A Fast Multi-Objective Optimization Approach to Solve the Continuous Network Design Problem with Microscopic Simulation

Raphaël A. F. Lamotte

A Thesis

In

The Department

Of

Building, Civil and Environmental Engineering

Presented in Partial Fulfillment of the Requirements

for the Degree of Master of Applied Science (Civil Engineering) at

Concordia University,

Montréal, Québec, Canada

April 2014

CONCORDIA UNIVERSITY

School of Graduate Studies

This is to certify that the thesis prepared

By: Raphaël Ali Francis Lamotte

Entitled: A Fast Multi-Objective Optimization Approach to Solve the Continuous Network Design Problem with Microscopic Simulation

and submitted in partial fulfillment of the requirements for the degree of

Master of Applied Science (Civil Engineering)

complies with the regulations of this university and meets the accepted standards with respect to originality and quality.

Signed by the final examining committee:

_____ Chair

Amruthur S. Ramamurthy

_____ Examiner

Abdessamad Ben Hamza

_____ Examiner

Attila Zsaki

_____ Supervisor

Ciprian Alecsandru

Approved by _____

Maria Elektorowicz
Graduate Program Director

_____ 2014 _____

Christopher Trueman
Interim Dean of Engineering and
Computer Science

ABSTRACT

A Fast Multi-Objective Optimization Approach to Solve the Continuous Network Design Problem with Microscopic Simulation

Raphaël Ali Francis Lamotte

The capacity of microscopic traffic simulation to estimate the environmental and road safety impacts opens the possibility to address the Network Design Problem from a new multi-objective point of view. Computation time, however, has hindered the use of this tool. The aim of this thesis was to find a continuous optimization method that would require only a very limited number of evaluations, and thus reduce the computation time. For this purpose, the most recent optimization literature was studied and two algorithms were selected: PAL and SMS-EGO. Both these algorithms rely on Gaussian process meta-models, but they are distinct with respect to the assumptions, criteria and methods used. They were then compared on a real-world case-study with NSGA-II, a genetic algorithm considered as state-of-the-art. Within the very limited computational budget allowed, SMS-EGO was found to outperform PAL and NSGA-II in the three configurations studied. However, the computational time required was still too important to allow for large scale optimization. To further accelerate the optimization process, three main adjustments were proposed, based on variable noise modeling, gradient-based optimization and conditional updates of the meta-models. Considering 20 runs for each optimization process, only variable noise modeling exhibited a statistically significant positive impact. The two other modifications also accelerated the optimization process on average, but high variability in the results led to p-values in the order of 0.15. Overall, the proposed optimization methodology represents a useful tool for transportation researchers to solve multi-objective optimization problems of limited scale.

Acknowledgments

First and foremost, I would like to express my gratitude to my supervisor, Dr. Alecsandru, for his academic guidance, his trust, and for the independence he gave me.

Besides my supervisor, I would also like to thank the *Ministère des Transports du Québec* (MTQ) for partially supporting this work under the contract number R706.1, Dr. Miranda-Moreno for his precious advice, my colleagues and especially my friend Matin, for the continuous support he gives to everyone.

Finally, I would like to thank my family who supported me in my decision to do research, even in Canada, Julie and my friends who listened to me.

Table of Contents

Table of Figures

Table of Tables

Table of Abbreviations

ANN: Artificial Neural Network

CMA-ES: Covariance Matrix Adaptation Evolution Strategy

CMEM: Comprehensive Modal Emission Model

CNDP: Continuous Network Design Problem

DACE: Design and Analysis of Computer Experiments

EGO: Efficient Global Optimization

LHS: Latin Hypercube Sampling

MARS: Multivariate Adaptive Regression Splines

NDP: Network Design Problem

NSGA-II: Non-dominated sorting Genetic Algorithm – II

PAL: Pareto Active Learning

PTNDSP: Public Transit Network Design and Scheduling Problem

RBF: Radial Basis Functions

RNDP: Road Network Design Problem

RSM: Response-Surface Methodology

SMS-EGO: S-Measure Selection-based Efficient Global Optimization

SSAM: Surrogate-Safety Assessment Model

SVM: Support Vector Machines

SVR: Support Vector Regression

TSP: Transit Signal Priority

Special Symbols

$\mathbb{R}$: Set of all real numbers

S_{++}^{n}: Set of all matrixes of dimension $n \times n$ that are symmetric, definite, positive

$x \sim N(\mu, \sigma^2)$: The random variable x follows a Gaussian distribution with mean μ and variance σ^2

$E[x]$: Expected value of the random variable x

$cov(y, z)$: Covariance between the random variables y and z

$corr(y, z)$: Pearson's correlation coefficient between the random variables y and z

Introduction

A transportation network is the result of decisions taken at the strategic level (e.g. road building), at the tactical level (e.g. lane allocation) and at the operational level (e.g. scheduling traffic lights) [1]. All these decisions define a configuration of the network and to each configuration can be associated a performance in terms not only of efficiency, but also of cost, safety, and environment. Assuming that the performance of a configuration can be predicted, various criteria could be used to select the optimal configuration: the most efficient, the safest, the least expensive, or a "right balance" of these objectives. While the last criteria would be the best in theory, it is in practice difficult for decision makers to agree on appropriate weights. Consequently, since transportation authorities have historically been mostly concerned with operational aspects, a standard practice is to optimize the network with respect to a measure of efficiency (e.g. capacity or average travel time) and then to check that the system performs sufficiently well with respect to the other objectives. Thus, the priority is implicitly given to efficiency.

In this thesis, we take the opposite approach and choose multi-objective optimization. Instead of defining a set of preferences prior to the search of the optimal solution, we first search all the solutions that are optimal for at least one set of preferences (the Pareto front). With two objectives, this set of solutions is a (possibly discontinuous) curve connecting the optimum of each objective. With three objectives, it is a surface connecting the three optima. Assuming derivability, the trade-off at one point between two objectives is then given by the tangent at this point in the plan defined by the two objectives. Thus, by providing an insight of the underlying trade-offs, this approach helps understanding what is at stake in the transportation Network Design Problem (NDP).

Until now, several reasons have deterred transportation practitioners from using multi-objective optimization. First, transportation authorities were mainly interested in operational

aspects. Second, the traditional queuing models used to predict the operational performance were ill-adapted to estimate other objectives such as those related to environmental or road safety impacts. Finally, multiple-objective optimization is much more demanding in terms of computation time.

Nevertheless, transport authorities are now pressed to give more weight to environmental and safety issues. The first attempts to address these concerns were based on simple analytical models similar to the queuing models. However, these simple models turned out to be very limited since they cannot take into account all the variability between drivers for instance. Fortunately, computing facilities have greatly improved and have made possible the development of very-detailed microscopic simulation models, which can be used to estimate all sorts of objectives related to efficiency, pollution or safety. By including microscopic simulation models in the optimization framework, some authors have proposed innovative and promising multi-objective approaches of the NDP. Nevertheless, as the computation time required for these simulation models is much more important, the algorithms that were commonly used for multi-objective optimization now take days or sometimes weeks to find good results. Although this problem is relatively new in the transportation field, it has already been extensively studied in the optimization literature, where it is referred to as *expensive optimization of black-box processes*.

The aim of this thesis was to review this literature and identify the most suitable solution for the Continuous Network Design Problem, which is a certain type of NDP. In simple terms, solving a NDP is computationally-expensive when there is a wide range of possible configurations. If there are only a few configurations (for example with and without speed bumps), they can all be evaluated so there is no need for complex optimization algorithms. We are faced with a wide range of configurations when the number of decision variables (the dimensionality) exceeds the possible number of evaluations or when some decision variables can take more values than can be evaluated. Very often, the complexity stems from a combination of both. In this thesis, we have decided to focus only on reducing the computational cost associated

with continuous decision variables, which is an extreme case of variables that can take many values. With minor adaptations, the same approach could be used with benefit for discrete variables that can take many values.

A literature review was conducted and it was found that the most promising optimization methods are those based on meta-models, and especially those based on Gaussian processes. Two algorithms, previously never compared to each other nor applied to the NDP, were selected, tested on a very simple case-study and compared with the state-of-the-art. The algorithm that displayed the best performances was then further adapted to the NDP.

This thesis is organized as follows:

- In Chapter I, we introduce the NDP, the potential contribution of microscopic simulation and state the optimization problem to be addressed.
- In Chapter II, we provide some background on multi-objective and on expensive optimization. We then review the literature on expensive multi-objective optimization of black-box processes, introduce Gaussian process meta-models and identify two approaches that were found to perform extremely well in situations similar to the one studied in this work.
- In Chapter III, we test these two approaches on a simple case-study and compare them with NSGA-II, a commonly used state-of-the-art genetic algorithm.
- In Chapter IV, SMS-EGO is modified to take into account stochastic evaluations and the impact on the performance is analyzed.
- In Chapter V, the repartition of computation time among the different tasks is analyzed and some adjustments to accelerate the creation of the meta-models are proposed and tested.
- In Chapter VI, the computational complexity is evaluated and potential avenues to reduce it are identified.

Finally, we go over the different results obtained and draw some conclusions.

Chapter I Literature review and problem statement

I.1 The Network Design Problem

I.1.1 Diverse applications

The NDP consists in helping the decision-makers to choose the best solution among a set of possible designs for a transportation network. However, many different classes of NDP have been defined in the past.

The type of transportation network is probably the most widely used classification criterion. In fact, the Road Network Design Problem (RNDP) and the Public Transit Network Design and Scheduling Problem (PTNDSP) are almost always treated separately. Distinct literature reviews can be found on these two problems, with for instance Yang and Bell [2] for the RNDP and Guihaire and Hao [3] or Kepaptsoglou and Karlaftis for the PTNDSP [4]. In addition, many authors focused on even more specific problems such as pricing problems [5] or traffic signal setting problems [6]. However, Magnanti and Wong [1] and Farahni et al. [7] showed that the same solutions can often be used for different types of problems and that each individual problem could benefit from a more global approach.

Another very common classification criterion is the decision level (strategic, tactical or operational). Indeed, most authors implicitly consider only one of these levels because the choices available at the tactical or operational level often depend on the decision taken at the higher level(s). For example, in their review of urban transportation network design problems, Farahani et al. [7] focused only on the strategic level and on tactical decisions related to the network topology. Nevertheless, Magnanti and Wong [1] showed that the methods used at the different decision levels are often the same.

Other criteria include the number of transportation modes that are modeled (one or several) and whether the model is time-dependent or not. For all of these models however, the optimization methods that can be used remain very similar [7].

I.1.2 Common characteristics

As highlighted by the remarks of Magnanti and Wong [1] or Farahani et al. [7], the different classes of NDP have much in common from the optimization point of view. Indeed, for all of these types of NDP:

- The real-life problem is stochastic because of many stochastic inputs (demand, driving behavior, environment, etc.).
- They have similar objectives (e.g. minimizing delay, accidents, or emissions).
- They often include both discrete and continuous decision variables.
- They can be considered as bi-level problems [7]. Indeed, while traffic authorities try to optimize some objectives at the network level, every user of the network also tries to optimize his/her own trip. Thus, when optimizing the network configuration (the upper-level part of the problem), the traffic authorities must predict the response of all users to new configurations (the lower-level part of the problem). Usually, solving the lower level part is done according to Wardrop's first principle that states that each user will minimize his own cost function (e.g. transportation time), without any cooperation between users [8].
- The real-life process can rarely be used to evaluate the objectives and approximate models have to be used in the optimization problem.

Thus, fundamentally, the NDP is a bi-level Mixed-Integer Multi-Objective Optimization problem with stochastic objective functions. In practice however, transportation practitioners almost always address a simpler version of the NDP, which depends mostly on the model selected to approximate the real world process. For instance, some microscopic simulation packages now include dynamic traffic assignment modules that automatically solve the lower-level part of the NDP. Thus, the NDP can be reduced to its upper-level.

I.2 Benefits of micro-simulation for the NDP and its limitations

As mentioned previously, microscopic simulation can be used to solve the lower-level part of the NDP. However, it also has important advantages in terms of accuracy and diversity of the objectives that can be evaluated.

I.2.1 More realistic models

As opposed to macroscopic simulation that is based on aggregated data, microscopic simulation models each user of the network separately (e.g. vehicles, pedestrians, bicycles, etc.). To account for the natural variations that are observed in the real world between different network users, their characteristics are attributed stochastically, according to observed real-world distributions. Besides, simulating individual network users also means simulating their interactions among themselves, with the infrastructure (e.g. pavement markings) and with traffic control devices (e.g. traffic signs and signals). Similar to other microscopic traffic flow simulators, the model chosen in this thesis, PTV VISSIM [9], addresses these issues separately with many different sub-models controlling for instance the car-following behavior, the lateral movements and the general tactical driving behavior [10].

The potential applications of microscopic simulation are numerous but before everything else, they are predominantly used to analyze what has traditionally been the first concern of transportation engineers: operational efficiency. While analytical queuing models are used for prediction of waiting times and queue lengths, microscopic simulation allows traffic engineers to test the effects of advanced measures such as adaptive traffic signal control, transit priority schemes, reserved lanes, etc. Due to its higher computational cost, microscopic modeling has mostly been used for analysis purposes but recent studies have endeavored to include it in network optimization. For instance, Osorio [11] used a traffic flow simulator to mitigate congestion in a network while Stevanovik et al. [12] and Robles [13] optimized traffic signals settings relying on VISSIM micro-simulations.

Even though this thesis is more focused on optimization using microscopic simulation, the optimization issues are very similar for the optimization of larger networks based on complex macro-simulation models. For instance, Fikse [14], who used a dynamic traffic assignment software to assess the effect of different dynamic traffic management schemes, also encountered problems related to high computation time.

I.2.2 New objectives

Thanks to a more realistic modeling, microscopic simulation also allows research practitioners to include a wide range of objective functions in the NDP.

I.2.2.1 Pollution

Researchers have endeavored to include pollution in the NDP for a long time. Even before the development of detailed and efficient microscopic simulation software, some authors developed analytical models to predict emissions depending on the speed [15, 16]. However, these macroscopic models were intrinsically limited since they could not take into account variations in the accelerations, decelerations and idling times caused by changes in the network.

Alternative simulation-based approaches have been proposed to address these limitations. For instance, Robles [13] coupled the two emission models CMEM and VT-Micro with VISSIM to optimize traffic signals with respect to both efficiency and emissions. Similarly, Stevanovik et al. [17] used VISSIM and CMEM to optimize fuel consumption and CO_2 emissions. Wismans et al. [18] dynamically optimized traffic management with respect to several objectives, including a measure of the emissions that was obtained using a model based on the traffic flow simulator ARTEMIS.

I.2.2.2 Safety

Road safety is one of the most important objectives of traffic engineers. However, difficulties in predicting the frequency and severity of crashes have long prevented engineers from optimizing transportation networks with regard to these objectives. Indeed, when no previous studies allowed engineers to estimate the safety of a design, they had to wait until a significant amount of crashes

had happened to assess its performance. In order to permit a faster reaction, surrogate safety measures such as conflict analysis have been proposed (see for instance Perkins and Harris [19]). Even though no strong correlation between conflicts and crashes has been established yet [20], conflict analysis not only yields similar results to previously used prediction techniques, but also provides a better understanding of the reasons leading to crashes.

With the recent improvement of microscopic simulation capacities, Gettman and Head [21] proposed to assess the safety of a design *in advance*, by extracting the frequency and severity of conflicts from the modeled vehicle trajectories. Gettman et al. [22] developed a corresponding software called Surrogate Safety Assessment Model (SSAM), that is compatible with four major microscopic simulation packages.

Stevanovic et al. [12] integrated VISSIM and SSAM in a multi-objective Genetic-Algorithm based optimization framework and obtained very promising results. However, computation time remained a serious issue. Indeed, according to the authors, "optimization experiments took several months to complete" [12].

I.2.3 Limitations

Despite the many benefits of microscopic simulation, it also has a few drawbacks, starting with its computation time. As explained previously, microscopic simulation models every network user independently, which is an extremely resource-consuming task. While this complexity may sometimes be necessary, the cost in terms of computation-time is prohibitive in many applications that focus on wide areas with many users.

Another limitation concerns the nature of variables that can be studied. Indeed, since the distribution of speeds has to be specified by the modeler, microscopic simulation cannot be used to assess for instance the effect of geometric design on speed. However, assuming that the effect of a geometric design on driver behavior has been studied beforehand and that the micro-simulator has been properly calibrated, the effect of a design on traffic flow as a whole can be evaluated.

Last but not least, calibration is another important limitation of microscopic simulation. In fact, the estimations of the objective functions that can be obtained with microscopic simulation are only as accurate as the model used to obtain them. While analytic models typically require a reduced and basic representation of the network and a unique driving behavior, microscopic simulation requires a more detailed representation and a statistical description of driving behaviors. If this is not done, the results provided by microscopic simulation will not be more accurate than those obtained by an analytical evaluation of the network even though they require more computation time. Thus, microscopic simulation has open new possibilities, but to fully exploit them, an additional calibration and computation work is required.

I.3 Associated optimization problem

I.3.1 Characteristics of the problem studied

As mentioned earlier, the NDP is a very complex problem that has to be reduced and approximated to be solved. Depending on these approximations, very different optimization problems can emerge. However, the choice to use microscopic simulation naturally leads to a more specific problem.

On the one hand, using microscopic simulation to evaluate the objective functions means that the analytic expressions of the objectives are not known (i.e. the objective functions are considered as *black-boxes*). In fact, if we were able to predict objective functions analytically, there would be no need to run a microscopic simulation. Furthermore, the evaluations are necessarily stochastic and computationally expensive. On the other hand, microscopic simulation allows transportation practitioners to consider multiple objectives and to address only the upper level of the NDP.

Finally, the only decision to be taken is the type of decision variables considered. Here, for simplification purposes, we chose to focus on the Continuous Network Design Problem (CNDP), which only handles continuous decision variables.

I.3.2 Examples of CNDP applications

"Street capacity" is historically one of the most widely used decision variable [15, 16, 23] in CNDP but it is also maybe the most debatable continuous variable. It was typically related to the objective functions with simple models such as the one proposed by the US Bureau of Public Roads [24]:

$$T_i(x_i) = a_i \left[1 + 0.15 \left(\frac{x_i}{c_i}\right)^4\right] \quad \textbf{(1)}$$

In this equation, T_i represents the travel time per unit of flow on the link *i,* a_i is the mean free flow travel time on link *i* and c_i the capacity of link *i.*

In order to modify the capacity, traffic authorities could implement different strategies, such as lane widening or resurfacing. Based on real-life observations of the costs of such measures, simple models between the capacity improvement and the investment in dollars could be obtained. Thus, a typical problem was to determine for a given budget the road capacities that maximize the efficiency of the network.

Other examples of variables considered in the CNDP include toll pricing [5], optimal speed setting [14], or public transit frequency [25]. The Traffic signal setting problem also includes many continuous variables [12, 13, 26–28].

I.4 Conclusion of the chapter

To conclude this chapter, the NDP encompasses many different practical situations. However, from an optimization point of view, the mathematical characteristics of these problems do not depend on the applications but on the assumptions chosen to model the network, on the nature of decision variables and the number of objectives considered.

The use of microscopic simulation within the NDP brings two major benefits: a better accuracy and the possibility to optimize the network with respect to a wide variety of objectives. However, these advantages have a cost: the objective functions are stochastic, resource-

consuming and have to be considered as black-boxes. Thus, the choice of microscopic simulation leads to a much more specific optimization problem, for which no satisfying optimization methodology has been identified so far. The aim of this thesis is to find such a method for the case of continuous decision variables. In Chapter II, the optimization literature is reviewed for such methods.

Chapter II Background: expensive multi-objective optimization

This chapter presents two specific types of optimization problems: multi-objective optimization and expensive optimization. To be consistent with the literature and to ease the understanding, we first introduce these two types of problems separately before explaining how their different methods can be combined for multi-objective expensive optimization. Last, we give some background information on the approach selected in this thesis.

II.1 Multiple-objective optimization

II.1.1 Single/multi-objective optimization

On the one hand, the aim of a regular single-objective optimization problem is to determine the input values that minimize or maximize a single objective function. In mathematical terms, the objective space is unidimensional. On the other hand, multi-objective optimization aims at optimizing two or more objective functions.

Optimization in a unidimensional objective space such as the set of all real numbers $\mathbb{R}$ is relatively straight-forward since two solutions can always be compared, i.e. $\forall a, b \in \mathbb{R}, a \leq b \; or \; b \leq a$ (total order). Thus, there can be at most one minimum or maximum in the objective space (although it can be reached from several points in the design space). In most real-world situations, there is no such relationship for multi-objective optimization. Instead, the concepts of weak and strong Pareto dominance are used:

$Let\ X(x_1, x_2, \dots x_n)\ and\ \ Y(y_1, y_2, \dots, y_n) \in \mathbb{R}^n\ be\ two\ sets\ of\ solutions$

$X > Y\ iff\ \forall\ i\ \in \{1 \dots n\}, \qquad x_i > y_i\ (strong\ Pareto\ dominance)$

$X \geq Y\ iff\ \forall\ i\ \in \{1 \dots n\}, \qquad x_i \geq y_i\ and\ \exists\ i\ \in \{1 \dots n\},$

$$x_i > y_i\ (weak\ Pareto\ dominance)$$

In this thesis, we will consider only the concept of weak Pareto dominance. Indeed, if for instance, we try to optimize the safety and efficiency of a transportation network: between two configurations that yield the same efficiency but different safety performances, we are only

interested in the safer of these two. However, if one configuration is safer while the other one has a better efficiency, both situations might be acceptable, depending on the relative importance of safety and efficiency. Thus, there may be several solutions that are not weakly Pareto dominated by any other solution. These solutions are called *non-inferior* or *Pareto-optimal* and the set of all these solutions is called the *Pareto front* or *Pareto frontier*. Figure 1 represents graphically a hypothetical Pareto front for the minimization of two objective functions F_1 and F_2.

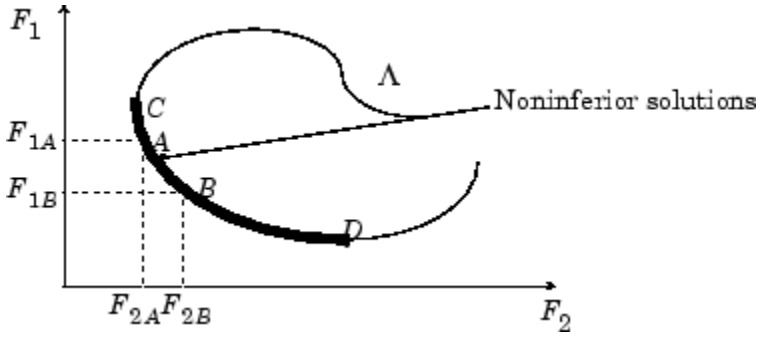

Figure 1: Illustration of the Pareto Front [29]

There are two main approaches to solve multi-objective optimization problems. A first simple approach, called *scalarizing*, consists in converting all the objectives into a single-objective function, for example by using a linear combination. However, this solution assumes a priori information about the relative importance of each objective. The second approach, chosen in this thesis, aims at approximating the complete Pareto front. Although more computationally expensive, it allows for the a posteriori articulation of the preferences, based on the knowledge of the existing trade-offs.

The main drawback of this approach is its cost in terms of computation time. For instance, a basic method to approximate the Pareto Front is to run many single-objective optimizations with varying weights on the different objectives. Another approach consists in approximating the complete Pareto Front with a single run of a Genetic Algorithm such as NSGA-II [30]. Despite being more efficient than the first approach, genetic algorithms-based solutions still require many thousands of evaluations of the objective functions [30]. Thus, for

simulation-based optimization of transportation networks, reducing the number of evaluations is paramount.

II.1.2 Note on the mathematical significance of the solutions found

In this thesis, we are looking for "good" configurations of the transportation network. Of course, we would like these configurations to be the best possible. However, since we are considering the objective functions as black-boxes, it is not even possible to guarantee the existence of such solutions. For simplification purposes, let us temporarily limit ourselves to the single-objective case. In order to guarantee the existence of a minimum and maximum, more information on the objective functions is required. For instance, since the design space is a Cartesian product of intervals of $\mathbb{R}$ that are closed and bounded, proving the continuity of the objective functions would prove the existence of extrema, according to the Extreme Value Theorem. However, without any additional information on the objective function, we cannot theoretically guarantee that extrema can be reached from the design space.

Besides, even if continuity was guaranteed, there would be no way to identify an extremum with a finite number of evaluations without the guarantee of any additional property, such as Lipschitz continuity. Thus, the output of black-box optimization algorithms are always the best configurations *among* the points evaluated, and they are only *approximations* of potential global extrema. However, keeping in mind these considerations, one can compare different optimization techniques by comparing the best solutions found.

II.2 Expensive optimization

The optimization of objective functions whose evaluations require significant resources either financially (e.g. crash-tests) or time-wise (e.g. complex simulation) is referred to as *expensive optimization*. This type of problem has been encountered in many fields, including for instance hardware design [31], industrial engineering [32], and biochemistry [33].

II.2.1 Meta-modeling

A popular approach to expensive optimization is to use the knowledge acquired from previous evaluations to predict the answer of the model to unevaluated configurations and thus guide the choice of the next configuration to be evaluated. This requires the creation of a *surrogate model* or *meta-model*, that imitates the real model but is less expensive to evaluate. In many algorithms, it also requires a function called the *figure of merit* that expresses our interest for each individual point of the decision space based on the meta-model. The general framework of an optimization strategy based on meta-modeling is schematically described in Figure 2.

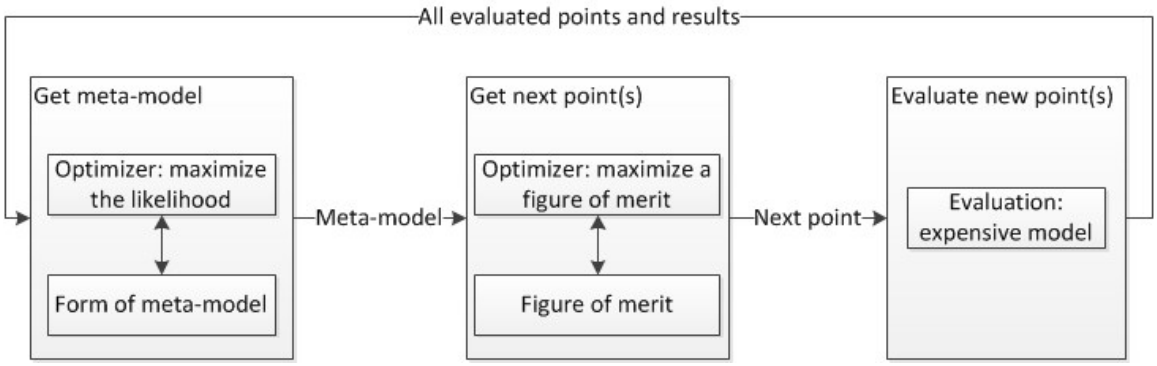

Figure 2: Schematic description of an optimization framework based on meta-modeling

II.2.1.1 Types of meta-models

Many different forms of meta-models have been proposed in the literature. The most widely used models are probably polynomial regression, artificial neural networks (ANN), Kriging or Gaussian process regression, Multivariate Adaptive Regression Splines (MARS), Radial Basis Functions (RBF) and Support Vector Machines (SVM). How these meta-models approximate black box functions from a few observations is very interesting. However, since this is not the scope of this thesis, we limit ourselves to a rapid description of their characteristics and explain why Gaussian process regression was chosen.

