UNIVERSIDADE D
COIMBRA

Francisco José Nibau Antunes

Active Learning Metamodels for Transport Simulation Problems

Thesis developed within the Doctoral Program in Transport Systems, supervised by Professors Bernardete Ribeiro and Francisco Pereira, and presented to the Department of Civil Engineering of the Faculty of Sciences and Technology of the University of Coimbra.

March 2021

1 2 9 0

UNIVERSIDADE D
COIMBRA

Francisco José Nibau Antunes

Active Learning Metamodels For Transport Simulation Problems

Thesis developed within the Doctoral Program in Transport Systems, supervised by Professors Bernardete Ribeiro and Francisco Pereira, and presented to the Department of Civil Engineering of the Faculty of Sciences and Technology of the University of Coimbra.

March 2021

Supervision

Bernardete Martins Ribeiro

Full Professor
Department of Informatics Engineering
Faculty of Sciences and Technology of the University of Coimbra

Francisco Câmara Pereira

Full Professor
Department of Technology, Management and Economics - Transport Division
Technical University of Denmark

Dedicated to my parents.

Abstract

Transport systems constitute a fundamental structure deeply hardwired into today's increasingly denser cities. The proper, voluntary, and efficient movement of people, goods, and services is quintessential to healthy and sustainable social and economic advancements.

Urban environments, and their inseparable transport infrastructures, are inherently intricate and highly dynamic. These deeply integrated systems frequently prove to be challenging to model and study due to the multitude of involved variables, external factors, unknown stochastic phenomena, and inevitably human behavior. This overwhelming complexity is not, in most cases, easily encapsulated into closed-form and tractable mathematical formulae, if not indeed infeasible in the first place. As a result, simulation approaches are traditionally employed as modeling tools to explore virtual representations of actual or planned systems to assess their performances ultimately.

Nevertheless, despite encompassing simplified representations of real-world systems, simulation models can too become rather complex and therefore computationally expensive to implement and run, especially if designed with sufficient detail. Even in situations where the simulation runtimes do not represent a significant hindrance to the computer experimentation, input variable spaces with reasonable dimensions can render the exploration of the simulator's output behavior tiresome to attain systematically. In this sense, simulation metamodels often emerge as easy-to-implement solutions to address the just mentioned shortcomings of simulation modeling. Simulation metamodels are essentially functions that aim at approximating the input-output mappings inherently defined by simulation models. These models, which are generally recognized by their computing speed and simple functional structure, are fitted to previously simulated data and then used for output prediction purposes.

In an effort to further minimize the computational burden of exhausting computer experimentation, active learning can be employed in strategic conjunction with simulation metamodels, as it proposes a more efficient modeling approach by aiming at high predictive performance with as few data points as possible. This is achieved by providing the metamodel, or associated algorithm, with the ability to choose the most informative data points in an iterative manner to be run by the simulation model, thereby reducing data redundancy and runtimes while increasing learning efficiency at the same time.

In this thesis, we develop and explore an integrated active learning metamodeling methodology in the context of transport systems simulation. This methodology seeks to improve the exploration process of the simulation input space in order to allow a more efficient description and understanding of the corresponding output behavior. To this end, the Gaussian Process modeling framework is employed as a simulation metamodel due to its non-linear and Bayesian properties, which in turn provide a natural platform for developing native active learning strategies. Several transport-related simulation models and active learning settings are analyzed and discussed.

Despite the clear advantages that both simulation metamodeling and active learning, in the context of expensive or systematic computer experiments, can provide to the transport simulation field, to the best of our knowledge, such techniques remain seldom applied in an integrated manner or even known to transport simulation modelers, practitioners, and related professionals. Many simulation-based studies still rely on somewhat manual processes, primarily focusing on the design of static what-if approaches. We are confident that the adoption of active learning metamodeling schemes can serve as a reliable and enhancing complement to the traditional simulation analysis both in research and industry. Furthermore, we believe that this work will stimulate further discussion, developments, and more applications within the field of transportation.

Keywords: Machine Learning, Active Learning, Transport Simulation Models, Simulation Metamodels, Gaussian Processes

Resumo

Os sistemas de transporte constituem uma estrutura fundamental ao funcionamento das cidades actuais cada vez mais densas e complexas. O movimento adequado, voluntário e eficiente de pessoas, bens e serviços é um elemento essencial para um desenvolvimento económico-social saudável e sustentável.

De uma forma geral, os ambientes urbanos e as suas respectivas infra-estruturas de transportes, são inerentemente intricados, muito dinâmicos e profundamente integrados. Estes sistemas são frequentemente complexos de modelar e de estudar devido às inúmeras variáveis envolvidas, factores externos, fenómenos estocásticos desconhecidos e, inevitavelmente, ao comportamento humano. Na maior parte dos casos, esta enorme complexidade é difícil, se não mesmo impossível, de transpor para fórmulas matemáticas fechadas ou expressões analíticas. Em alternativa, métodos numéricos de simulação são tradicionalmente usados como ferramentas de modelação que permitem, acima de tudo, a exploração virtual de representações conceptuais destes sistemas, actuais ou planeados para o futuro, a fim de avaliar os seus respectivos desempenhos. Todavia, apesar de constituírem representações simplificadas de sistemas do mundo real, os modelos de simulação podem tornar-se igualmente complexos e, por consequência, computacionalmente difíceis de implementar e de executar, especialmente se forem desenhados com detalhe e realismo suficientes. Adicionalmente, mesmo em situações onde os tempos de simulação não são propriamente elevados, os espaços de entrada de simulação, quando caracterizados por um número razoável de dimensões, o que é frequentemente o caso, podem, por si, só tornar as análises do comportamento de saída dos simuladores subjacentes em processos exaustivos quando conduzidos de forma sistemática.

Neste sentido, os metamodelos de simulação apresentam-se como soluções simples de implementar a fim de resolver os problemas computacionais usualmente impostos pelos métodos de simulação descritos anteriormente. Estes tipos de modelos constituem essencialmente funções cujo objectivo é aproximar os mapeamentos de entrada-saída inerentemente definidos pelos respectivos modelos de simulação. Por sua vez, estas funções são geralmente caracterizadas pela sua rapidez de computação e simplicidade, sendo ajustadas a dados simulados e posteriormente usadas para fins de previsão de valores de saída. Num esforço complementar de minimizar ainda mais o potencial entrave computacional associado a processos de análise de simulação exaustiva, estratégias de aprendizagem activa podem ser empregadas numa abordagem integrada com metamodelos de simulação, proporcionando, assim, uma modelação mais eficiente que visa atingir elevado desempenho de previsão com tão poucos dados de treino quanto possível. Isto é concretizável dando ao metamodelo, ou algoritmo associado, a capacidade de escolher, de uma forma iterativa, os pontos mais informativos a serem executados pelo modelo de simulação, reduzindo, deste modo, a redundância de dados e tempos de execução, e consequentemente, aumentando a eficiência da aprendizagem.

Nesta tese, desenvolvemos e exploramos uma metodologia integrada de aprendizagem activa e metamodelos no contexto geral da simulação de sistemas de transporte. Esta metodologia pretende melhorar o processo de exploração dos espaços de entrada de simuladores, no sentido de permitir uma descrição e conhecimento dos respectivos comportamentos de saída de uma forma mais eficiente. Para este fim, a modelação através de Processos Gaussianos foi usada enquanto metamodelo de simulação, devido às suas propriedades não-lineares e Bayesianas que, por sua vez, constituem uma plataforma natural para o desenvolvimento nativo de estratégias de aprendizagem activa. Vários modelos de simulação de transportes e configurações de aprendizagem activa são analisados e discutidos.

