Downloaded from orbit.dtu.dk on: Apr 19, 2024

DTU Library

Efficient Transport Simulation With Restricted Batch-Mode Active Learning

Antunes, Francisco; Ribeiro, Bernardete; Pereira, Francisco C.; Gomes, Rui

Published in:
I E E E Transactions on Intelligent Transportation Systems

Link to article, DOI:
10.1109/TITS.2018.2842695

Publication date:
2018

Document Version
Peer reviewed version

Link back to DTU Orbit

Citation (APA):
Antunes, F., Ribeiro, B., Pereira, F. C., & Gomes, R. (2018). Efficient Transport Simulation With Restricted Batch-Mode Active Learning. *I E E E Transactions on Intelligent Transportation Systems*, *19*(11), 3642-3651. https://doi.org/10.1109/TITS.2018.2842695

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

Efficient Transport Simulation With Restricted Batch-Mode Active Learning

Francisco Antunes, Bernardete Ribeiro, *Senior Member, IEEE*, Francisco C. Pereira, *Member, IEEE*, and Rui Gomes

***Abstract*— Simulation modeling is a well-known and recurrent approach to study the performance of urban systems. Taking into account the recent and continuous transformations within increasingly complex and multidimensional cities, the use of simulation tools is, in many cases, the only feasible and reliable approach to analyze such dynamic systems. However, simulation models can become very time consuming when detailed input-space exploration is needed. To tackle this problem, simulation metamodels are often used to approximate the simulators' results. In this paper, we propose an active learning algorithm based on the Gaussian process (GP) framework that gathers the most informative simulation data points in batches, according to both their predictive variances and to the relative distance between them. This allows us to explore the simulators' input space with fewer data points and in parallel, and thus in a more efficient way, while avoiding computationally expensive simulation runs in the process. We take advantage of the closeness notion encoded into the GP to select batches of points in such a way that they do not belong to the same high-variance neighborhoods. In addition, we also suggest two simple and practical user-defined stopping criteria so that the iterative learning procedure can be fully automated. We illustrate this methodology using three experimental settings. The results show that the proposed methodology is able to improve the exploration efficiency of the simulation input space in comparison with non-restricted batch-mode active learning procedures.**

***Index Terms*— Active learning, transport simulation, simulation metamodels, Gaussian processes.**

I. Introduction

URBAN environments are highly complex systems involving a multitude of both internal and external variables, and their respective interrelationships, which are not often easy to identify. Additionally, these systems also exhibit an inherent stochastic nature and other unknown random phenomena that cannot be realistically described by a closed and tractable mathematical formula.

Manuscript received December 13, 2017; revised March 30, 2018; accepted May 15, 2018. This work was supported in part by Fundação para a Ciência e Tecnologia (FCT) in the framework of the MIT-Portugal Doctoral Program in Transportation Systems under Grant PD/BD/128047/2016 and in part by the EU's Horizon 2020 Research and Innovation Programme under the Marie Sklodowska-Curie Individual Fellowship H2020-MSCA-IF-2016 under Grant 745673. The Associate Editor for this paper was R. Nair. *(Corresponding author: Francisco Antunes.)*

F. Antunes, B. Ribeiro, and R. Gomes are with University of Coimbra, 3030-790 Coimbra, Portugal (e-mail: fnibau@dei.uc.pt).

F. C. Pereira is with Technical University of Denmark, 2800 Kongens Lyngby, Denmark.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TITS.2018.2842695

To successfully overcome the problem of intractability, simulation approaches are often employed to virtually explore the behavior of such urban systems and to assess their performances. Urban dynamics require theoretical approaches and planning methodologies that are capable of modeling the subjacent spatio-temporal transformations processes from a multidimentional prespective. In fact, urban planning models usually encompass the possibility of predicting future scenarios of urban intervention focused on infrastructure improvement and service promotion. These models are simplified representations of the urban space, embedded into a computer-generated reality, considered as an experimental ground to understand the long-term performance of urban policies decisions and corresponding interventions [1]. Nevertheless, when detailed with enough realism, urban simulation models can become computationally expensive to run due to their overwhelming complexity. Moreover, if the simulator output space proves to have a complex functional structure, we might need to systematically explore the input domain with further detail, which often requires multiple and exhausting simulation experiments, turning the exploration process virtually intractable.

To address the problem of expensive simulation runs, i.e., simulations that require great computational workload and exhibit prohibitive runtimes, simulation metamodels are often used to approximate the simulation results and thus the simulation model itself. Futhermore, in experimental scenarios in which simulation data is computationally expensive to obtain, active learning also emerges as a powerful approach, as it aims to provide high prediction performance with fewer data points. Among many significant machine learning tools, the Gaussian Processes (GP) [2], a fully Bayesian modeling approach, allow for an intuitive way to develop active learning algorithms, by providing the posterior mean and variance which in turn can be eventually used to search for the most informative data points.

In this paper, we propose an active learning approach, based on GPs, that gathers the most informative simulation data points in batches, not only according to their variance but also taking into account the relative distance between them. This allows us to explore the simulation input space with fewer training points in a faster, more efficient and parallel manner, while avoiding computationally expensive simulation runs at the same time. Taking advantage of the closeness and similarity notions encoded into the GPs, mainly via the kernel function, our approach selects batches of points

1524-9050 © 2018 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

that do not simultaneously belong to the same high variance neighborhoods. In addition, we also suggest two simple and practical user-defined stopping criteria so that the iterative learning procedure can be fully automated.

We illustrate our methodology using three independent settings, within a controlled experimental environment. The first consists of a synthetic data generated by a known function, which plays the role of an arbitrary simulation model. Then, we proceed to a one-dimension study of a Demand-Responsive Transportation (DRT) simulator. Finally, we explore the behavior of a road intersection implemented in a micro traffic simulation software, expanding our study to a two-dimensional input case. The obtained results show that the proposed batch-mode approach is able to improve the exploration efficiency of the simulation input space in comparison with non-restricted batch-mode active learning procedures.

The remainder of the paper is organized as follows. In the next section we provide a brief review on the main background topics and related literature. The proposed approach is detailed in Section III, followed by the presentation of studied simulation data, experiments and discussion (see Section IV). Then we proceed to the validation of the obtained results in Section V. Finally, we end this paper with some conclusions and possible lines of future work.

II. Background

Active learning is a special case of supervised machine learning consisting of an iterative sampling scheme that allows the algorithm to choose the data points from which it learns, and an oracle, i.e., an instance label provider. It is particularly useful under scenarios where labeled data is expensive to obtain. Thus, the general idea of this learning paradigm is to actively select the most informative data points, as few as possible, in order to simultaneously boost the model training efficiency and its prediction performance [3]. When the active learning paradigm was first proposed, the oracle was typically represented by a human annotator. Nowadays, however, due to the development of technology, the oracle's role can be taken by an algorithm, a sensor, a simulator, etc. Most importantly is that the oracle is able to provide labeled instances from the ground truth underlying function describing the process of interest.

Depending on how the unlabeled data is presented to the oracle, the active learning algorithms can be divided into two classes, namely, stream-based and pool-based. Whereas in the latter the entire unlabelled data set is available for querying, in the former each data point is presented individually or in sucessive blocks [4]. In addition, because of its intrinsic iterative nature, active learning procedures must be stopped at a certain time. Although there is a vast research under the active learning paradigm, not many approaches suggest a stopping criteria [5], [6], and, according to [4], there is no best stopping rule that is suitable across all applications.