Polynomial regression Models

In the Response Surface Model (RSM) approach introduced by Box and Wilson [34], the objective functions are approximated by low order polynomial functions of the decision variables.

While this approach is very simple, many authors pointed out its lack of accuracy for nonlinear problems (see for instance Clarke et al. [35] or Jin et al. [36]).

Multivariate Adaptive Regression Splines (MARS)

The MARS approach, introduced by Friedman [37], divides the design space into sub-regions, which makes it well suited for problems with high dimensions, as highlighted by Clarke et al. [35]. On each sub-region, the process is modeled by a set of basis functions called splines. The splines are connected on the edges of these sub-regions in such a way that the resulting meta-model is continuous and has continuous derivatives on the entire design space [37].

Radial Basis Functions (RBF)

In the RBF approach, the objectives are interpolated as a linear combination of a polynomial function and a set of n independent radial basis functions. Each of those functions takes as a unique input the distance between one of the n points already evaluated and the point where the objective function has to be estimated. Gutmann [38] popularized this method which was further explored by other researchers like Regis and Shoemaker [39] and Mullur and Messac [40].

Kriging or Gaussian Process

This approach has been formalized by Matheron [41] after the idea of Danie G. Krige, a South-African mining engineer. It has been used for decades in geostatistics in order to predict the underground concentration of valuable minerals. The particularity of Kriging comes from the fact that it does not select one best function as a surrogate model but instead, it allocates a probability density to each function, through what is called a *Gaussian process*. Thus, it allows not only for a prediction of the output of the black-box, but also for an estimation of the *uncertainty* of this prediction.

Concretely, the black-box function is usually modeled as the sum of a very simple function (e.g. a constant or linear function) of the inputs and of an additional term, which is a realization of a Gaussian process. Based on the previous evaluations of the black-box process, the Gaussian process is updated to give more weights to the functions that take the same values as the

black-box function (or close values, if a noise is modeled). Additionally, the parameters of the Gaussian process (such as its length-scale) can also be updated by maximizing the likelihood of the previous observations. Gaussian processes and their use for expensive optimization are further detailed in the part II.4 of this chapter.

This approach has been adapted by Sacks et al. [42] for the design of (noise-free) computer experiments under the name "Design and Analysis of Computer Experiments" (DACE). Jones et al. [43] combined DACE with a figure of merit to create a single-objective optimization process called Efficient Global Optimization (EGO). These methods are explained more in depth in section II.2.2.

Support Vector Machine (SVM)

Although SVM was first developed for classification problems [44], it has then be extended to approximate black-box functions [45]. This extension is often referred to as Support Vector Regression (SVR). In its simplest form, the black-box is simply approximated by a linear function of the inputs. However, unlike for linear regression, in this case the linear coefficients ($\vec{w}$) selected are the smallest ones that allow for a maximum error inferior to a given threshold ϵ. The approximation problem is then:

$$Min\left(\frac{1}{2}|\vec{w}|^2\right) \; subject\ to\ \forall i = 1, \dots, n, |\vec{w}.\vec{x_i} + b - y_i| \leq \epsilon \qquad \textbf{(2)}$$

Where $\vec{x_i}$ is the vector of inputs for observation i and y_i is the corresponding observed output. This minimization leads to a model which is as "flat" as possible and which is easily solvable since it is convex. As the existence of such coefficients is not guaranteed, an extension has been proposed that allows for bigger errors but penalizes them. Finally, in order to approximate more complex non-linear processes, the dot product can be replaced by a non-linear application of the input vectors, called a *kernel function.*

Artificial Neural Networks (ANN)

Artificial Neural Network is an umbrella term covering many different techniques that are all inspired by the operation of organic brains. In general, an artificial neural network is made out of multiple nodes that are arranged in layers, and of connections between nodes from one layer to the nodes of the adjacent ones. By associating an activation function to each node and coefficients to each connection, an ANN can reproduce diverse types of functions. Since meta-modeling is often used in situations with no prior knowledge on the nature of the underlying function, this versatility offered by ANNs has led many authors to choose this technique.

II.2.1.2 Comparison

Several authors have endeavored to compare the performances of diverse meta-models to approximate a black-box function. While this thesis is focused on optimization more than on global approximation, the results of these studies are still interesting since many expensive optimization methods rely on a good meta-model.

One of these comparisons was made by Jin et al. [36], who compared the performances of MARS, RBF, Kriging and Polynomial regression in terms of R-square, Relative Average Error and Relative Maximum Absolute Error. In this study, the test cases consisted in either small design spaces (with two or three design variables) or quite large design spaces (ten or more design variables). The authors showed that RBF and Kriging outperformed polynomial regression and MARS in most situations but RBF was found to be slightly more robust.

However, two further studies with different design spaces found contradictory results. First, Clarke et al. [35] compared the same four algorithms but also included SVR in their comparison. The authors used similar performance measures and found that SVR and Kriging outperformed all the other models with regard to the three criteria used. Between SVR and Kriging, SVR was found to have better performances in terms of Root Mean Square Error (RMSE) and average error while Kriging had a smaller maximum error. In this study, RBF was found to perform very poorly but no explanation was proposed. Second, Kim et al. [46] compared

RBF, Kriging, SVR and Moving Least Squares (MLS) on six non-convex continuous functions with either 2, 4, 6 or 8 decision variables. They found that in terms of RMSE, the MLS and Kriging methods outperformed the two others, while SVR was the least accurate and RBF had very unstable performances.

While it is interesting to notice that Kriging performed consistently well in these three studies [35, 36, 46], the differences in the results obtained highlight to which extent the performance of each algorithm depends on the objective functions.

Finally, even though ANNs have been used for decades to model complex functions, there are very few studies that compare ANNs with other meta-modeling techniques such as Kriging, RBF, MARS or polynomial regression. A few comparisons have been found for very specific issues but the results found often show opposite trends [47, 48]. For instance, one can quote Paiva et al. [49], who compared Kriging, ANN and quadratic-interpolation-based response surfaces in four case-studies. The authors concluded that Kriging was in general "more robust and better performing than ANN". An explanation advanced by the authors is the difficulty in selecting the best ANN configuration (number of neurons for instance). Also, Matías et al. [50] compared different variants of Kriging, regularization networks, RBF networks and a kind of ANN called multilayer perceptron network on a mining application. They too found that Kriging provided the best results.

II.2.1.3 Mixtures of meta-models

The high variability in the performance of the different meta-models have led some authors to consider not only one meta-model, but a combination of several meta-models. In the approach defined by Müller and Piché [51], a black-box process is approximated by a linear combination of meta-models obtained by polynomial regression, MARS, RBF and Kriging. After each evaluation, the weights of each algorithm are updated based on a leave-one-out cross-validation process and on the Dempster-Shafer theory. Viana et al. [52] adopted a similar technique and included it the EGO optimization approach [43], which relies on estimations of both the output

value and of the uncertainty. Since all the meta-models cannot readily be used to estimate the uncertainty of the prediction, the authors imported uncertainty estimate from a Kriging meta-model into other meta-models, obtained for instance by SVR.

II.2.1.4 Use of meta-models in the NDP

Since genetic algorithms have been widely used with analytic queuing models, many of the authors who have endeavored to include complex simulation within the NDP have also relied on them [12, 13, 17, 53]. As explained previously, the most important limitation in this type of approach is the high number of evaluations that is required, making any optimization extremely resource-consuming. For instance, recently Stevanovic et al. [54] proposed a traffic signal timing optimization method for which the computations took several months to complete.

However, some meta-model based approaches have already been explored in the past. Before the wide development of computation-intensive simulation models, ANN meta-models had already been used to solve the traffic assignment problem. For instance, Xiong and Schneider [55] used a Neural Network instead of a traditional user-equilibrium trip assignment algorithm to solve the NDP with a cumulative genetic algorithm in a reasonable time. Bielli et al. [56] applied the same approach to optimize public bus networks.

Besides these early studies, at least three more recent attempts have been made to incorporate meta-modeling within the NDP framework. Chow [5] addressed a multi-objective toll-pricing problem using an adapted version of Metric Stochastic Response Surface (MSRS) [57], which is itself based on RBF meta-models. Fikse [14] tackled the dynamic road capacity management problem with different meta-model assisted genetic algorithms. As the main optimization algorithm, the author selected NSGA-II. However, he reduced the number of expensive evaluations required by NSGA-II by pre-evaluating the values of the objectives using a meta-model and then only evaluating the most interesting ones with the real process. The author compared three versions of NSGA-II assisted with methods based on polynomial regression, RBF and Kriging. A similar approach was then undertaken by Wismans et al. [18] with SPEA2+

(another genetic algorithm) instead of NSGA-II. Finally, Osorio [11] proposed a meta-model combination, consisting of an analytic queuing model and of a simple quadratic polynomial. This approach was also described in other studies by Osorio and Bierlaire [58] and Osorio and Chong [59].

However, these three methods have been designed for very specific situations and are not easily applicable to the case investigated in this study. On the one hand, the methods proposed by Chow [5] and Fikse [14] were designed for objective functions whose computation time is not negligible, but still much inferior to the computation time required by a microscopic simulation. Thus, the approaches they proposed required about 6,250 and 10,000 evaluations. On the other hand, the method proposed by Osorio [11] was designed for a very limited computational budget (150 evaluations) but is based on the use of an analytic queuing meta-model. Although this allows for a reduction in the number of evaluations required, it might also be a very restrictive constraint for objective functions such as surrogate safety measures, which cannot be easily approximated analytically.

The EGO approach [43] that is proposed to be applied to the NDP in this study is essentially based on Kriging, without any evolutionary algorithm. It is a true approach of global expensive optimization that can approximate the solution of non-convex problems within a few hundred evaluations, without any analytical model. To the best of our knowledge, it is the first time that such an approach is applied to the NDP.

II.2.2 The EGO approach

The risk with meta-models that only provide an estimate of the objective function is to ignore a part of the design space and to find only a local optimum. To avoid this pitfall, some basic tricks can be implemented to ensure that no region of the design space is left unexplored. However, given that the total number of evaluations is limited, more exploration means less exploitation of the meta-model. Thus, finding the right equilibrium between these two trends can be cumbersome.

An interesting alternative is provided by the EGO approach [43], that is based on Gaussian process regression. As mentioned earlier, Kriging, or Gaussian process regression, does not only allow us to estimate objective functions at unevaluated points, but it also provides an estimation of the uncertainty of this prediction. This is probably why Kriging has also aroused interest in the field of global optimization, where it is known as "Gaussian process regression", "Bayesian global optimization", or as "random function approach" - see references in Jones et al. [43]. In this field, one important goal was to identify an appropriate *figure of merit* that would automatically balance exploration and exploitation. Unlike many other heuristic propositions, the figure of merit introduced by Močkus [60] and then included in the EGO approach by Jones et al. [43] is based on a rigorous statistical basis. In fact, this figure of merit, called the "Expected Improvement" (EI), is simply defined for the point x as:

$$EI(x) = E[I(x)] = E[\max(f_{min} - Y, 0)] \quad \textbf{(3)}$$

Where f_{min} stands for the best (minimum) value of the objective function evaluated so far and Y is the stochastic estimation of the value of the objective function at x.

The EGO approach combines the Kriging modeling described in DACE [42] with the EI to create an algorithm that can be summed up in five steps [43]:

1) Creation of an initial set of initial points with a space-filling design obtained for instance by Latin Hypercube Sampling (LHS),

2) Evaluation of the initial points,

3) Selection of the parameters of the DACE model based on Maximum Likelihood Estimation (MLE),

4) Selection of a new sample that maximizes the EI on the design space,

5) Evaluation of the new sample and return to step 3).

Although the rigorous definition of the Expected Improvement cannot be easily transposed to multi-objective optimization, several authors have proposed alternative heuristic solutions inspired from the same concept.

II.3 Expensive multi-objective optimization based on Gaussian Process meta-model

Kriging has often been included within multi-objective optimization frameworks, but in most situations it was coupled with a genetic algorithm. Fikse [14] demonstrated the interest of such a strategy for the dynamic management of road networks. However, as mentioned previously, such strategies involve something in the order of half the iterations required by a traditional genetic algorithm, which can still represent a sizeable amount of computation if the evaluations rely on time-consuming simulations. For a tighter computational budget (a few hundreds simulations), two approaches have been proposed.

II.3.1 Multi-objective EGO adaptations

The first approach, chosen by many authors, is to adapt the EGO framework to multiple-objective optimization. With one objective, selecting the next point to be evaluated was relatively straight-forward: one simply had to find the point that maximized the Expected Improvement. As always with multiple objectives, one has two possible approaches: either combining all the objectives into a single one or keeping them separate.

The most well-known multiple-objective implementation of the EGO approach may be ParEGO, proposed by Knowles [61]. In order to handle multiple objectives, the author proposed to create one Kriging meta-model for each objective, and to combine them in a single objective via randomly chosen scalarizing weight factors that are modified after each iteration. In short, this method is “multi-objective” since it ultimately provides an approximation of the Pareto front, but each new point is obtained by scalarizing the objectives.

Alternatively, authors chose other solutions that avoid scalarizing the objectives. For instance, Jeong and Obayashi [62] proposed to compute the Expected Improvement for every objective separately and then to carry out a multi-objective optimization taking as fitness functions the Expected Improvement of each objective. Eventually, one only evaluates the best designs found by the multi-objective optimization.

Ponweiser et al. [32] chose to adapt EGO to multiple objectives in still another way. Their approach is based on the S-measure (or hypervolume), which is the Lebesgue measure enclosed by a reference point and the Pareto front, or, as it was originally named by Zitzler and Thiele [63], "the size of the space covered". With two objectives, the hypervolume is in fact an area (see Figure 3). Since this unary index had been shown to be especially good to assess the quality of a Pareto front [64], the authors had the idea to maximize its expected improvement to select the next point to be evaluated. In this elegant way, the initial multi-objective problem is transformed into a single-objective that can be addressed using standard optimization techniques. However, for simplification purposes, the authors adopted a very simplified version of the Expected Improvement. To avoid integrating the improvement of the S-measure on all the objectives, the authors simply computed the S-measure for one single point, the lower confidence bound value $y_{LCB} = \mu - \alpha * \sigma$ (where μ is the vector of expected values, α is a scalar coefficient and σ is the vector of standard deviations). In the case where the lower confidence bound value would improve the Pareto front, the Expected Improvement was defined as the hypervolume increment produced by the addition of this point to the set. Otherwise, a penalty would be applied depending on the distance of y_{LCB} from the Pareto front. In both cases, the value of the Expected Improvement depends only on the vector μ of the expected values, the vector σ of the standard deviations (both provided by the DACE model) and the constant α. The authors named the resulting algorithm SMS-EGO, which is short for "S-Measure-Selection-based Efficient Global Optimization".

As a legacy of the EGO framework, all these multi-objective optimization approaches assume noise-free evaluations.

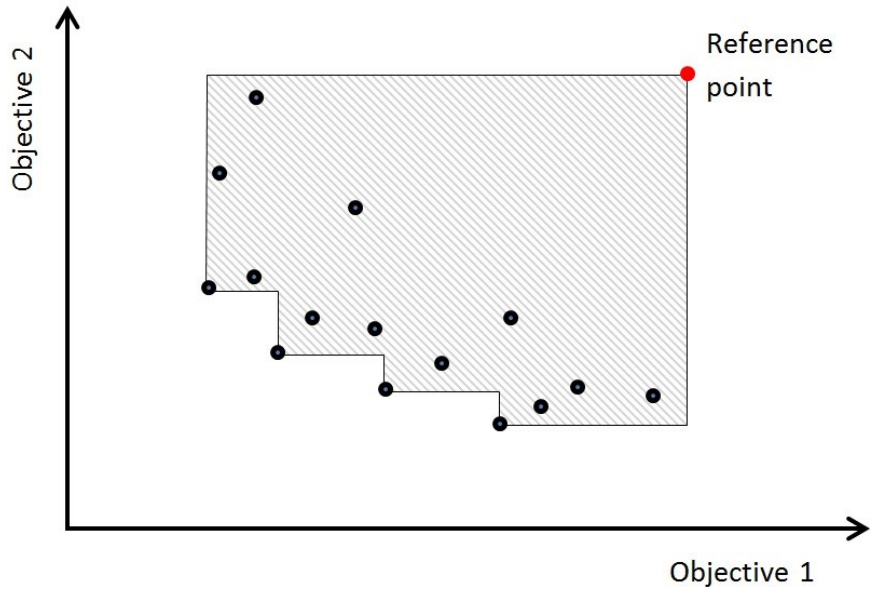

Figure 3: Illustration of the concept of hypervolume (hatched)

II.3.2 Pareto Active Learning (PAL)

The second very distinct approach to expensive evaluations based on Gaussian Processes has been proposed by Zuluaga et al. under the name of Pareto Active Learning (PAL) [31]. PAL also relies on the same theoretical principles as SMS-EGO, but it is not built on the foundations laid by DACE and EGO. Instead, Zuluaga et al. used directly the theoretical work of Rasmussen and Williams [65] and the associated Gaussian Process Regression and Classification Toolbox for MATLAB [66]. With this toolbox, users are able to define their own form of covariance, mean, likelihood and inference function. Thus, unlike the other authors who built upon the existing EGO framework, Zuluaga et al. created an entirely different optimization framework, specific for their needs.

While the multi-objective EGO adaptations aim at approximating the Pareto-curve in a continuous (infinite) design space, PAL aims at identifying and evaluating *all* the Pareto-optimal points in a *finite* set. In order to reach this goal with a minimum number of evaluations, the points

of an initial training set are evaluated and then every iteration consists of three steps. First, every objective function is modeled as a Gaussian process using knowledge from the previous evaluations. In the second step, the algorithm attempts to classify each point as Pareto-optimal or Pareto-dominated. This classification relies on the values of the mean and standard deviation computed in the previous step and on a pre-defined confidence level. In the third step, the point with the biggest variance among the points not classified yet or classified as Pareto-optimal is chosen for the next evaluation. The algorithm stops when all points are classified or when the maximum number of evaluations is reached.

Other differences between PAL and the EGO approach include the type of covariance function chosen and the way the mean function is determined. In addition, PAL assumes that the evaluations of the objective functions differ from the real values by an additive noise which follows an independent, identically distributed Gaussian distribution with zero mean [65].

II.3.3 Previous comparisons

In order to assess the performance of an algorithm, its performance has to be compared with a reference, an already validated algorithm. In multi-objective optimization, the reference often used is NSGA-II, the genetic algorithm proposed by Deb et al. [30]. In the subdomain of expensive multi-objective optimization based on Gaussian process meta-models, ParEGO has been used for comparison [61].

Knowles [61] compared ParEGO with NSGA-II on several noise-free test problems and ParEGO was found to perform significantly better. In terms of robustness to stochastic evaluations, Knowles et al. [67] found that ParEGO outperforms other algorithms not based on meta-models in situations where a Gaussian noise is added to the evaluations. Nevertheless, the authors suggested modifying the EGO approach to account for noisy evaluations.

Among the different adaptations of the EGO approach, SMS-EGO performs relatively well. Indeed, SMS-EGO was found in Ponweiser et al. [32] to outperform ParEGO [61] and the other multi-objective adaptation of the EGO approach proposed by Jeong and Obayashi [62]. In

another study focused on the Expected Improvement criteria, Wagner et al. [68] slightly modified the definition of the penalty in the Expected Improvement proposed by Ponweiser et al. [32] and showed that SMS-EGO compares favorably with other multi-objective algorithms based on the EGO approach.

However, these studies do not include PAL, which has been proposed more recently and which is not based on the same paradigm. Indeed, PAL has, to the best of our knowledge, only been compared once: in Zuluaga et al, [31], PAL was found to outperform ParEGO in most situations. However, these case-studies were exactly the type of situations for which PAL had been designed – a finite set of configurations and noisy evaluations. For the case evaluated in this thesis (unlimited number of configurations and noisy evaluations), it is not clear whether the same trend would be observed.

Based on this literature review, we identified SMS-EGO and PAL as the most promising algorithms to reduce the computation time of the multi-objective, micro-simulation-based, optimization of the CNDP.

II.4 The Bayesian framework and Gaussian processes

Before assessing the performance of these algorithms on specific case-studies, we introduce first some basic elements of the Bayesian framework and Gaussian processes. These concepts are at the core of Kriging and of the two algorithms used in this thesis.

II.4.1 The Bayesian framework

In general, regression analysis is done within a previously specified class of approximating functions and often consists in finding the function that has the best fit to the unknown function. However, in the Bayesian framework, there is no selection of the best approximating function. Instead, a Bayesian regression consists in associating to each approximating function a probability density that reflects how well it fits the data. Thus, one can then estimate the probability of a specific output at any un-evaluated point. If one wishes to have a single output value for this un-evaluated point, one can simply take the expected value or the value associated

with the biggest probability. Note that if the distribution is symmetric as it is the case with Gaussian processes, these two values are equal.

More formally, let us consider the following general problem: we want to approximate the real process H, but we can only measure the output of H with a random noise ϵ, as shown in the equation below;

$$y_i = H(x_i) + \epsilon_i \quad \textbf{(4)}$$

Where x_i is a vector that represents a point of the decision space X and y_i is the measured output (a real number). Let us assume that for any set I of indices i, $(\epsilon_i)_{i\in I}$ is independent, identically distributed and follows a distribution D_ϵ. We will write $\epsilon \sim D_\epsilon$ and the associated probability density function (PDF) will be written p_ϵ. In the Bayesian framework, we start by defining what is called a *prior distribution* over the set of possible models $\mathcal{H}$, which contains all the prior information we might have. We will write this $h \sim D_h$. Thus, initially, we have: $\forall h^* \in \mathcal{H}$,

$$p(y_i|h^*, X_i) = p_\epsilon\big(y_i - h^*(X_i)\big) \quad \textbf{(5)}$$

And, by marginalizing out h^*:

$$p(y_i|X_i) = \int_{h^*\in\mathcal{H}} p_\epsilon\big(y_i - h^*(X_i)\big).p(h^*)dh^* \quad \textbf{(6)}$$

Let $\{(X_i, y_i)_{i=1..N}\}$ be a set of training data points. We will also write $\overrightarrow{y_{1\dots N}} = \begin{pmatrix} y_1 \\ \cdots \\ y_N \end{pmatrix}$.

Using Bayes' rule, a *posterior distribution* over the possible models can be computed:

$$p(h^*|\overrightarrow{y_{1\dots N}}) = p(h^*).\frac{p(\overrightarrow{y_{1\dots N}}|h^*)}{p(\overrightarrow{y_{1\dots N}})} = p_h(h^*).\frac{\prod_{i=1}^N p_\epsilon\big(y_i - h^*(X_i)\big)}{\prod_{i=1}^N \int_{h^*\in\mathcal{H}} p_\epsilon\big(y_i - h^*(X_i)\big).p(h^*)dh^*} \quad \textbf{(7)}$$

Thus, the prediction can now be updated based on this posterior distribution:

$$p(y_{N+1}|X_1, X_2 \dots, X_{N+1}, \overrightarrow{y_{1\dots N}}) = \int_{h^*\in\mathcal{H}} p_\epsilon\big(y_{N+1} - h^*(X_{N+1})\big).p(h^*|\overrightarrow{y_{1\dots N}})dh^* \quad \textbf{(8)}$$

This framework allows to include all the knowledge acquired to predict not only an estimation (such as the one that has the maximum likelihood) of y_{N+1} but also an estimation of the uncertainty. Generally, the integrals in the previous equations involve computing complex approximations and repeating this process for every evaluation of the meta-model is prohibitive in terms of computation time. However, this issue can be solved very easily by selecting adapted types of meta-models and distributions.

II.4.2 Introduction to Gaussian processes

The integration of Gaussian Processes in the Bayesian framework is thoroughly explained in Rasmussen and Williams [65]. For interested readers, the theory is also presented in a more condensed way in Do [69]. Here, we will try to present only the basics necessary to understand the algorithms used in this thesis.

We can introduce Gaussian Processes as an extension of multivariate Gaussian distributions. It is recalled that the vector x of n random variables has a joint Gaussian distribution with mean $\mu \in \mathbb{R}^n$ and covariance $\Sigma \in S_{++}^n$ if

$$p(x|\mu,\Sigma) = \frac{1}{(2\pi)^{n/2}\sqrt{|\Sigma|}} \exp\left(-\frac{1}{2}(x-\mu)^t\, \Sigma^{-1}(x-\mu)\right) \quad \textbf{(9)}$$

Starting from this definition, we could try to describe the probabilities associated with a given model by sampling this model and computing the joint Gaussian distribution of all these samples. However, we would be limited to a finite number of samples. Gaussian processes allow us to consider all the possible samples as they represent *distributions over functions*.

Definition: A Gaussian Process is a collection of random variables, any finite number of which have a joint Gaussian distribution.

Gaussian processes are completely defined by their mean and covariance functions. Thus, if f is a Gaussian process, we write $f \sim GP(m,k)$. Here, m and k are functions defined as:

$\forall x, x' \in X$,

$$m(x) = E[f(x)] \quad \textbf{(10)}$$

$$k(x, x') = E\big[\big(f(x) - m(x)\big)\big(f(x') - m(x')\big)\big] \quad \textbf{(11)}$$

Rasmussen and Williams [65] explain that to be a valid covariance function, a function only has to satisfy

$$\forall x, x' \in X, k(x, x') = k(x', x)\ (symmetry), \quad \textbf{(12)}$$

$$\forall n \in \mathbb{N}, \forall\, D = \{x_i | i = 1, \dots, n\}, \forall v \in \mathbb{R}^n, v^t K v \geq 0\ (positive\ semidefiniteness) \quad \textbf{(13)}$$

Where K is a matrix such that $K_{ij} = k(x_i, x_j)$ (i.e. K is the Gram matrix associated to D and k).

Nevertheless, depending on the situation, some forms of covariance functions make more sense than others. For instance, covariance values that continuously decrease when two points get further will yield to smooth functions. The squared exponential is a very classic example of such covariance functions:

$$cov\big(f(x), f(x')\big) = k(x, x') = \sigma^2 \exp\left(-\frac{|x - x'|^2}{l^2}\right) \quad \textbf{(14)}$$

II.4.3 Gaussian processes within the Bayesian framework

In this part, we are going to show how Gaussian processes can be particularly convenient within a Bayesian framework. Again, more detailed explanations can be found in Rasmussen and Williams [65]. Let us first assume that in the original model described in Equation 4, the realizations of the noises, ϵ^i, are independent, identically distributed with a Gaussian, zero-mean, distribution which is also independent from the value of the output of the model:

$$\epsilon \sim N(0, \sigma^2) \quad \textbf{(15)}$$

In addition, we define the prior distribution as a Gaussian process:

$$h \sim GP(m, k) \quad \textbf{(16)}$$

Where m can be any mean function and k is a valid covariance function that contains the prior knowledge (for instance a squared exponential function with a given length-scale l).