Apesar das vantagens que a modelação por metamodelos de simulação e aprendizagem activa apresentam para o ramo da simulação de sistema de transportes, estes tipos de técnicas não são frequentemente aplicados de uma forma integrada ou até conhecidas por parte de modeladores, praticantes e demais profissionais associados à simulação de sistemas de transportes. Muitos dos estudos baseados em técnicas de simulação ainda dependem de processos manuais, focando-se sobretudo no desenho estático de abordagens hipotéticas. Estamos confiantes que a adopção conjunta de esquemas de aprendizagem activa e de metamodelos de simulação pode servir como complemento confiável de modelação às análises de simulação mais tradicionais, tanto na investigação como na indústria. Além disso, acreditamos que este trabalho irá estimular mais discussão, desenvolvimentos e aplicações no ramo de estudo de sistemas de transportes.

Palavras Chave: Aprendizagem Máquina, Aprendizagem Activa, Modelos de Simulação de Transportes, Metamodelos de Simulação, Processos Gaussianos

Acknowledgments

Without any hesitation and second thoughts, my first words of utmost gratitude must go to my supervisors, Professors Bernardete Ribeiro and Francisco Pereira. Thank you for your unmeasurable support, guidance, and patience. Unquestionably, I would not be now writing and submitting this manuscript without your collaboration, experience, and everlasting wise words. You constitute the critical foundations of my training as a researcher and an idyllic reference on how to become one. When I grow up, I want to be like you.

To Professor António Pais Antunes, coordinator of the Doctoral Program in Transport Systems, I express my gratitude for supporting my application and encouraging me since the beginning. Thank you also for all your pieces of advice and all the paperwork on my behalf.

To Professor Christopher Zegras, who kindly hosted me as an international visiting student at MIT, thank you for your insights and knowledge, and for introducing me to some of your students and to the Intelligent Transportation Systems (ITS) Lab. A special thanks also go to Professor Moshe Ben-Akiva, for allowing me to attend his 1.205 Advanced Demand Modeling class during Fall 2018. Additionally, I am also grateful to all MIT-Portugal staff, both in Portugal and in the USA, for their smooth processes, promptness, and general support.

To my American family, Ralph and Steven, and Katharine, I cannot thank you all enough for receiving me in your homes with such warmth and love. If all had gone according to the original plan, I would never have met you. Eventually, and within a cosmic plot twist, the Universe changed its mind and ended up wishing better for me. Living with you and being fortunate enough to meet your beautiful families was one of the best experiences of my life. You are all responsible for that. I miss you. I miss Boston. Thank you for treating me, in the past and present, as one of your own. Not least, to my friend Vahram, whom I also met during my stay on the other side of the Atlantic, thank you for your serenity, wisdom, and, of course, your cooking abilities. And to Ana, for your companionship during our adventure trip to North America.

To all the staff at DTU Transport Division, that made my life extremely easy, especially upon my arrival as a guest Ph.D. student. To all my colleagues at the Machine Learning for Smart Mobility (MLSM) Group, thank you for all conversations during the lunch and coffee breaks. Special thanks are directed to Vishnu and Renming for joining me at the gym, for the endless discussions across various themes and controversial subjects, and for all the walks in Copenhagen. Thank you for your company and friendship.

To all my closest friends and colleagues at the Department of Civil

Engineering (DEC), thank you for all the great moments we spent together, working or not, having fun or just serious conversations about life and what there is. To my oldest best friend, Gonçalo, whom I have ever since regarded as a reference of perfectionism, hard work, and intelligence, thank you for the enthusiastic encouragement and firm belief in my capabilities. And to those who, by design or not, are not mentioned herein, know this: you may not be acknowledged, but you remain in my heart. In one way or another, either by pure randomness or on purpose, you all contributed to defining who I am today, and thus I am immensely grateful to you.

The Fundação para a Ciência e Tecnologia (FCT), and its associated national and European partners, are greatly acknowledged for financing this project within the scope of the MIT-Portugal Doctoral Program in Transportation Systems, under grant no. PD/BD/128047/2016, which additionally made my visits to MIT and DTU a reality. The Research Center for Territory, Transports, and Environment (CITTA) and the Center for Informatics and Systems of the University of Coimbra (CISUC) are also equally acknowledged for their support.

Lastly, but definitely not least, and for supporting everything else and beyond, to my closest family members, as they encompass the most important and irreplaceable people I have in my life. Their names do not require to be called out, as they certainly know whom I am talking about. You are my home, my safe haven, my comfort zone. Thank you for your unconditional support and protection.

Nonetheless, special final words ought to be expressively addressed to my mother, Augusta, to my father, Francisco, and to my late grandparents, Joana, Joaquina, Mário, and Hermínio. You were, you are, and you will always be the ultimate role models I constantly compare myself to and struggle to imitate regularly. More than a trivial direct line of genetic heritage, you raised me, taught me, educated me, and, most importantly, you gave me the opportunity and freedom to pursue what I believe in. Consequently, you irreversibly planted the seeds of the potential of what I can yet become in the future.

Hence, to my grandparents, for the values they have embedded into me. They were the furthest living ancestry I had the honor to meet and directly learn from, a bridge to a collective memory I am so proud of, and to a past that I obviously did not experience but from which I partially derive. To my mother, for enclosing the materialized pinnacle of unconditional love and endless care. And, finally, to my father for being the kindest person I have ever met and the supreme archetype of what a man and a father ought to be like.

Content

List of Figures

List of Tables

Chapter 1

Introduction

"He who has begun is half done. Dare to know, begin!"

Quintus Horatius Flaccus, 65 - 8 BC

According to the World Cities Report 2020, recently released by the United Nations (2020), more than half of the world's population live, as of today, in urbanized areas. This proportion is projected to reach 60.4% by 2030. Generally, it is expected that every region will become increasingly more urban, especially in the developing countries, as today's already highly urbanized areas are expected to experience a slowdown in their urban growths. This steady migration from the countryside-like areas to urban centers, as well as the creation of new ones, will place unprecedented pressure on the existing infrastructures, consequently leading to further and, in many cases, unpredictable urban transformations. For this reason, it is essential to understand the main driving forces underlying urbanization so that we make our increasingly denser cities into safe and sustainable human settlements (Martine et al., 2007; United Nations, 2018).

Human migration alone does not explain this worldwide urban growth. Many other factors, such as global economic forces and technological impacts, have conjointly contributed to the generation of remarkable dynamic transformations within urban environments. In order to maximally exploit these transformations for our own benefit and, at the same time, to minimize or even avoid their potential negative impacts on society, we should somehow be capable of planning for this inevitable urban growth by taking into proper account the subjacent spatiotemporal transformations through a multidimensional prespective (Marengo, 2014).

Urbanization progress typically follows the evolution of transportation (Rodrigue et al., 2016), meaning that, as the urban populaces increase in size, so does too the need for mobility, either within or outside the urban areas. Therefore, transportation systems play a particularly vital role since they constitute the backbone that supports sophisticated social and economic advancements. A healthy economic activity inevitably requires and allows the voluntary movement of people, goods, and services (Moore and Pulidindi, 2013).

Urban environments, and their associated transport infrastructures, are inherently characterized by an overwhelming complexity. In addition to involving numerous interrelated variables and entities, these integrated systems often exhibit stochastic behavior and unknown random phenomena that are not easily encapsulated into closed-form and tractable mathematical formulae. As a result, simulation approaches are traditionally employed as modeling tools, allowing the exploration of virtual representations of actual or planned systems so that their performances can be ultimately assessed.

Simulation modeling is a well-known and established tool to study real-world transport systems or to plan new ones (Fishburn et al., 1995), especially those that prove to be particularly complex to be described as a whole by conventional and pure analytic approaches (Law, 2015). In fact, with the recent and continuous transformations of today's increasingly complex and multidimensional cities, the use of simulation approaches is, in many cases, the only feasible and reliable modeling option to study such systems. Due to their intrinsically exploratory nature, simulation tools prove to be highly advantageous for planning and policy analysis (Bankes, 1992, 1993).