In most of the proposed active learning algorithms, the queries are presented sequentially, i.e., one at a time. This strategy can become quite inefficient for some heavy learning tasks. One way to address this issue is to select data points in batches in order to speed-up the learning process by parallelizing it. However, to avoid redundancy within each batch, it should simultaneouly account for diversity and informativeness among the selected data points [4]. Several batch-mode schemes have addressed this challenge, most of them within classification problems [7]–[10].

Although active learning has been applied in many different fields, we are particularly interested in those of simulation metamodels [11] applications. Their main purpose is to serve as surrogates for simulation models so that expensive simulations can be avoided. These models are essentially simple functions that approximate the true and more complex unknown function inherently defined by the simulation model itself [12]. They are fitted to the input-output data generated by computer experiments and then can be used for prediction purposes, among others [13]. Hence, these simulation metamodels provide a practical framework to explore the behavior of complex and time-consuming simulation experiments in a rather less expensive way.

In our case, we assume that the simulation model is perfectly validated and calibrated with respect to the problem of interest. The metamodel, which should be a valid approximation of the simulation model, is then used to effortlessly explore its inner structure and, consequently, to provide yet another analysis tool for the problem under study. The Gaussian Processes (GP) framework has been widely used for simulation metamodelling [14]–[17], and other works combining GP metamodeling with active learning strategies have also been conducted [18], [19]. Its Bayesian formalism provides an easy way to develop algorithms that actively learn with uncertainty. Other machine learning techniques, such as Artificial Neural Networks (ANN), have also been employed [20], [21].

Within the transportation literature, the application of simulation metamodels is relatively recent. Only a handful of works is currently available and, with respect to their application domain, they can be essentially divided into two groups, namely, traffic prediction and network optimization [22]. In 2013, Ciuffo *et al*. [23] developed a methodology that applies a GP-based metamodel to conduct a sensitivity analysis using the mesoscopic traffic simulator AIMSUN as a case study. Using 512 different input combinations, the authors concluded that the metamodel estimated outputs and the real simulation outputs were significantly similar, thereby showing the strength and parsimony of their methodology. In [24] Zhang *et al.* developed a Bayesian stochastic Kriging metamodel that simultaneously optimizes travel behavior and dynamic traffic management using, as case study, the real-world corridors of I-270 and MD355 in the state of Maryland, USA. Using a similar case study, Chen *et al*. [25] used a GP metamodel to approximate the response surface of a transportation simulation with expensive-to-evaluate objective functions and random noise. The goal was to minimize the network-wide average travel time by implementing the optimal toll rates predicted by the metamodel. Similarly, in [26] a metamodel-based optimization framework was developed to solve the bilevel Mixed Network Design Problem (MNDP).

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

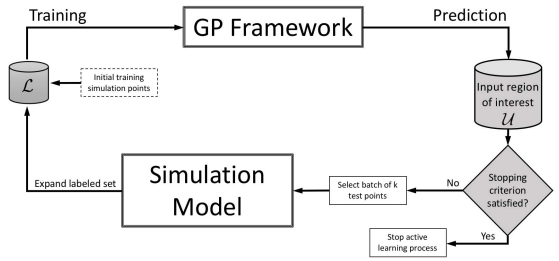

Fig. 1. Pool-based batch-mode active-learning scheme with a simulation model serving as the oracle.

A mesoscopic Dynamic Traffic Assignment (DTA) simulator, DTALite, was used to evaluate the system response to several network design strategies. The authors showed that the optimal investment could reduce the network average travel time in approximately 18% during the morning period. Finally, and very recently, Song *et al*. [22] presented a GP-based metamodeling framework that approximates Dynamic Network Loading (DNL) models. The authors show that the tested DNL metamodels attain high accuracy, providing prediction errors below 8%, and superior computational efficiency, up to 455 times faster than the traditional DNL approaches. Although these and other works constitute important applications of simulation metamodeling in transport problems, the research in this area is still scarce.

III. Approach

In this work, we adopt a pool-based batch-mode active learning approach in which a simulator plays the role of oracle, as depicted in Fig. 1. The machine learning model, used as metamodel, is a GP and the unlabeled pool, $\mathcal{U}$, is the input space where we want to explore the simulation behavior. The pool of labeled instances, $\mathcal{L}$, is formed by the results of the simulation runs already performed. From a regression perspective, we want to establish a functional relationship between the simulation inputs, $\mathbf{x} \in \mathbb{R}^D$, where D is the number of inputs, and its output, $y \in \mathbb{R}$, by assuming a GP prior over this relation, $y = f(\mathbf{x})$. Then, taking advantage from the Bayesian formalism from which the GP frameworks derives, we aim to infer the conditional distribution of the output given a set of unlabeled inputs. By doing so, we are not only bypassing the simulation computational workload but also providing a way to effortlessly approximate and therefore analyze the simulator behavioral structure. Notice that the simulation model is treated as a black box from which we aim to get better insights in terms of its functional behavior, while avoiding as many simulation runs as possible.

In the following we present the basis of Gaussian Processes and then we detail the proposed active learning procedure which is built upon this modeling framework.

A. Gaussian Processes

A GP [2] is a stochastic process completely characterized by a mean and a covariance function, respectively denoted as $m_f(\mathbf{x})$ and $k_f(\mathbf{x}, \mathbf{x}')$, with $\mathbf{x}$ and $\mathbf{x}'$ being two input data points and simply denoted by $\mathcal{GP}(m_f(\mathbf{x}), k_f(\mathbf{x}, \mathbf{x}'))$ where

Input: $\alpha_1, \alpha_2, \beta \in [0,1], k \in \mathbb{N}, \mathcal{L}, \mathcal{U}$.

While $\frac{ITV - CTV}{ITV} < \alpha_1$ **or** $\frac{AVT_r}{AVT_s} < \alpha_2$ **do**

1: Train a GP with $\mathcal{L}$ and obtain the predicted labels for each point in $\mathcal{U}$, $k_{f*}^\top [K_y]^{-1}\mathbf{y}$, and their corresponding variances, $k_{f**} - k_{f*}^\top [K_y]^{-1} k_{f*}$.
2: Determine the top k highest variance test points, top_k, so that the minimum distance between them is $\beta \times 100\%$ of the diameter of $\mathcal{U}$.
3: Obtain the true labels for top_k, via simulator, and define $\mathcal{L}_+$ as the new labeled set.
4: Set $\mathcal{L} = \mathcal{L} \cup \mathcal{L}_+$.
5: Update stopping criterion.

end

Fig. 2. General batch-mode active learning approach with spatial restriction.

$m_f(\mathbf{x}) = \mathbb{E}[f(\mathbf{x})]$ and $k_f(\mathbf{x}, \mathbf{x}') = \mathbb{E}[(f(\mathbf{x}) - m_f(\mathbf{x}))(f(\mathbf{x}') - m_f(\mathbf{x}'))]$. More formally, the GP framework assumes a prior over functions, i.e., $y = f(\mathbf{x}) + \epsilon$, where $\epsilon \sim \mathcal{N}(0, \sigma^2)$ and $f(\mathbf{x}) \sim \mathcal{GP}(m_f(\mathbf{x}), k_f(\mathbf{x}, \mathbf{x}'))$. For simplicity, it is common practice to fix $m_f(\mathbf{x}) = 0$. Thus, the prior over the latent function is given by $p(\mathbf{f}|\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n) = \mathcal{N}(0, K_f)$ where $\mathbf{f} = [f_1, f_2, \ldots, f_n]^\top$, $f_i \triangleq f(\mathbf{x}_i)$ and K_f is the covariance matrix, with its elements given by $[K_f]_{ij} = k_f(\mathbf{x}_i, \mathbf{x}_j)$.