Let us rewrite the previous problem which consisted in predicting y_{N+1} given the values of the previous evaluations $y_1, \dots y_N$. For a better readability, we adopt the following notations:

$$K(X_{1\dots N}, X_{1\dots N}) = \begin{bmatrix} k(X_1, X_1) & \dots & k(X_1, X_N) \\ \dots & k(X_i, X_j) & \dots \\ k(X_N, X_1) & \dots & k(X_N, X_N) \end{bmatrix}$$

$$K(X_{N+1}, X_{1\dots N}) = [k(X_{N+1}, X_1) \quad \dots \quad k(X_{N+1}, X_N)]$$

$$K(X_{1\dots N}, X_{N+1}) = K(X_{N+1}, X_{1\dots N})^t$$

$$K(X_{N+1}, X_{N+1}) = k(X_{N+1}, X_{N+1}).$$

By applying the definition of a Gaussian process:

$$\begin{bmatrix} h(X_1) \\ \dots \\ h(X_N) \\ h(X_{N+1}) \end{bmatrix} | X_1, X_2 \dots, X_{N+1} \sim N\left(\begin{bmatrix} m(X_1) \\ \dots \\ m(X_{N+1}) \end{bmatrix}, \begin{bmatrix} K(X_{1\dots N}, X_{1\dots N}) & K(X_{1\dots N}, X_{N+1}) \\ K(X_{N+1}, X_{1\dots N}) & K(X_{N+1}, X_{N+1}) \end{bmatrix} \right) \quad \textbf{(17)}$$

Besides, from the assumption of an independent identically distributed zero-mean Gaussian noise:

$$\begin{bmatrix} \epsilon_1 \\ \dots \\ \epsilon_N \end{bmatrix} \sim N\left(\vec{0}, {\sigma_n}^2 I_N\right) \quad \textbf{(18)}$$

Where I_N represents the matrix identity of dimension N. As the noise and the model are assumed to be independent, the distribution of their sum can be obtained by adding their respective means and variances:

$$\begin{bmatrix} \overrightarrow{y_{1\dots N}} \\ h(X_{N+1}) \end{bmatrix} | X_1, \dots X_{N+1} \sim \quad N\left(\begin{bmatrix} m(X_1) \\ \dots \\ m(X_{N+1}) \end{bmatrix}, \begin{bmatrix} K(X_{1\dots N}, X_{1\dots N}) + \sigma_n^2 . I_N & K(X_{1\dots N}, X_{N+1}) \\ K(X_{N+1}, X_{1\dots N}) & K(X_{N+1}, X_{N+1}) \end{bmatrix} \right) \textbf{(19)}$$

Note that we are interested in the real value of the process at the point X_{N+1}, not in its noisy measure.

In general, within the Bayesian framework, in order to obtain the distribution of y_{N+1}, we need to compute the posterior distribution over the possible models and then integrate over this

distribution. However, with Gaussian Processes, this simply corresponds to the conditioning of a distribution, which obeys the following rule:

$$if \begin{bmatrix} A \\ B \end{bmatrix} \sim N\left(\begin{bmatrix} \mu_A \\ \mu_B \end{bmatrix}, \begin{bmatrix} \Sigma_{AA} & \Sigma_{AB} \\ \Sigma_{BA} & \Sigma_{BB} \end{bmatrix}\right)$$

$$then\ B \mid A \sim N(\mu_B + \Sigma_{BA}\Sigma_{AA}^{-1}(A - \mu_A), \Sigma_{BB} - \Sigma_{BA}\Sigma_{AA}^{-1}\Sigma_{AB}) \quad \textbf{(20)}$$

Thus, in our case:

$$h(X_{N+1}) \mid y_1, \dots y_N \sim N(\mu_{N+1}, \Sigma_{N+1}) \quad \textbf{(21)}$$

Where

$$\mu_{N+1} = m(X_{N+1}) + K(X_{N+1}, X_{1\dots N})(K(X_{1\dots N}, X_{1\dots N}) + \sigma_n^2.I_N)^{-1}(\overrightarrow{y_{1\dots N}} - \overrightarrow{m_{1\dots N}}) \quad \textbf{(22)}$$

$$\Sigma_{N+1} = K(X_{N+1}, X_{N+1}) - K(X_{N+1}, X_{1\dots N})(K(X_{1\dots N}, X_{1\dots N}) + {\sigma_n}^2.I_N)^{-1}K(X_{1\dots N}, X_{N+1}) \quad \textbf{(23)}$$

This is the result which is at the core of both PAL and SMS-EGO. We can see that thanks to the use of Gaussian Processes, it has been obtained without computing any integration. However, these equations should not be implemented as such since direct matrix inversion can be highly time-consuming and can lead to important numerical errors. Instead, the method used by the authors of the two toolboxes at the core of the algorithms studied in Chapter III, relies on the Cholesky decomposition, which is faster and more accurate. To minimize the number of calculations, the resulting Cholesky decomposition is saved and re-used every time a new point has to be evaluated with the same model.

Nevertheless, in practice one rarely knows in advance what covariance function in the prior distribution will best fit the data. Thus, one might, for instance, make several hypotheses and test which one gives the best results. In order to compare different prior distributions, the probability of the output observed, given the input and under this prior distribution, can be used.

As this probability is obtained by marginalizing out the function h^*, it is called the *marginal likelihood*:

$$p(\overrightarrow{y_{1\dots N}}|X_1,\dots X_N) = \int_{h^*\in\mathcal{H}} p(\overrightarrow{y_{1\dots N}}|h^*,X_1,\dots X_N)p(h^*|X_1,\dots X_N)dh^* \quad \textbf{(24)}$$

However, as this expression is rather unwieldy, $\overrightarrow{y_{1\dots N}}$ will instead be considered as the sum of two independent Gaussian distribution, as it has already been done in equation 19. Besides, since all the points are treated together, there is no confusion possible, so we will simply write $K = K(X_{1\dots N}, X_{1\dots N})$, $Y = \overrightarrow{y_{1\dots N}}$, $M = \overrightarrow{m_{1\dots N}}$. Thus:

$$Y|X_1,\dots X_N \sim \quad N(M, K+\sigma^2 I_N) \quad \textbf{(25)}$$

And for the marginal log-likelihood:

$$\log\big(p(Y|X_1,\dots X_N)\big) = \log\left(\frac{1}{(2\pi)^{\frac{N}{2}}|K+\sigma^2 I_N|^{\frac{1}{2}}} exp\left(-\frac{1}{2}(Y-M)^t\big(K+\sigma^2 I_N\big)^{-1}(Y-M)\right)\right) \quad \textbf{(26)}$$

Or

$$\log\big(p(Y|X_1,\dots X_N)\big) = -\frac{N}{2}\log(2\pi) - \frac{1}{2}\log\left(\left|K+\sigma^2 I_N\right|\right) - \frac{1}{2}(Y-M)^t\big(K+\sigma^2 I_N\big)^{-1}(Y-M) \quad \textbf{(27)}$$

This concludes this introduction to the use of Gaussian processes within a Bayesian optimization framework. While the equations 22 and 23 can be used to estimate the expected value and variance at any point, the equation 27 allows us to evaluate how well the hyper-parameters describe the data. Together, these three equations form the core of any optimization strategy based on Gaussian processes. These equations can be used as such, for instance in PAL, or under a different form, for instance in SMS-EGO. For the variants of these equations used in the EGO framework, the reader is referred to the Appendix. These technical differences play an

important role in Chapter IV and Chapter V, where we endeavor to adapt the original program for the specific problem proposed by this thesis.

II.5 Conclusion of the chapter

In this chapter, the optimization literature has been reviewed to identify the algorithms that are best fitted for the problem defined in Chapter I. Based on this literature review, two algorithms based on Gaussian process meta-models have been selected. According to previous work, both algorithms can be expected to perform especially well for multi-objective expensive optimization. However, the types of problems for which they were designed slightly differ from the one studied in this thesis, either because the evaluations were deterministic, or because the problems were based on a finite design space. The main characteristics of the algorithms selected are summed up in Table 1. For more details, the reader is referred to the corresponding articles.

		PAL [31]	SMS-EGO [32]
	Design space	Finite	Hyperrectangle (box)
çObtaining meta-model	Optimizer	Minimize.m [70] (gradient-based)	CMA-ES [71] (gradient-free)
	Form of meta-model	$f = d + z_l(x)$ Where: - d is a constant (hyperparameter), - z_l is a zero-mean Gaussian Process with the covariance function k	$f = \beta + z_l(x)$ Where: - β is a Generalized Least Squares estimate, - z_l is a zero-mean Gaussian Process with the covariance function $k = \sigma^2 r$
	Form of covariance / correlation function	$cov\left(f(X_p), f(X_q)\right) = k(X_p, X_q) = \sigma^2 . \prod_{j=1}^{n} e^{-\theta_j . (X_{p,j} - X_{q,j})^2} + \sigma_n \delta_{pq}$ Where $\delta_{pq} = \begin{cases} 1 \; if \; p = q \\ 0 \; otherwise \end{cases}$	$corr\left(f(X_p), (X_q)\right) = r(X_p, X_q) = \prod_{j=1}^{n} e^{-\theta_j . ((X_{p,j} - X_{q,j})^{\gamma_j}}$
	Hyper-parameters	$d, \theta_{1\ldots n}, \sigma, \sigma_n$	$\theta_{1\ldots n}, \gamma_{1\ldots n}$
Choosing next point	Optimizer	Exhaustive comparison	CMA-ES [71] (gradient-free)
	Figure of merit	Variance	Expected Improvement of the hypervolume [68]

Table 1: Characteristics of PAL and SMS-EGO

Chapter III Comparison of existing algorithms (case-study)

This chapter is based on a paper [72] presented at the Transportation Research Board Annual Meeting 2014 which has been modified and enriched for this thesis. The objective of this work is to evaluate the potential of meta-model-based algorithms to accelerate the approximation of the Pareto front. For this purpose, the two algorithms selected (PAL and SMS-EGO) were evaluated on different situations and compared with NSGA-II, a genetic algorithm considered as the state-of-the-art. First, in this chapter, the case-study considered is presented and all the necessary practical details concerning the simulation and the algorithms are provided. Then, the framework used to compare the different algorithms is described. Finally, the results are presented and conclusions are drawn.

III.1 Case-study

In order to compare the performances of the three algorithms, a case study was carried out on a section of *Chemin de la Côte des Neige* (Montréal, Canada), a four-lane arterial with two intersections controlled by traffic signals. This location was selected because it is representative of the part-time operated reserved bus lanes utilized in Montreal. Indeed, there is currently on this part of the arterial a bus lane that operates only in peak-hour in the direction with most traffic. By changing the lane allocation, we were able to generate three different case-studies without multiplying the networks to be modeled. Furthermore, a calibrated microscopic simulation model was already available for this research.

III.1.1 Configurations

For the same part of the *Côte des Neiges* arterial, three alternatives were considered for the off-peak period:

- The “base” configuration: two “General Purpose” (GP) lanes in each direction with a fixed-time signal control strategy;

- The “bus-lane” configuration: one GP and bus lane in each direction with a fixed-time signal control strategy ;
- The “TSP” configuration: one GP and bus lane in each direction with Transit Signal Priority (TSP) on the bus lane.

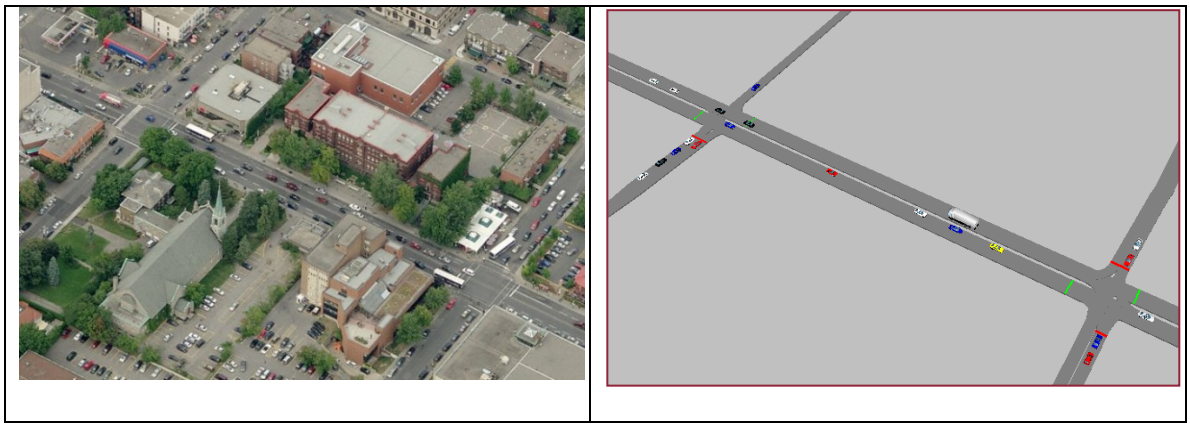

Figure 4: Photograph of the road section studied [73] **(left) and picture from the corresponding Vissim model (right)**

In the three cases, the demand was chosen according to a prior calibration. The actual accuracy of this calibration was not evaluated.

III.1.2 Design space

For the first two alternatives, three decision variables were considered: the green split at the two intersections and the relative offset between them. In the “TSP” scenario, detectors were added on the bus-lanes 50 m before each intersection and six decision variables were defined: the minimum durations of each of the four green phases, the relative offset between the two intersections, and the duration of the green extension when buses are detected. Within the traffic simulator, VISSIG and VAP (Vehicle Actuated Programming), two Vissim add-ons, were used to implement the fixed-time and vehicle-actuated traffic control strategies, respectively. The cycle length was assumed to be imposed by the other intersections of the network for all alternative scenarios.

III.1.3 Objectives

The multi-objective approach is only useful when the objectives are conflicting. An example of such a situation could be to minimize the frequency and severity of crashes while minimizing

travel time. Emissions could also be minimized simultaneously. This thesis is mainly focused on the optimization methodology, which is independent of the type or number of objectives (as long as there are at least two). Thus, we have limited ourselves to only two objectives that can be readily obtained with Vissim and that characterize the operational efficiency: minimizing the average delay per vehicle and minimizing the average delay per bus.

III.2 Practical details

III.2.1 Source code and software used

The source code for PAL [31] and SMS-EGO [32] was provided by their authors in MATLAB. Except for some light modifications such as minimizing two objectives instead of minimizing one and maximizing the other, they were kept as designed by their authors. An implementation of NSGA-II is also available in MATLAB in the Global Optimization Toolbox. As all the programs were available in MATLAB, this programming language was chosen for the optimization part of this thesis. For the traffic simulation, PTV Vissim (version 5.4), a widely-used commercial micro-simulation software, was used.

III.2.2 Integration Vissim/MATLAB

In order to set simulation parameters and to run new simulations from the MATLAB optimization program, the COM interface available in Vissim was used. However, since the traffic control schemes were written in text files (.sig) read by Vissim, it is more efficient to modify them directly from the MATLAB program. Similarly, to read the results, the MATLAB program was directly accessing the .npe files written by VISSIM. This framework is illustrated in Figure 5.

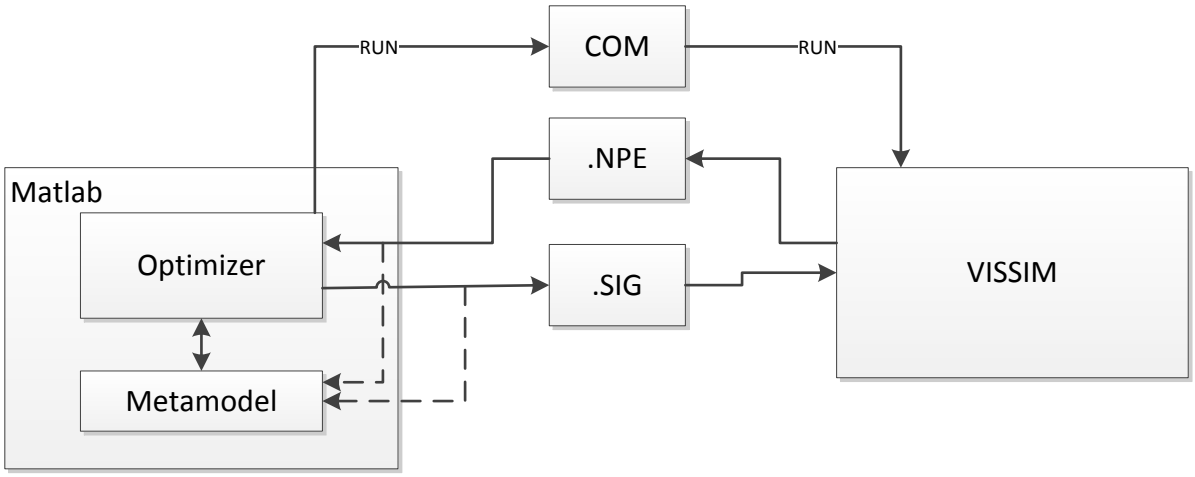

Figure 5: Integration of VISSIM and MATLAB

III.2.3 Simulation parameters

In order to accommodate a sufficient number of simulations within a reasonable amount of time, the number of simulation steps per second was fixed to two. While Vissim is capable to update the state of the system every tenth of the second, the selected time step represents a reasonable compromise since the objective functions chosen did not require a very precise time resolution. In addition, all the simulations were run with a fixed traffic assignment strategy and without enabling the graphical animation of vehicles interactions.

To ensure that the evaluation corresponds to a steady state of the network, each simulation was preceded by a 15 min warming-up period and then was run during 60 min, as in Robles [13].

III.2.4 Accounting for stochastic evaluations

The randomness in Vissim stems from many stochastic processes, such as the arrival of new vehicles in the network, their characteristics, their route, etc. In practice, all these inputs are generated based on a unique random number, called the random seed. Thus, two simulations that are run with the same seed provide exactly the same results. A stochastic system has to be modeled with different random seeds in order to have a more representative view of the network

performance. As in Robles [13], each simulation was run five times with different random seeds and the results were averaged to reduce the noise of the evaluations.

III.2.5 Initial sets and population size

PAL and SMS-EGO need an initial training set to develop meta-models of the objective functions, while NSGA-II needs an initial population. In order to have a well-defined global model, no region of the decision space should be left unexplored. Therefore, the points of the initial training set or initial population should be spread as evenly as possible. As suggested by Jones et al. [43] in the EGO approach and by Ponweiser et al. [32], a Latin Hypercube design was chosen with $11d - 1$ initial points for NSGA-II and SMS-EGO, where d is the dimension of the decision space. For PAL [31], the entire decision space was created in this research with a Latin Hypercube design. Then, the $11d - 1$ initial points were selected using an algorithm created by Zuluaga et al. [31], aiming at maximizing the distances between the points selected.

III.2.6 Algorithm parameters

In PAL, several parameters determine the degree of confidence and the number of iterations which are necessary to identify the Pareto-optimal subset. In most of the cases, values recommended by the authors were chosen. The only parameter that was not specified (ϵ) was fixed to zero after trying several values. In SMS-EGO, the only parameter to be specified is α, which appears in the expressions of the low-confidence bound values of the objective functions, $y_{LCB} = \mu + \alpha * \sigma$, used to compute the Expected Improvement. The value of α was chosen such that there was a probability $p_\alpha = 0.75$ that the true value of the objective function was in the interval $[\mu - \alpha * \sigma\,; \mu + \alpha * \sigma]$.

III.2.7 Hardware

To allow for a comparison between the algorithms, all the optimization processes were run on a Intel® Xeon ® E3110 3.00 GHz processor for the “bus lane” and “base” configurations and on a Intel® Xeon® E5405 2.00 GHz processor for the “TSP” configuration.

III.3 Performance assessment

The evaluation of optimization algorithms is usually based on one or several indices. For instance, in single-objective optimization, one might compare several algorithms based on the best value found within a pre-defined computational budget. However, in the case of expensive multi-objective optimization, defining the computational budget and the best value is not as straight-forward.

III.3.1 Performance index

In multiple-objective optimization, the output of each algorithm is a set of several points approximating the Pareto front. The idea is to associate to this set of point a performance index that could be used as a basis for comparison with other algorithms. The variation of this index over time would also allow us to analyze the dynamic behavior of the algorithms.

Different performance indices have been proposed in the literature regarding multi-objective optimization algorithms, measuring for instance the diversity of the approximation set or the closeness to the real Pareto front. Zitzler et al. [64], however, showed that no comparison method based on a finite combination of unary indices (i.e. depending only on one approximation set) can predict whether an approximation set is better than another. Thus, the only way available to us is to compare the complete Pareto front approximations. This task was carried out on one configuration with one output from each algorithm.

Nevertheless, unary indices are still useful as they allow for a synthetic representation of the results. Zitzler et al. [64], compared different unary indices and identified the hypervolume enclosed by the Pareto front approximation and a reference point as one of the most meaningful (see Figure 3). Indeed, if a first approximation set dominates a second one, then the first hypervolume dominates the second one. Conversely however, if the hypervolume of the first set dominates the second one, it only proves that the first set is not dominated by the second one. This index was selected to evaluate the evolution of the quality of the Pareto front during the optimization process.

III.3.2 Computational budget

When expensive evaluations are involved, the question of the definition of the computational budget is also fundamental. Indeed, in this case, the computational time consists of two parts: the time required by the algorithm itself and the time required by the evaluations. Depending on the application, the computational time associated with the evaluations may vary from a few minutes by evaluation to days, weeks, or more. With microscopic simulation, this time depends for instance on the number of agents, on the number of simulation steps, and on the number of simulations per evaluation. It usually varies between a few minutes and an hour. Consequently, defining a computational budget is difficult. Instead, we chose to study the evolution of the performance index, both over computation time and over the number of evaluations. While the former representation is more adapted for adapted for applications with "reasonably cheap" evaluations, the latter makes more sense for applications with very expensive evaluations.

Regardless of how the performances are evaluated, it is still necessary to define maximum computational budgets to keep the optimization process duration within reasonable boundaries. On one hand, as there is no meta-model in NSGA-II, the total computation time is directly proportional to the product of the number of generations and the population size. In cases with three decision variables, the number of 32-solution generations was fixed to 20 as in Robles [13], leading to a total of 640 evaluations of the objective functions. When TSP was used, there were six decision variables and, therefore, each generation had 65 solutions. In this case, the number of generations was fixed to 15, leading to a total of 975 evaluations of the objective functions.

On the other hand, both PAL and SMS-EGO rely on meta-models. While meta-models are well adapted when only a few hundred evaluations are possible, computing and optimizing them becomes highly time-consuming as more information is acquired. Thus, in addition to the 32 evaluations previously made in the training phase in cases with three decision variables, the number of evaluations was limited to 230 for SMS-EGO. With six decision variables, the

computation time was still bigger and therefore, the number of evaluations after the 65 initial was limited to 100. In total, 262 evaluations were made for the “base” and “bus-lane” configurations and 165 for the “TSP” configuration. For PAL, the number of iterations necessary to determine the Pareto-optimal points depends on the values of the evaluations, but it can be influenced by modifying the size of the decision space. In the different optimization processes ran with PAL, the number of evaluations ranged between 61 and 285 for three variables and between 203 and 315 for six.

III.4 Results

III.4.1 Comparison of Pareto fronts

In order to compare the final outputs of the three algorithms in each situation, the best Pareto fronts obtained are presented in Figure 6. With NSGA-II and SMS-EGO, the quality of the Pareto front improves with the number of evaluations. Therefore, only the Pareto front obtained at the last iteration was presented for these two algorithms. With PAL, the Pareto front is only defined once the algorithm has stopped. Therefore, the optimization process was run several times with design spaces of different sizes and only the Pareto front with the biggest hypervolume was presented.

First, the performances of the algorithms can be evaluated by comparing the different approximations they provide for the same Pareto front. Based on the results illustrated in Figure 6, one can see that the approximations of the Pareto sets determined by SMS-EGO dominated the ones obtained by NSGA-II and PAL in almost all of the objective space. Indeed, SMS-EGO found traffic signal settings that, for the same average car delay, led to lower bus delay than the solutions found by NSGA-II or PAL. PAL also significantly outperformed NSGA-II in the “TSP” configuration, but this advantage was less significant for the “bus lane” and “base” configurations. In fact, in these two situations, NSGA-II locally outperformed PAL for small values of car delay. Importantly, one can see that these differences in Pareto Front approximations could lead to different design choices. Indeed, while the “bus-lane” configuration

would be chosen over the "base" configuration regardless of the algorithm used, one can see that the addition of the TSP would be judged detrimental with NSGA-II but potentially beneficial with PAL or SMS-EGO.

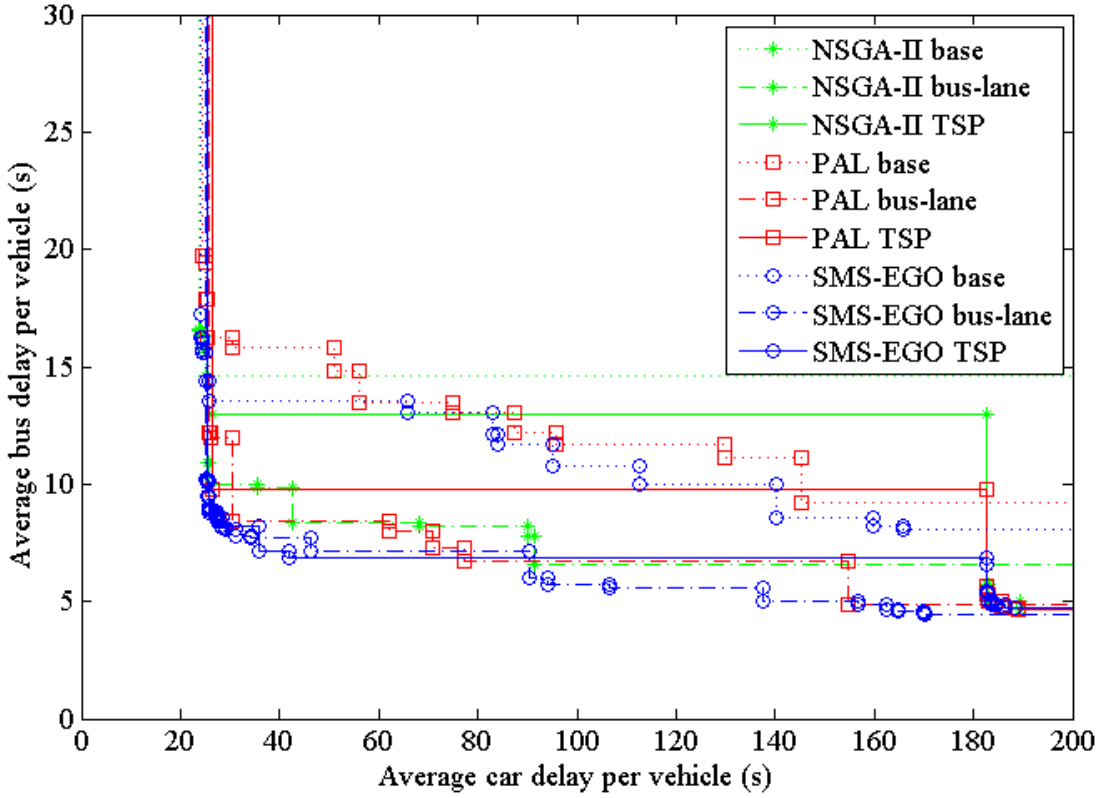

Figure 6: Comparison of the Pareto fronts obtained with NSGA-II, PAL, and SMS-EGO in the "base", "bus-lane", and "TSP" configurations.