Even though simulation models are simplified representations of reality, they can still and most of the time do become quite complex and computationally expensive to run, especially when embedded with enough detail and realism and when systematic and exhausting experimentation is required. Obviously, this computational hindrance will additionally depend upon the simulation's objectives and the properties of the real-world problem it aims to describe and study.

Within the transportation field, many simulation-based studies primarily focused on forecasting objectives still rely on somewhat manual what-if approaches in an effort to characterize and ultimately discretize uncertainty into a finite and intelligible set of possible futures. Since planning for the future inevitably involves risk and uncertainty, the use of scenarios as descriptive narratives of potential future states of the system under study is widely used in planning and robust decision-making processes (Mietzner and Reger, 2004).

In the context of simulation modeling, an arbitrary combination of input values can be regarded as a specific scenario, as pointed out by Kleijnen (2009). Therefore, a particular scenario can be assigned to a well-defined initial state of a simulated system. From this set of initial values, the system evolves according to the simulation model, and the output values are eventually computed, usually in the form of Key Performance Indicators (KPIs). Moreover, multiple simulation experiments might be required for stochastic simulators, naturally depending on the stochasticity degree, so that a complete description of the entire simulation output distribution is obtained. Consequently, the obtained distribution is then used to explicitly characterize the final simulation state and corresponds to a possible future state of the underlying system under study.

Unsurprisingly, scenario design supported by computer experiments can turn out to be a challenging task, especially when combined with highly dimensional simulation models exhibiting reasonable execution times. Furthermore, although these kinds of what-if approaches are typically guided by expert and domain knowledge, they still risk failing to properly and entirely explore the simulation model as a whole and to identify relevant input values that were not even considered in the first place. By narrowing down the simulation possibilities to a pre-defined and finite set of input value combinations specified according to the hypothesized situations, the simulation analysis becomes restricted to particular small regions of the input space, thereby curbing the reach of its insights and conclusions.

Therefore, the core motivation of this research is to investigate methodological solutions that are capable of overcoming, to a certain extent, the computational hindrances resulting from exhausting computer experiments in the context of transport simulation settings. Additionally, closely in line with the previous, it is also a matter of interest within this work to investigate ways of automatizing the traditional manual workflows usually employed to study the output behavior of simulation models via exploration of its corresponding input spaces.

1.1. Research Approach

Our approach to tackling the computational challenges usually posed by simulation analysis procedures is twofold. On the one hand, we can address the computational burden by providing a framework for approximating the simulation results and enabling prediction over unlabeled regions of the simulation input space. Here, unlabeled regions refer to those parts of the input space that have not been simulated yet or, in other words, not explored through the simulation model. By predicting simulation output values for unlabeled input data points, a reasonable amount of simulation runs and time can, up to a certain extent, be avoided, provided that high prediction performances are guaranteed.

On the other hand, modeling approaches that specifically aim at carefully selecting the data to be simulated for posterior fitting ends can also be adopted. As simulation results are always required for the fitting procedure, simulation models assume an indispensable role in providing ground truth data. Notice that our observations are not part of the real world but instead contained within the digital abstraction generated by the simulation model. Thus, the experimental setting for obtaining new data is obviously not entirely avoidable. However, it can be simplified if the number of required simulation runs are minimized, thereby restricting the simulation execution process to only an essential set of input values while maintaining, at the same time, acceptable levels of fitting performance and generalization capabilities.

Simulation metamodels (Friedman, 2012) constitute a common approach to deal with the computational challenges associated with a systematic exploration of simulation input spaces, being generally employed with the goal of approximating the behavior of simulators. Known for their simple and intelligible formal structure, as well as computational speed, these models are fitted to the input-output data previously sampled from simulation models and are then used for explanatory and prediction purposes, among others (Kleijnen and Sargent, 2000; Kleijnen, 2009). As a result, metamodels provide a modeling framework that allows the exploration of the simulation models' behavior while minimizing the computational burden associated with the computer experiments they aim to mimic.

Nevertheless, as these metamodels explicitly rely on data generated by simulators, they are still dependent, to a great degree, on the complexities exhibited by the underlying simulation models they are designed to approximate. In this sense, feeding a simulation metamodel with previously generated data is as expensive and time-consuming as running the associated simulation model itself. Thus a proper data sampling strategy is necessary to further reduce the impact of this direct dependency. In these specific experimental settings where data is particularly difficult to obtain, Active Learning emerges to play a fundamental role by proposing a more efficient modeling paradigm.

As a sub-field of Machine Learning (Carbonell et al., 1983; Alpaydin, 2020), Active Learning (Settles, 2010), aims at providing high predictive performance with as few data points as possible. The main idea is essentially to increase learning efficiency by reducing redundancy in data. This is attained by designing into the underlying model, or algorithm, the ability to choose the most informative data points from which it learns. Within a simulation setting, such as that described above, one data point corresponds to an input-output n-tuple resulting from a single simulation run.

Active learning approaches are designed as iterative sampling strategies that seek to maximize information gain while maintaining careful and conservative management toward the number of sampled points. Most of these approaches are essentially based on data informativeness concepts, which in turn are typically defined as functions of uncertainty and entropy (Wang and Zhai, 2016). From a modeling perspective, the degree of informativeness of an arbitrary data point can be measured to what extent it constitutes an incremental contribution to prediction performance. In this sense, low informativeness can be broadly categorized as data redundancy, as the contributions to the models' performance improvement are potentially minimal. On the contrary, high informativeness points are potentially associated with higher performance gains.

Furthermore, uncertainty can also be regarded as unexplored and unexploited information potential. Consequently, these types of information-driven sampling strategies can be directly employed in simulation settings, thereby driving the simulation experiments towards the most informative points to minimize the associated computational drawbacks as much as possible.

In the context of transport-related simulation problems, we adopt an approach that integrates simulation metamodeling and active learning, effectively combining the best of both worlds. Whereas metamodels serve as computationally fast auxiliary approximations of simulation models themselves, active learning provides a parsimonious way of selecting the simulation data to which they are posteriorly fitted. A pictorial description of the integrated research approach conducted in this thesis is shown in Figure 1.1, where we can identify the core elements on which our approach is built

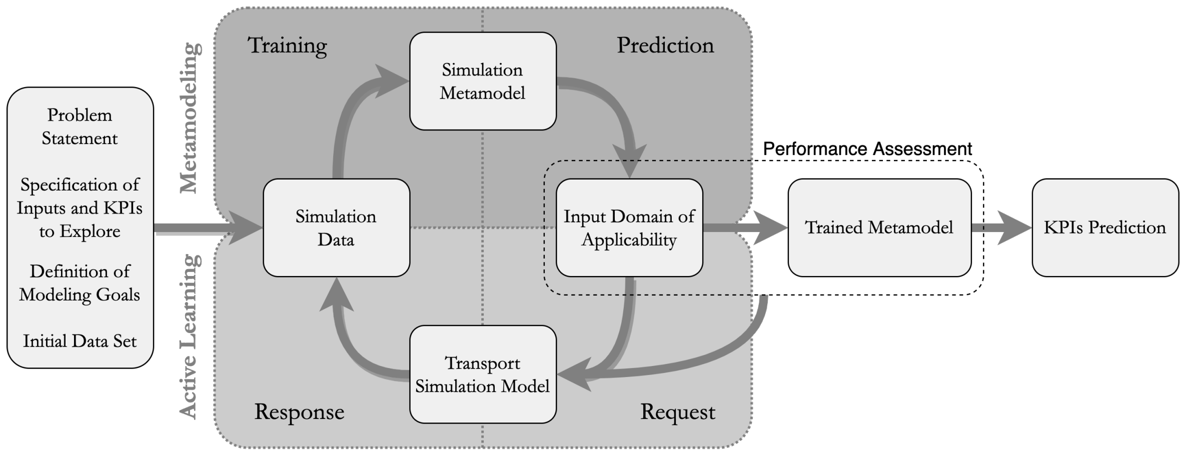


Figure 1.1: Core elements of the research approach and their corresponding relationships.

upon, fundamentally split into two phases, namely, metamodeling and active learning.