Most of the covariance functions, or kernels, have several free parameters which can be optimized to fit the training data by maximizing the marginal likelihood. After these parameters are obtained, the conditional distribution at a new test point $\mathbf{x}_*$ is given by $f_*|X, \mathbf{y}, \mathbf{x}_* \sim \mathcal{N}(k_{f*}^\top [K_y]^{-1}\mathbf{y}, k_{f**} - k_{f*}^\top [K_y]^{-1} k_{f*})$, where $k_{f*} = k_f(X, \mathbf{x}_*)$, $k_{f**} = k_f(\mathbf{x}_*, \mathbf{x}_*)$ and $\mathbf{y}$ is the vector of the target values. Thus, we can naturally use the predicted Gaussian distribution at any given test point to guide the active learning process and therefore learn with its associated uncertainty.

B. Active Learning Procedure

The algorithm presented in Fig. 2 refers to the general batch-mode active learning procedure used in this work and it serves as the base for other algorithms forwardly presented and tested. Here, $\mathcal{L}$ represents the set of labeled training points, whereas $\mathcal{U}$ the set of unlabeled ones. The latter is defined according to input space region we aim to explore. This algorithm selects batches of k test points with the highest variance values (provided by the GP) in a way that each point does not originate from the same high variance neighborhood. Taking into account the spatial notion of closeness and similarity encoded into the GP via its kernel, it is expected that spatially closer input points are more likely to have similar output values. Therefore, the hypothesis is that sampling multiple batch points from the same region may not be efficient. To avoid this situation we introduce β to therefore control the minimum distance between the selected active learning points. This parameter is a ratio with respect to the maximum possible distance between any two points (diameter) of the input space. So, if $\beta = 0.4$, then the minimum distance between the points is 40% of the maximum distance. We call it space or spatial restriction induced by β. Notice that this approach is only valid for continuous input metric spaces. The parameter k defines the size of the batch.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

We also propose two simple variance-based stopping criteria controlled by α_1 and α_2, respectively. The first, which we call Criterion A, states that the algorithm stops when the total current variance (TCV) of the test points, at iteration i, is less than $(1-\alpha_1)\%$ of the initial total variance (ITV) at iteration 0. Thus, if, for example, $\alpha_1 = 0.3$, the process stops when TCV is reduced by 70% with respect to ITV, i.e., when $ITV(1-\alpha_1) \geq TCV$. Note that, instead of the total variance, which is simply the sum of the variances of all test points, we could have considered the average variance per iteration. However, since this criterion is defined as a ratio, the total number of test point would cancel out. On the other hand, Criterion B is defined as a ratio between the average variance of the training ($AVTr$) points and the average variance of the test ($AVTs$) points at each iteration i. Here, contrary to Criterion A, since the total number of training and testing points is not the same, it makes sense to consider the average. When $AVTr/AVTs \geq \alpha_2$, the algorithm stops. The average variance at training points is less than the average variance at testing points, so this ratio lies in $[0, 1]$. Moreover, as the process advances, $AVTs$ is likely to decrease, while $AVTr$ is expected to approximately maintain its values. However, this is a more demanding criterion to be satisfied if α_2 is close to 1. If the model, in our case the GP, is very certain at the training points and the contrary at the test points, $AVTr/AVTs \approx 0$, which will prevent the algorithm from converging at an acceptable speed. Its performance will also depend on the noise structure of the underlying function being estimated.

In each iteration a new GP model is fitted to $\mathcal{L}$. Its hyper-parameters are obtained by maximizing the log likelihood function conditional on this training set in a Leave-One-Out Cross-Validation (LOOCV) scheme. This constitutes a simple scheme to minimize potential overfitting problems during the training stage. Afterwards, the trained GP is used to predict the simulation output values (labels) associated to the unlabeled points in $\mathcal{U}$, therefore avoiding many simulations runs. Then, several testing points are selected according to the approach described in Fig. 2, their respective true labels are obtained via oracle, i.e., the simulator, and finally $\mathcal{L}$ is expanded. This iterative process is repeated until the chosen stopping criterion is satisfied.

Given the parametric nature of the proposed algorithm, many variants can be derived depending on the concrete values assigned to α_1, α_2, β and k, as well as on the used stopping criteria. We test the four following combinations which are built upon the base structure of this general approach:

- **Algorithm 1**: base approach + Criterion A with $\beta = 0$.
- **Algorithm 2**: base approach + Criterion A with $\beta > 0$.
- **Algorithm 3**: base approach + Criterion B with $\beta = 0$.
- **Algorithm 4**: base approach + Criterion B with $\beta > 0$.

The parameters used in these algorithms, α_1, α_2 and β, vary according to the different simulators in study and were obtained from a series of preliminary experiments using line search. The size of the batch was fixed at $k = 3$. Moreover, note that in Algorithms 1 and 3 we assume that $\beta = 0$, which means that there is no spatial restriction during the batch selection. Thus, these two cases correspond to the standard batch-mode schemes, serving as the baseline approaches in our comparative study. In the following, we present and discuss the results obtained within three experimental settings.

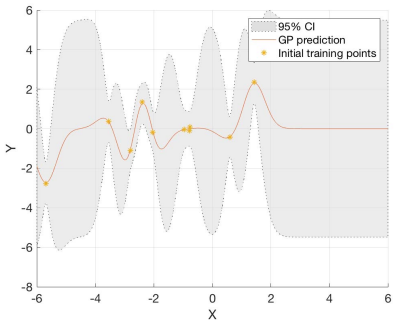

Fig. 3. Initial learning state for the toy data set, where the first 10 training points were randomly scattered in the input domain $[-6, 6]$. This corresponds to iteration 0, which is shared by the four algorithms in study.

IV. Experiments

In this section, we test the proposed methodology using three independent experimental settings. The first consists of a synthetic data generated by a known function playing the role of an arbitrary simulation model. Then, we proceed to a one-dimension study of a Demand-Responsive Transportation (DRT) simulator. Finally, we move to a two-dimensional study of a simple road intersection implemented in a free and open-source microtraffic simulation software. We use the GP implementation from [2] and select the Squared Exponential function as the GP kernel. Our study focus on the analysis of the number of iterations required for each algorithm to satisfy the stopping criteria, rather than on the observation of the corresponding CPU times, despite its great importance from a practical point-of-view. This allows us to conduct a more meaningful and hardware-independent performance analysis, making it possible to compare the results originated from different simulation experiments with different hardware specifications.

A. Toy Data Set

For the first set of experiments, we used a toy data set generated by $f(x) = cos(5x) + \frac{1}{2}x + 2\,U(0, 1)$, which plays the role of oracle, where $U(a, b)$ denotes the Uniform distribution in the interval $[a, b]$. Fig. 3 shows the initial GP learning state. The initial labeled pool, $\mathcal{L}$, is comprised of 10 randomly generated training points, $\mathcal{U}$ is formed by 10000 unlabeled test points uniformly spaced in $[-6, 6]$. Fig. 4 and 5 present the results.