Second, Figure 6 can be used to estimate the diversity of the solutions provided by the three algorithms. Even though the Pareto front approximations obtained with PAL and SMS-EGO were based on fewer evaluations than the approximations obtained with NSGA-II, they contained significantly more points, especially for the "bus-lane" and "base" configurations. Thus, the use of PAL and SMS-EGO enabled to define more clearly the trade-offs between car delay and bus delay. This observation remains valid for the "TSP" configuration, even though all the estimations of the Pareto front were less precisely defined. This can be explained by the additional decision variables that made the situation more complex.

III.4.2 Comparison of efficiency

Efficiency is the second important criterion for the choice of an algorithm. A multi-objective optimization process should provide a good Pareto front approximation in a reasonable time. Because the computation time highly depends on the application, it is necessary to provide some efficiency measure that is somehow independent of the application. For this reason, as explained in section III.3, the performances of each algorithm were evaluated both in terms of number of simulations required (Figures 7a, 7c and 7e) and computation time (Figures 7b, 7d and 7f). By analyzing the different graphs and comparing them, one can make several observations.

First, the compatibility of the hypervolume index with the dominance relations observed in Figure 6 can be verified in Figure 7. Indeed in the three configurations, the final hypervolume of the solutions obtained by SMS-EGO was bigger than the best hypervolume obtained with PAL, which was itself bigger than the final hypervolume obtained with NSGA-II.

Second, based on the evidence of the evaluations carried out, it seems that in the three configurations, SMS-EGO found better approximations of the Pareto fronts, and did so very fast. Indeed, in the three case studies, SMS-EGO progressed very fast at the very first iterations and then seemingly converged toward a final state. On the contrary, NSGA-II was quite slow to find good approximations and in the “bus lane” and “base” configurations at least, the quality of the approximation was still substantially improving when the optimization process was stopped. The analysis of the optimization processes run with PAL showed that within a limited number of evaluations, PAL provided better results than NSGA-II in the “bus lane” and “base” configurations, but not as good as SMS-EGO. The results of PAL and NSGA-II were similar for the “TSP” configuration. Thus, in these three configurations and with a very small computational budget, SMS-EGO globally outperformed PAL, which itself performed similarly to or better than NSGA-II.

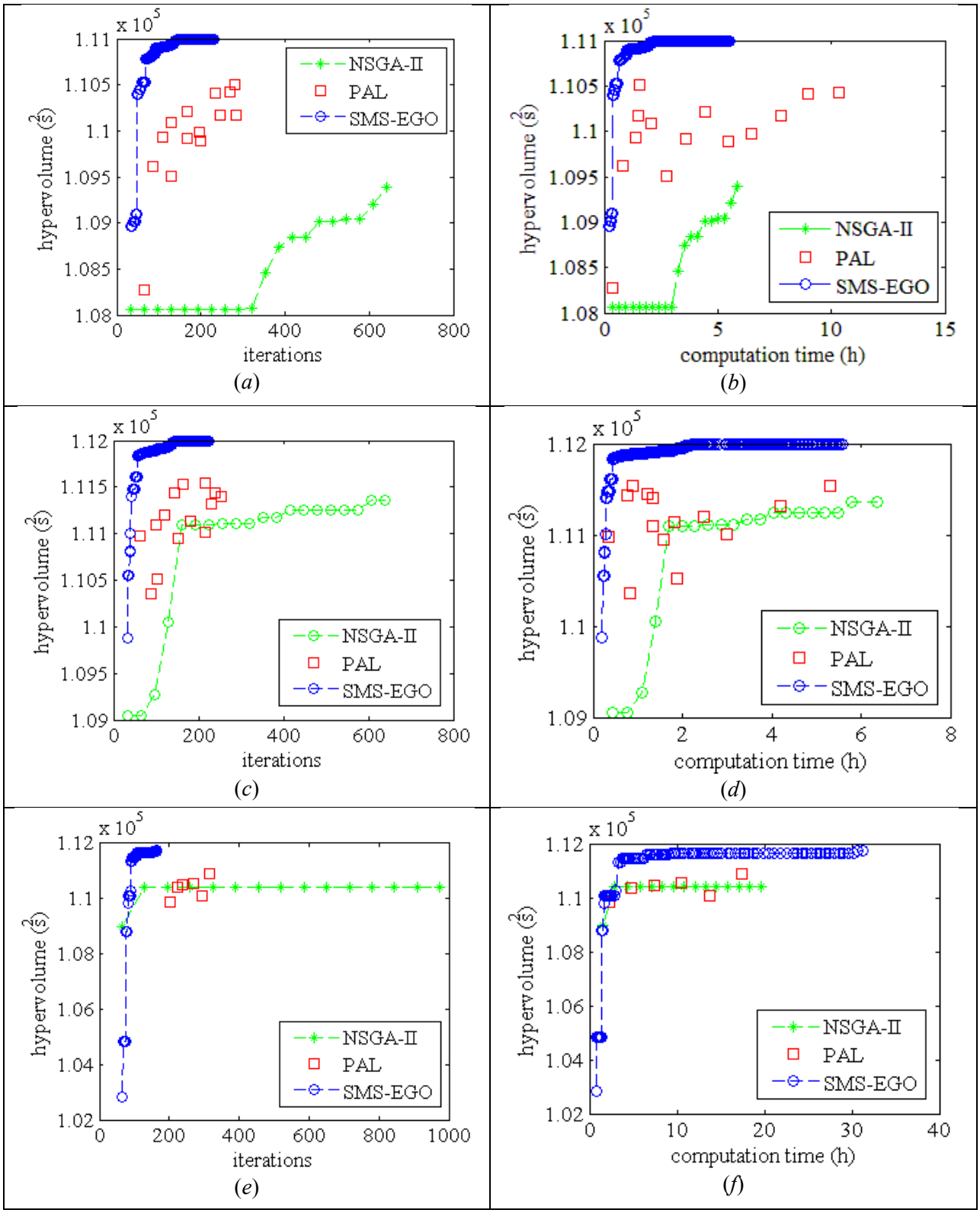

Figure 7: Comparison of the hypervolume of the Pareto fronts approximations obtained with NSGA-II, PAL, and SMS-EGO in function of the number of iterations and of the computation time for the "base" (a, b), "bus-lane" (c, d) and "TSP" (e, f) configurations.

Finally, the comparison of the figures 7a, 7c and 7e with the figures 7b, 7d and 7f, shows that the advantage of SMS-EGO and PAL over NSGA-II was more significant in terms of number of iterations than in terms of computation time. This observation can be explained by the additional complexity required first to compute the meta-models and then to determine the best point for the next evaluation.

III.4.3 Robustness to noise

In the three algorithms, the final output is an approximation of the Pareto front consisting of objective values obtained using noisy evaluations. In fact, even though PAL takes into account the noise in its meta-model, it still relies on the values from the evaluations to classify points and does so without accounting for the uncertainty. In order to have a more accurate picture of the Pareto front, it is necessary to re-evaluate the selected configurations.

In the optimization process, the performance measures were obtained by averaging the results of five simulations. In this part, the objective values of the best points identified by PAL and SMS-EGO in the "bus-lane" configuration were re-evaluated by averaging the results of 50 simulations. The values obtained from these two series of measures are shown in Figure 8 and Figure 9, where the values corresponding to a same configuration are connected with a dotted line.

In Figures 8 and 9, one can see that the differences between the two series of measures are quite substantial. In addition, one can see that most of the posterior evaluations are "inside" the hypervolume, which means that the optimization process tends to over-estimate the performance of the points on the Pareto front. In fact, this bias can be explained by the way the points are selected in these algorithms. Since only the points with the best evaluations are selected in the final output, the configurations that have benefited from the noise are more likely to be selected than the ones that were penalized by it. Eventually, this type of selection necessarily leads to a bias in the performance measures of the points selected. This phenomenon is evidenced

in Figure 10 and Figure 11, which show in the same plots the estimates of the Pareto front approximations obtained with the optimization process and with the new series of measures.

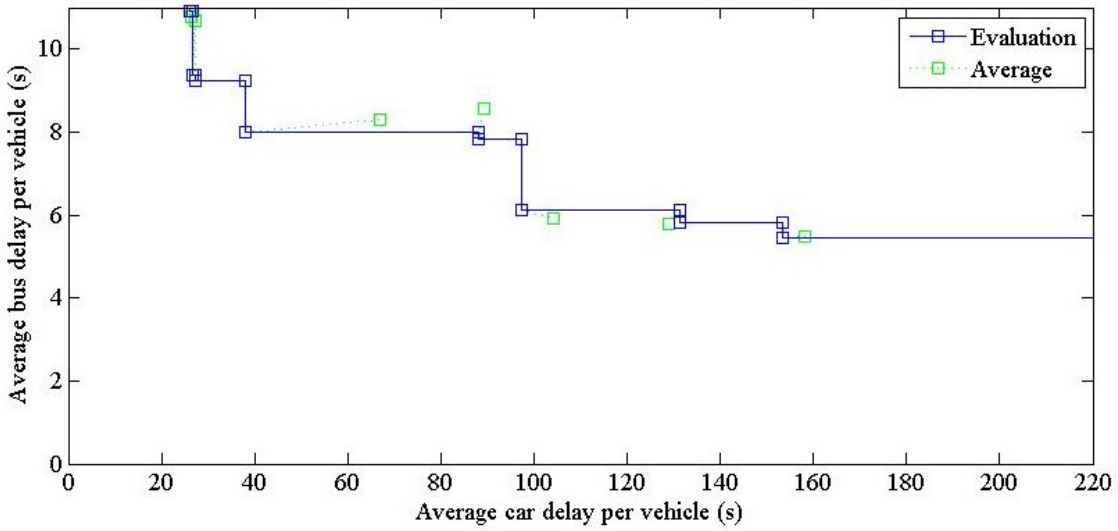

Figure 8: Estimates of the best objective values obtained in PAL with 5 simulations (in blue) and with a posterior evaluation (in green) of the same designs based on 50 simulations.

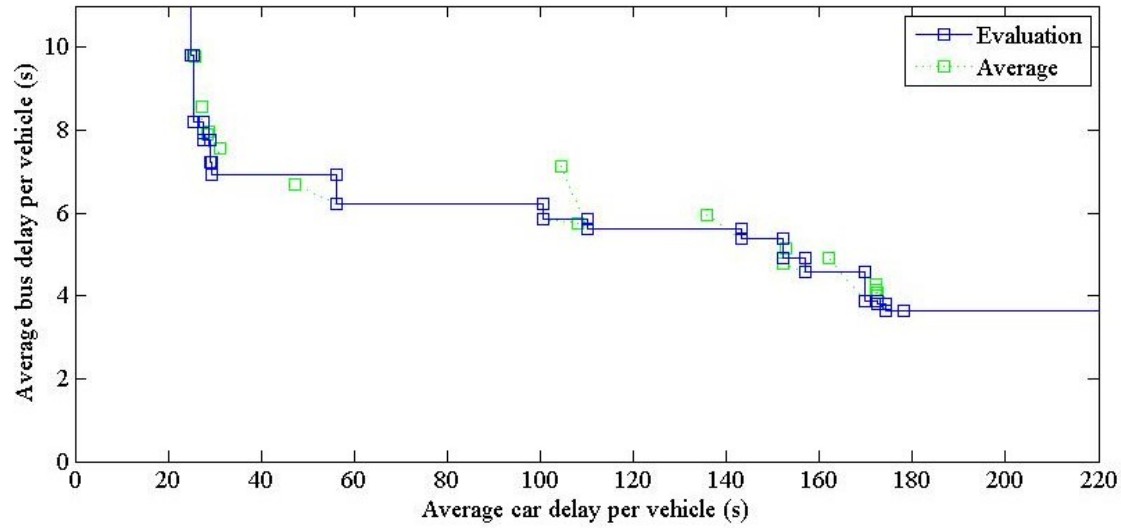

Figure 9: Estimates of the best objective values obtained in SMS-EGO with 5 simulations (in blue) and with a posterior evaluation (in green) of the same designs based on 50 simulations.

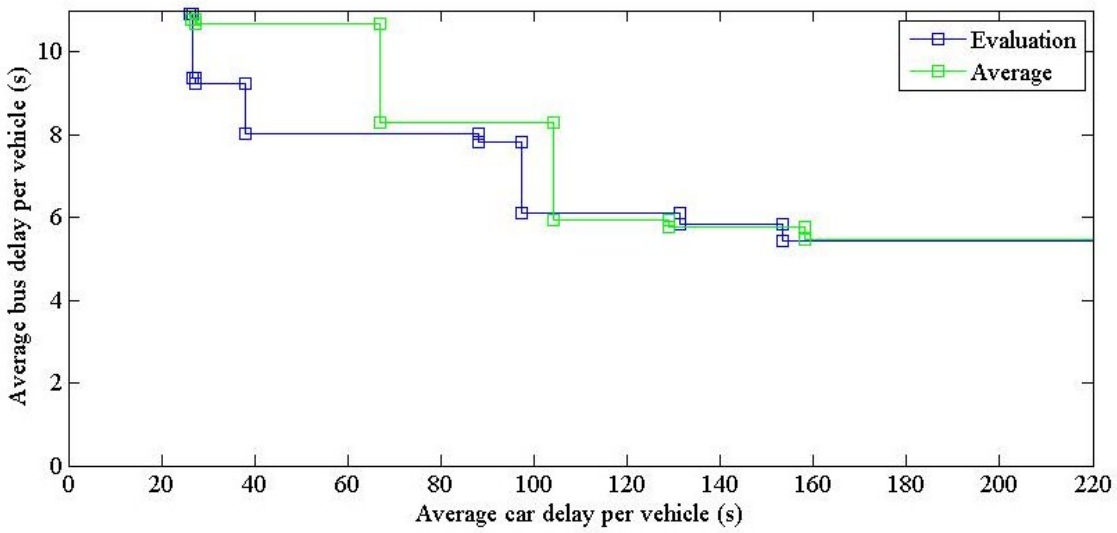

Figure 10: Estimates of the Pareto fronts obtained in PAL with 5 simulations (in blue) and with a posterior evaluation (in green) of the same designs based on 50 simulations.

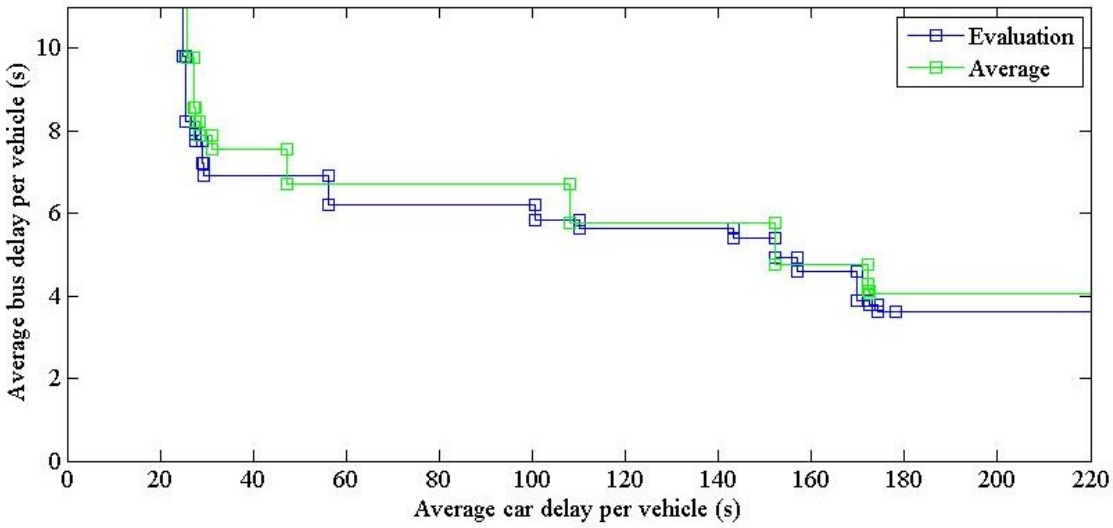

Figure 11: Estimates of the Pareto fronts obtained in SMS-EGO with 5 simulations (in blue) and with a posterior evaluation (in green) of the same designs based on 50 simulations.

Re-evaluating only the best configurations is not a major problem since it can be done in a relatively short amount of time. However, the substantial differences between the evaluated values and the real average values raise two other issues. First, since the selection of the best configuration is based on noisy evaluations, the configurations selected might not be the best evaluated so far in terms of average values. Thus, the approximation of the Pareto front might not

be optimal, given the points that have been evaluated. Second, the efficiency of the optimization process relies mostly on the accuracy of the meta-model. If the meta-model is not accurate because it cannot account for the noise, the optimization process is slowed down. Thus, even though SMS-EGO outperformed PAL without modeling the noise, this might eventually be another limitation.

III.5 Conclusion of the chapter

In this chapter, PAL and SMS-EGO were evaluated on a very small computational budget on three variations of a same road network and compared with NSGA-II. Based on the analysis of the final Pareto front and of the evolution of the hypervolume, SMS-EGO was found to outperform PAL, while PAL was found to perform as well as NSGA-II, if not better. Overall, these results are very encouraging.

However, two limitations still restrict the wide scale adoption of simulation-based multi-objective optimization. First, the computation time required is still sizeable. Indeed, in cases with only three decision variables, it took nearly 150 iterations for SMS-EGO to reach a sort of "final approximation". With six decision variables, we had to limit the number of iterations to 100 because the computation time was already exceeding 30 hours. Thus, in cases with many decision variables, SMS-EGO can only provide a first approximation of the Pareto front. To obtain finer results, it might be necessary to limit the decision space or to convert the problem into a single-objective one.

Second, the analysis of the final output showed that stochastic evaluations lead to a bias between the predicted approximation of the Pareto front and the actual average values. While it can be corrected by re-evaluating many times the best configurations identified, the selection process of the best configurations remains biased. We explained in the last part of this chapter how this phenomenon can lead to sub-optimal approximations of the Pareto front. In addition, inaccuracies in the meta-model may render it less useful for the optimization process. Thus,

although this bias stems from a more realistic stochastic model, it might penalize simulation-based optimization.

To conclude, meta-model based algorithms, and especially SMS-EGO, have showed very promising results for simulation-based multi-objective optimization. Nevertheless, as long as the optimization processes studied have difficulties handling large-scale and stochastic problems, they remain ill-adapted for transportation network design. Thus, we believe that these two issues are the first obstacles to overcome on the road towards a more realistic solution to the NDP.

Chapter IV Accounting for stochastic evaluations

IV.1 Introduction

The stochastic nature of evaluations raises two types of issues. First, as highlighted in the previous part, stochastic evaluations lead to an approximation of the Pareto front that is necessarily biased. Second, if the meta-model cannot account for some noise, a meta-model that maximizes the likelihood has to fit all the points exactly, even though some very fast variations in the output may in fact be caused by the noise. Thus, when several points have been evaluated in a very small part of the decision space, the meta-model is bound to under-estimate the length-scale of the variations and will make less accurate predictions. To limit these effects, the approach commonly taken is to increase the number of simulations run for each evaluation, so that the variance of each evaluation is reduced.

This standard approach has two obvious drawbacks. First, the number of simulations required is often important. Indeed, if the simulations are considered independent and identically distributed, the standard deviation of the average value of the objective decreases with the square root of the number of simulations run. For instance, in order to reduce the standard deviation to one tenth, then 10^2 simulations would be required. Second, since the decision-makers are ultimately only interested in the Pareto front, determining with precision objective values which are very far from the Pareto Front can appear as a waste of computation resources.

Instead, it has been decided to take inspiration from the approach suggested by Rasmussen and Williams [65] and already implemented in the algorithm PAL. This approach consists in including into the model the fact that evaluations are stochastic. Assuming an independent identically distributed Gaussian noise, it was shown in II.4.3 that the noise can be modeled simply by adding $\sigma_n{}^2.I_N$ to the covariance matrix ($\sigma_n{}^2$ represents the variance of the noise and I_N the identity matrix). Thus, since this approach would take into account the randomness of the evaluations, it would not be necessary to run multiple simulations for every

single configuration. However, since the assumption of an independent identically distributed Gaussian noise is very strong, we first study its validity.

IV.2 Analysis of the stochastic variations

In order to model the randomness of the evaluations as accurately as possible, we first suggest analyzing the characteristics of this randomness: is the assumption of a normally-distributed noise acceptable? Does this noise depend on the point considered?

To answer the first question, a single configuration was chosen in the previous *Cote des Neiges* case-study and 5000 evaluations (based on one simulation only) of the two objectives (average travel time per car and average travel time per bus) were carried out.

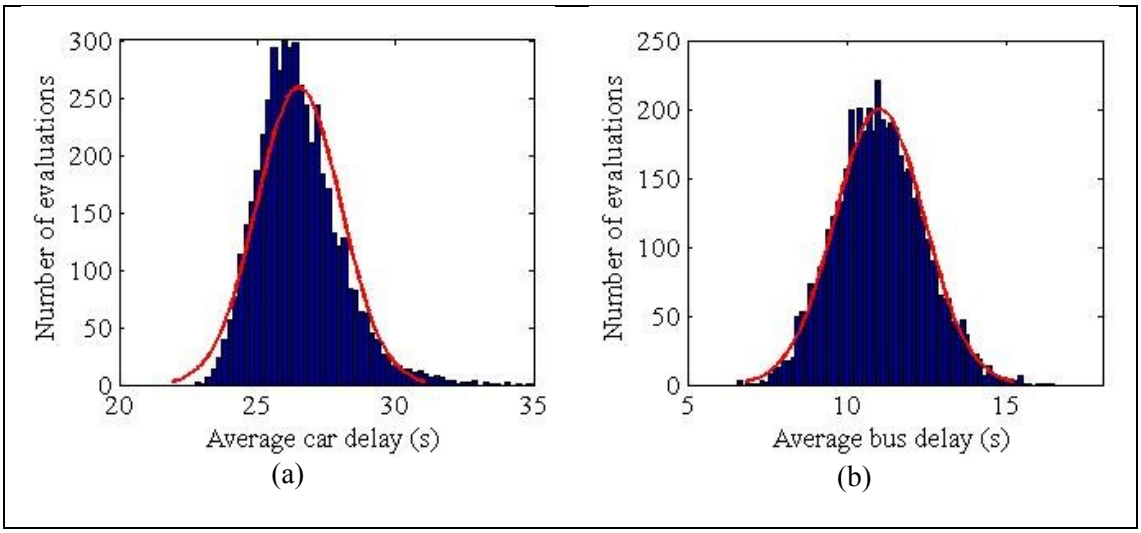

Figure 12: Distribution of the evaluations of average car delay (a) and average bus delay (b) for the "bus lane" configuration of the *Cote des Neiges* case-study and their approximations with normal distributions.

As shown in Figure 12, the normal approximation seems very reasonable for the average bus delay, but the distribution of the first objective appears to be positively skewed. To confirm these trends, one-sample Kolmogorov-Smirnov tests were performed on the normalized data for the null hypothesis that the data comes from a standard normal distribution. The p-values obtained were 0.1192 for the average bus delay and 2×10^{-14} for the average car delay. Thus, considering the 5000 evaluations, the Kolmogorov-Smirnov test rejects the Gaussian hypothesis for the average car delay but not for the average bus delay. To be more rigorous, the same

analysis should have been completed at many different points However, since Gaussian processes were chosen as meta-models, it would be highly computationally demanding to model a non-Gaussian noise. Thus, even though we knew that a Gaussian noise was a rough approximation, we still selected it because of computational benefits.

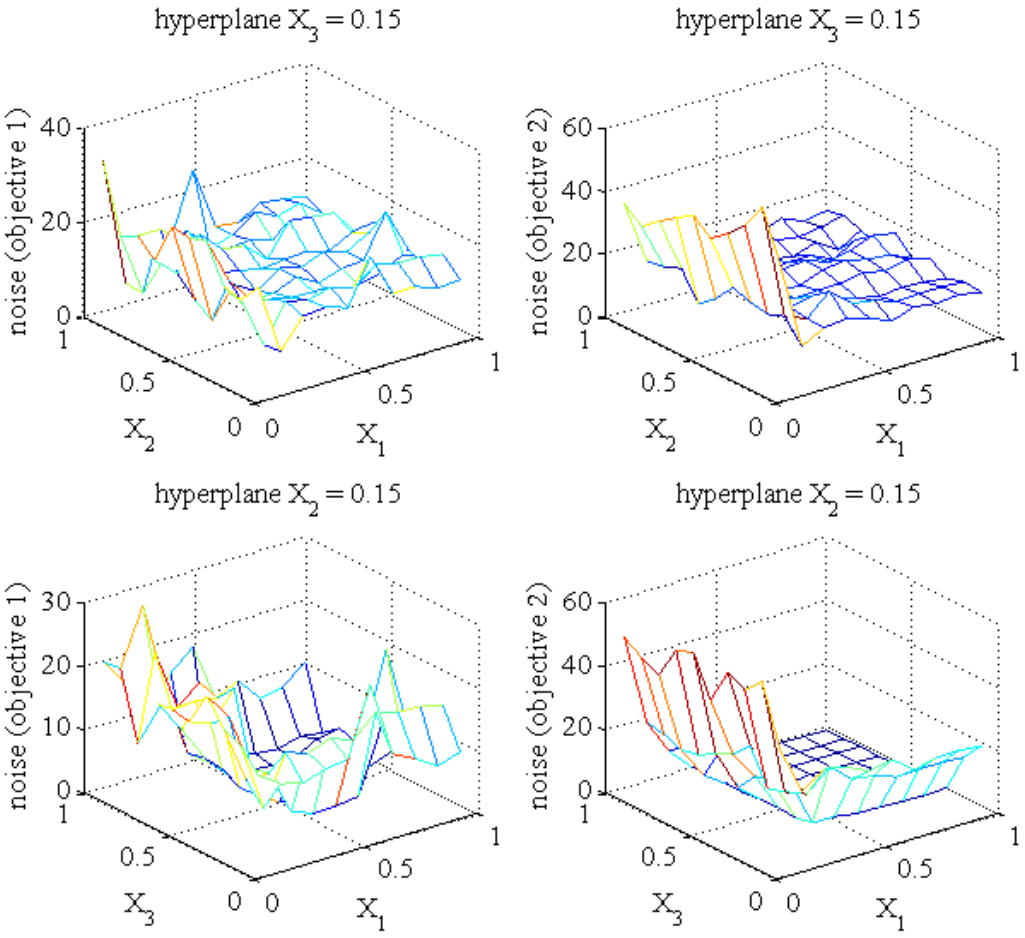

Figure 13: Representations of noise estimates for the first and second objectives in two different hyperplanes

In addition, the magnitude of this noise could depend on the point considered. Thus, to answer to the second question, the noise was mapped across the design space in the "bus-lane" configuration of the *Côte des Neiges* case-study from the previous chapter. Since the design space is of dimension 3, a grid of 10^3 points was created with values equal to 0.05, 0.15, … 0.95 on the three axes. For each point, 10 simulations were run to estimate the average value and the standard deviation of the noise. The results are shown for two hyperplanes in Figure 13.