The first step of this approach falls on the precise definition of the problem to be addressed through simulation analysis, and along with it, the specification of the relevant input variables to consider, and the corresponding output variables, typically in the form of performance indicators (or KPIs). This step additionally establishes the particular goals of the active learning metamodeling strategy to be employed. Following the specification of the input and output spaces of interest, an initial data set comprising of prior simulation results is constructed so that the metamodel can be trained. Then, the prediction step follows, where the fitted metamodel is used to predict over a given input region of interest, corresponding to the applicability domain in which we aim to explore the simulator's behavior and for which we do not have the associated simulation results. Prediction performance assessment is additionally carried out, effectively ending the metamodeling phase.

Subsequently, the active learning procedure is initiated by requesting the simulation model for new data according to a pre-defined sampling strategy, essentially enclosing some information-based criterion. Note that the simulation model must be constantly accessible for on-demand requests to allow this phase to work correctly. After the latter selectively generates the new data, the training set is expanded, and the metamodel is retrained accordingly. The entire process is then repeated sequentially until a specific stopping criterion (or set of criteria) is satisfied. This criterion should consider the metamodel's relative performance and recognize when no further improvements are attainable solely via the expansion of the training set. Naturally, the user is always capable of fine-tuning the performance criteria, as well as the stopping rule, if these prove to be unsatisfactory for the modeling goals previously set. Finally, the process ends with a trained metamodel specially tailored for the experimental input domain specified at the beginning.

As a side note, notice that we use the terms simulation model and simulator interchangeably for the sake of language simplicity. Whereas the simulation model is the formal abstract representation of the real-world system or problem, the simulator corresponds to its material implementation via a specific simulation platform or programming language. In the context of this work, for all intents and purposes, the differences between both entities are not particularly relevant for our approach. In most cases, however, we mean the latter rather than the former.

It is also worthwhile to mention that it is not our intent to entirely replace the underlying simulation model with the obtained metamodel, and certainly not its crucial role in providing ground truth data. The latter should always be regarded as an approximation for the former, so much so that the critical trade-off balance between computational speed and accuracy loss should be constantly tracked and adjusted according to the metamodeling's ultimate objectives. The metamodel is mainly employed to discover insights from the simulation model's behavior limited to the input region of interest, which is defined, as mentioned earlier, as a function of a particular task or problem to be addressed. It is also essential to constantly keep in mind its approximate nature throughout the entire modeling process so that the correct conclusions are drawn, and improvements are undertaken if required.

In a sense, this approach aims to employ the metamodel side-by-side with the simulation model, never actually discarding the latter. Whereas the metamodel helps to reduce the redundancy of the exploration process, the simulation model guarantees that this exploration is heading in the right direction. This is attained by providing ground truth data whenever the metamodel's prediction uncertainty is high enough. Note that the generalization capabilities of the metamodel are somewhat restrained to the data points which it has already "seen" in previous iterations and their similarities with the uncharted regions of the simulation input space.

We can easily observe the automatic nature of this approach. This feature does not waive, however, proper supervision from an expert in the field along the entire process, especially concerning the final obtained metamodel, whose predictions and general behavior are expected to broadly follow the domain knowledge main premises. Fine-tuning and minor adjustments of the methodology might be required in more complex settings, essentially consisting of the revisiting of the sampling strategies, stopping criteria, and even functional structure of the metamodel itself.

Given a particular transport simulation problem to be addressed, both the type of metamodel employed and the active learning scheme can be regarded as parameters of this integrated approach. Changing such parameters will inevitability change the approach's outcome. On the other hand, the option for the metamodel is closely intertwined with the designed active learning strategy.

Among a wide range of possible machine learning tools, Gaussian Processes (Rasmussen and Williams, 2006) proves to be a complete and versatile modeling approach successfully linking active learning and simulation metamodeling. In fact, it has been traditionally and vastly applied in both fields, making it an obvious choice for this work. As we discuss in Section 2.1.2, the Gaussian Process framework proves to be a flexible modeling platform to develop active learning metamodeling strategies due to its nonlinear and nonparametric properties. In addition, and as a Bayesian tool, its predictions are generated in the form of probability distributions rather than pointwise ones, thereby successfully enclosing a notion of uncertainty and entropy within its own prediction system, which can be ultimately exploited to design native active learning strategies.

At this point, the multidisciplinary nature of this integrated approach should be evident. In Figure 1.2, this thesis is depicted as an intersection of several major research fields of interest. We briefly expand on some of these topics later in Chapter 2, where we provide a succinct background review by visiting each topic individually and in a sequential manner that fits the logic of this thesis.

In summary, and following the motivation of this work, as well as the research approach itself, our research hypothesis is that the combination of both active learning and simulation metamodeling in a single integrated methodology has a great potential to aid and complement the more manual-driven simulation analyses. Furthermore, it also should provide a framework where the simulation input space exploration process can be, according to certain criteria and specific goals, fully automated while easily

