots & \varepsilon_{n[m]}^{\mathrm{M}} \end{bmatrix} \tag{13}$$

At this point all the elements necessary to generate traffic counts are described. Traffic counts can be computed using the system of linear equations from Eq. (**14**).

$$\begin{aligned} y_1 &= a_{1,1} \cdot \varepsilon_{1,1}^{\mathrm{A}} \cdot x_1 + a_{1,2} \cdot \varepsilon_{1,2}^{\mathrm{A}} \cdot x_2 + \ldots + a_{1,n[v]} \cdot \varepsilon_{1,n[v]}^{\mathrm{A}} \cdot x_{n[v]} + \varepsilon_1^{\mathrm{M}} \\ y_2 &= a_{2,1} \cdot \varepsilon_{2,1}^{\mathrm{A}} \cdot x_1 + a_{2,2} \cdot \varepsilon_{2,2}^{\mathrm{A}} \cdot x_2 + \ldots + a_{2,n[v]} \cdot \varepsilon_{2,n[v]}^{\mathrm{A}} \cdot x_{n[v]} + \varepsilon_2^{\mathrm{M}} \\ &\vdots \\ y_{n[m]} &= a_{n[m],1} \cdot \varepsilon_{n[m],1}^{\mathrm{A}} \cdot x_1 + a_{n[m],2} \cdot \varepsilon_{n[m],2}^{\mathrm{A}} \cdot x_2 + \ldots + a_{n[m],n[v]} \cdot \varepsilon_{n[m],n[v]}^{\mathrm{A}} \cdot x_{n[v]} + \varepsilon_{n[m]}^{\mathrm{M}} \end{aligned} \tag{14}$$

To have a more general view, a generic traffic count y_i can be written as follows:

$$y_i = a_{i,1} \cdot \varepsilon_{i,1}^{\mathrm{A}} \cdot x_1 + a_{i,2} \cdot \varepsilon_{i,2}^{\mathrm{A}} \cdot x_2 + \ldots + a_{i,n[v]} \cdot \varepsilon_{i,n[v]}^{\mathrm{A}} \cdot x_{n[v]} + \varepsilon_i^{\mathrm{M}} \tag{15}$$

where $i = 1, \ldots, n_m$ and $j = 1, \ldots, n_v$. A traffic count is calculated as the sum of all the O-D flows passing over a specific detector during a specific time interval. The function describing the relationship between an O-D flow and detector (or link) can be linear, given by Eq. (**16**) and used in Eqs. (**14**) and (**15**) above, or any other (non-linear) function $f(x)$ describing their dependencies, as given by Eq. (**17**).

$$y_i = \sum_{j=1}^{n[v]} \left(a_{i,j} \cdot \varepsilon_{i,j}^{\mathrm{A}} \cdot x_j \right) + \varepsilon_i^{\mathrm{M}} \tag{16}$$

$$y_i = \sum_{j=1}^{n[v]} \left(a_{i,j} \cdot \varepsilon_{i,j}^{\mathrm{A}} \cdot f(x_j) \right) + \varepsilon_i^{\mathrm{M}} \tag{17}$$

where $i = 1, \ldots, n_m$ and $j = 1, \ldots, n_v$. On the other side, a function evaluation f_i is computed as an assignment of the demand to the network. Function evaluation consists of two parts: returned evaluated value $\hat{f}$ and model error ε^{F}. Model errors are incorporated here to represent approximations and inaccuracies that come from model simulations. They are given by Eq. (**18**).

$$\left(\boldsymbol{\varepsilon}^{\mathrm{F}}\right)^{\mathrm{T}} = \begin{bmatrix} \varepsilon_1^{\mathrm{F}} & \varepsilon_2^{\mathrm{F}} & \cdots & \varepsilon_{n[m]}^{\mathrm{F}} \end{bmatrix} \tag{18}$$

This way we are able to represent all the non-linearity that are embedded in a dynamic system as one we are dealing with. For all the links with count locations we apply the following:

$$f_i = \hat{f}_i + \varepsilon_i^{\mathrm{F}}, \ \forall i, \ i = 1, \ldots, n_m \tag{19}$$

Effectiveness and efficiency is assessed using the following goodness of fit measures and indicators such as:

- Normalized Root Mean Square Error (*RMSN*) – to track the objective function value.

$$RMSN = \frac{\sqrt{n_m \sum_{i=1}^{n[m]} \left[r_i (f_i - y_i)^2 \right]}}{\sum_{i=1}^{n[m]} y_i} \tag{20}$$

It incorporates the accuracy information of each measurement represented with a vector **r**.