For a more global overview, one can analyze the following statistics:

Objective 1	Noise	Ratio Noise/Objective
Minimum value	0.5179	0.0099
Maximum value	36.5247	0.4302
Mean value	9.2786	0.0643
Standard deviation	7.5111	0.0540
Objective 2	Noise	Ratio Noise/Objective
Minimum value	0.6896	0.0316
Maximum value	60.9095	0.3947
Mean value	11.5305	0.1043
Standard deviation	13.1414	0.0486

Table 2: General statistics for the noise estimates

As it can be seen in Table 2 and in Figure 13, the noise varies on a large scale. Thus, the hypothesis of a constant noise made in PAL is clearly not adapted for this problem. As an alternative, one could have thought that the noise might be proportional to the objective values. However, the analysis of the Noise/Objective ratio shows that this blatantly false too, with ratios varying for instance for objective 1 between 1% and 43%, with a standard deviation of 5.4%, compared to a mean of 6.4%.

To conclude this analysis, it turned out that the noise could be roughly approximated as Gaussian, but not as an identically distributed signal. In addition, the independence of noise measurements for different configurations is guaranteed by the choice of new random seeds for every configuration.

IV.3 Adaptation of the meta-model to stochastic evaluations

In section II.4.3 we presented how an independent identically distributed Gaussian noise can be included within a Gaussian process meta-model, as it had been done in PAL. To include an independent Gaussian noise that is not identically distributed, one can simply follow the same method and instead of equations 22 and 23, one obtains:

$$\mu_{N+1} = m(X_{N+1}) + K(X_{N+1}, X_{1\dots N})(K(X_{1\dots N}, X_{1\dots N}) + \Sigma)^{-1}(\overrightarrow{y_{1\dots N}} - \overrightarrow{m_{1\dots N}}) \quad \textbf{(28)}$$

$$\Sigma_{N+1} = K(X_{N+1}, X_{N+1}) - K(X_{N+1}, X_{1\dots N})(K(X_{1\dots N}, X_{1\dots N}) + \Sigma)^{-1}K(X_{1\dots N}, X_{N+1}) \quad \textbf{(29)}$$

Where $\begin{bmatrix}\epsilon_1 \\ \dots \\ \epsilon_N\end{bmatrix} \sim N(\vec{0}, \Sigma)$ and $\Sigma = \begin{pmatrix}\sigma_1^2 & 0 & \dots & 0 \\ 0 & \sigma_2^2 & \dots & 0 \\ \dots & \dots & \dots & 0 \\ 0 & 0 & 0 & \sigma_N^2\end{pmatrix}$.

However, these equations cannot be directly integrated within the EGO approach, which relies on the correlation coefficients rather than on the covariance. Moreover, the analysis of the literature shows that at least four research teams have endeavored to adapt the EGO approach to stochastic evaluations [74–77] but that they all considered the noise as identically distributed. Thus, the EGO approach was adapted to variable noise by rewriting the previous equations with Pearson's correlation coefficients. The results are given directly here and the detailed calculation steps are included in the Appendix.

$$\mu_{N+1} = m(X_{N+1}) + corr(X_{N+1}, X_{1\dots N})\left(R + \frac{\Sigma}{\sigma^2}\right)^{-1}(\overrightarrow{y_{1\dots N}} - \overrightarrow{m_{1\dots N}}) \quad \textbf{(30)}$$

$$\Sigma_{N+1} = \sigma^2\left(corr(X_{N+1}, X_{N+1}) - corr(X_{N+1}, X_{1\dots N})\left(R + \frac{\Sigma}{\sigma^2}\right)^{-1} corr(X_{1\dots N}, X_{N+1})\right) \quad \textbf{(31)}$$

Where $R = corr(X_{1\dots N}, X_{1\dots N})$ and σ^2 is the variance of the Gaussian process (see Appendix).

Interestingly, these results have exactly the same form as in the noise-free DACE/EGO framework, except for the term $\frac{\Sigma}{\sigma^2}$ that is added to the correlation matrix R. Thus, accounting for

the noise within the SMS-EGO source code seems relatively straight-forward. However, since neither Σ nor σ^2 is known, approximations of these terms are required.

First, Σ is a diagonal matrix with the variance of the noise at each of the configurations evaluated so far. In PAL, all these coefficients were chosen equal and their common value was considered as an hyper-parameter, determined for every model by maximizing the likelihood. With a variable noise, modeling the N coefficients by N new hyper-parameters would make the optimization of the likelihood a very large scale problem, which would be in itself already extremely time-consuming. An alternative, although still time-consuming, would be to describe this noise across the design space with another model, for instance polynomial.

Instead, since here the ratio noise/objective can take very high values (up to 43% according to Table 2), a hybrid approach was defined. To reduce this noise, we chose to keep evaluating each configuration based on five simulations, as it is usually done. However, to further improve the precision of our meta-model, the uncertainty was modeled as suggested by the previous part. This hybrid approach had two assets: not only it allowed us to access averages of objective values which were less noisy, but it also allowed us to estimate the standard deviation of this noise with the sample standard deviation. Thus, by including this value in the covariance matrix, we were able to model the noise without any additional hyper-parameter. In case a point is evaluated several times, the sample mean of its objectives and the associated sample standard deviation can simply be updated.

Second, one need to estimate σ^2. As a very simple solution, one could simply rely on the unbiaised sample variance $s^2 = \frac{1}{N-1}\sum_{i=1}^{N}(y_i - \bar{y})^2$ (where $\bar{y}$ is the sample mean). However, such an estimate would give more weights to the parts of the design space that have been more thoroughly explored, thus leading to an important bias. In the DACE/EGO approach, this variance is approximated by its Maximum Likelihood Estimate (MLE). To derive it, the

expression of the log-likelihood in the DACE/EGO framework is given directly here (it can be easily derived from equation 27):

$$\log\left(p(Y|X_1,\dots X_N)\right) = -\frac{1}{2}\left(Nlog(\sigma^2) + \log(|R|) + \frac{(Y-M)\,{}^tR^{-1}(Y-M)}{\sigma^2}\right) + constant \quad \textbf{(32)}$$

Thus, $\frac{\partial \log\left(p(Y|X_1,\dots X_N)\right)}{\partial(\sigma)} = 0$ is equivalent to:

$$\frac{\sigma.N}{\sigma^2} - \frac{(Y-M)\,{}^tR^{-1}(Y-M)}{\sigma^3} = 0 \quad \textbf{(33)}$$

And we obtain the MLE:

$$\widehat{\sigma^2} = \frac{(Y-M)^tR^{-1}(Y-M)}{N} \quad \textbf{(34)}$$

One can see that this is a generalization of the sample variance that takes into account the correlation between the different points.

In the case studied here, the addition of a noise implies the addition of $\frac{\Sigma}{\sigma^2}$ to the matrix R, which would make the derivation much more computationally intensive. For simplification purposes, we have decided to simply replace σ^2 in this expression by σ^2_{old}, the estimate of the variance obtained at the previous iteration. Thus, $R + \frac{\Sigma}{\sigma^2_{old}}$ is independent of σ and the MLE of the variance is:

$$\widehat{\sigma^2} = \frac{(Y-M)^t\left(R + \frac{\Sigma}{\sigma^2_{old}}\right)^{-1}(Y-M)}{N} \quad \textbf{(35)}$$

For the first iteration, since the initial points are chosen randomly with a LHS design, the sample variance was used.

IV.4 Other adjustments

In order to integrate this new meta-model into SMS-EGO other adjustments were necessary. Indeed, when SMS-EGO was first run a few times with the new meta-model, it systematically stopped progressing after approximately 120 iterations because the same point kept being evaluated. This point was also the initial point of the algorithm used to select points based on the expected improvement (CMA-ES). The analysis of the error showed that CMA-ES was stopping prematurely at its first iteration because all the 8 points of the first generation had a fitness value of 0.

To understand why CMA-ES stops, a minimum knowledge of its working principles is necessary. CMA-ES is an evolution strategy, i.e. it progresses towards some local minimum by evaluating a new generation of points at every iteration. These points are sampled based on a multivariate normal distribution that is supposed to be centered around the area with the best fitness and which is modified at each generation depending on the new data. To start, CMA-ES requires an initial distribution and a population size. In SMS-EGO, the initial distribution was centered around the center of the decision space and its standard deviation is one fourth of the range of the decision space. After the first generation, the multivariate normal distribution is modified based on the new evaluations. If all the evaluations give equal results, the fitness function is assumed to be flat and in its default configuration, SMS-EGO stops.

To explain why all the initial points had a fitness value of zero, we need to explain how the expected improvement is computed. One of the reasons of this phenomenon is the ε-dominance relationship used in SMS-EGO. When minimizing two objectives, a point in the objective space is said to be ε-dominated by another point if it is dominated by this point translated by a vector $(-\varepsilon, -\varepsilon)$ $(\varepsilon > 0)$. Thus, four cases can occur:

- Case 1: the predicted objective point is dominated by the current Pareto front,

- Case 2: the predicted objective point neither dominates nor is dominated by any point of the current Pareto front but it is ε-dominated by at least a point of the current Pareto front,
- Case 3: the predicted objective point is ε-dominated by the current Pareto front but it also dominates at least one point of the Pareto front,
- Case 4: the predicted objective point is not ε-dominated by the current Pareto front.

In the two first cases, a penalty (negative expected improvement) was given to the point, depending on its relative position compared to the current Pareto front. In the fourth, the fitness value was the expected improvement (positive). However, in the third case, a fitness value of zero was given. These different cases were plotted for a hypothetical case-study in Figure 14.

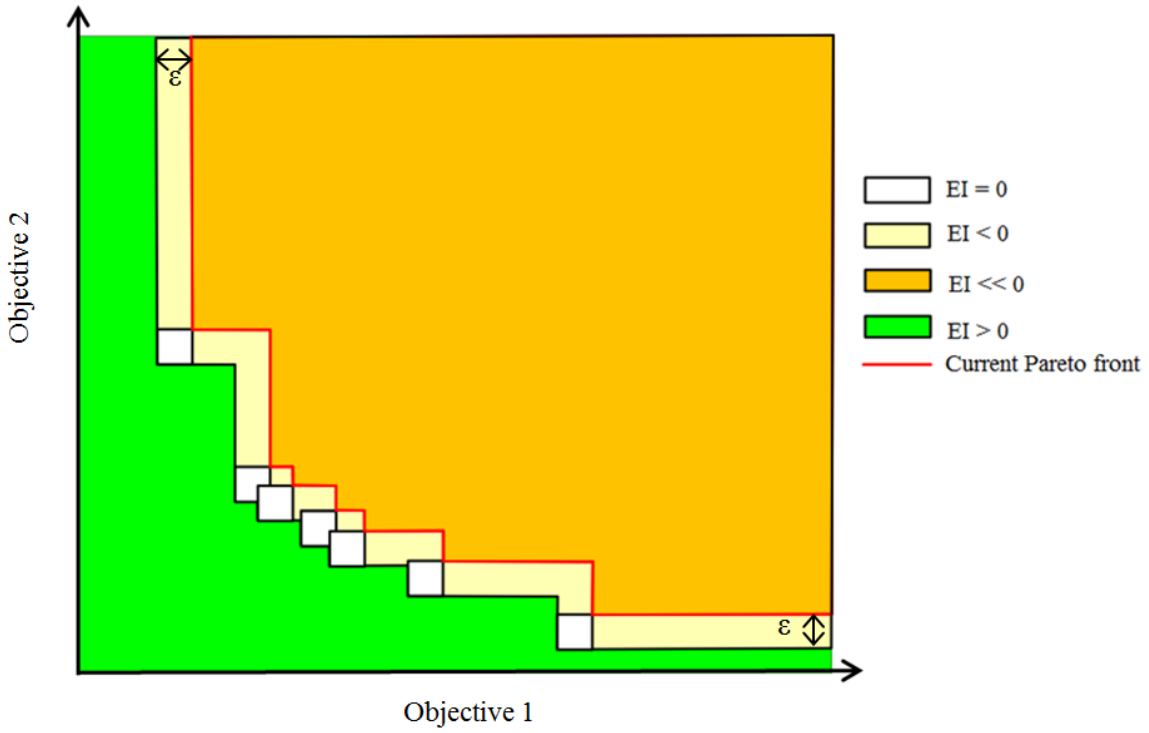

Figure 14: Graphical representation of the expected improvement for a hypothetical case-study

One can see that in the areas where the Pareto front is more finely defined, the fitness function is equal to zero on a substantial part of the objective space. Thus in some very specific

cases, the concept of *ε-dominance*, introduced to guarantee some diversity in the Pareto front, can backfire and stop the optimization process.

At least one reason could potentially explain why this phenomenon commonly occurs with the new meta-model but does not with the original one. As the new meta-model accepts some noise, it does not strictly interpolate the points on the Pareto front. Thus, since these evaluations are likely to be especially biased, the meta-model may predict less good performances than the original one. Consequently, the lower confidence bound might fall in the epsilon dominated area more often with the new algorithm than with the original.

To prevent this phenomenon, two adjustments have been implemented. First, the calculation of the current Expected Improvement was done based on the predicted values at the evaluated points. Second, the initial point of CMA-ES has been replaced by a random variable that changed at every iteration.

IV.5 Case-study

In the remainder of the thesis, the optimization process resulting from the combination of SMS-EGO with the new meta-model accounting for noisy evaluations is referred to as the "variant 1".

Since the aim of variant 1 was to improve the precision of the meta-model, we first assessed potential changes in the global accuracy and evaluated their impact on the optimization process. Then, we analyzed whether this meta-model could also be used to select the best points for the final approximation of the Pareto front.

IV.5.1 Global accuracy of the meta-models

In order to evaluate the accuracy of the meta-models in the optimization process, 50 test points (i.e. sets of values for the decision variables) were chosen at random for the "bus lane" configuration and precise estimates of their performances were obtained by averaging the results of 20 simulations for each point. Then, the variant 1 and the original version of SMS-EGO were run 20 times for 150 iterations on the same case-study and all the meta-models were saved.

Finally, the 150 meta-models obtained for each of the 40 runs were tested on the 50 test points to compute the relative errors displayed in Figure 15.

The analysis of Figure 15 shows that the two meta-models had a very similar accuracy for the first 150 iterations and that their accuracy globally improved with the iterations. On average however, it seems that for both objectives, the meta-model used in Variant 1 is globally slightly more accurate than the original meta-model.

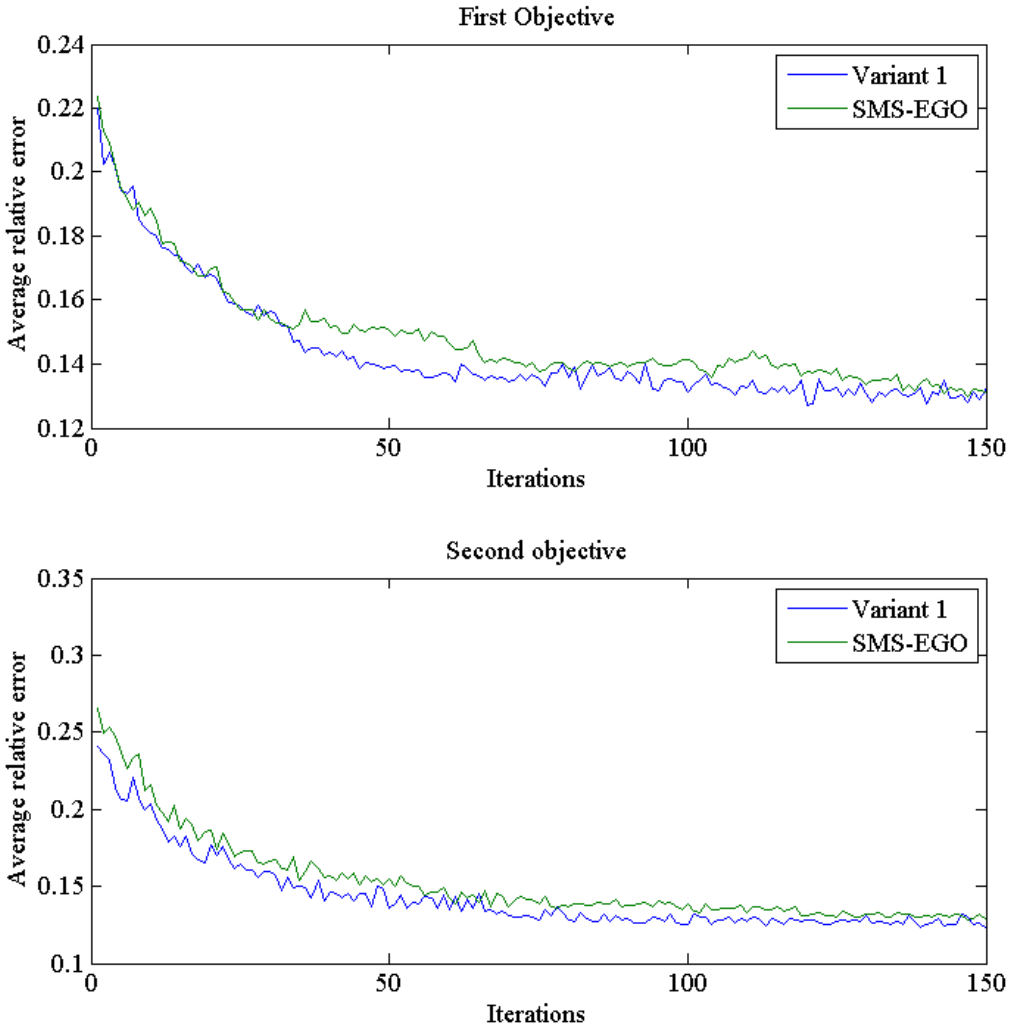

Figure 15: Evolution of the average relative errors of the meta-models for 20 runs of SMS-EGO and Variant 1 on the "bus-lane" case study with three decision variables

IV.5.2 Impact of the new meta-model on the optimization process

Changes in the meta-model influence the choice of the points to be evaluated and therefore, they are also expected to impact the efficiency of the optimization process. However, in the previous study, the accuracies of the two meta-models were very similar, even after 150 iterations. Thus, it was not clear a priori whether the new meta-model would significantly accelerate the optimization process. To answer this question, the evolution of the average hypervolume of the Pareto front approximations obtained over the 150 iteration of the 20 previous runs of SMS-EGO and variant 1 were plotted in Figure 16. The evolution of the average hypervolume was also plotted over computation time in Figure 17. To avoid any bias caused by the selection process, the hypervolume should have been computed at every iteration based on new evaluations of the best solutions. However, if the bias is the same for the two optimization processes, it does not impact their comparison. Thus, to avoid additional computations, the calculations of the hypervolumes were based on the values from the evaluations made during the optimization process.

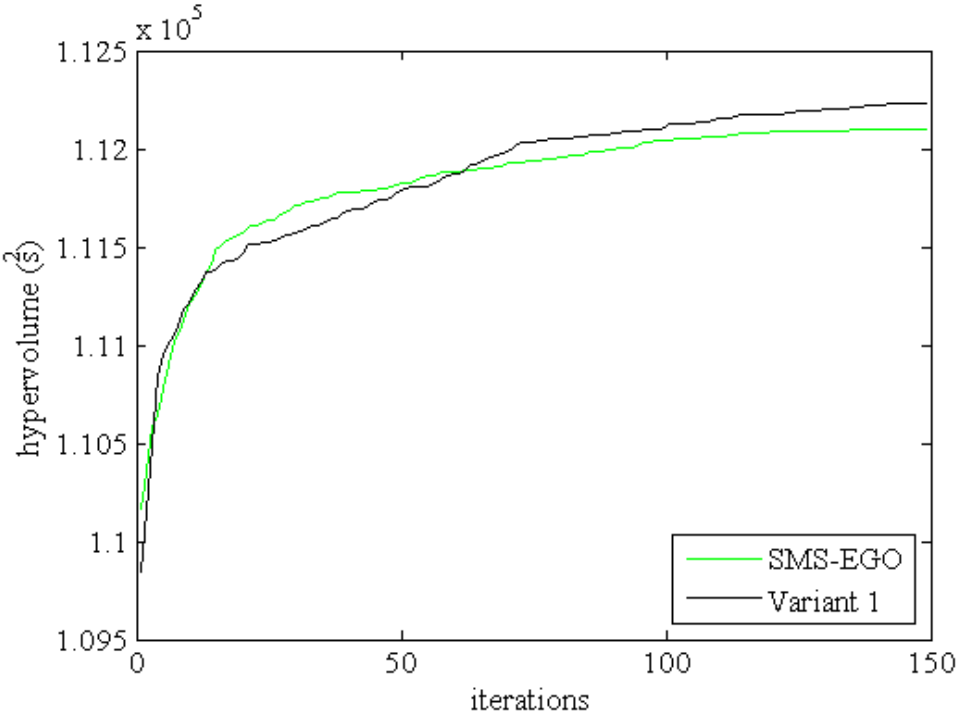

Figure 16: Comparison of the evolution of the average hypervolume over 150 iterations in 20 runs of SMS-EGO and of the variant 1, on the "bus lane" configuration.

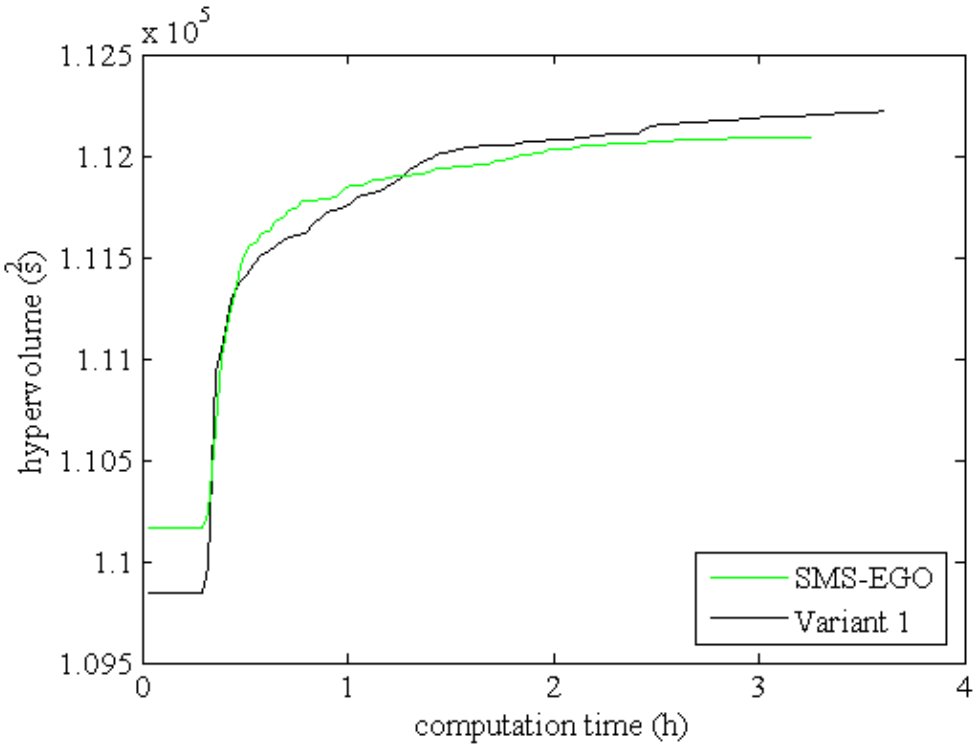

Figure 17: Comparison of the evolution of the average hypervolume over time in 20 runs of SMS-EGO and of the variant 1, on the "bus lane" configuration.

The analysis of Figure 16 and Figure 17 shows that the optimization process can be split in three parts. At the very beginning (before 20 iterations), the two algorithms had very similar performances. Then, between approximately the 20th and the 60th iteration, the original version of SMS-EGO performed better than the variant 1. Finally, after the 60th iteration, the variant 1 outperformed the original version of SMS-EGO. In order to compare the two algorithms on a more rigorous basis, two-tailed Student's *t* tests were performed at every iteration and at regular time intervals. For independent samples of equal size and different variances, the formula of the *t* statistic is:

$$t = \frac{\overline{X_1} - \overline{X_2}}{\sqrt{\frac{(s_1^2 + s_2^2)}{n}}} \tag{36}$$

And the degree of freedom to be used in significance testing is given by:

$$\boldsymbol{d.f.} = \frac{\left(\boldsymbol{s_1^2 + s_2^2}\right)^2 (\boldsymbol{n-1})}{\boldsymbol{s_1^4 + s_2^4}} \tag{37}$$

Where s_1^2 and s_2^2 represent the unbiased estimators of the variances.

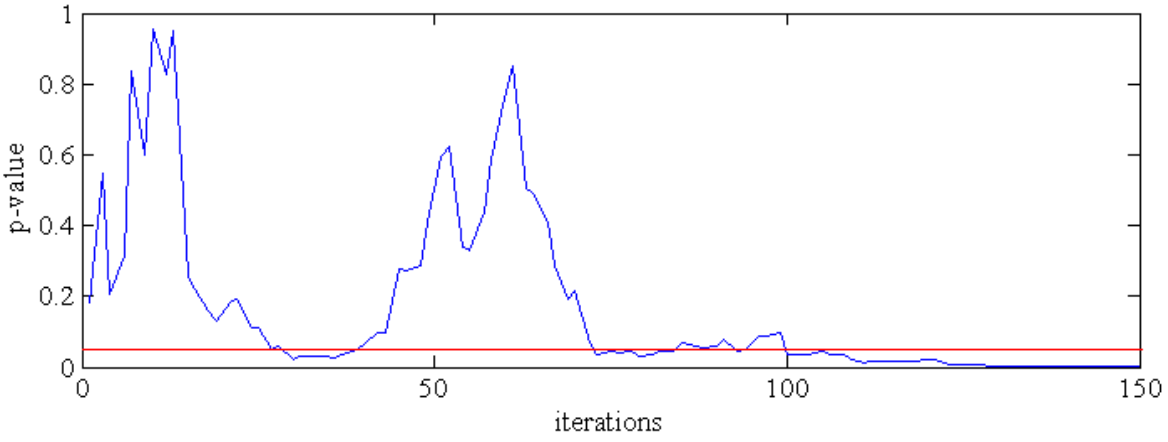

Figure 18: Evolution over the iterations of the p-value obtained with a two-tailed Student's *t* test comparing the hypervolumes obtained with SMS-EGO and variant 1

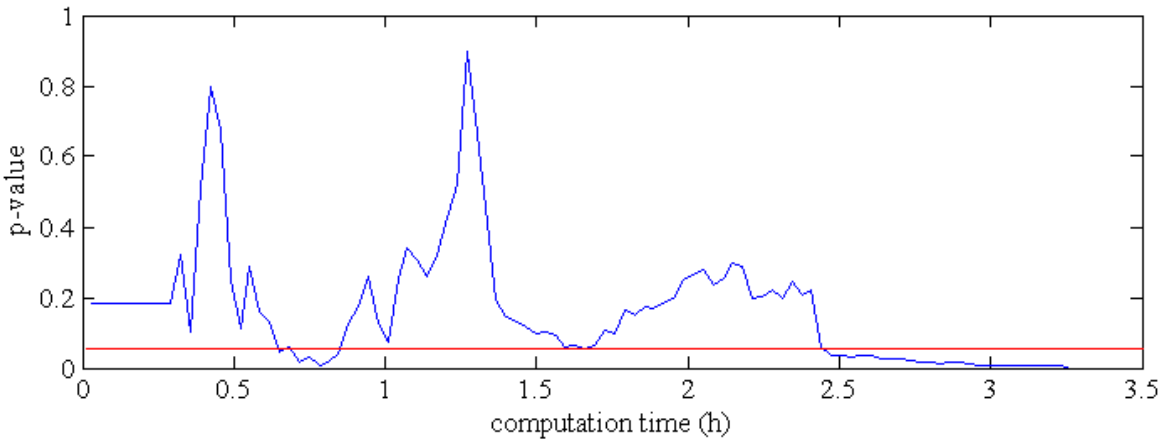

Figure 19: Evolution over the computation time of the p-value obtained with a two-tailed Student's *t* test comparing the hypervolumes obtained with SMS-EGO and variant 1

The analysis of Figure 18 and Figure 19 confirmed the trends observed in Figure 16 and Figure 17. Indeed, around the 30th iteration (or after 45 min), the traditional significance level of 0.05 (in red on the figures) was reached and it was also reached for all the tests after the 100th iteration (or after 2 h 30 min). The tests after the 125th iteration (or 3h) even reached the more rigorous significance level of 99%.