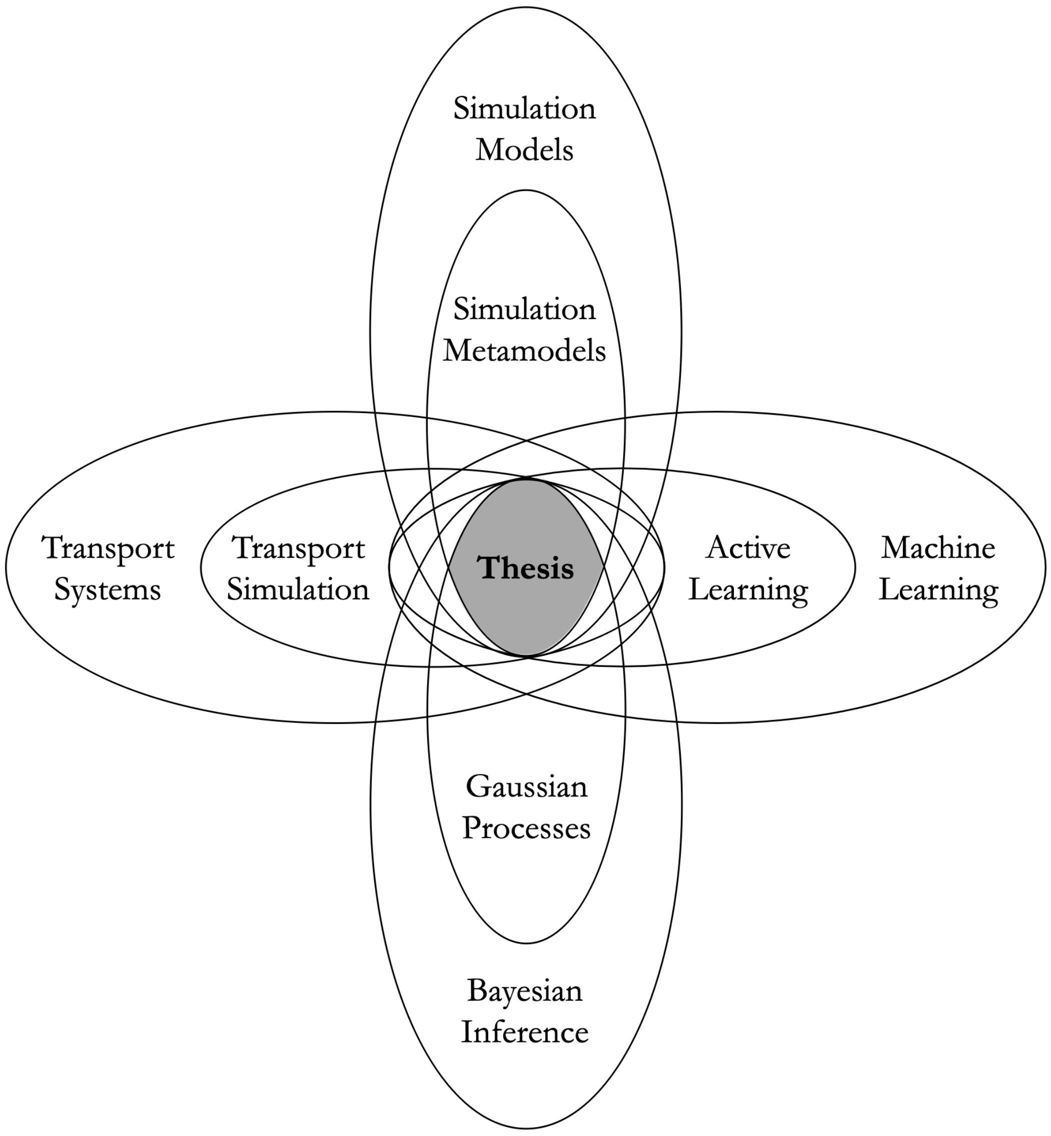


Figure 1.2: Multidisciplinary view of the research approach.

taking into account the proper expert domain knowledge and supervision. As a result, this integrated methodology should reduce the computational workload often associated with exhausting and systematic simulation runs.

These hypotheses should help us to answer the following research questions.

- To what degree is it possible to reduce the computational hassles associated with systematic simulation runs in a significant manner? What are these limits?
- How to use active learning to automatize the metamodel learning process? How to stop it? Which criteria can be used?
- How to sample the simulation model in an efficient fashion, which particularly

focuses on avoiding redundancy between simulation runs?

- Which trade-off thresholds should be set to specifically balance the metamodel's accuracy loss and computing speed? How to define a "good" metamodel approximation as a function of the metamodeling goals? When should we stop expanding the training set?

- How to decrease the number of required simulation runs while maintaining a reasonable understanding of the underlying model to draw correct and meaningful conclusions?

1.2. Research Goals and Contributions

Despite the potential advantages that both simulation metamodeling and active learning, in the context of expensive and systematic computer experiments, can provide to the transport simulation field, it is our understanding that such techniques remain seldom applied by or even known to transport simulation modelers and practitioners, as well as to other related professionals. As mentioned earlier, many simulation-based studies still rely on rather manual processes that end up reducing the entire simulation input space into intelligible and finite sets of input values combinations in order to understand particular regions of the simulators' output behavior. This often poses a real practical hindrance to the exploration of the simulation model's behavior.

Accordingly, the main goal of this work is to develop and explore the feasibility of active learning metamodeling methodologies focused on transportation-related simulation settings. Consequently, another explicit objective is to contribute to closing the gap between the transport simulation and the machine learning communities. The strong multidisciplinary nature of this thesis is a testimony of these aims, hoping that it stimulates further discussion and applications in the field.

Summarily, the goals of this research are as follows.

- Explore the feasibility of active learning metamodeling strategies in transport-related simulation problems.

- Automatize, to a certain extent, the metamodeling process through active learning strategies in order to complement or reduce the need for traditionally manual processes, decreasing the associated computational hindrances.

- Show the potential of the methodology to aid traditional exploratory analyses of simulated transport systems, thereby enhancing what-if planning approaches and policy analysis processes.
- Serve as a discussion platform and foster the collaboration between machine learning, transport research, and the specialized industry of transport simulation.
- Promote and further develop integrated approaches combining simulation metamodeling and active learning within the field of transportation.

Essentially, the contributions of this thesis are of methodological nature. The explored integrated methodology to deal with the potential shortcoming of simulation-driven exploratory analysis has not been properly addressed within the transport research field. On the one hand, metamodels aim to approximate the function defined by the simulation models themselves. On the other hand, active learning tries to attain higher fitting performance in a rather economical manner. Our approach is to combine these techniques in order to offer a fully integrated methodology where i) metamodels are used as parsimonious approximations of simulation models and ii) trained via active learning schemes to boost fitting efficiency, the latter being tantamount to executing the simulation model to the least possible extent. This methodology also provides a framework to automatize the exploration processes often manually employed in simulation analysis.

In the following, we briefly present the core contributions of this work organized into four independent chapters.

- Chapter 3: a batch-mode active learning metamodeling scheme is proposed, along with two user-defined stopping criteria. A toy data set, a Demand Responsive Transportation simulator and a traffic simulator were used as case studies. The results show that accounting for diversity within each batch can make the exploration process more efficient. Furthermore, a user-defined parameter is introduced, forcing each batch to comprise points originating from different high variance regions. Batch-mode active learning strategies also allow, up to some extent, the parallelization of the simulation requests across different machines or multi-threat processes, thereby allowing to further speed up the metamodeling process.

- Chapter 4: in the context of a simulated emergency medical system, an active learning metamodeling approach was explored to assess its performance outcome. The recommended guideline of eight minutes for the maximum emergency response time, and its relation with the survival rate, was studied. The results show a great utility potential for decision-making processes and policy analysis applications, as the methodology provides a parsimonious approach for fast policy testing and design. It can be easily expanded to allow multiple policy analyses to be carried out at once if required.

- Chapter 5: a heuristic based on simulation metamodeling was designed in order to specifically address the problem of exhausting simulation exploration within a simulation-optimization setting. The metamodeling framework, which implements a local search heuristic approach, provides a fast track for evaluating alternative solutions with a minor but controlled accuracy loss. Therefore, a metamodel was used to replace the more computationally expensive simulation model, allowing empirical evidence from the simulated system to be inferred instead of computed, saving a reasonable amount of computational hassle in the process.

- Chapter 6: an algorithm based on a sequence of training grids, iteratively suggested by the metamodel, steers the simulation exploration process towards input regions that trigger a specific pre-defined output value of interest, resembling a reserve engineering scheme. Here, for a given simulation output value, or set of values, this active learning metamodeling approach aims at identifying, in an efficient way, the set of input values that trigger the latter. This approach is of great applicational value, as it provides an auxiliary tool to discover input regions in which the degree of success or failure of a given policy is defined according to a pre-defined simulation output value, or sets of values, of interest.

1.3. Research Dissemination

Within the framework of no. 2 of article 31º of the Decree-Law no. 74/2006 from March 24th of the Portuguese Law, parts of this thesis have been previously submitted to international peer-reviewed journals of recognized merit across the fields of transportation, simulation, and machine learning, namely, IEEE Transactions in Intelligent Transportation Systems, T-ITS (published), Simulation Modeling Practice and Theory, SIMPAT (published), Computers & Industrial Engineering, CAIE (published), and Transportmetrica A: Transportation Science (under review). These four papers are presented in this manuscript as Chapters 3, 4, 5, and 6, respectively. Therefore, the present manuscript is partly a compilation of published or submitted work.

Moreover, during the course of this doctoral project, albeit not directly related to it, work has been developed and further published in an international peer-reviewed journal, essentially encompassing Gaussian Processes within nonparametric machine learning approaches as those used herein.

In the following, we present the details of the aforementioned works published and submitted during the course of this thesis, essentially encompassing the period from 2017 to 2020.

- Antunes, F., Ribeiro, B., Pereira, F. and Gomes, R. (2018) Efficient Transport Simulation with Restricted Batch-Mode Active Learning, IEEE Transactions on Intelligent Transportation Systems (T-ITS), Vol. 19, pp. 3642 - 3651, ISSN 1558-0016.

- Antunes, F., Amorim, M., Pereira, F. and Ribeiro, B. (2019) Active Learning Metamodeling for Policy Analysis: Application to an Emergency Medical Service simulator, Simulation Modelling Practice, and Theory (SIMPAT), Vol. 97, pp. 101947, ISSN 1569-190X.