Comparing Algorithms 1 and 2, the superiority of the latter is clear from an efficiency point-of-view (see Fig. 4). For the same criterion threshold ($\alpha_1 = 0.6$), Algorithm 2 is over than four times faster than Algorithm 1, due to the space restriction induced by $\beta = 0.2$. This means that the testing points constituting the batches were not selected from the same high variance neighborhoods, scattering the input exploration process and thus turning it more efficient. Moreover, it is important to note the number of iterations alone does not

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

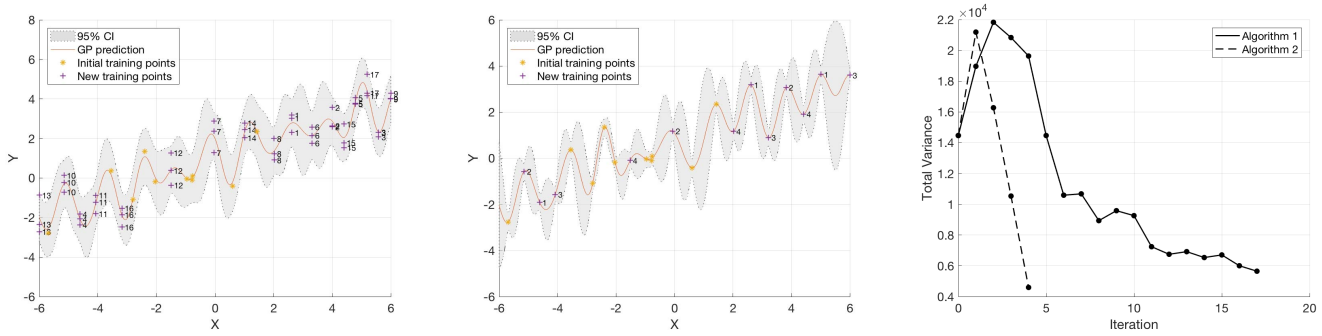

Fig. 4. Final results for the toy data set using (a) Algorithm 1 with user-defined parameters $\alpha_1 = 0.6$, $\beta = 0$ and $k = 3$, and (b) Algorithm 2 with $\alpha_1 = 0.6$, $\beta = 0.2$ and $k = 3$. The active points are labelled with the number of the iteration in which they were selected. Panel (c) compares the total variance reduction between both algorithms, displaying it as a function of the iteration number.

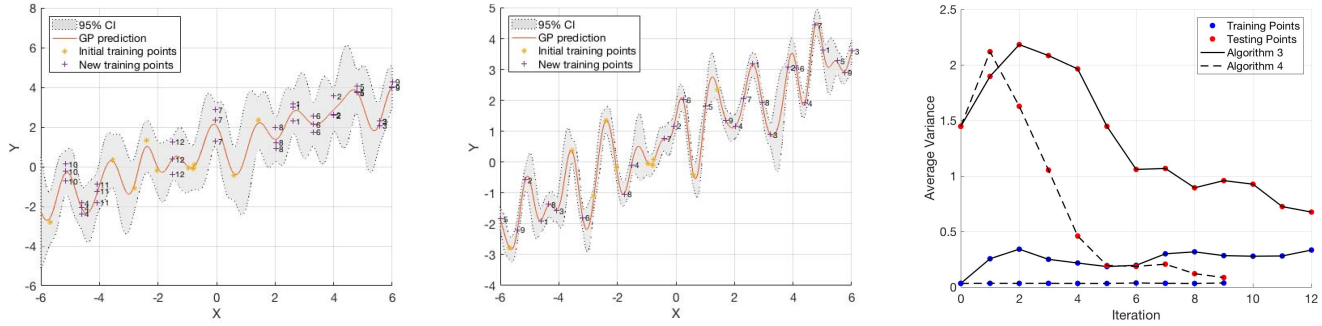

Fig. 5. Final results for the toy data set using (a) Algorithm 3 with user-defined parameters $\alpha_2 = 0.4$, $\beta = 0$ and $k = 3$, and (b) Algorithm 4 with $\alpha_2 = 0.4$, $\beta = 0.2$ and $k = 3$. The active points are labelled with the number of the iteration in which they were selected. Panel (c) compares the training and test average variance between both algorithms, displaying them as a function of the iteration number.

directly measure the real involved computational workload, which is intrinsically related with the simulation model. As we adopted a batch-mode active learning scheme, we should additionally take into account the size of the batch, k. Therefore, whereas Algorithm 1 requested the oracle $17 \times 3 = 51$ times, in Algorithm 2 this number decreased to $4 \times 3 = 12$.

For Algorithms 3 and 4, a similar scenario has occurred. Using the same stopping rule, based on Criterion B with $\alpha_2 = 0.4$, Algorithm 4 required three less iterations than Algorithm 3. This represents a total difference of $3 \times 3 = 9$ simulation requests. However, note that, as previously mentioned in Section III-B, Criterion B, may be harder to satisfy, which explains the lesser performance in comparison in both Algorithms 1 and 2.

B. DRT Simulator

Demand Responsive Transportation (DRT) systems are a kind of hybrid transportation approach between the taxi and bus solutions that address the problems that emerge from the use of fixed routes and schedules, typically found in regular public road transportation. From the transport operators' point-of-view, this traditional approach can prove to be quite expensive and inefficient in lower population density zones, such as rural areas, and in certain periods of the day. Both cases are characterized by a low, variable and unpredictable demand. Thus, DRT systems aim to provide transportation solutions that are able to adapt, in real-time, its routes and frequencies to match the actual observed demand.

Service design is critical for the success of DRT systems, so decision-makers need to understand well how the different ways of operating the service affect its performance. The flexibility of DRTs may cause organizational problems such as, a) the number and type of requests may involve an exceedingly high number of vehicles, b) very sparse requests that are hard to combine efficiently or c) the quality of the service in terms of pickup/delivery time and travel duration might not be guaranteed with the available resources or when unpredictable events occur. These effects are often studied through simulation, whose purpose is to obtain a better understanding of the behavior of a system under a given set of input conditions, even with uncertain events. The performance of these systems can be determined by observing what happens on the network, during simulation, for different input conditions. However, when dealing with real-world events (especially with high degree of dynamism) and extremely complex road networks and demand, simulation models can become very time-consuming.

We now consider the problem of exploring the outcome of the DRT simulation model developed by Gomes *et al*. [27].

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

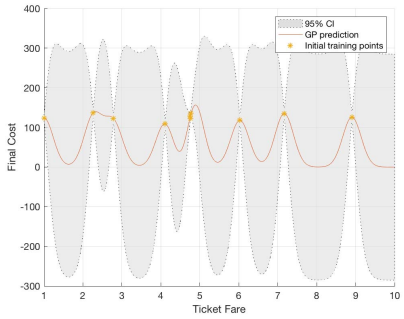

Fig. 6. Initial learning state for the DRT simulation data set, where the first 10 training points were randomly scattered in the input domain [1, 10]. This corresponds to iteration 0, which is shared by the four algorithms in study.