IV.5.3 Selection of points for the final Pareto front approximation

Since evaluations are noisy and only the best of them are kept for the final approximation of the Pareto front, this approximation is bound to be biased. To avoid this bias, the final output should be a set of points that are selected and then re-evaluated a statistically significant number of times.

Nevertheless, one still has to choose the points to be re-evaluated. In this thesis, for the original version of SMS-EGO, only the points that had already been evaluated and that were not dominated by any other evaluated point (the Pareto front based on evaluations) were kept. Such an approach has already been followed in Stevanovik et al. [54]. Alternatively, since the meta-model can provide an estimated value at any point, one could also select points that have never been estimated. However, this would lead to a new optimization problem on a continuous design

space with a number of decision variables equal to the initial dimension of the problem multiplied by the number of points chosen to describe the Pareto front.

For algorithms that do model uncertainty however, one can choose between selecting the best points based on their evaluations or based on their predicted value (these two values were identical in the original version of SMS-EGO). To guide the choice, it is proposed to compare the accuracy and bias of both estimates *for the best configurations only*. For this purpose, only the 20 runs of variant 1 were used. For each run, among all the evaluated configurations, the non-dominated ones were selected, either based on their evaluation (themselves based on 5 simulations), or based on their predicted value. For each of these configurations, 20 new simulations were carried out. Then, the average values of the new 20 simulations were compared with the two estimates (predicted values or averages from 5 simulations).

Statistics	Formula	Average (5 simulations)	Prediction
Average relative error (Objective 1)	$\overline{\left\lvert\frac{Y^1_{pred} - Y^1_{AVG}}{Y^1_{AVG}}\right\rvert}$	4.0 %	4.2%
Average relative error (Objective 2)	$\overline{\left\lvert\frac{Y^2_{pred} - Y^2_{AVG}}{Y^2_{AVG}}\right\rvert}$	7.5 %	5.3 %
Average relative bias (Objective 1)	$\overline{\left(\frac{Y^1_{pred} - Y^1_{AVG}}{Y^1_{AVG}}\right)}$	-1.3 %	-1.7 %
Average relative bias (Objective 2)	$\left(\overline{\frac{Y^2_{pred} - Y^2_{AVG}}{Y^2_{AVG}}}\right)$	-5.9 %	-3.4 %

Table 3: Comparison of the average relative error and average relative bias between the meta-model of variant 1 and the average of the evaluated values

Statistics for the 20 runs are given in Table 3. The trends observed concerning the average relative error and the average relative bias are very similar. Indeed, the new meta-model was both slightly less accurate (0.2 %) and slightly more biased (-0.4 %) for the first objective but

it was significantly more accurate (2.2 %) and less biased (2.5 %) for the second objective. Thus, around the best evaluated points, the new meta-model seemed overall beneficial. These local results were consistent with the results already obtained in the previous part at a more global scale.

Ultimately, what matters the most is the quality of the final Pareto front approximation. To estimate it, the hypervolume of the Pareto fronts that would have been obtained when the best configurations were selected based on their evaluations and when they were selected based on their predicted values were computed for the 20 runs of Variant 1. To avoid any bias, the hypervolumes were computed using the new evaluations of the objective functions. With a selection process based on previous evaluations, the average hypervolume was found to be 1.1166×10^5. With a selection process based on predicted values, the average hypervolume was found to be slightly bigger: 1.1172×10^5. In order to determine the significance of this difference, a two-tailed Student's t test was carried out using the equations 36 and 37 and a p-value of 0.22404 was obtained. Thus, although prediction-based selection seemed to lead to better approximations of the Pareto front, the results were not significant enough and more tests would be required. Possible explanations for this relatively large p-value are the small number of optimization processes, their high variability and the fact that only 20 simulations were used to determine the new average values.

For illustration purposes, the estimates of the Pareto front based on the two indicators obtained for one run of variant 1 were plotted in Figure 20. As it was expected, one can see that the Pareto front found based on the evaluated values seemed to dominate the one obtained with the predicted values. Nevertheless, the analysis of Figure 21 which shows the approximations obtained after 20 new simulations, proves that it was in fact the opposite and that evaluated values were simply more biased. As another benefit, the Pareto fronts based on the predicted values were slightly more precisely detailed in almost all the runs.

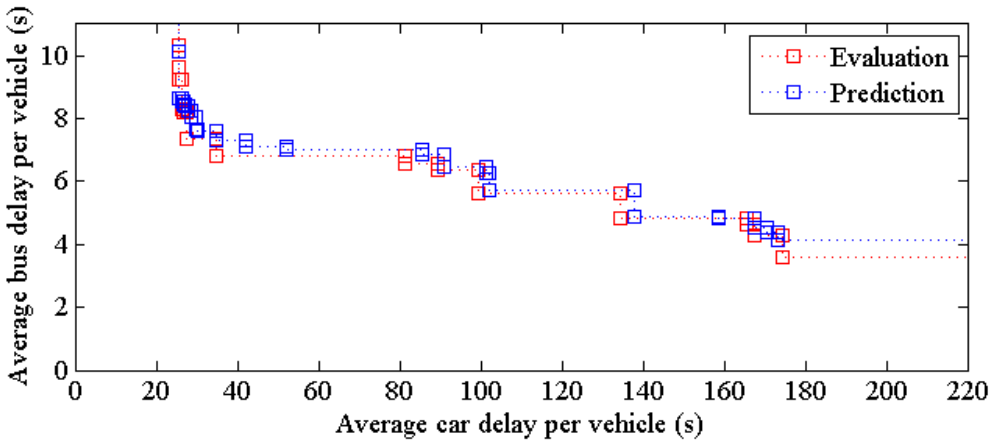

Figure 20: Estimates of the Pareto fronts approximations obtained in variant 1 when the selection process is based on evaluations (red) and when the selection process is based on predicted values (blue)

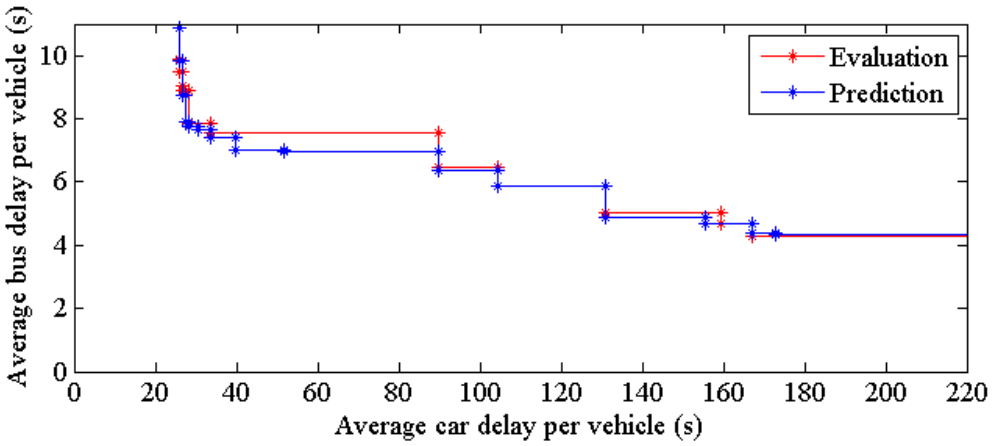

Figure 21: Comparison of the Pareto front approximations obtained in variant 1 with 20 new simulations when the selection process is based on evaluations (red) and when the selection process is based on predicted values (blue)

IV.5.4 Combination of the impacts on the optimization process and on the final selection

Finally, we analyzed the combined benefits of this new meta-model for the optimization process. For this purpose, we compared:

- The average hypervolume (based on new evaluations) obtained at the 150th iteration of 20 runs of the original version of SMS-EGO with a selection of the best configurations based on the evaluated values.

- The average hypervolume (based on new evaluations) obtained at the 150th iteration of 20 runs of the variant 1 with a selection of the best configurations based on the predicted values.

A new two-tailed Student's *t* test gave the following results: $t = -2.0476$, $df = 37.58$, $p = 0.047635$. Thus, the significance level of 0.05 was reached. However, one can note that the p-value is not as small in this case as in section IV.5.2. To explain this phenomenon, it is suggested to compare the mean difference and standard deviations presented in Table 4.

	Student's *t* test of part IV.5.2		Student's *t* test of part IV.5.4	
Algorithm	SMS-EGO	Variant 1	SMS-EGO	Variant 1
Selection based on	Evaluations	Evaluations	Evaluations	Predictions
Hypervolume based on	Evaluations	Evaluations	New average values	New average values
Standard deviation	102.78	114.82	158.83	142.84
Mean difference	132.38		97.806	
p-value	5×10^{-4}		4.8×10^{-2}	

Table 4: Comparison of the two-tailed Student's t tests realized at the 150th iteration in part IV.5.2 and IV.5.4

The analysis of the data in Table 4 showed that the increase in the p-value from 5×10^{-4} in part IV.5.2 to 4.8×10^{-2} in the previous t-test was caused both by an increase in the standard deviations and by a smaller difference between the mean values of the hypervolumes. Thus, when new average values were used to compute the hypervolume, the difference in terms of performance between SMS-EGO and variant 1 was not as marked. A possible explanation for this phenomenon might be the small number of simulations (20) that were used to compute the average.

IV.6 Conclusion of the chapter

In this chapter, we proposed to account for noisy evaluations in the meta-model used in SMS-EGO. Based on a rapid analysis, it turned out that the noise could be roughly modeled as a

Gaussian independent, but not identically distributed signal and SMS-EGO was modified consequently to create another version that was called the "variant 1".

A case-study was then carried out to assess the impacts of this modification. First, it showed that the new meta-model was on average slightly more accurate than the original one, both globally and for the best solutions. The analysis of the evolution of the hypervolume over time then demonstrated that this gain in the accuracy of the meta-model allowed for a more efficient optimization process with a significance level of 0.01 after 150 iterations. Second, the potential benefits of this new meta-model for the selection of the points used in the final Pareto front approximation were investigated based on new average values. Although the results were not statistically significant, selecting the best solutions based on their predictions seemed to lead on average to Pareto fronts with better hypervolumes. Finally, the impact on the optimization process of these two combined modifications was evaluated by comparing the hypervolumes obtained using the new average values. Based on 20 optimization processes run with each algorithm and on average values obtained from 20 new simulations, the variant 1 was found to outperform the original version of SMS-EGO at the 150^{th} iteration at the traditional 0.05 significance level.

Chapter V Reducing the optimization computation time

V.1 Introduction

In Chapter IV, the optimization process SMS-EGO was adapted to the problem studied in this thesis by accounting for stochastic evaluations. In the case-study, this modification was found to accelerate the optimization process by improving the quality of each iteration. However, even though the number of evaluations was very limited, the computation time remained considerable and it seems difficult to further reduce the number of evaluations required significantly.

In this chapter, the opposite approach was investigated: instead of improving the quality of each iteration, different ways to reduce their computation time were investigated. As explained in Figure 2, each iteration of any meta-model based optimization process can be split in three main tasks:

- Finding a meta-model that maximizes the likelihood of previous data,
- Finding a new configuration that maximizes the expected improvement,
- Evaluating the new configuration (with Vissim in this thesis).

In order to determine the tasks that should be accelerated in priority, the repartition of the computation time among them was analyzed for a run of the original SMS-EGO algorithm. For this purpose, the MATLAB functions *tic* and *toc* were used to implement internal stopwatch timers and record the computation time required by each of these tasks and by an entire iteration. The results are shown in Figure 22 for the *Côte des Neiges* "bus-lane" case-study.

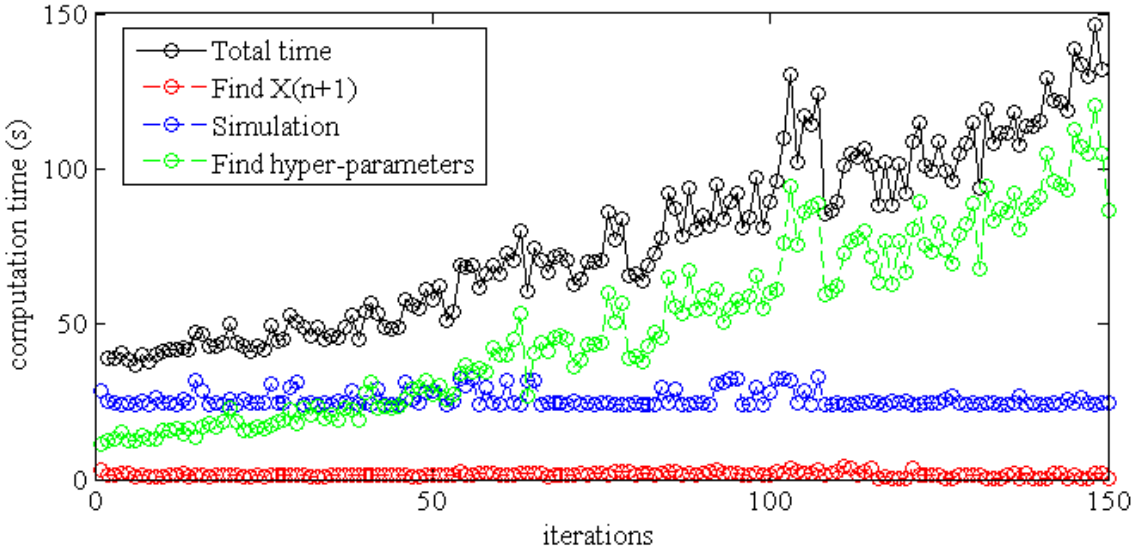

Figure 22: Repartition of computation time among the main tasks

Figure 22 shows that while actual simulations represented most of the computation time at the first iterations, very rapidly, the major time-consuming task became determining the hyper-parameters that best explained the observed data. Indeed, not only the time associated with this task tended to increase with the number of iterations but the increases were also more important at every iteration. Finally, it accounted for almost the entire computation time. Thus, in this chapter, efforts were focused on reducing the time required to determine new hyper-parameters.

V.2 Possible avenues to reduce the computation time

Although highly time-consuming, the determination of the hyper-parameters is a crucial step in the optimization process and it should not be overlooked. Indeed, the hyper-parameters are used to describe the length-scales within the correlation matrix. If these length-scales are under-estimated, this allows for variations that are shorter than the real-process variations and thus increases the uncertainty of the predictions. Conversely, if the length-scales are over-estimated, rapid variations within the real-process might not be described by the meta-model, which would decrease the goodness of fit. This phenomenon is very clearly illustrated in the Figure 23, taken from Rasmussen and Williams [65], where l represents the length-scale. Thus, a botched

determination of the hyper-parameters would lead to a less efficient algorithm and more iterations.

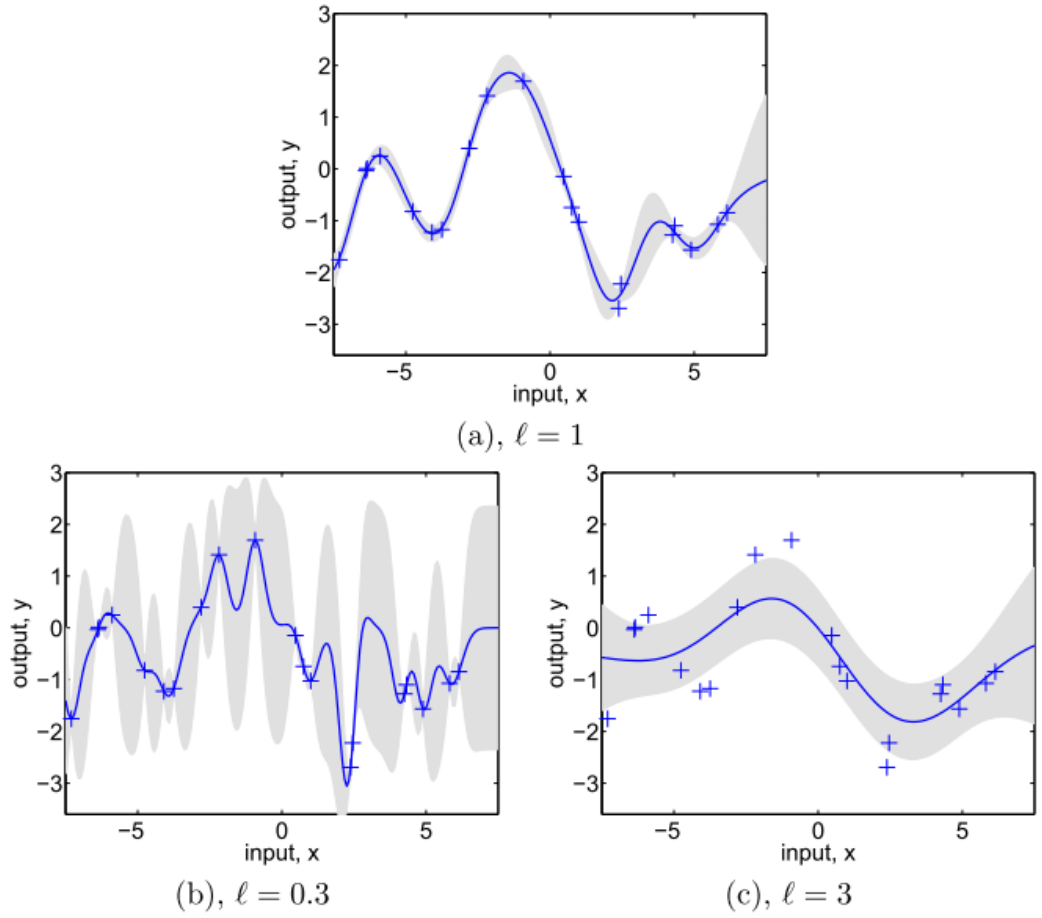

Figure 23: Influence of the length-scale of the Gaussian process on the meta-model [65]

To accelerate the determination of the hyper-parameters without compromising on the quality, two options were considered. First, one could replace the algorithm used to maximize the likelihood of the previous data by a more efficient one. Finding the most adequate algorithm is usually a task that requires a very good knowledge of the many diverse optimization techniques existing. However, in our case, the comparison of PAL and SMS-EGO is very instructive. Indeed, both algorithms determine the values of hyper-parameters by maximizing the likelihood of previous data but while SMS-EGO relies on a gradient-free optimizer, a gradient-based method is used in PAL. Since computing the gradient might be time-consuming, it is not immediately clear

which approach is the most time-efficient. In fact, according to Rasmussen and Williams [65]: *"Once K^{-1} is known, the computation of the derivatives [...] requires only time $O(N^2)$ per hyper-parameter. Thus, the computational overhead of computing derivatives is small, so using a gradient based optimizer is advantageous."* However, according to Lophaven et al., authors of the DACE toolbox for MATLAB, the *"computation of the gradient [...] with respect to the components of θ [the hyper-parameters] is possible, but would involve considerable extra effort"*[78]. Thus, it might depend on other choices, such as the mean function chosen for the Gaussian process. This is the first avenue that was explored in this chapter.

Second, one could choose to update the hyper-parameters less often. While this would in general lead to a decrease in the quality of the meta-model, it might not be the case if the hyper-parameters are updated at the "right" iterations. In fact, as explained in the first paragraph, the maximum likelihood function is used to select the hyper-parameters that best explain the knowledge of the real process that we have acquired so far. Thus, if a newly observed value is perfectly in agreement with what was observed earlier, it is very likely that the new "best" hyper-parameters will be very close to the previous "best" ones. Therefore, especially toward the end of the optimization process, one might expect the hyper-parameters to converge toward some values associated with the real underlying process. This is the second alternative that was investigated in this chapter to reduce computation time.

V.3 Gradient-based algorithms

V.3.1 Computing the derivatives

In Rasmussen and Williams [65], the authors show that the derivatives of the likelihood in equation 27 can be simply obtained by

$$\frac{\partial}{\partial\theta}\log\big(p(y|X)\big) = = \frac{1}{2}tr\left((\alpha\alpha^t - (K+\sigma^2 I_N)^{-1})\frac{\partial K}{\partial\theta}\right), \quad \textbf{(38)}$$

Where θ is an hyper-parameter of the covariance function and $\alpha = (K+\sigma^2 I_N)^{-1}Y$. As the authors explained, one can see that the computational overhead required to compute these

derivatives is small since $(K + \sigma^2 I_N)^{-1}$ has already been determined to estimate the log-likelihood.

In the DACE/SMS-EGO framework, the log-likelihood takes a different form. Indeed, by injecting the MLE of the variance obtained in equation (34) into equation (32), the "concentrated" log-likelihood is obtained:

$$\log_{\text{EGO}}\left(p(Y|X_1, \dots X_N)\right) = -\frac{1}{2}(Nlog(\hat{\sigma}^2) + \log(|R|)) + constant\,. \quad \textbf{(39)}$$

Then, as explained by Sacks, Welch, Mitchell and Wynn [42], maximizing the concentrated log-likelihood is equivalent to minimizing

$$\Psi(\theta) = |R|^{\frac{1}{N}}.\hat{\sigma}^2\,. \quad \textbf{(40)}$$

Because of its simplicity, Ψ is the function that is minimized in SMS-EGO to determine the values of the hyper-parameters. Let us compute its derivatives with respect to the hyper-parameter θ:

$$\frac{\partial \Psi}{\partial \theta} = \frac{\partial |R|^{\frac{1}{N}}}{\partial \theta}.\hat{\sigma}^2 + |R|^{\frac{1}{N}}.\frac{\partial \hat{\sigma}^2}{\partial \theta} \quad \textbf{(41)}$$

Let us compute these two summands separately. For the first term:

$$\frac{\partial |R|^{\frac{1}{N}}}{\partial \theta}.\sigma^2 = \frac{1}{N}\frac{\partial |R|}{\partial \theta}.|R|^{\frac{1}{N}-1}.\hat{\sigma}^2 = \frac{1}{N}|R|.tr\left(R^{-1}.\frac{\partial R}{\partial \theta}\right).|R|^{\frac{1}{N}-1}.\hat{\sigma}^2 \quad \textbf{(42)}$$

i.e.

$$\frac{\partial |R|^{\frac{1}{N}}}{\partial \theta}.\sigma^2 = \frac{1}{N}tr\left(R^{-1}.\frac{\partial R}{\partial \theta}\right).|R|^{\frac{1}{N}}.\hat{\sigma}^2 \quad \textbf{(43)}$$

For the second term:

$$|R|^{\frac{1}{N}}.\frac{\partial\hat{\sigma}^2}{\partial\theta} = |R|^{\frac{1}{N}}.\frac{\partial\dfrac{(\mathrm{Y}-\mathrm{M})^t R^{-1}(\mathrm{Y}-\mathrm{M})}{N}}{\partial\theta} \quad \textbf{(44)}$$

The question of the mean function, which had previously been left aside, should now be addressed. In PAL as in SMS-EGO, the mean is chosen as a constant. In PAL, the value of this constant is determined as any other hyper-parameter by maximizing the likelihood. In SMS-EGO however, this constant, written β, is determined by the method of the Generalized Least Squares:

$$\beta = (F^t R^{-1} F)^{-1} F^t R^{-1} Y \quad \textbf{(45)}$$

Where F is simply the $N \times 1$ vector filled with ones. Since the inverse of the correlation matrix has already been computed, this is a very fast way to determine the mean. However, as the mean depends on the correlation matrix, it also depends on the correlation hyper-parameters, which complicates the computation of the derivatives. To avoid any computational overhead, it was proposed in this thesis to use as a mean the value β_{old} computed with the Generalized Least Squares at the previous iteration, which is a constant independent from the new hyper-parameters. For the first iteration, since the points were well distributed and no previous value was available, the sample mean was used.

Thus:

$$M = \begin{bmatrix} \beta_{old} \\ \dots \\ \beta_{old} \end{bmatrix}_{N\times 1} \quad \textbf{(46)}$$

And

$$|R|^{\frac{1}{N}}.\frac{\partial\hat{\sigma}^2}{\partial\theta} = \frac{1}{N}|R|^{\frac{1}{N}}(\mathrm{Y}-\mathrm{M})^t\frac{\partial R^{-1}}{\partial\theta}(\mathrm{Y}-\mathrm{M}) = -\frac{1}{N}|R|^{\frac{1}{N}}(\mathrm{Y}-\mathrm{M})^t R^{-1}\frac{\partial R}{\partial\theta}R^{-1}(\mathrm{Y}-\mathrm{M}) \quad \textbf{(47)}$$

Finally, by adding the two summands:

$$\frac{\partial\Psi}{\partial\theta} = \frac{1}{N}|R|^{\frac{1}{N}}\left(tr\left(R^{-1}.\frac{\partial R}{\partial\theta}\right).\hat{\sigma}^2 - (\mathrm{Y}-M)^t R^{-1}\frac{\partial R}{\partial\theta}R^{-1}(\mathrm{Y}-M)\right) \quad \textbf{(48)}$$

$$\frac{\partial \Psi}{\partial \theta} = \frac{1}{N}|R|^{\frac{1}{N}}\left(tr\left(\left(\frac{\hat{\sigma}^2}{N}.R^{-1} - \alpha\alpha^t\right).\frac{\partial R}{\partial \theta}\right)\right) \quad \textbf{(49)}$$

Where $\alpha = R^{-1}(\mathrm{Y} - M)$.

One can observe that most of the terms of this equation ($|R|$, $\hat{\sigma}^2$, the Choleski decomposition of R) have already been computed to obtain the log-likelihood. In fact, one can obtain the derivatives of the likelihood by computing $\frac{\partial R}{\partial \theta}$, which has a complexity of $O(N^2)$, and by applying this last equation, which also has a complexity of $O(N^2)$ providing that only the diagonal elements of the matrix product $\left(\frac{\hat{\sigma}^2}{N}.R^{-1} - \alpha\alpha^t\right).\frac{\partial R}{\partial \theta}$ are computed.