- Amorim, M., Antunes, F., Ferreira, S. and Couto, A. (2019) An Integrated Approach for Strategic and Tactical Decisions for The Emergency Medical Service: Exploring Optimization and Metamodel-Based Simulation for Vehicle Location, Computers and Industrial Engineering (CAIE), Vol. 137, pp. 106057, ISSN 0360-8352.

- Antunes, F., Amorim, M., Pereira, F. and Ribeiro, B. (2020) Active Learning Metamodeling for Survival Analysis of a Simulated Emergency Medical System, Transportmetrica A: Transportation Science, under review.

- Antunes, F., Ribeiro, B. and Pereira, F., (2017) Probabilistic Modeling and Visualization for Bankruptcy Data Analytics, Applied Soft Computing, Elsevier, Vol. 60, pp. 831-843, ISSN 1568-4946.

Most of the contents developed in this work have also been totally or partially presented and discussed in several conferences and meetings, both national and international, as chronologically listed below. Some of these works have additionally been published as part of conference proceedings, when available. One technical report was also produced.

- Antunes, F., Ribeiro, B. and Pereira, F., (2017) Active Learning for Scenario Exploration in Transport & Land-Use Simulation Models, 14th Annual Transports Study Group Conference (GET), February 20-21, Fátima, Portugal.
- Antunes, F., Ribeiro, B., Pereira, F. and Zegras, C., (2017) Active Learning Metamodeling for Scenario Discovery in Transportation Simulators, Poster Session, Transport Summer Summit at Technical University of Denmark (DTU), May 31st, Copenhagen, Denmark.
- Antunes, F., Ribeiro, B. and Pereira, F., (2017) Active Learning GP-based Metamodels for Input Exploration in Transportation Simulation Models, in the Proc. of the 23rd Portuguese Conference on Pattern Recognition (RECPAD), October 27th, Lisbon, Portugal.
- Antunes, F., Ribeiro, B., Pereira, F. and Gomes, R., (2017) Active Learning Metamodels for Transportation Simulators, Tech. Report no. 2017-005, CISUC, University of Coimbra.
- Antunes, F., Ribeiro, B. and Pereira, F., (2018) Restricted Batch-Mode Active Learning Metamodels for Transport Simulators, 15th Annual Transports Study Group Conference (GET), February 19-20, Fátima, Portugal.
- Antunes, F., Ribeiro, B. and Pereira, F., (2018) Active Learning for Input Space Exploration in Traffic Simulators, in the Proc. of the International Joint Conference on Neural Networks (IJCNN) in IEEE World Congress on Computational Intelligence (WCCI), July 8-13, Rio de Janeiro, Brazil, ISSN 2161-4407.

- Coimbra, L., Antunes, F., Seco, A. and Ribeiro, R. (2018) Exploratory Scenario Design in Transportation Planning, in the Proc. of the 8th Luso-Brazilian Congress for Urban, Regional, Integrated and Sustainable Planning (PLURIS 2018), October 24-26, Coimbra, Portugal, ISSN 2525-7390.

- Antunes, F., Ribeiro, B. and Pereira, F., (2018) Exploring the Output Behavior of Traffic Simulators using Batch-mode Active Learning, Poster Session, National Science Summit'18 (Ciência 2018), July 2-4, Lisbon, Portugal.

- Antunes, F., Ribeiro, B. and Pereira, F., (2018) Application of Active Learning Metamodels and Clustering Techniques to Emergency Medical Service Policy Analysis, in the Proc. of the 24th Portuguese Conference on Pattern Recognition (RECPAD), October 26th, Coimbra, Portugal.

- Antunes F., Amorim, M., Ribeiro, B. and Pereira, F., (2019) An Active Learning Metamodeling Approach for Policy Analysis: Application to an Emergency Medical Service Simulator, paper no. 19-05677, Transport Research Board (TRB) 98th Annual Meeting, January 13-17, Washington D.C., USA.

- Antunes, F., Amorim, M., Pereira, F., and Ribeiro, B. (2019) Policy Analysis for Emergency Medical Systems using Simulation Metamodels, Poster Session, National Science Summit'18 (Ciência 2019), July 2-4, Lisbon, Portugal.

- Antunes, F., Amorim, M., Pereira, F. and Ribeiro, B. (2019) Metamodeling based on active learning: application to an emergency medical service simulator, 12th CITTA International Conference on Planning Research, September 19-20, Porto, Portugal.

- Antunes, F., Pereira, F. and Ribeiro, B. (2019) Studying the behavior of a simulated Emergency Medical System using Active Learning, in the Proc. of the 25th Portuguese Conference on Pattern Recognition (RECPAD), October 31st, Porto, Portugal.

- Antunes, F., Pereira, F. and Ribeiro, B. (2019) Directional Grid-based Search for Simulation Metamodeling using Active Learning, in Proc. of the 3rd EAI International Conference on Intelligent Transport Systems (INTSYS 2019), December

4-6, Braga, Portugal, Intelligent Transport Systems from Research and Development to the Market Uptake, Chapter 3, pp. 32-46, ISBN 978-3-030-38822-5.

Furthermore, some of the ideas explored in this thesis are being further developed and implemented within the ongoing EU-funded research project NOSTROMO - Next-generation Open-Source Tools for ATM performance Modeling and Optimization, under grant agreement no. 892517 of the Single European Sky ATM Research (SESAR) Joint Undertaking under the European Union's Horizon 2020 research and innovation programme. Among other objectives, NOSTROMO aims to develop novel approaches to the field of Air Traffic Management (ATM) modeling, mostly relying on simulation metamodels and active learning to aid the study of ATM performance at the European Civil Aviation Conference (ECAC) level. The active learning metamodeling methodology being adopted within NOSTROMO heavily builds upon the contents of this thesis which constitute a starting point for the exploratory nature of the project.

This initiative, which officially kicked off in June 2020, is led by Centro de Referencia de Investigación Desarrollo e Innovación ATM - CRIDA (Spain), involving several other research partners and industry key players in the field, namely, Nommon Solutions and Technologies - NOMMON (Spain), Universitat Politècnica de Catalunya - UPC (Spain), ISA Software Limited - ISA (United Kingdom), University of Westminster (UoW, United Kingdom), and Technical University of Denmark - DTU (Denmark).

1.4. Thesis Structure

The present manuscript is organized into eight chapters. Except for Chapters 1 (Introduction), 2 (Research Background), and 7 (Conclusion), all the remaining chapters are written in the format of self-contained scientific papers and can thus be read independently. Due to this type of organization, a partial overlap between some chapter's contents is present, especially at the introduction and background revision levels. The order of each of these chapters reflects a chronological and iterative sequence that synchronizes with the work developed throughout the entire doctoral project. In the following, we summarize the contents of this thesis.

Chapter 1 introduces the motivation, aims, and scope of the thesis. It also frames the current manuscript as a compilation of previously presented, published, and submitted works during the doctoral project's period from 2017 to 2020.

Chapter 2 provides the essential background check on active learning and simulation metamodels. Although brief, the adequate presentation of these concepts is fundamental for the understanding of the subsequent chapters and especially the presented and explored methodologies. Some of these contents are also presented to some extent in the introductory sections of the following chapters.

Chapter 3 presents a batch-mode active learning strategy based on simulation metamodeling using three different experimental settings. In order to take advantage of parallelization processing technologies increasingly available nowadays, an active learning metamodeling strategy that queries the simulator in batches of simulation requests, which can then be executed independently in different machines. We also introduce a parameter that forces the batches to be formed by points from different spatially distanced high variance neighborhoods, thus speeding up the overall variance reduction.