This simulation system integrates four submodels, covering the service area, trip requests (demand), vehicles, and real time events. It has 22 inputs/parameters and, among a few outputs that measure the system overall performance, we focus on the DRT system total operating cost. For a given demand structure, different input combinations will lead to different costs and service quality levels. We loaded the simulator with a real road-network structure, within the metropolitan region of Oporto (Portugal), with symbolic parameters values. For the sake of illustration, we then considered the Ticket Price as the input domain for which we aim to explore the outcome, Total Cost, of the simulator, maintaining the remaining inputs unchanged. From previous experiments, we concluded that, for our particular application, the simulation running times do not vary significantly from each other. Therefore, it does make sense to focus only on the number of iterations to assess the performances of the studied algorithms.

Similarly to the previous experimental setting, we decided to explore the input domain contained in the interval $[1, 10]$ using 10000 uniformly spaced test points. This test set corresponds to $\mathcal{U}$, whereas $\mathcal{L}$ is constituted by 10 random simulation data points within the same interval, as presented in Fig. 6. The results for this simulation data are generally aligned with the results obtained for the toy set experiment. Again, and as depicted in Fig. 7, Algorithm 2 is more than two times faster than Algorithm 1. For the same level of variance reduction, induced by $\alpha_1 = 0.60$ (about 40% of the initial total variance), the former requested the simulator $5 \times 3 = 15$ times, against $2 \times 3 = 6$ requests in the latter. It is obvious that the less runs the simulator executes the better. Therefore, for the same stopping criteria and threshold, Algorithm 2 proved to be more efficient as it was able to scatter the learning points forming the batch along different high variance neighborhoods.

For Algorithms 3 and 4 we obtained an interesting result which highlights our concerns presented in Section III-B regarding Criterion B. In Fig. 6 we can obverse that for the first approximation (Iteration 0), the GP model assigned very little variance to the training points, meaning that, in the context of Criterion B, $AVTr \approx 0$. As the process iteratively evolves, i.e., as more simulation data points are added to the expanding data set, $AVTr$ seems to remain close to zero. On the other hand, $AVTs$ clearly shows a decreasing behavior towards zero. In several preliminary experiments, which we do not present due to space restrictions, we noticed that setting, for example, $\alpha_2 = 0.6$, as we did for α_1, was too ambitious for both Algorithms 3 and 4 to yield competitive performance. This means that these algorithms were taking too many iterations to converge to be fairly comparable to Algorithms 1 and 2. Therefore, after fixing $\alpha_2 = 0.1$, which is equivalent to say that the stopping criterion is satisfied when $AVTr$ is approximately 10% of $AVTs$, we concluded that both Algorithms 3 and 4 converged in reasonable and comparable running times. Despite these initial configuration problems, these algorithms attained similar results to those of Algorithm 1 and 2. It is worthwhile to mention that, once again, the restriction applied during the formation of the batches was a decisive factor in the reduction of the number of iterations.

C. Traffic Simulator

In this section we move to a microscopic traffic simulation example, by exploring a road intersection implemented with the Simulation of Urban Mobility (SUMO) [28], in which the traffic flows in three directions only, North-South (NS), West-East (WE) and East-West (EW).[1] The vertical axis is dedicated to important vehicles, whereas in the horizontal axis we only have light passenger car traffic. Moreover, the model is designed to prioritize the NS traffic over the remaining flows. Therefore, it is expected that if this flow increases, the horizontal traffic flows will potentially form more and longer queues, consequently increasing the overall total waiting time.

During each simulation run, the demand, or traffic flow, generated from each operational axis is randomly generated according to a Poisson distribution, approximated by a Binomial distribution with parameter $p \in [0, 1]$. This parameter sets how many vehicles are generated, on average, within a certain period of time. For example, if $p = 1/s$, then it means that one vehicle is expected every interval of s seconds. Notice that the traffic flow actually increases when $s \to 0$.

The simulated example encompasses three input parameters that have a direct influence in the intersection performance, namely, the NS, WE and EW demands, each of which associated with different Binomial parameters. Moreover, it is assumed that the three different traffic flows are mutually independent. To assess the performance of the simulated road networks, SUMO has a large number of different output measures. Raw vehicle positions, trip and route information and simulation state statistics are just a few examples of possible outputs. For the sake of illustration of our methodology, we decided to focus on the total waiting time spent by all the vehicles crossing the intersection as our aggregated traffic performance measure. Our objective is to use the proposed active learning scheme to explore the simulation input space and to evaluate how it affects the total waiting time. Therefore, following a similar experimental design of the one-dimensional analyses presented in Sections IV-A and IV-B, we now extend our study to a two-dimensional case where the traffic demands

[1] See http://sumo.dlr.de/wiki/Tutorials/TraCI4Traffic_Lights

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

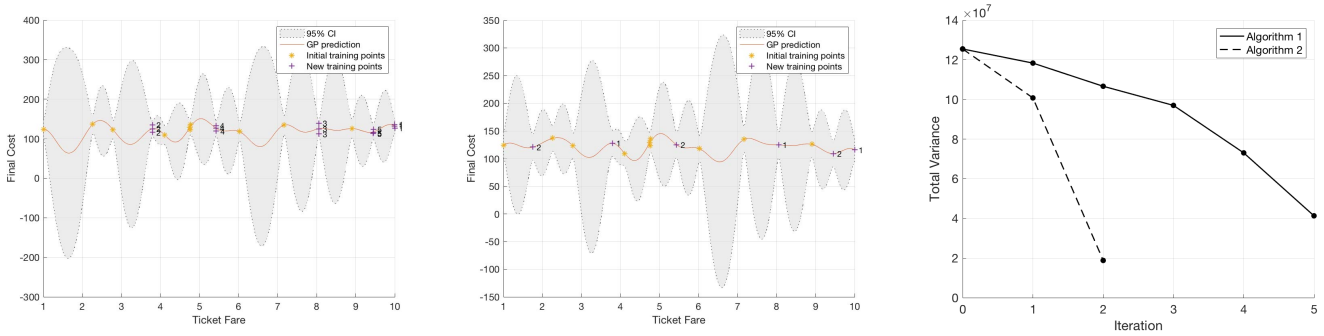

Fig. 7. Final results for the DRT simulation data using (a) Algorithm 1 with user-defined parameters $\alpha_1 = 0.6$, $\beta = 0$ and $k = 3$, and (b) Algorithm 2 with $\alpha_1 = 0.6$, $\beta = 0.2$ and $k = 3$. Each active point is labelled with the number of the iteration in which it was selected. Panel (c) compares the total variance reduction between both algorithms, displaying it as a function of the iteration number.