V.3.2 Choice of a gradient-based algorithm

Finding the most adapted gradient-based optimization algorithm to maximize the log-likelihood would require an in-depth study of the literature and many tests. Since this was not the scope of this thesis, it was decided to simply use one of the available algorithms. A first idea was to use the same algorithm that was used in PAL. This algorithm was created by Rasmussen [70] for unconstrained, gradient-based optimization. However, when it was coupled with SMS-EGO, errors were generated. The analysis of these errors led us to the conclusion that the form of correlation function used in SMS-EGO could not be used with unconstrained optimization algorithms. Indeed, it was observed that SMS-EGO generated errors when the hyper-parameters varied on a too big scale because this led to ill-conditioned correlation matrices that were then very difficult to inverse. Although PAL was based on very similar equations, we believe that it was not suffering from the same issue because the type of hyper-parameters chosen had a smaller impact on the correlation coefficients. Thus, it was finally decided to use an algorithm from MATLAB toolboxes.

Based on MATLAB documentation, the *GlobalSearch* algorithm was selected, together with the *fmincon* solver and the *trust-region-reflective* algorithm. In short, the *trust-region-reflective* algorithm is recommended when gradient information is available and when the

constraints are only in the form of bounds [79], which is the case in this thesis. Concerning the choice of the solver, *fmincon* is the solution recommended for multi-variable objective functions that are smooth but non-linear with constraints of type "finite bounds" [79]. Finally, while the *fmincon* solver aims at finding a local minimum, the *GlobalSearch* algorithm ensures that multiple start points are used so that a global optimization is carried out.

Concerning the options of the algorithm, most of the default values were kept but it was decided to decrease the number of start points and trial points in *GlobalSearch*. Indeed, with the defaults values (respectively 200 and 1000), the total computation time required for the *GlobalSearch* algorithm was bigger than the time required by CMA-ES. Since the aim in this part was to reduce this time, it was chosen instead to set the number of start points in stage 1 to 4 and the number of trial points to 10. This was also in agreement with plots of the likelihood that were realized for a smaller case-study with only two decision variables and that showed a likelihood function with only a few local minima. The termination tolerance on the function value and on the hyper-parameters were set to the same values as in the original version of SMS-EGO for CMA-ES (respectively 10^{-6} and 10^{-12}). Finally, the constraints that were defined on the hyper-parameters were the same as in the original version of SMS-EGO (with the same notations as in Table 1, $10^{-3} \leq \theta_j \leq 10$ and $0.1 \leq \gamma_j \leq 2$).

To conclude, the algorithm selected was certainly not the most well suited for our case-study. Nevertheless, it was expected to give us an insight about the potential benefits of gradient-based optimization.

V.4 Reducing the update frequency

The second alternative that was proposed to reduce the computation time was to compute the hyper-parameters less often. Although relatively obvious, this alternative may be more detrimental than anything if it is not implemented cautiously. Indeed, even though reducing the modeling time would leave more time for evaluations, these evaluations might not be well

chosen. More evaluations might represent a waste of time and it might also slow down the next meta-modeling step, since there would be more data to take into account.

Based on these observations, it was proposed to update the meta-model at least at pre-determined iterations and at other iterations only when there are reasons to believe that the meta-model is ill-adapted. A very simple way to decide when to update the meta-model is to compare the predicted value with the newly observed value. If the current meta-model is not able to predict the behavior of the underlying process accurately, it is ill-adapted and should be updated. In this work, the following update condition was adopted:

$$update\ condition: \exists i \in \{1, \dots n_{obj}\}, \left| {y_{predicted}}^{i} - {y_{measured}}^{i} \right| > {\sigma_{predicted}}^{i}, \quad \textbf{(50)}$$

Where ${y_{predicted}}^{i}$ and ${y_{measured}}^{i}$ represent the predicted and measured values of the objective *i* and ${\sigma_{predicted}}^{i}$ the standard deviation for the prediction. The main drawback of such a strategy is that meta-models that have unnecessary big standard deviations are kept for the next iterations. Thus, as mentioned earlier, conditional updates were associated with a pre-determined automatic update process. In this work, the following automatic update strategy was chosen:

$$automatic\ update\ condition$$

$$iteration \equiv 0 \left(mod \left(\left\lfloor \frac{N}{5} \right\rfloor + 1 \right) \right) \quad \textbf{(51)}$$

Where $\lfloor \dots \rfloor$ is the floor function, N is the number of points already evaluated, and mod represents the congruence relation. Thus, the meta-model is updated for the first five iterations at every iteration, then every two iterations for the next five, then every three iterations, etc. This framework was design to account for the fact that the meta-model is rapidly evolving at the beginning and then more slowly.

To conclude, many different strategies could be implemented to decide when the hyper-parameters should be updated. However, the objective of this thesis was only to explore some potential avenues to improve SMS-EGO.

V.5 Case-study

In this part, the original SMS-EGO algorithm was compared with the two alternatives presented in this chapter. To refer to these two variants, we use the following notations:

- "Variant 2" for the version of SMS-EGO described in section V.3, which relied on a gradient-based optimization algorithm to determine the values of the hyper-parameters.
- "Variant 3" for the version of SMS-EGO described in section V.4, in which the hyper-parameters are only updated when at least one of the update conditions is verified.

As in Chapter IV, the new variants were run 20 times each on 150 iterations of the "bus-lane" configuration of the *Côte des neiges* case-study.

In this chapter, the objective was to accelerate the optimization process by reducing the time required for each iteration without significantly degrading their quality. In this case-study, it was first verified that iterations had indeed been accelerated. For this purpose, the average total computation time over all the runs was plotted in Figure 24 as a function of the number of iterations for SMS-EGO and for the two new variants. As expected, the two new variants went in average noticeably faster than the original version.

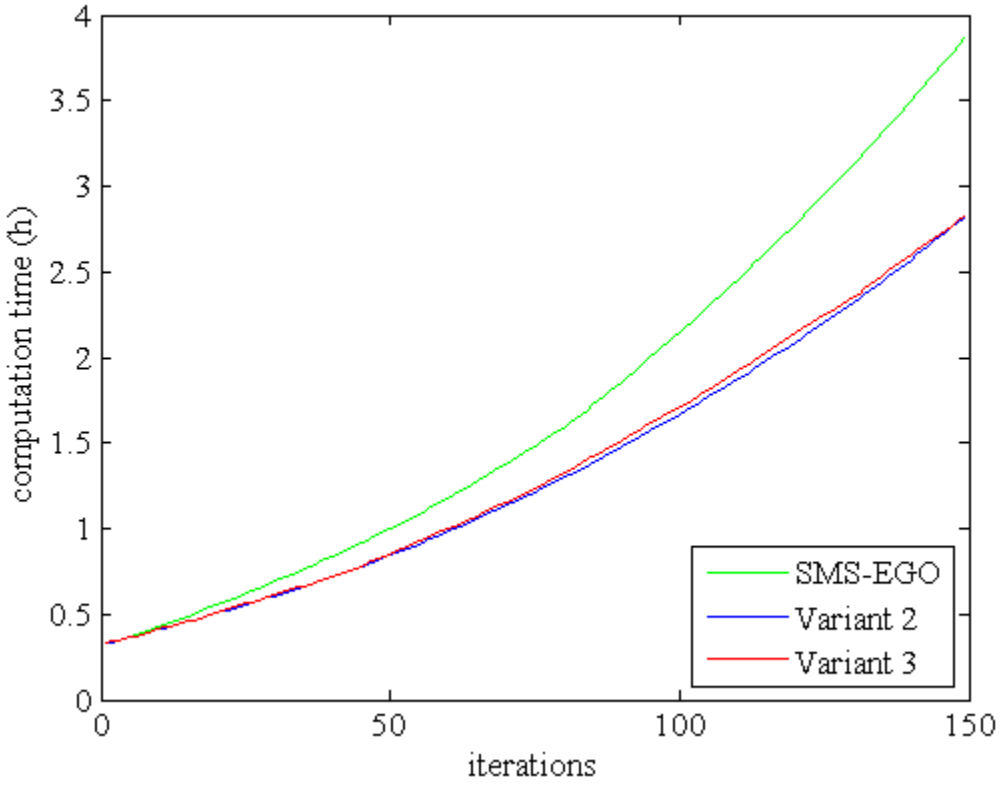

Figure 24: Average total computation time as a function of the number of iterations for SMS-EGO and the variants 2 and 3

However, the final objective was to accelerate the approximation of the Pareto front. Thus, to analyze the impact of the modifications made, the evolution of the hypervolume was plotted as a function of the number of iterations in Figure 25 and as a function of the computation time in Figure 26. In addition, as in Chapter IV, a series of two-tailed Student's *t* tests were run to determine the significance level of the results. The results of these tests are displayed in Figure 27 and Figure 28.

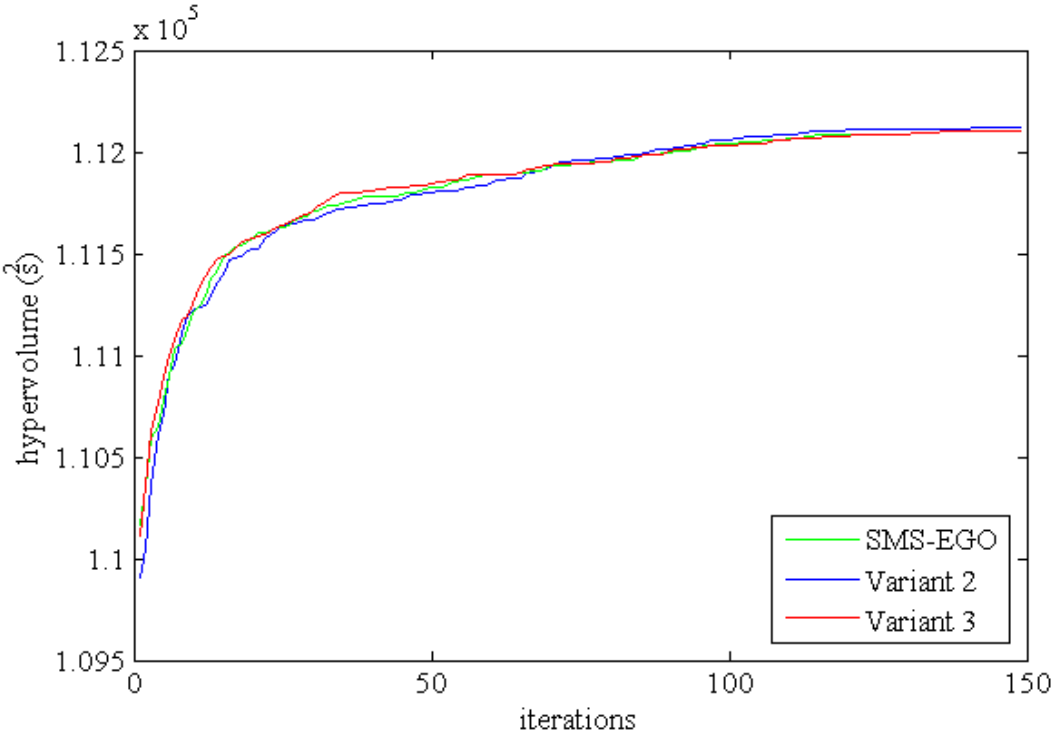

Figure 25: Evolution of the average hypervolume obtained over ten runs of SMS-EGO, variant 2 and variant 3 as functions of the number of iterations.

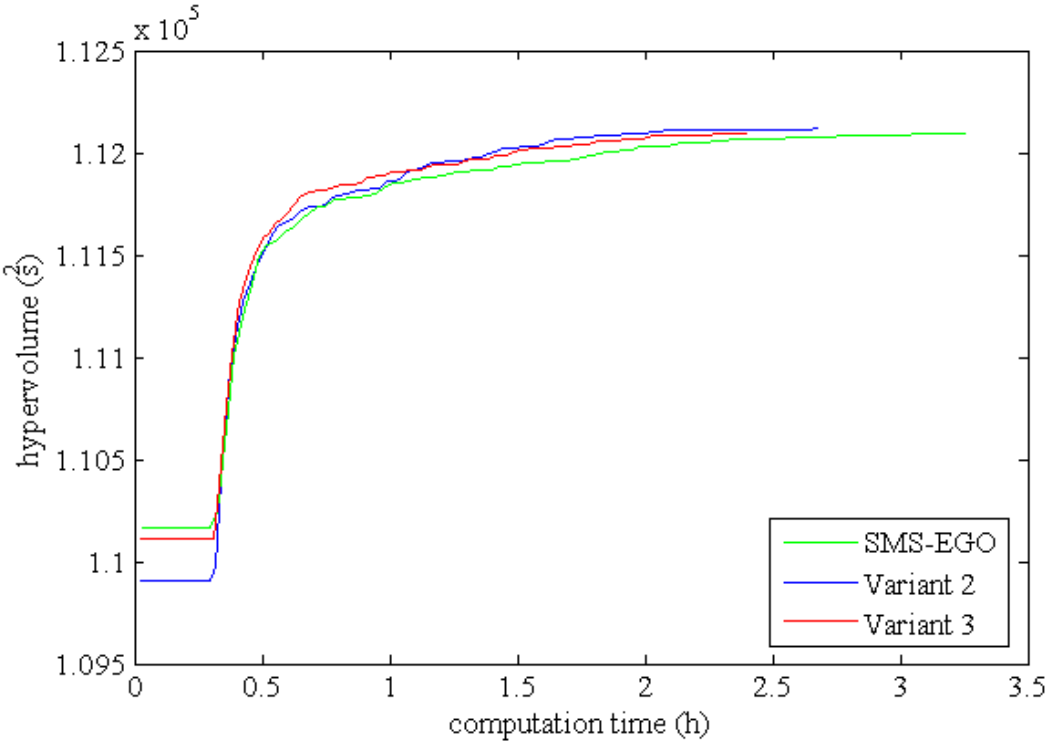

Figure 26: Evolution of the average hypervolume obtained over ten runs of SMS-EGO, variant 2 and variant 3 as functions of the computation time

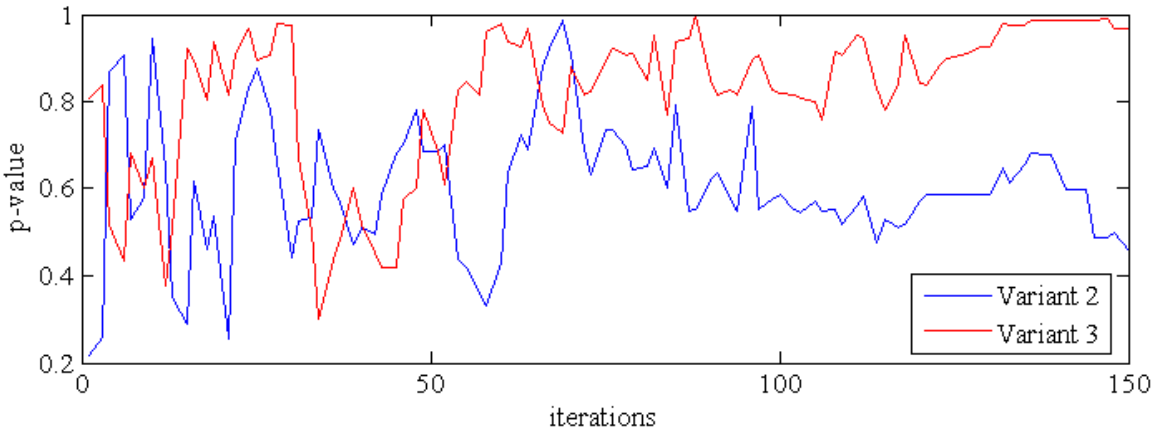

Figure 27: Evolution over the iterations of the p-values from two-tailed Student's *t* tests comparing on the one hand the hypervolumes obtained with SMS-EGO and variant 2 and on the other hand the hypervolumes obtained with SMS-EGO and variant 3

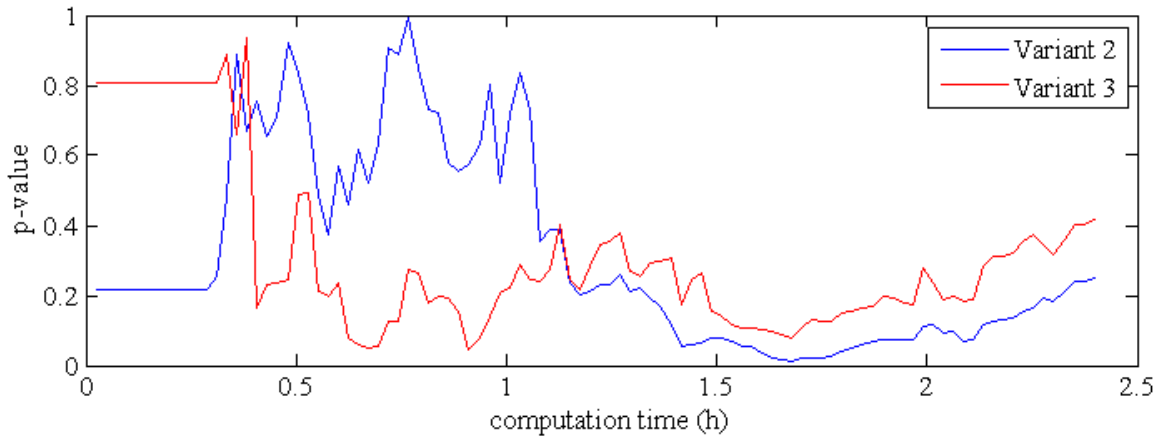

Figure 28: Evolution over the computation time of the p-values from two-tailed Student's *t* tests comparing on the one hand the hypervolumes obtained with SMS-EGO and variant 2 and on the other hand the hypervolumes obtained with SMS-EGO and variant 3

Several trends can be observed by studying together the plots of the average hypervolume and the p-values from the two-tailed Student's *t* tests. First, based on both Figure 25 and Figure 27, the modifications made did not significantly degrade the quality of each iteration. Indeed, the three curves plotted in Figure 25 were extremely close to each other and the p-values never stayed consistently under 0.4 for any of the two variants.

Since the two modifications reduced the computation time necessary for each iteration without degrading their quality, one could deduce that they also accelerated the entire

optimization process. In fact, in Figure 26, one can see that the curves associated to the variants 2 and 3 were consistently higher than the curve associated to SMS-EGO after about 30 min for variant 3 and 1 h for variant 2. In terms of p-values, the difference with SMS-EGO seemed to reach a maximum for the two variants after approximately 1h40min, before decreasing as the optimization process started stagnating. Around this time, the p-value associated to the variant 2 remained inferior to 0.20 during 25 min and the p-value associated to the variant 3 remained inferior to 0.10 during 35 min and inferior to 0.05 during 13 min. Thus, although it was not as clear as in Figure 24, these results tend to confirm that the two variants did accelerate the optimization process. The high variability that was observed from one optimization process to another might explain why the p-values remained relatively large.

V.6 Conclusion of the chapter

In this chapter, the repartition of computation time among the three elementary tasks of SMS-EGO was analyzed and the determination of the hyper-parameters was found to be the most time-consuming. Following this analysis, two different avenues to accelerate the determination of the hyper-parameters were proposed.

The first of these avenues consisted in accelerating this optimization process by relying on gradient information. This modification was implemented by replacing the algorithm used in SMS-EGO by an algorithm from the MATLAB library. In the case-study, the resulting modified version of SMS-EGO was found to reduce the computation time of each iteration without significantly reducing the quality. When analyzing the hypervolume as a function of computation time, the modified version was found to outperform the original version of SMS-EGO, even though the p-value only reached the significance level of 0.05 during 13 min. These results are highly encouraging since they were obtained without any in-depth comparison of gradient-based optimization processes and with only 20 runs for each optimization process. Thus, gradient-based optimization is an avenue that should be more thoroughly explored in the future.

The second avenue explored in this chapter was to implement conditional updates of the hyper-parameters. In the original version of SMS-EGO, the hyper-parameters are systematically updated at every iteration. In this second part, SMS-EGO was accelerated by updating the hyper-parameters only at some pre-determined iterations or when the newly evaluated values were too different from the predicted ones. As expected, this modification was found to accelerate each iteration. In addition, the case-study showed that conditional updates did not noticeably decrease the quality of each iteration. Finally, based on these observations and on the evolution of the hypervolume over time, it was concluded that this modification most likely did accelerate the optimization process, although the p-value resulting from the two-tailed Student's *t* test remained relatively large, around 0.20. For future work, it is recommended to analyze different update conditions and carrying out more optimization runs in diverse situations.

Chapter VI Complexity analysis and future work

In this last chapter, the computation time is addressed from the angle of the computational complexity. Even though computational complexity alone does not allow us to predict which algorithm would be the most efficient, it does allow us to understand why iterations take more and more time and how it is likely to evolve if even more iterations were carried out. In the second part of this chapter, different approaches explored in the literature to reduce this complexity are rapidly presented.

VI.1 Complexity analysis

VI.1.1 For a given network

As shown by the repartition of computation time in Figure 22, the most time-consuming task is to determine the values of the hyper-parameters, which is done by maximizing the likelihood of the previously observed data. Its duration, written $T(HP\ determination)$, can be roughly decomposed as the product between the number of evaluations of the log-likelihood N_{eval} and the time required for one evaluation, $T(LL\ evaluation)$:

$$T(HP\ determination) \cong N_{eval} \times T(LL\ evaluation) \quad \textbf{(52)}$$

In order to evaluate the complexity of one evaluation, we recall here the expression of the concentrated log-likelihood in the DACE/EGO framework:

$$\log_{\text{EGO}}\left(p(\overrightarrow{y_{1\dots N}}|X_1,\dots X_N)\right) = -\frac{1}{2}\left(Nlog(\sigma^2) + \log(|corr(X_{1\dots N}, X_{1\dots N})|)\right) + constant\ . \quad \textbf{(53)}$$

As explained by Sacks et al. [42], maximizing this expression of the log-likelihood is equivalent to minimizing

$$\Psi(\theta) = |corr(X_{1\dots N}, X_{1\dots N})|^{1\backslash N}\hat{\sigma}^2 \quad \textbf{(54)}$$

Where $\hat{\sigma}^2$ is the MLE of the variance obtained in equation 34:

$$\widehat{\sigma^2} = \frac{(\overrightarrow{y_{1\dots N}} - \overrightarrow{m_{1\dots N}})^t Corr(X_{1\dots N}, X_{1\dots N})^{-1}(\overrightarrow{y_{1\dots N}} - \overrightarrow{m_{1\dots N}})}{N}. \quad \textbf{(34)}$$

The complexity of each elementary operation is given in Table 5.

Operation	Complexity
Choleski decomposition	$O(N^3)$
$corr(X_{1\dots N}, X_{1\dots N})$	$O(N^2)$
$\lvert corr(X_{1\dots N}, X_{1\dots N})\rvert^{1\backslash N}$ (using the Choleski decomposition)	$O(N)$
$Corr(X_{1\dots N}, X_{1\dots N})^{-1}(\overrightarrow{y_{1\dots N}} - \overrightarrow{m_{1\dots N}})$ (using the Choleski decomposition)	$O(N^2)$
$\frac{(\overrightarrow{y_{1\dots N}} - \overrightarrow{m_{1\dots N}})^t Corr(X_{1\dots N}, X_{1\dots N})^{-1}(\overrightarrow{y_{1\dots N}} - \overrightarrow{m_{1\dots N}})}{N}$	$O(N)$

Table 5: Computational complexities of the elementary operations used in the likelihood estimation.

Thus, the overall complexity of one evaluation of the likelihood is the complexity of the Choleski decomposition:

$$T(LL\ evaluation) = O(N^3)\,. \quad \textbf{(55)}$$

Assuming that the number of evaluations N_{eval} is bounded, the overall complexity of one entire optimization process with a budget of N evaluations is:

$$T(optimization) = \sum_{i=1}^{N} N_{eval} \times O(i^3) = O\left(\sum_{i=1}^{N} i^3\right) = O\left(\frac{1}{4}(N^4 + 2N^3 + N^2)\right) = O(N^4) \quad \textbf{(56)}$$

Finally, in case the derivatives of the log-likelihood are also computed, it was shown in section V.3.1 that the computational overhead of estimating the derivatives of the likelihood is a $O(N^2)$. Thus, computing the derivatives does not change the overall complexity.

VI.1.2 Network size

For a given network, it was shown that the complexity of the overall optimization process is $O(N^4)$. However, the value of N that can be expected to provide acceptable results depends on the size of the network, and more precisely on the dimension of the problem. Although it is difficult to quantify it, the dimension of the problem impacts the computation time in two ways. First, if the design space is bigger, more evaluations are required to create a good approximation of the whole design space. Second, because of the form of correlation matrix used in SMS-EGO, the number of associated hyper-parameters is twice the dimension of the problem. Thus, the constant N_{eval} that determines the number of evaluations required to maximize the log-likelihood of the previous data is also very likely to increase with the dimension of the problem.

VI.2 Future work

The wide-scale adoption of simulation-based multi-objective optimization is conditional upon the design of a simple procedure that could be implemented without an in-depth knowledge of the optimization field. Thus, future research should focus on general solutions that can handle mixed-integer objective functions and large decision spaces. However, before addressing mixed-integer functions, we believe that the main obstacle remains the computation time. The different approaches proposed in this thesis do have a potential to reduce it, but they would be even more useful if associated with progress on other fronts. For most obstacles, technical solutions already exist and the only difficulty is to include them in a multi-objective optimization framework. Here, a few strategies that could be integrated within an algorithm based on Gaussian process meta-models are listed.

First, a simple yet very effective idea to reduce the cost associated with the Gaussian processes is to use them only to model phenomena that are difficult to predict otherwise. For instance, when minimizing delays, Gaussian processes could be paired with analytical predictions based on queuing models. In fact, Osorio [11] already showed that the association of an analytical

model with a quadratic polynomial accelerated the optimization process. However, such an approach demands more preliminary work since it requires designing and calibrating both a microscopic simulation model and a macroscopic model.

Second, many different ideas have been proposed to reduce the computational complexity of Gaussian process regression. As explained earlier, the most demanding task is to compute the Cholesky decomposition of the correlation matrix, which takes time $O(N^3)$.

A first idea to reduce this complexity is to choose covariance functions with compact support that still guarantee positive definiteness. Indeed, this would lead to a sparse correlation matrix, which would be easier to decompose. For instance, Vanhatalo and Vehtari [80] proposed a Gaussian process model consisting of both a global sparse approximation for long length-scale variations and of a covariance model with compact support for short-length scale variations. With these two additive models, the authors were able to solve the computational issues linked to the traditional full Gaussian process regression and to avoid discontinuities often caused by sparse correlation matrixes. However, according to the authors, the computational complexity depends on how sparse the correlation matrix is.

Another idea studied by many authors is to use reduced-rank approximations of the covariance matrix. Indeed, even in cases where the covariance matrix is not sparse, its Cholesky decomposition can be accelerated by reducing it to its "main" components, which is done by approximating its eigenvalues/vectors. Several research groups proposed different methods based on these concepts and they usually obtained an overall complexity of $O(m^2N)$, where m is the size of the subset used to compute the reduced-rank approximation matrix. For a review and comparison of such methods, the reader is referred to the chapter 8 of Rasmussen and Williams [65].