- Number of algorithm iterations (n_{ite}) – to assess the importance of the modifications of a single iteration that each perturbation has and their influence on the convergence.

Assignment matrix fractions are initialized with a random number sampled from uniform distribution, $a_{i,j} = rand$, which is done for $\forall(i, j)$, where $i = 1,\ldots,n_m$ and $j = 1,\ldots,n_v$. Then a value of 0/1 is assigned based on a predefined *threshold*:

$$a_{i,j} = \begin{cases} 1, & rand \geq threshold \\ 0, & \text{otherwise} \end{cases} \tag{21}$$

Finally a random number is assigned where the value was 1:

$$a_{i,j} = \begin{cases} rand, & a_{i,j} = 1 \\ 0, & \text{otherwise} \end{cases} \tag{22}$$

Creating errors for assignment matrix, measurements and model

Any type of errors can be created in various ways. We adopt here a straightforward way where the errors are created using random numbers (ρ), noise (ν) and constants (α).

- Assignment matrix errors, calculated using Eq. (**23**) as a random noise around the assignment matrix fraction value, where $\nu \in [0,1]$.

$$\varepsilon_{i,j}^{\mathrm{A}} = a_{i,j} + \rho \cdot 2 \cdot \nu - \nu \tag{23}$$

which is calculated for $\forall(i, j)$, where $i = 1,\ldots,n_m$ and $j = 1,\ldots,n_v$.

- Measurement errors, calculated using Eq. (**24**) as a random noise around the constant value, where α should represent average link flow in veh/h for urban links and $\nu \in [0,1]$.

$$\varepsilon_{i}^{\mathrm{M}} = \alpha \cdot (\rho \cdot 2 \cdot \nu - \nu) \tag{24}$$

which is calculated for $\forall(i, j)$, where $i = 1,\ldots,n_m$ and $j = 1,\ldots,n_v$.

- Model errors, calculated using Eq. (**25**) as a random noise around the constant value, where α should represent average error in link flow value for urban links and $\nu \in [0,1]$.

$$\varepsilon_{i}^{\mathrm{F}} = \alpha \cdot (\rho \cdot 2 \cdot \nu - \nu) \tag{25}$$

which is calculated for $\forall(i, j)$, where $i = 1,\ldots,n_m$ and $j = 1,\ldots,n_v$.

4. Demand perturbation

Travel demand, as a critical part in mobility analysis, intrinsically contains spatial and temporal correlation among its components. In demand modelling and calibration, initial matrices provide us with starting demand structure. Taking this into consideration, we here test the hypothesis that O-D demand components, as they are not independent of each other, should be treated aggregately in the calibrating process. Instead of calibration every demand component individually, we here pursue an aggregate approach that does not require simultaneous change of all matrix entries separately, as it can have significant consequences on the convergence to a solution. Therefore following our approach single O-D pairs are not optimized individually but we rather adopt different levels of aggregation that are employed in calibration. A detailed list of all demand correlation objects on which the aggregation can be conducted is described below, starting from ‘aggregate time-constant’ demand components, as the highest level of aggregation, to ‘single time-dependent’ demand components, as the lowest level of aggregation in demand representation. Two or more of the correlation objects can be combined in a single calibration run, but the detailed tests and analyses are outside the scope of this paper.

- **Time-constant zone generation** – perturbation of the total origin demand flow of a zone as a result of the land use change, new building in the zone, new shopping mall, new car park, etc., or simply because of a poor estimation in the starting matrices. The temporal distribution of the demand inside the analysis period is kept unchanged. Total zone generation is changed and it proportionally scales all the time-dependent zone generations by the same rate. In Eq. (**26**) a multiplier m_o changes zone generation for all temporal profiles and towards all destinations:

$$d_{od}^{j} = m_o \cdot d_{od}^{j},\ \forall j,\ j = 1,...,n_t, \forall d \in D \tag{26}$$

which is valid for $\forall o \in O$, O being a set of origin zones and D being a set of destination zones.

- **Time-constant zone attraction** – perturbation of the total destination demand of a zone for the same reasons as for zone generation. Similarly as in the previous case, the temporal distribution of the demand inside the analysis period is kept unchanged. Total zone attraction is changed and it proportionally scales all the time-dependent zone attractions by the same rate. In Eq. (**27**) a multiplier m_d changes zone attraction for all temporal profiles and from all origins, which is valid for $\forall d \in D$:

$$d_{od}^{j} = m_d \cdot d_{od}^{j}, \forall j,\ j = 1,...,n_t, \forall o \in O \tag{27}$$