Chapter 4 explores the application of active learning metamodeling to policy analysis. Building upon the work presented in the previous chapter, a metamodel was employed to study the performance outcome of a simulated medical emergency system. In particular, we analyze in what conditions such a system fulfills a specific policy regarding a recommended maximum response time threshold. The methodology shows that active learning metamodels constitute a promising auxiliary tool that can complement simulation analysis concerning the designing and testing of transport-related policies.

Chapter 5 integrates simulation metamodeling with optimization through the development of a metamodel-based heuristic. A metamodel was designed to explore and validate alternative solutions as a parsimonious substitute for the original and more computationally expensive simulation model. Although with a minor accuracy loss, resulting from the metamodeling itself, an increased amount of solutions can be tested more efficiently.

Chapter 6 proposes a directional active learning scheme that explores the simulation input space for regions that trigger specific pre-defined output values, resembling a reverse engineering approach. Here, our aim is to narrow down the input space exploration effort to the region, which, via simulation, produces output values close to that defined a priori by the user. Policy analysis can be effectively carried out in a less computational-intense manner if such output value of interest is specified as a consequence of a particular decision boundary from which, for example, certain strategy beings to fail or succeed.

Finally, Chapter 7 concludes this thesis. Here, we discuss several limitations of the proposed methodology. Furthermore, and motivated by the latter, we point out some challenges yet to be addressed, as well as other alternative directions for future research.

Chapter 2

Research Background

"Remember that all models are wrong; the practical question is how wrong do they have to be to not be useful."

George E. P. Box, 1919 - 2013

This chapter provides a succinct overview of the concepts, models, and techniques to properly frame this work's intrinsically multidisciplinary nature, complementing the dedicated background review already present within Chapters 3 to 6.

In Section 2.1 we review the background modeling paradigms and tools that are part of the methodology we explore, namely Machine Learning, Bayesian Inference, Active Learning, and Gaussian Processes. Closing this chapter, Section 2.2 presents a primer on the Simulation Modeling and the corresponding Metamodels. We do not intend to conduct a thorough coverage since most of these topics can quickly become quite vast and fall out of scope. Instead, we focus on the details that fit this thesis's purpose, framing it in the context of a broader and multidisciplinary field.

2.1. Machine Learning

In 1950, the mathematician Alan Turing introduced a test, originally called the Imitation Game but later named after him, in an effort to answer the question "Can machines think?" (Turing, 1950). This test constituted the first formal and operational way to assess intelligent behavior exhibited by computers. It was primarily devised to determine if a computer program could learn from experience and communicate with the outside world in such a way that it would pass for a human being. In other words, Turing's goals were to somehow measure the intelligence of a computer and to what extent it could indistinguishably imitate human behavior. This and many other contemporary works, which tried to systematically define and tackle human consciousness and intelligence from a computational perspective, marked the intellectual foundations for the posterior emergence of the Artificial Intelligence (AI) field (French, 2000).

It is unanimous that a computer performs exceptionally well in logic-based processes, such as arithmetic and other intrinsically mechanical operations. However, the same does not hold true with respect to their ability to reproduce human thinking.

AI can be broadly defined as the intelligence exhibited by computers, and it is used to describe those situations where a computer program seeks to mimic the human mind and its associated cognitive functions. Here, the crucial assumption is that if we are able to understand how the human mind works, we can somehow embed its functional architecture into computer programs, thereby enabling them to perceive the world from a human perspective and model it accordingly. Therefore, AI research is a generic approach aiming to develop intelligent machines that try to replicate human cognitive tasks, such as reasoning, natural language processing, visual perception, and prediction, among others. A popular approach to achieving such goals is Machine Learning (ML).

ML is a subfield of AI that focuses on the study and development of models and algorithms that can learn from data and consequently solve specific tasks. The main principle behind ML is that of enhancing the machines, or computer programs, with the ability of continuous learning from the exposure to new data and through the experience without being explicitly programmed. In fact, Samuel (2000), one of the pioneers in ML studies, pointed out that designing computers to learn from experience should lead to the reduction of detailed programming work.

Inspired by Mitchell et al. (1997), we can generally describe a given machine learning task as the following quadruple

$$(\mathcal{M}, \mathcal{T}, \mathcal{P}, \mathcal{E})$$

where $\mathcal{M}$ is any model, computer program or algorithm designed to perform task $\mathcal{T}$, whose performance is assessed by the metric $\mathcal{P}$ and improved via experience $\mathcal{E}$. Notice that the performance metric $\mathcal{P}$ tracks the learning success of the model, ultimately driving the entire learning process.

In terms of terminology, the data to which the ML model is fitted is commonly called the training set. Conversely, the test set, which should be disjoint from the latter, is used to assess their prediction and generalization performances. These learning and test phases do not waive proper data treatment and pre-processing, usually tailored for both the model input structure and the problem of interest. Depending on the type of tasks and their associated goals, ML can be divided into three major categories, namely, supervised, unsupervised, and reinforcement learning. Whereas the first learns from explicitly labeled data by extracting shared properties (or features) associated with each label or target, the second aims to discover groups of similarity within unlabeled observations automatically. Finally, reinforcement learning is concerned with taking actions that maximize a reward function given specific conditions. In contrast to supervised learning, the algorithm searches for the optimal outputs based on a trial and error interaction with a particular environment (Bishop, 2006; Goodfellow et al., 2016).

ML is an actively growing and promising field with many applications spanning across the majority of engineering and technological fields. Since the beginning of the 21st century, the proliferation of ubiquitous intelligent technologies (Kindberg and Fox, 2002) and the development of the Internet of Things (IoT) paradigm (Li et al., 2015), supported upon the Information Technology revolution, has culminated in an exponential data growth never experienced before. In the most different contexts, data is being generated at unprecedented speeds and volumes every day. The intense use of network-enabled devices and interfaces is just one of the many causes of this explosion and data availability. On the other hand, however, these significant volumes of data are oftentimes characterized by a high degree of heterogeneity, incompleteness, and unstructured noise, ultimately calling for an urgent increase in the employment of data-driven approaches and methodologies, especially those embedded with uncertainty-handling properties (Antunes et al., 2017).

The transportation industry is currently one of the most popular and visible areas of ML applications and research. Together with the mentioned advent of IoT and Ubiquitous Sensing, ML has been playing a fundamental role in the development of Intelligent Transportation Systems (Zantalis et al., 2019). Smart transport and communication infrastructures, self-driving vehicles, smart ticketing, and real-time demand prediction encompass a few examples of the technological impacts that will definitively change our everyday lives and as well as our relationship towards transportation itself.

From an epistemological perspective, ML can be divided into several major lines of thought, depending on the view of the world, assumptions, problems of interest, and how these are ultimately addressed. Domingos (2015) brilliantly proposes five tribes, namely, symbolists, connectionists, evolutionaries, analogizers, and the Bayesians.

The symbolists believe that intelligence manifests itself through symbols and, consequently, that the logical manipulation of such symbols leads to further insights into the problems. Their main approach is inverse deduction, meaning that it comprises the filling of gaps in the current knowledge in order to make deductions and eventually generalize it as much as possible. For the connectionists, the main goal is to reverse engineer the brain as they believe that, in order to make sense out of the data, we have to emulate the human mind by understanding how neural networks work. Backpropagation is their main learning framework, essentially consisting of applying continuous changes at the level of the neuron connections until the model's output compares favorably with the desired one.

As for the evolutionaries, they assert that natural selection is the supreme learning scheme. Thus, understanding and simulating evolution is their main goal. Their primary approach is genetic programming, which incorporates many evolutionary biology concepts making computer programs able to evolve and improve themselves in similar ways to that of Nature. The analogizers' primary aim is to find similarities, or analogies, between entities and situations so that other similarities can be inferred. Their preferred learning approach is supported by kernel-based techniques, namely the support vector machine. Kernels are functions that essentially evaluate the similarity between two arbitrary data points, constituting the base for many of the generalization capabilities of such methods.