Finally, a last idea to reduce the complexity is to divide the decision space in different areas with their own meta-models. If each area includes a maximum of m points, the complexity

for each of the $\left(\left[\frac{N}{m}\right]+1\right)$ meta-models would be $O(m^3)$ and the overall complexity would a priori become $O(m^3 N)$. However, continuity issues might arise at the boundaries and the criteria used to group the points and to select the adequate meta-model may also be challenging to define and computationally intensive. Rasmussen and Ghahramani [81] proposed a solution based on the Mixture of Experts model that includes a gating network for an infinite number of models. If m meta-models are used, the resulting complexity was found to be $O(N^3/m)$ for one meta-modeling step. As another benefit, this framework would allow for different length-scales in different parts of the decision space, which could improve the goodness of fit of the meta-model.

VI.3 Conclusion of the chapter

In this chapter, a better understanding of the computation time required by the optimization process was gained by analyzing its computational complexity. The task responsible for the overall complexity was found to be the inversion of the correlation matrix necessary to create a Gaussian process meta-model. This operation is based on a Cholesky decomposition and requires $O(N^3)$. Thus the complexity of each iteration is $O(N^3)$ and the overall complexity of the optimization process is $O(N^4)$. Nevertheless, the number of iterations N which is necessary depends on the size of the problem.

The next challenge is to reduce this complexity to allow for large-scale optimization. As shown in the last part of this chapter, diverse methods have already been proposed to reduce the complexity of Gaussian-process meta-modeling, although they were not specifically created for large-scale multi-objective optimization. To conclude, more research is necessary to determine which approach would be the most adapted to the NDP and to include it within a multi-optimization framework such as SMS-EGO.

Conclusion

In summary, the ultimate objective of this thesis was to find algorithms that would enable transportation engineers to use microscopic simulation to solve the multi-objective CNDP in a reasonable time and with a better accuracy than with macroscopic models.

To answer this issue, the literature on expensive multi-objective optimization was reviewed in Chapter II and two algorithms relying on Gaussian process meta-models were selected: PAL and SMS-EGO. In Chapter III, the algorithms were compared on a small case-study and on a very small computational budget (a few hundred evaluations) to the genetic algorithm identified as the state-of-the-art, NSGA-II. SMS-EGO emerged from this case-study as the most efficient algorithm, while PAL was found to perform at least as well as NSGA-II. However, besides these encouraging results, the case-study also revealed some potential limitations for problems with more decision variables, especially in terms of computation time required by the optimization process itself.

In an attempt to overcome these limitations, three main modifications were proposed. In Chapter IV, SMS-EGO was adapted to take into account a variable noise in evaluations. A case-study revealed that by improving the quality of the meta-model, the approximation of the Pareto front was actually accelerated. Then, two additional avenues were explored to reduce the computation time associated with the meta-modeling process. First, the evolutionary algorithm used in SMS-EGO was replaced by a gradient-based algorithm. Second, tests were implemented to determine whether it was necessary to update the hyper-parameters of the meta-model. While both these adjustments seemed to accelerate the optimization process on average, their impact was not found significant over the 20 optimization processes run. With p-values around 0.15, these results were nonetheless encouraging and other gradient-based algorithms or different update criteria should be explored.

Eventually, a complexity analysis was carried out and several approaches already investigated by other authors were listed. Indeed, the use of Gaussian process meta-models is not restricted to expensive multi-objective optimization and many authors already endeavored to reduce its computational complexity. Although these techniques may be difficult to implement, their integration into a multi-objective algorithm such as SMS-EGO could lead to great computational benefits.

To conclude, the technical limitations that slow down the adoption of simulation-based multi-objective optimization are rapidly fading away. In fact, most of the mathematical and programming tools necessary to reduce its computational requirements to acceptable levels have probably already been unveiled. With this thesis, we hope that we have contributed in enriching the knowledge of tools that could benefit the NDP.

Bibliography

1. Magnanti, T. L., and R. T. Wong. Network Design and Transportation Planning : Models and Algorithms. *Transportation Science*, Vol. 18, No. 1, 1984, pp. 1–55.
2. Yang, H., and M. G. H. Bell. Models and algorithms for road network design: a review and some new developments. *Transport Reviews: A Transnational Transdisciplinary Journal*, Vol. 18, No. 3, 1998, pp. 257–278.
3. Guihaire, V., and J.-K. Hao. Transit network design and scheduling: A global review. *Transportation Research Part A: Policy and Practice*, Vol. 42, No. 10, 2008, pp. 1251–1273.
4. Kepaptsoglou, K., and M. Karlaftis. Transit Route Network Design Problem : Review. *Journal of Transportation Engineering*, Vol. 135, No. 8, 2009, pp. 491–505.
5. Chow, J. Y. J. *Flexible Management of Transportation Networks under Uncertainty*. Ph. D. diss., University of California - Irvine, 2010.
6. Cascetta, E., M. Gallo, and B. Montella. Models and algorithms for the optimization of signal settings on urban networks with stochastic assignment models. *Annals of Operations Research*, Vol. 144, No. 1, 2006, pp. 301–328.
7. Farahani, R. Z., E. Miandoabchi, W. Y. Szeto, and H. Rashidi. A review of urban transportation network design problems. *European Journal of Operational Research*, Vol. 229, No. 2, 2013, pp. 281–302.
8. Correa, J. R., and N. E. Stier-Moses. Wardrop Equilibria. In *Wiley Encyclopedia of Operations Research and Management Science*, ed J. J. Cochran, John Wiley & Sons, Inc., 2010.
9. PTV Planung Transport Verkehr AG. *VISSIM 5.40 - User Manual*. Karlsruhe, Germany, 2012.

10. Fellendorf, M., and P. Vortisch. Microscopic Traffic Flow Simulator VISSIM. In *Fundamentals of Traffic Simulation*, ed J. Barceló, Springer New York, New York 2010.

11. Osorio Pizano, C. *Mitigating Network Congestion : Analytical Models, Optimization Methods and their Applications*. Ph. D. diss., Ecole Polytechnique Fédérale de Lausanne, 2010.

12. Stevanovic, A., J. Stevanovic, and C. Kergaye. Optimization of traffic signal timings based on surrogate measures of safety. *Transportation Research Part C: Emerging Technologies*, Vol. 32, 2013, pp. 159–178.

13. Robles, D. *Optimal signal control with multiple objectives in traffic mobility and environmental impacts*. M. Sc. Diss., Royal Institute of Technology (KTH) Stockholm, 2012.

14. Fikse, K. *Accelerating the Search for Optimal Dynamic Traffic Management*. M. Sc. Diss., University of Twente, 2011.

15. Guldmann, J.-M., and W. Kim. Urban transportation network design, traffic allocation, and air quality control: an integrated optimization approach. *European Regional Science Association 36th European Congress, ETH Zurich, Switzerland*, No. August, 1996, pp. 1–42.

16. Sharma, S., and T. V Mathew. Multiobjective network design for emission and travel-time trade-off for a sustainable large urban transportation network. *Environment and Planning B: Planning and Design*, Vol. 38, No. 3, 2011, pp. 520–538.

17. Stevanovic, A., J. Stevanovic, K. Zhang, and S. Batterman. Optimizing Traffic Control to Reduce Fuel Consumption and Vehicular Emissions. *Transportation Research Record: Journal of the Transportation Research Board*, Vol. 2128, No. -1, 2009, pp. 105–113.

18. Wismans, L., E. van Berkum, and M. Bliemer. Accelerating solving the dynamic Multi-Objective Network Design Problem using Response Surface Methods. *Journal of*

Intelligent Transportation Systems: Technology, Planning, and Operations, Vol. 18, No. 1, 2013, pp. 17–29.

19. Perkins, S. R., and J. I. Harris. *Criteria for traffic conflict characteristics*. Research Laboratories, General Motors Corporation, 1967.

20. Tarko, A., G. Davis, N. Saunier, T. Sayed, and S. Washington. *Surrogate Measures of Safety*. 2009.

21. Gettman, D., and L. Head. *Surrogate Safety Measures From Traffic Simulation Models*. Federal Highway Administration, 2003.

22. Gettman, D., L. Pu, T. Sayed, and S. Shelby. *Surrogate Safety Assessment Model and Validation*. Federal Highway Administration, 2008.

23. Chen, A., J. Kim, S. Lee, and Y. Kim. Stochastic multi-objective models for network design problem. *Expert Systems with Applications*, Vol. 37, No. 2, 2010, pp. 1608–1619.

24. Bureau of Public Roads. *Traffic Assignment Manual*. U.S. Department of Commerce, Urban Planning Division, Washington, D.C., 1964.

25. Shimamoto, H., J.-D. Schmöcker, and F. Kurauchi. Optimisation of a Bus Network Configuration and Frequency Considering the Common Lines Problem. *Journal of Transportation Technologies*, Vol. 02, No. 03, 2012, pp. 220–229.

26. Cantarella, G. E., and A. Vitetta. The multi-criteria road network design problem in an urban area. *Transportation*, Vol. 33, No. 6, 2006, pp. 567–588.

27. Hu, H., Y. Gao, and X. Yang. Multi-objective Optimization Method of Fixed-Time Signal Control of Isolated Intersections. *2010 International Conference on Computational and Information Sciences*, 2010, pp. 1281–1284.

28. Sun, D., F. Benekohal, and S. T. Waller. Multi-objective Traffic Signal Timing Optimization Using Non-dominated Sorting Genetic Algorithm. *2003 IEEE Intelligent Vehicles Symposium, Columbus, Oh.*, 2003, pp. 198–203.

29. MathWorks. What is Multiobjective Optimization? Available at: http://www.mathworks.com/help/gads/what-is-multiobjective-optimization.html [Accessed November 20, 2013].

30. Deb, K., A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, Vol. 6, No. 2, 2002, pp. 182–197.

31. Zuluaga, M., A. Krause, E. T. H. Zurich, G. Sergent, and P. Markus. Active Learning for Multi-Objective Optimization. *JMLR: W&CP*, Vol. 28, 2013.

32. Ponweiser, W., T. Wagner, D. Biermann, and M. Vincze. Multiobjective Optimization on a Limited Budget of Evaluations Using Model-Assisted S-Metric Selection. *PPSN X, LCNS 5199, Dortmund, Germany*, Vol. LCNS 5199, 2008, pp. 784–794.

33. Knowles, J., and E. J. Hughes. Multiobjective Optimization on a Budget of 250 Evaluations. In *Evolutionary Multi-Criterion Optimization, Third International Conference*, eds Coello CA, Aguirre AH, Zitzler E, 2005.

34. Box, G. E. P., and K. B. Wilson. On the Experimental Attainment of Optimum Conditions. *Journal of the Royal Statistical Society, Series B (Methodological)*, Vol. 13, No. 1, 1951, pp. 1–45.

35. Clarke, S. M., J. H. Griebsch, and T. W. Simpson. Analysis of Support Vector Regression for Approximation of Complex Engineering Analyses. *Journal of Mechanical Design*, Vol. 127, No. 6, 2005, pp. 1077.

36. Jin, R., W. Chen, and T. W. Simpson. Comparative Studies of Metamodeling Techniques Under Multiple Modeling Criteria. *Structural and Multidisciplinary Optimization*, Vol. 23, No. 1, 2001, pp. 1–13.

37. Friedman, J. H. Multivariate Adaptive Regression Splines. *The Annals of Statistics*, Vol. 19, No. 1, 1991, pp. 1–67.

38. Gutmann, H.-M. A Radial Basis Function Method for Global Optimization. *Journal of Global Optimization*, Vol. 19, 2001, pp. 201–227.

39. Regis, R. G., and C. a. Shoemaker. Constrained Global Optimization of Expensive Black Box Functions Using Radial Basis Functions. *Journal of Global Optimization*, Vol. 31, 2005, pp. 153–171.

40. Mullur, A. A., and A. Messac. Extended Radial Basis Functions : More Flexible and Effective Metamodeling. *AIAA Journal*, Vol. 43, No. 6, 2005, pp. 1306–1315.

41. Matheron, G. Principles of geostatistics. *Economic Geology*, Vol. 58, 1963, pp. 1246–1266.

42. Sacks, J., W. J. Welch, T. J. Mitchell, and H. P. Wynn. Design and Analysis of Computer Experiments. *Statistical Science*, Vol. 4, No. 4, 1989, pp. 409–423.

43. Jones, D. R., M. Schonlau, and J. William. Efficient Global Optimization of Expensive Black-Box Functions. *Journal*, Vol. 13, 1998, pp. 455–492.

44. Cortes, C., and V. Vapnik. Support-Vector Networks. *Machine Learning*, Vol. 20, No. 3, 1995, pp. 273–297.

45. Vapnik, V., S. E. Golowich, and A. Smola. Support Vector Method for Function Approximation, Regression Estimation, and Signal Processing. *Proceedings of the 1996 Neural Information Processing Systems Conference*, 1996, pp. 281–287.

46. Kim, B.-S., Y.-B. Lee, and D.-H. Choi. Comparison study on the accuracy of metamodeling technique for non-convex functions. *Journal of Mechanical Science and Technology*, Vol. 23, No. 4, 2009, pp. 1175–1181.

47. Yuan, R., and B. Guangchen. *2009 Eighth IEEE International Conference on Dependable, Autonomic and Secure Computing*, 2009.

48. Chowdhury, M., A. Alouani, and F. Hossain. Comparison of ordinary kriging and artificial neural network for spatial mapping of arsenic contamination of groundwater. *Stochastic Environmental Research and Risk Assessment*, Vol. 24, No. 1, 2008, pp. 1–7.

49. Paiva, R. M., A. R. D. Carvalho, C. Crawford, and A. Suleman. Comparison of Surrogate Models in a Multidisciplinary Optimization Framework for Wing Design. *AIAA Journal*, Vol. 48, No. 5, 2010, pp. 995–1006.

50. Matias, J. M., A. Vaamonde, J. Taboada, and W. Gonzales-Manteiga. Comparison of Kriging and Neural Networks With Application to the Exploitation of a Slate Mine 1. *Mathematical Geology*, Vol. 36, No. 4, 2004, pp. 463–486.

51. Müller, J., and R. Piché. Mixture surrogate models based on Dempster-Shafer theory for global optimization problems. *Journal of Global Optimization*, Vol. 51, No. 1, 2010, pp. 79–104.

52. Viana, F. a. C., R. T. Haftka, and L. T. Watson. Efficient global optimization algorithm assisted by multiple surrogate techniques. *Journal of Global Optimization*, Vol. 56, No. 2, 2013, pp. 669–689.

53. Chen, A., K. Subprasom, and Z. Ji. A simulation-based multi-objective genetic algorithm (SMOGA) procedure for BOT network design problem. *Optimization and Engineering*, Vol. 7, No. 3, 2006, pp. 225–247.

54. Stevanovic, A., J. Stevanovic, and C. Kergaye. Optimization of traffic signal timings based on surrogate measures of safety. *Transportation Research Part C: Emerging Technologies*, Vol. 32, 2013, pp. 159–178.

55. Xiong, Y., and J. B. Schneider. Transportation Network Design Using a Cumulative Genetic Algorithm and Neural Network. *Transportation Research Record: Journal of the Transportation Research Board*, No. 1364, 1992, .

56. Bielli, M., P. Carotenuto, and G. Confessore. A New Approach for Transport Network Design and Optimization. *38th Congress of the European Regional Science Association*, 1998.

57. Regis, R. G., and C. a. Shoemaker. A Stochastic Radial Basis Function Method for the Global Optimization of Expensive Functions. *INFORMS Journal on Computing*, Vol. 19, No. 4, 2007, pp. 497–509.

58. Osorio Pizano, C., and M. Bierlaire. *A simulation-based optimization framework for urban traffic control*. Transport and Mobility Laboratory (Ecole Polytechnique Fédérale de Lausanne), 2010.

59. Osorio, C., and L. Chong. An efficient simulation-based optimization algorithm for large-scale transportation problems. *Proceedings of the 2012 Winter Simulation Conference (WSC)*, 2012, pp. 1 – 11.

60. Mockus, J. On Bayesian methods for seeking the extremum. *Optimization techniques IFIP International Conference Novosibirsk*, 1974, pp. 400–404.

61. Knowles, J. ParEGO : A Hybrid Algorithm With On-Line Landscape Approximation for Expensive Multiobjective Optimization Problems. *IEEE Transactions on Evolutionary Computation*, Vol. 10, No. 1, 2005, pp. 50–66.

62. Jeong, S., and S. Obayashi. Efficient Global Optimization (EGO) for Multi-Objective Problem and Data Mining. *2005 IEEE Congress on Evolutionary Computation*, Vol. 3, 2005, pp. 2138–2145.

63. Zitzler, E., and L. Thiele. Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach. *IEEE Transactions on Evolutionary Computation*, Vol. 3, No. 4, 1999, pp. 257–271.

64. Zitzler, E., L. Thiele, M. Laumanns, C. M. Fonseca, and V. Grunert. Performance Assessment of Multiobjective Optimizers : An Analysis and Review. *IEEE Transactions on Evolutionary Computation*, Vol. 7, No. 2, 2003, pp. 117–132.

65. Rasmussen, C. E., and C. K. I. Williams. *Gaussian processes in machine learning*. The MIT Press, 2006.

66. Rasmussen, C. E., and H. Nickisch. Gaussian Process Regression and Classification Toolbox Version 3.1 for Matlab 7.x., 2010.

67. Knowles, J., D. Corne, and A. Reynolds. Noisy Multiobjective Optimization on a Budget of 250 Evaluations. *Evolutionary Multi-Criterion Optimization, 5th International Conference, Nantes, France*, 2009, pp. 36–50.

68. Wagner, T., M. Emmerich, A. Deutz, and W. Ponweiser. On Expected-Improvement Criteria for Model-based Multi-objective Optimization. *PPSN XI, Part I, LNCS 6238, Krakow, Poland*, 2010, pp. 718–727.

69. Do, C. B. Gaussian processes. 2007, pp. 1–13. Available at: http://see.stanford.edu/materials/aimlcs229/cs229-gp.pdf.

70. Rasmussen, C. E. minimize.m. 2010, Available at: http://learning.eng.cam.ac.uk/carl/code/minimize/minimize.m.

71. Hansen, N., and A. Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary computation*, Vol. 9, No. 2, 2001, pp. 159–195.

72. Lamotte, R. A. F., and C. Alecsandru. Fast Multi-Objective Optimization for Continuous Network Design Problems Based on Gaussian Process Models. Proceedings of the *93rd Annual Meeting of the Transportation Research Board, Washington, D.C.*, 2014.

73. Pictometry International. Aerial photograph. 2012.

74. Huang, D., T. T. Allen, W. I. Notz, and N. Zheng. Global Optimization of Stochastic Black-Box Systems via Sequential Kriging Meta-Models. *Journal of Global Optimization*, Vol. 34, No. 3, 2006, pp. 441–466.

75. J. Forrester, A. I., A. J. Keane, and N. W. Bressloff. Design and Analysis of "Noisy" Computer Experiments. *American Institute of Aeronautics and Astronotics Journal*, Vol. 44, No. 10, 2006, pp. 2331–2339.

76. Ankenman, B., B. L. Nelson, and J. Staum. Stochastic Kriging for Simulation Metamodeling. *Operations Research*, Vol. 58, No. 2, 2010, pp. 371–382.

77. Sasena, M. J. *Flexibility and Efficiency Enhancements for Constrained Global Design Optimization with Kriging Approximations*. Ph. D. diss., University of Michigan, 2002.

78. Lophaven, S. N., H. B. Nielsen, and J. Søndergaard. *Aspects of the matlab toolbox dace*. 2002.

79. The MathWorks. Documentation Center - R2013b Documentation - Choosing a Solver. 2014, Available at: http://www.mathworks.com/help/optim/ug/choosing-a-solver.html#bsbqd7i [Accessed February 1, 2014].

80. Vanhatalo, J., and A. Vehtari. Modelling local and global phenomena with sparse Gaussian processes. *Proceedings of the Twenty-Fourth Conference on Uncertainty in Artificial Intelligence*, 2008.

81. Rasmussen, C. E., and Z. Ghahramani. Infinite Mixtures of Gaussian Process Experts. *Advances in Neural Informaiton Processing Systems 14*, 2002, pp. 881–888.

Appendix

1. EGO equations

While the approach described by Rasmussen and Williams and summed up in the part II.4.3 is very general and quite straight-forward, it may be somehow inappropriate in practice. Indeed, considering all the hyper-parameters together to maximize the log-likelihood might lead to an unnecessary complicated optimization problem. The EGO approach, based on DACE, proposes a seducing alternative. Just as in PAL, the mean and variance of the Gaussian process are chosen as constant:

$$m(X) = \mu \tag{57}$$

$$cov\big(f(x), f(x')\big) = \sigma^2 corr\big(f(x), f(x')\big) \tag{58}$$

In the EGO and DACE approaches however, μ and σ are treated separately from the hyper-parameters θ_j and γ_j that appear in the correlation function:

$$corr\left(f(X_p), f(X_q)\right) = \prod_{j=1}^{n} e^{-\theta_j \cdot \left((X_{p,j} - X_{q,j}\right)^{\gamma_j}} \tag{59}$$

In fact, instead of determining μ and σ with a costly optimization process, the DACE approach analytically computes their maximum likelihood estimates. For more details on how this is done, the reader is referred to the original papers. Thus, all the equations at the core of the DACE approach rely on Pearson's correlation coefficients rather than on the covariance.

In the end, the equations obtained for the expected value and the estimated mean squared error at any point are very close to those obtained in the equations (22) and (23):

$$\mu_{N+1} = m(X_{N+1}) + corr(X_{N+1}, X_{1\dots N}) corr(X_{1\dots N}, X_{1\dots N})^{-1} (\overrightarrow{y_{1\dots N}} - \overrightarrow{m_{1\dots N}}) \tag{60}$$

$$\Sigma_{N+1} = \sigma^2 \big(corr(X_{N+1}, X_{N+1}) - corr(X_{N+1}, X_{1\dots N})\, corr(X_{1\dots N}, X_{1\dots N})^{-1} corr(X_{1\dots N}, X_{N+1})\big) \tag{61}$$

Besides, the prior determination of the standard deviation allows for a simplified expression of the log-likelihood (the concentrated log-likelihood):

$$\log_{\mathrm{EGO}}\left(p(\overrightarrow{y_{1\dots N}}|X_1,\dots X_N)\right) = -\frac{1}{2}\left(Nlog(\sigma^2) + \log(|corr(X_{1\dots N}, X_{1\dots N})|)\right) + constant \quad \textbf{(62)}$$

2. Including the noise in SMS-EGO

Let us assume that we observe $y = h + \epsilon$ and that:

- The noise is an independent zero-mean Gaussian process but not identically distributed:

$$\begin{bmatrix}\epsilon_1\\ \dots\\ \epsilon_N\end{bmatrix} \sim N\left(\vec{0}, \Sigma\right) \quad \textbf{(63)}$$

Where $\Sigma = \begin{pmatrix}\sigma_1^2 & 0 & \dots & 0\\ 0 & \sigma_2^2 & \dots & 0\\ \dots & \dots & \dots & 0\\ 0 & 0 & 0 & \sigma_N^2\end{pmatrix}$

- The underlying process is a Gaussian process:

$$h \sim GP(m,k) \quad \textbf{(64)}$$

By applying exactly the same reasoning as in part II.4.3, one can obtain:

$$\mu_{N+1} = m(X_{N+1}) + K(X_{N+1}, X_{1\dots N})(K(X_{1\dots N}, X_{1\dots N}) + \Sigma)^{-1}(\overrightarrow{y_{1\dots N}} - \overrightarrow{m_{1\dots N}}) \quad \textbf{(65)}$$

$$\Sigma_{N+1} = K(X_{N+1}, X_{N+1}) - K(X_{N+1}, X_{1\dots N})(K(X_{1\dots N}, X_{1\dots N}) + \Sigma)^{-1}K(X_{1\dots N}, X_{N+1}) \quad \textbf{(66)}$$

However, the formulae used in the EGO approach rely on Pearson's correlation coefficients, rather than on the covariance. The definition of Pearson's correlation coefficients is recalled:

$$corr(X,Y) = \frac{cov(X,Y)}{\sigma_X \sigma_Y} \quad \textbf{(67)}$$

Thus, we have:

$$K(X_{N+1}, X_{1\dots N}) = [k(X_{N+1}, X_1) \quad \dots \quad (X_{N+1}, X_N)] = \sigma^2\, corr(X_{N+1}, X_{1\dots N}) \quad \textbf{(68)}$$

$$K(X_{1\dots N}, X_{N+1}) = [k(X_{N+1}, X_1) \quad \dots \quad (X_{N+1}, X_N)]^T = \sigma^2\, corr(X_{1\dots N}, X_{N+1}) \tag{69}$$

And similarly,

$$K(X_{1\dots N}, X_{1\dots N}) + \Sigma = \sigma^2 \left(corr(X_{1\dots N}, X_{1\dots N}) + \frac{\Sigma}{\sigma^2} \right) \tag{70}$$

$$K(X_{N+1}, X_{N+1}) = \sigma^2 corr(X_{N+1}, X_{N+1}) \tag{71}$$

The equations of the EGO approach under noisy conditions can now be obtained by combining the previous equations:

$$\mu_{N+1} = m(X_{N+1}) + \sigma^2 corr(X_{N+1}, X_{1\dots N}) \left(\frac{1}{\sigma^2}\right) \left(corr(X_{1\dots N}, X_{1\dots N}) + \frac{\Sigma}{\sigma^2} \right)^{-1} (\overrightarrow{y_{1\dots N}} - \overrightarrow{m_{1\dots N}}) \tag{72}$$

i.e.

$$\mu_{N+1} = m(X_{N+1}) + corr(X_{N+1}, X_{1\dots N}) \left(corr(X_{1\dots N}, X_{1\dots N}) + \frac{\Sigma}{\sigma^2} \right)^{-1} (\overrightarrow{y_{1\dots N}} - \overrightarrow{m_{1\dots N}}) \tag{73}$$

Similarly for the variance:

$$\begin{aligned} \Sigma_{N+1} = {} & \sigma^2 corr(X_{N+1}, X_{N+1}) \\ & - \left(\sigma^2 corr(X_{N+1}, X_{1\dots N})\right) (\sigma^2)^{-1} \left(corr(X_{1\dots N}, X_{1\dots N}) \right. \\ & \left. + \frac{\Sigma}{\sigma^2} \right)^{-1} \left(\sigma^2 corr(X_{1\dots N}, X_{N+1})\right) \end{aligned} \tag{74}$$

$$\begin{aligned} \Sigma_{N+1} = \sigma^2 \Bigg(& corr(X_{N+1}, X_{N+1}) \\ & - corr(X_{N+1}, X_{1\dots N}) \left(corr(X_{1\dots N}, X_{1\dots N}) \right. \\ & \left. + \frac{\Sigma}{\sigma^2} \right)^{-1} corr(X_{1\dots N}, X_{N+1}) \Bigg) \end{aligned} \tag{75}$$

The only difference with the equations obtained in the noise-free case is the addition of the term $\frac{\Sigma}{\sigma^2}$ to the correlation matrix $corr(X_{1\ldots N}, X_{1\ldots N})$.