- **Temporal profile** – perturbation of the total demand departing at the same temporal profile because of the under-estimation or over-estimation of temporal distribution of the demand. It is also reasonable from technical point of view in cases when the matrices used in static assignment are used as staring dynamic demand matrices with denser time discretization, which is often the case. In Eq. (**28**) a multiplier m^j changes the whole demand of a temporal profile, which is valid for $\forall j,\ j = 1,...,n_t$:

$$d_{od}^{j} = m^{j} \cdot d_{od}^{j},\ \forall o \in O,\ \forall d \in D \tag{28}$$

- **Time-dependent zone generation** – perturbation of the time-dependent slices of a zone generation towards all destinations. This way the splitting shares towards destinations are kept fixed and proportionally changed. In Eq. (**29**) a multiplier m_o^j changes the whole zone generation of a temporal profile towards all destinations, which is valid for $\forall o \in O$ and for $\forall j,\ j = 1,...,n_t$:

$$d_{od}^{j} = m_o^{j} \cdot d_{od}^{j},\ \forall d \in D \tag{29}$$

- **Time-dependent zone attraction** – perturbation of the time-dependent slices of a zone attraction from all origins. Here the splitting shares from all origins are kept fixed and proportionally changed. In Eq. (**30**) a multiplier m_d^j changes the whole zone attraction of a temporal profile from all origins, which is valid for $\forall d \in D$ and for $\forall j,\ j = 1,...,n_t$:

$$d_{od}^{j} = m_d^{j} \cdot d_{od}^{j},\ \forall o \in O \tag{30}$$

- **Time-constant O-D pair** – perturbation of the demand of a single O-D component. The splitting shares regarding departure times remain fixed and all are proportionally scaled. In Eq. (**31**) a multiplier m_{od} changes the whole O-D demand, which is valid for $\forall (o,d)$ where $o \in O$ and $d \in D$:

$$d_{od}^{j} = m_{od} \cdot d_{od}^{j}, \ \forall j, j = 1,...,n_t \tag{31}$$

- **Time-dependent O-D pair** – perturbation of the demand of a time-dependent O-D component. This dynamic O-D pair is the smallest correlation object of demand on which the aggregation can be performed. This is valid for $\forall (o,d)$ where $o \in O$ and $d \in D$, and for $\forall j, \ j = 1,...,n_t$.

$$d_{od}^{j} = m_{od}^{j} \cdot d_{od}^{j} \tag{32}$$

In urban networks there is a very large number of O-D pairs and it is not possible to calibrate them individually neither effectively nor efficiently. There is also additional time requirement for an optimization algorithm to sample a lot of new points for every function evaluation. When multiplied with the number of function evaluations needed this can lead to the visible increase in the overall computation time. Up to now in the literature O-D pair calibration was the preferred way of calibrating demand. Because of the network structure, zoning system in urban context comprises much higher number of zones, where an increase in the number of zones is followed by quadratic growth in the number of O-D pairs. As our goal is to apply the methodology to large-scale urban networks, we test using aggregate approach in the calibration. Because of the stochasticity of demand and its individual or disaggregate nature, which the effect of its behavioral side, we assume the main reason why this approach has been used is because the demand is represented in the model in this way; theoretically only time-dependent O-D demand calibration can lead to precise results and global optimum.

5. Experimental results

We conducted several sets of tests to compare the performance of the standard SPSA algorithm (using single O-D components as parameters for calibration) and our SPSA method (using aggregate demand values with time-constant zone generation and time-constant zone attraction and correlation objects). The test network consists of 300 zones, which gives 90000 O-D components. A problem of this scale is very difficult to solve because of the non-convex response surface due to the non-linear correlations among demand and link flows. On the other side, the aggregate approach deals with 600 variables for calibration, which is twice the number of zones (origins plus destinations). True demand was generated in a way that flow measurements take values in the range [0, 4000] veh/h. Initial demand was a perturbation of the true demand vector with a certain distance level. Here we also included the correlation among demand components. For various correlation objects we included additional noise to model the correlation among O-D pairs that belong to the same object.

$$\tilde{x}_i = x_i + \rho \cdot \nu \cdot x_i - \frac{\nu \cdot x_i}{2}, \ \forall i, \ i = 1,...,n_d \tag{33}$$

The tests are conducted for several cases regarding different measurement errors. Results are shown in Figure 1. First, no measurement errors are considered in the generation of flow measurements. Here we clearly demonstrate the ability of our approach to converge to a good solution (with *RMSN* of less than 2%), with reasonable number of iterations. In the same test case, due to a lot of degrees of freedom, standard SPSA is unable to find good gradient direction to explore the search space effectively. The consequence is a very bad solution with *RMSN* equal to 25% (which is a decrease of only 5% of the starting value). Successive tests include measurement errors of different magnitude (e.g. 5%, 10% and 20%). All these cases confirm that SPSA with aggregate demand perturbation approach converge to a solution, although with higher *RMSN* (3%, 5% and 10% respectively), whereas standard SPSA continues to break.