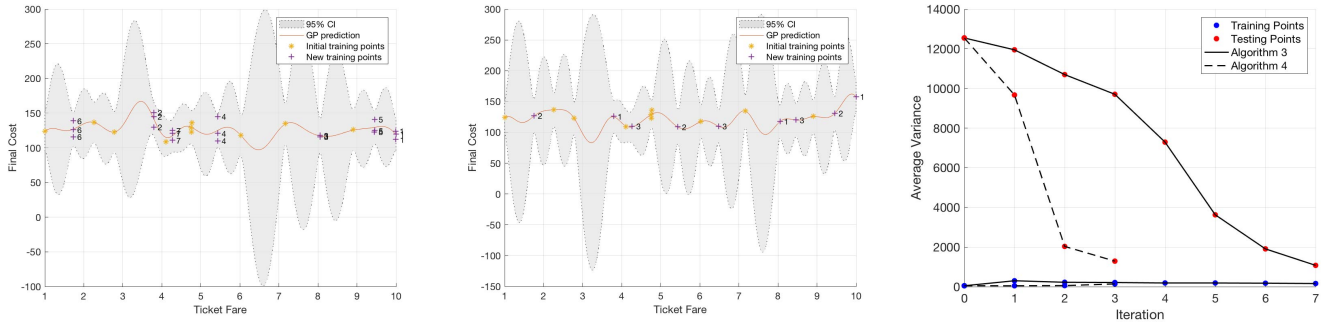

Fig. 8. Final results for the DRT simulation data using (a) Algorithm 3 with user-defined parameters $\alpha_2 = 0.1$, $\beta = 0$ and $k = 3$, and (b) Algorithm 4 with $\alpha_2 = 0.1$, $\beta = 0.2$ and $k = 3$. Each active point is labelled with the number of the iteration in which it was selected. Panel (c) compares the training and test average variance between both algorithms, dislplaying them as a function of the iteration number.

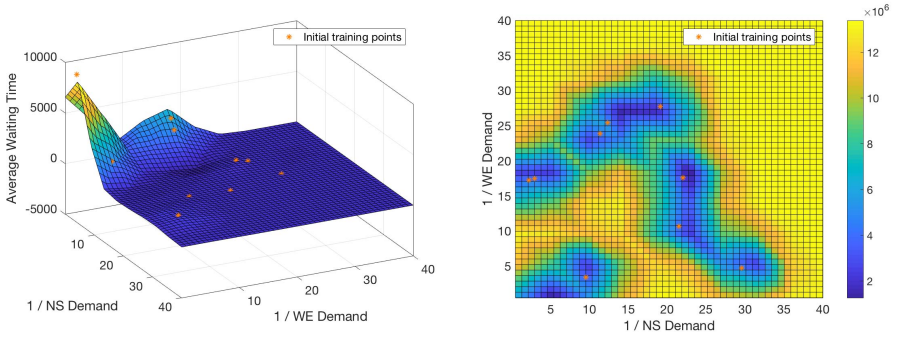

Fig. 9. (a) Initial learning state for the traffic simulation data, where the first 10 training points were randomly scattered in the input space $[0, 40] \times [0, 40]$. This corresponds to iteration 0, which is shared by the four algorithms in study. (b) Variance behavior across the input region.

from NS and WE operational axes are considered as inputs, and the expected vehicular waiting time is our output performance of interest.

The new input region of interest ($\mathcal{U}$) is defined by the square $[0, 40] \times [0, 40]$, from which 10 random training points were selected, corresponding to the initial set of simulation runs ($\mathcal{L}$), as depicted in Fig. 9(a). On the other hand, Fig. 9(b) shows the variance across the entire test region. As expected, the variance near the training points is lower than the variance associated to the test points. Starting from this initial learning stage, our approach is designed to actively search for the top k highest variance neighborhoods (yellow tones regions), that are not mutually within a radius of $\beta \times 100\%$ of the diameter of the input region. In any case, in this first approximation we can already observe that the values of the average waiting time (z-axis) tend to increase when both NS and WE demands increase, matching our initial guess regarding the simulation output behavior.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

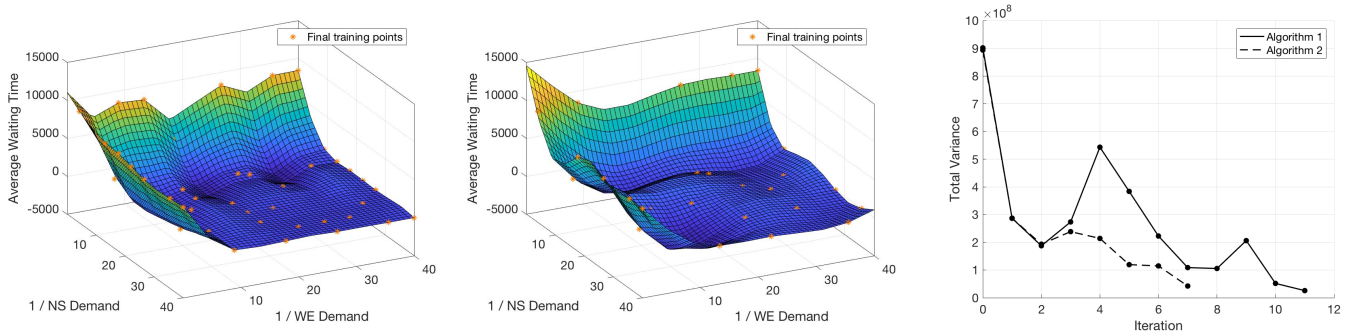

Fig. 10. Final results for the traffic simulation data using (a) Algorithm 1 with user-defined parameters $\alpha_1 = 0.95$, $\beta = 0$ and $k = 3$, and (b) Algorithm 2 with $\alpha_1 = 0.95$, $\beta = 0.3$ and $k = 3$. Panel (c) compares the training and test average variance between both algorithms, dislplaying them as a function of the iteration number.

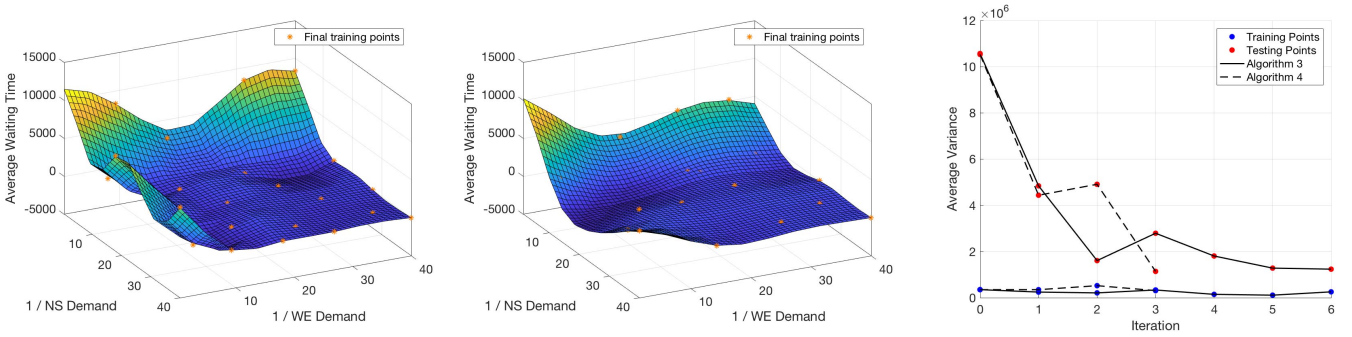

Fig. 11. Final results for the traffic simulation data using (a) Algorithm 3 with user-defined parameters $\alpha_2 = 0.2$, $\beta = 0$ and $k = 3$, and (b) Algorithm 4 with $\alpha_2 = 0.2$, $\beta = 0.2$ and $k = 3$. Panel (c) compares the training and test average variance between both algorithms, dislplaying them as a function of the iteration number.

TABLE I

AVERAGE RESULTS OBTAINED FROM 30 RANDOM COMPUTER RUNS USING 20 TEST POINTS FOR THE DIFFERENT EXPERIMENTAL SETTINGS AND ALGORITHMS

Data set	Algorithm	RMSE	MAE	RAE	RRSE	Cor.	Difference	p-value
Toy set	1	01.03	00.83	00.49	00.52	00.91	07.70	05.52E-15
	2	01.01	00.87	00.51	00.51	00.94		
	3	01.13	00.91	00.54	00.57	00.90	03.83	09.59E-12
	4	01.02	00.85	00.51	00.52	00.93		
DRT	1	04.13	02.85	00.55	00.65	00.84	05.93	06.34E-29
	2	04.14	02.89	00.55	00.65	00.77		
	3	03.84	02.29	00.44	00.60	00.81	08.60	01.43E-29
	4	03.70	02.73	00.52	00.58	00.81		
Traffic	1	2194.60	1366.68	00.30	00.44	00.91	05.21	02.89E-09
	2	2513.91	1498.23	00.33	00.51	00.89		
	3	1244.19	888.92	00.35	00.37	00.92	03.23	01.15E-08
	4	1405.02	939.51	00.37	00.41	00.91		

Fig. 10 and 11 present the final results, showing fairly identical GP approximations across the four algorithms. We observe that Algorithm 1 took 11 iteration to achieve a total variance reduction of 95%, against seven iterations from Algorithm 2 (see Fig. 10(c)). Both are based on Criterion A and on the same stopping threshold. However, due to the proposed space restriction (imposed by $\beta = 0.3$), the latter presents a more efficient performance than the former, with a difference of $(11 - 7) \times 3 = 12$ simulation runs. Finally, we can see from Fig. 11(c) that Algorithm 4 took three iterations to stop, whereas Algorithm 3, whose batch formation is not restricted by β, required six to satisfy Criterion B with $\alpha_2 = 0.2$. Although these differences in the iteration numbers may seem to be of little significance for the current academic example, they can prove to be quite relevant in real-world and thus computationally heavy simulation models, which can take up to several days to finish.

V. VALIDATION

We now conduct a series of computer runs to not only compare the algorithms' results but also to assess the quality of the obtained GP approximations with respect to the underlying simulation functions in study. These results are summarized in TABLE I.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

To evaluate the performance of the studied algorithms we used the Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), Root Relative Squared Error (RRSE) and Pearson's linear correlation coefficient (Cor.). Within each experimental setting, these metrics were computed by comparing the predicted values provided by the GP (in the last iteration of the algorithms) and the true simulation output values over a set of 20 test points scattered along the different studied input regions. We also obtained the average iteration difference between Algorithms 1 and 2 (Criterion A), and Algorithms 3 and 4 (Criterion B), respectively. Additionally, in order to evaluate the statistical significance of this difference, we conducted a paired sample t-test. Each computer run is determined by a different set of 10 random initial training points. The values for the parameters α_1, α_2 and β were maintained the same as in the previous section.

Overall, the results show that the GP was able to provide rather good approximations for the underlying simulation functions associated with the three experimental settings and across the four studied algorithms. However, the main difference resides in the average number of iterations required to satisfy the stopping criteria. Additionally, the reported p-values indicate that there are no evidences to support the acceptance of the null hypothesis, meaning that such iteration number differences seem to be statistically different from zero (criterion-wise). This shows that the introduction of the parameter β which induces a space restriction during the formation of the active point batches can improve the exploration performance while maintaining a reasonably good approximation to the underlying simulation function at the same time.

VI. Conclusion and Future Work

In this paper, we proposed a restricted batch-mode active learning approach, along with two practical user-defined stopping criteria, in the context of transport simulation metamodeling. The proposed algorithm seeks for the most informative test points in spatially restricted batches. The parameter β, which represents a fraction of the maximum possible distance (diameter) within the input region of interest, controls the minimum distance between each gathered point. This prevents each batch from being formed with points from the same high variance regions, making the learning process faster and parallelizable, and thus more efficient. Our objective is to obtain a reasonable understanding regarding the behavior of the simulator of interest with as few simulation runs as possible.

The results obtained from three independent experimental settings show that the introduction of a spatial restriction, induced by β, in the formation of the batch is able to turn the metamodeling process more efficient, while maintaining a good approximation to the underlying simulation functions at the same time. Additionally, we concluded that different data contexts and experimental settings require different parameter values configurations so that comparable results are attained.

There are several directions in which this work can be improved. The different values for the algorithms' parameters (α_1, α_2 and k) were obtained from a series of preliminary experiments and approximations using a simple line search, essentially to test the proposed active learning approach. Thus, we plan to develop proper strategies to fine tune these parameters, as well as to conduct the associated sensitivity analysis. Moreover, we intend to not only expand the current study with more parameter configurations but also to explore their relationship with the size, shape and dimensionality of the simulation input regions of interest.

On the other hand, the problem of metamodeling bounded simulation outputs, which is the case of the DRT simulator (costs should not be negative) used in this work, constitutes an interesting challenge to address in a near future, as well as the combination of both simulation and optimization metamodels using active learning strategies. Ultimately, we aim to apply our approach using a large-scale transportation simulation problem, whose runs can take up to days to finish, in order to assess its feasibility in comprehensive real-world applications.

Acknowledgment

The anonymous reviewers are also gratefully acknowledged for their interesting comments and suggestions.

References

[1] M. C. Marengo, "Urban simulation models: Contributions as analysis-methodology in a project of urban renewal," *Current Urban Stud.*, vol. 2, no. 3, p. 298, 2014.

[2] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning* (Adaptive Computation and Machine Learning). Cambridge, MA, USA: MIT Press, 2005.

[3] B. Settles, "Active learning literature survey," Comput. Sci., Univ. Wisconsin-Madison, Madison, WI, USA, Tech. Rep. 1648, 2010.

[4] X. Wang and J. Zhai, *Learning With Uncertainty*. Boca Raton, FL, USA: CRC Press, 2016.

[5] A. Vlachos, "A stopping criterion for active learning," *Comput. Speech Lang.*, vol. 22, no. 3, pp. 295–312, 2008.

[6] W. Wang, W. Cai, and Y. Zhang, "Stability-based stopping criterion for active learning," in *Proc. IEEE Int. Conf. Data Mining (ICDM)*, Dec. 2014, pp. 1019–1024.

[7] K. Brinker, "Incorporating diversity in active learning with support vector machines," in *Proc. ICML*, vol. 3, 2003, pp. 59–66.

[8] S. C. H. Hoi, R. Jin, and M. R. Lyu, "Large-scale text categorization by batch mode active learning," in *Proc. ACM 15th Int. Conf. World Wide Web*, 2006, pp. 633–642.

[9] Z. Xu, R. Akella, and Y. Zhang, "Incorporating diversity and density in active learning for relevance feedback," in *Proc. Eur. Conf. Inf. Retr.* Berlin, Germany: Springer-Verlag, 2007, pp. 246–257.

[10] Y. Guo and D. Schuurmans, "Discriminative batch mode active learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2008, pp. 593–600.

[11] L. W. Friedman, *The Simulation Metamodel*. Norwell, MA, USA: Kluwer, 1996. [Online]. Available: https://www.springer.com/la/book/9780792396482

[12] J. P. C. Kleijnen and W. C. M. van Beers, "Application-driven sequential designs for simulation experiments: Kriging metamodelling," *J. Oper. Res. Soc.*, vol. 55, no. 8, pp. 876–883, 2004.

[13] J. P. C. Kleijnen and R. G. Sargent, "A methodology for fitting and validating metamodels in simulation," *Eur. J. Oper. Res.*, vol. 120, no. 1, pp. 14–29, 2000.