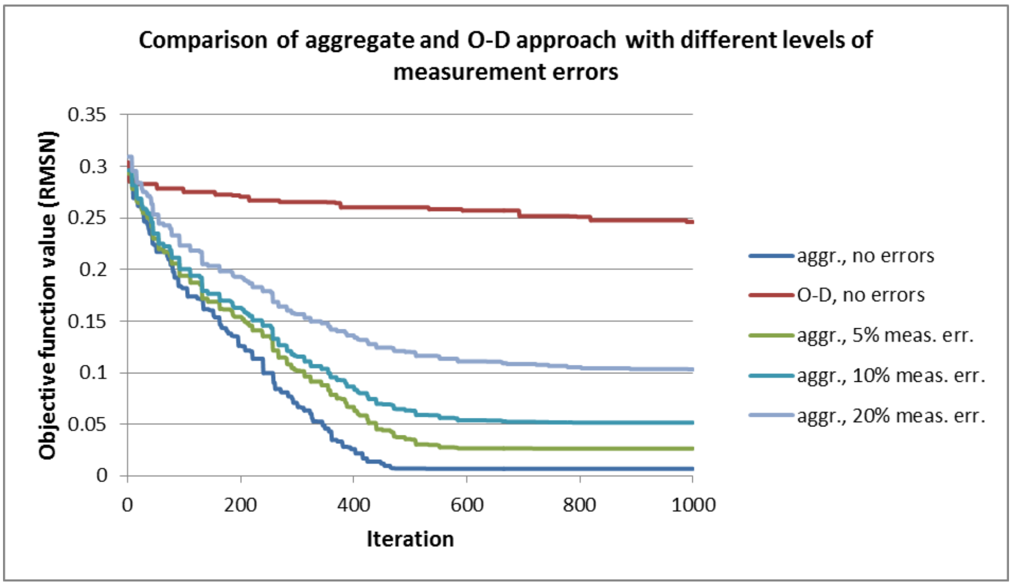

Figure 1. Comparison between two approaches for different levels of measurement errors

6. Conclusion

The contribution of this paper is that is provides a novel approach for demand calibration in DTA that allow for feasible application of SPSA to large-scale networks. We demonstrated on the synthetic model that SPSA using single demand components brakes in high-dimensional spaces. Using the aggregate approach we changed the scale of the problem and created setting where SPSA achieves good converging properties even in cases when measurement errors are involved. Beside small approximation it assumes, its simplicity has a number of advantages. First, the restriction of the search space (decreases degrees of freedom due to aggregation and lower number of variables) provides good response surface where SPSA can effectively search for global optimum. Then, faster calculation as it requires lesser internal computation cost of a single iteration. Finally, it is easy to implement and it does not require any additional preparation effort of the input data. Further research will involve real-world test cases, investigating the use of different aggregation schemes and their impact on the convergence.

Acknowledgments

The authors would like to express their gratitude to the COST Action TU1004 'TransITS' which supported the collaboration among the authors.

References

A. Tympakianaki, H. N. Koutsopoulos, and E. Jenelius, "c-SPSA: Cluster-wise simultaneous perturbation stochastic approximation algorithm and its application to dynamic origin–destination matrix estimation," Transp. Res. Part C, vol. 55, pp. 231–245, June 2015.

B. Kostic, G. Gentile, "Using Traffic Data of Various Types in the Estimation of Dynamic O-D Matrices," Proc. MT-ITS 2015, Budapest, Hungary, 2015.

E. Cipriani, A. Gemma, and M. Nigro, "A bi-level gradient approximation method for dynamic traffic demand estimation: sensitivity analysis and adaptive approach," Proc. IEEE ITSC 2013, The Hague, The Netherlands, 2013.

J. C. Spall, "Multivariate stochastic approximation using a simultaneous perturbation gradient approximation," IEEE Trans. Automat. Control, vol. 37 (3), pp. 332–341, 1992.

J. C. Spall, "Implementation of the simultaneous perturbation algorithm for stochastic optimization," IEEE Trans. Aerosp. Electron. Syst., vol. 34 (3), pp. 817–823, 1998.

L. Lu, Y. Xu, C. Antoniou, and M. Ben-Akiva, "An enhanced SPSA algorithm for the calibration of Dynamic Traffic Assignment models," Transp. Res. Part C, vol. 51, pp. 149–166, January 2015.