[14] A. Boukouvalas, "Emulation of random output simulators," Ph.D. dissertation, Dept. Eng. Appl. Sci., Univ. Aston, Birmingham, U.K., 2010. [Online]. Available: http://www.aston.ac.uk/study/, https://research.aston.ac.uk/portal/en/theses/emulation-of-random-output-simulators(3a584d57-bad2-4a87-9e3f-b242f6b2b7ed).html, https://research.aston.ac.uk/portal/en/persons/alexios-boukouvalas(6fe997c9-b00f-48dc-b982-d5a0059ee290).html, http://www.aston.ac.uk/departments/, and http://www.aston.ac.uk/eas/

[15] T. Chen, K. Hadinoto, W. Yan, and Y. Ma, "Efficient meta-modelling of complex process simulations with time–space-dependent outputs," *Comput. Chem. Eng.*, vol. 35, no. 3, pp. 502–509, 2011.

This article has been accepted for inclusion in a future issue of this journal. Content is final as presented, with the exception of pagination.

[16] S. Conti and A. O'Hagan, "Bayesian emulation of complex multi-output and dynamic computer models," *J. Stat. Planning Inference*, vol. 140, no. 3, pp. 640–651, 2010.
[17] J. P. C. Kleijnen, "Kriging metamodeling in simulation: A review," *Eur. J. Oper. Res.*, vol. 192, no. 3, pp. 707–716, 2009.
[18] J. A. Christen and B. Sansó, "Advances in the sequential design of computer experiments based on active learning," *Commun. Stat.-Theory Methods*, vol. 40, no. 24, pp. 4467–4483, 2011.
[19] T. Pfingsten, "Bayesian active learning for sensitivity analysis," in *Proc. Eur. Conf. Mach. Learn.* Berlin, Germany: Springer, 2006, pp. 353–364.
[20] D. A. Cohn, "Neural network exploration using optimal experiment design," in *Proc. Adv. Neural Inf. Process. Syst.*, 1996, pp. 1071–1083.
[21] D. J. Fonseca, D. O. Navaresse, and G. P. Moynihan, "Simulation metamodeling through artificial neural networks," *Eng. Appl. Artif. Intell.*, vol. 16, no. 3, pp. 177–183, 2003.
[22] W. Song, K. Han, Y. Wang, T. Friesz, and E. del Castillo, "Statistical metamodeling of dynamic network loading," *Transp. Res. Procedia*, vol. 23, pp. 263–282, Jul. 2017. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352146517302934
[23] B. Ciuffo, J. Casas, M. Montanino, J. Perarnau, and V. Punzo, "Gaussian process metamodels for sensitivity analysis of traffic simulation models: Case study of AIMSUN mesoscopic model," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 23, pp. 263–282, Dec. 2013. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2352146517302934
[24] L. Zhang, X. He, C. Xiong, and Z. Zhu, "Bayesian stochastic Kriging metamodel for active traffic management of corridors," in *Proc. IIE Annu. Conf.* Georgia, U.K.: IISE, 2014, p. 1790.
[25] X. Chen, L. Zhang, X. He, C. Xiong, and Z. Li, "Surrogate-based optimization of expensive-to-evaluate objective for optimal highway toll charges in transportation network," *Comput.-Aided Civil Infrastruct. Eng.*, vol. 29, no. 5, pp. 359–381, 2014.
[26] X. Chen, Z. Zhu, X. He, and L. Zhang, "Surrogate-based optimization for solving a mixed integer network design problem," *Transp. Res. Rec., J. Transp. Res. Board*, vol. 2497, pp. 124–134, Dec. 2015. [Online]. Available: https://trrjournalonline.trb.org/doi/10.3141/2497-13
[27] R. Gomes, J. P. de Sousa, and T. Galvão, "An integrated approach for the design of demand responsive transportation services," in *Computer-Based Modelling and Optimization in Transportation*. Cham, Switzerland: Springer, 2014, pp. 223–235.
[28] D. Krajzewicz, J. Erdmann, M. Behrisch, and L. Bieker, "Recent development and applications of SUMO-simulation of urban mobility," *Int. J. Adv. Syst. Meas.*, vol. 5, nos. 3–4, pp. 128–138, 2012.

Francisco Antunes is currently pursuing the Ph.D. degree in transportation systems with University of Coimbra, within the framework of the MIT-Portugal Doctoral Program. He started his research activity in the Center for Informatics, University of Coimbra, in 2013. He is currently a member of the Research Centre for Territory, Transports and Environment. His main research interests include statistical models, intelligent transportation systems, machine learning, and transport/urban simulation models.

Bernardete Ribeiro (SM'15) received the Ph.D. and Habilitation degrees in informatics engineering from University of Coimbra. She is currently a Full Professor with University of Coimbra, also the Director of the Center of Informatics and Systems, University of Coimbra, also the President of the Portuguese Association of Pattern Recognition, and also the Founder and the Director of the Laboratory of Artificial Neural Networks for over 20 years. Her research interests are in the areas of machine learning, and pattern recognition and their applications to a broad range of fields. She has been responsible/participated in several research projects both in international and national levels in a wide range of application areas. She is an IEEE SMC Senior Member, a member of the IARP International Association of Pattern Recognition, a member of the International Neural Network Society, a member of the APCA Portuguese Association of Automatic Control, a member of the Portuguese Association for Artificial Intelligence, a member of the American Association for the Advancement of Science, and a member of the Association for Computing Machinery. She received several awards and recognitions.

Francisco C. Pereira (M'05) received the Ph.D. degree from University of Coimbra, Portugal, in 2005. He was a Senior Research Scientist with the MIT/CEE ITSLab, where he was involved in real-time traffic prediction, behavior modeling, and advanced data collection technologies, both in Boston and Singapore, as part of the Singapore-MIT Alliance for Research and Technology, Future Urban Mobility Project. He was also a Professor with University of Coimbra. He is currently a Full Professor with Technical University of Denmark, where he leads the Machine Learning for Mobility Research Group. He has authored or co-authored a long list of journal and conference papers in areas such as pattern recognition, transportation, knowledge-based systems, or cognitive science. His main research focus is on applying machine learning and pattern recognition to the context of transportation systems with the purpose of understanding and predicting mobility behavior, and modeling and optimizing the transportation system as a whole. The thesis relates Artificial Intelligence with Creativity and originated a book. In 2013, he was awarded the prestigious Singapore Challenge prize for a collaborative paper with researchers from A*Star, SUTD, and MIT.

Rui Gomes received the degree in informatics engineering, the M.Sc. degree in informatics engineering from the Faculty of Engineering, University of Porto (FEUP), and the Ph.D. degree in transport systems from the MIT-Portugal Program in 2013. He was a Visiting Graduate Student with the Massachusetts Institute of Technology in 2011 and an Invited Assistant Professor with FEUP from 2008 to 2012. He has been a teacher at several higher education institutions. He is currently a Senior Researcher and a Project Coordinator with the Centre for Informatics and Systems, University of Coimbra. He has authored or co-authored several book chapters, journal articles, and over 25 conference papers. His research interests include intelligent transport systems, demand responsive transport systems, combinatorial optimization, metaheuristics, ambient intelligence, and artificial intelligence. He has been a reviewer for several international scientific conferences and journals in the area of transportation, logistics, and artificial intelligence.