Finally, for the Bayesians, understanding the world cannot be conducted if uncertainty is not taken into proper account and carefully handled, as they believe that all knowledge and the learning process itself are inevitably uncertain. Hence, the ultimate approach is mostly based on probabilistic inference through the lens of Bayes' theorem, thereby providing an optimal learning framework to face uncertainty and incorporate new upcoming evidence into their prior belief system.

It is worthwhile noting that some problems might not be uniquely approachable via one of these views alone. Instead, integrated solutions comprising two or more different perspectives might be required to properly address them, each of them contributing with their own set of unique beliefs and distinctive tools.

In this thesis, we mostly follow a Bayesian perspective combined with that of the analogizers' as our primary approach for obtaining new insights from data within an integrated modeling framework. In the next section, we briefly expand on the Bayesian inference formalism as a starting point to introduce several intertwined key components that compose the methodology explored in this work.

2.1.1. Bayesian Inference

A fundamental concept residing at the core of most machine learning approaches is that of uncertainty. The ability to deal with uncertainty is often built over probability and statistical theories, which provide consistent modeling systems able to quantify and manipulate various forms of the unknown. These frameworks facilitate the development of powerful inference tools, even when the available information is incomplete or ambiguous (Bishop, 2006; Murphy, 2012; McElreath, 2020).

Within the field of statistical inference, there are two major perspectives on probability, namely, frequentist (or classical) and Bayesian inference. The divergences between both approaches are historical and of philosophical nature with respect to statistical reasoning (Vallverdú, 2015). In the frequentist view, the notion of probability is defined as the limit of the relative frequency of a particular event over a large number of well-defined and systematic random experiments. This theoretical limit is fixed and unknown beforehand, and it can be inferred by statistical analysis of random observations. On the other hand, in Bayesian inference, probabilities are used to measure the degree of belief (or certainty) that one has with respect to a given statement or evidence of interest, whether it is random or not. Consequently, it essentially assigns probabilities to possible states of the unknown truth while providing, at the same time, an explicit formula to revise them in a simple iterative manner as new data is available for analysis, as well as keeping track of the prior knowledge (Box and Tiao, 2011).

The Bayesian inference framework derives primarily from the Bayes' theorem, also called Bayes' law or rule, which provides a structured way to update probability statements given new pieces of evidence. Let us consider two random events, A and B, exhibiting some form of mutual dependency. In its simplest form, the Bayes' theorem states that

$$P(A|B) = \frac{P(B|A) \times P(A)}{P(B)},$$

deriving directly from the definition of conditional probability applied twice, i.e.,

$$P(A|B) = \frac{P(A \cap B)}{P(B)} = \frac{P(B \cap A)}{P(B)} = \frac{P(B|A) \times P(A)}{P(B)}.$$

This rule expresses the conditional probability of the event A given the observation, or realization, of the event B. This probability is usually called posterior probability, or posterior distribution, of A with respect to the observation of B. It is defined in terms of the prior probabilities of A and B, and also of the conditional probability of B given A. The prior probability of A, $P(A)$, refers to what is known of A regardless of B, whereas the probability of B given A, $P(B|A)$, is the likelihood of the observed B for all possible values of A. Finally, the probability of B, $P(B)$, can be viewed as a scaling factor, and often called the marginal likelihood, or evidence, that guarantees that $P(A|B)$ remains a probability distribution in $[0, 1]$. For this reason, Bayes' rule is often simplified to

$$P(A|B) \propto P(A) \times P(B|A),$$

where '$\propto$' stands for "proportional to".

Now following Box and Tiao (2011), consider a vector of observations $\mathbf{y}$ sampled from an arbitrary probability distribution, $p(\mathbf{y}|\Theta)$, depending on a set of parameters Θ. After replacing A, B and $P(.)$ respectively by Θ, $\mathbf{y}$ and $p(.)$, we obtain the refurbished Bayes' formula

$$p(\Theta|\mathbf{y}) = \frac{p(\mathbf{y}|\Theta) \times p(\Theta)}{p(\mathbf{y})}.$$

Here, $p(\Theta)$ represents the prior distribution over the parameter set, which reflects our beliefs regarding the parameters before $\mathbf{y}$ is observed. On the other hand, $p(\mathbf{y}|\Theta)$ is the likelihood of the parameters given the observed data when viewed as a function of Θ and not of $\mathbf{y}$, and for this reason oftentimes rewritten to $l(\Theta|\mathbf{y})$. We can do this because the data is fixed. Lastly, the posterior distribution, which combines the likelihood and the prior, is given by $p(\Theta|\mathbf{y})$. This distribution essentially encloses the uncertainty about the parameters in Θ, given the observed data and the prior. Again, the marginal likelihood, $p(\mathbf{y})$, is a normalizing constant defined by the following integral

$$p(\mathbf{y}) = \int p(\mathbf{y}|\Theta) \times p(\Theta) \; d\Theta.$$

Notice that, since the integration is over the set of parameters, $p(\mathbf{y})$ is independent from Θ, thus it can be set to an unknown constant of proportionality. Eventually, the

Bayes' formula can again be simplified to

$$\underbrace{p(\Theta|\mathbf{y})}_{posterior} \propto \underbrace{l(\Theta|\mathbf{y})}_{likelihood} \times \underbrace{p(\Theta)}_{prior}.$$

From this expression, we observe that the likelihood function plays a rather crucial role, as it is through it that the observed data, or evidence, revises the prior knowledge of Θ. Accordingly, it can be regarded as the information about Θ contained within the observations.

As mentioned earlier, the iterative nature of the Bayesian formalism allows for an elegant and recurrent formulation for incorporating new information into the old knowledge as more observations are taken into consideration. For illustration, consider a initial set of observations $\mathbf{y}_1$. Hence, via Bayes' rule we obtain

$$p(\Theta|\mathbf{y}_1) \propto l(\Theta|\mathbf{y}_1) \times p(\Theta).$$

We can combine a new set of observations, independent from the first, via

$$\begin{aligned} p(\Theta|\mathbf{y}_2, \mathbf{y}_1) &\propto l(\Theta|\mathbf{y}_2, \mathbf{y}_1) \times p(\Theta) \\ &\propto l(\Theta|\mathbf{y}_2) \times l(\Theta|\mathbf{y}_1) \times p(\Theta) \\ &\propto l(\Theta|\mathbf{y}_2) \times p(\Theta|\mathbf{y}_1). \end{aligned}$$

Note that the prior for $\mathbf{y}_2$ is exactly the posterior for $\mathbf{y}_1$, $p(\Theta|\mathbf{y}_1)$. As expected, this step can be repeated sequentially as more observations are available. Consider n independent samples. Thus, the posterior distribution for Θ given m previous observations comes in the sequential form of

$$p(\Theta|\mathbf{y}_1, \dots, \mathbf{y}_m) \propto p(\Theta|\mathbf{y}_1, \dots, \mathbf{y}_{m-1}) \times l(\Theta|\mathbf{y}_m),$$

with $m = 2, \dots, n$.

In sum, we see that Bayesian analysis is based on a framework that strives to make use of all the available information present within the data, thereby embedding prior knowledge in its inference system. Such a system allows for evidence originating from newly observed data to be elegantly integrated into our assumptions, modifying them whenever required. Additionally, it provides a consistent iterative model building procedure which is a fundamental characteristic of the scientific method, explicitly miming the process of learning from observation and experience (Box and Tiao, 2011).

Bayesian methods have recently emerged from a quite specialized field of statistics, becoming more mainstream and accessible to the data science and machine learning communities and related research fields. Furthermore, its practical applicability has also been enhanced by the dissemination of kernel-based models that provide flexible ways to conduct high-dimensional data analysis, offering new data-oriented approaches to today's ever-increasing complex problems (Bishop, 2006). In the following, we present one of these kernel-based modeling approaches, which represents a state-of-the-art and sophisticated nonlinear prediction tool within the Bayesian reasoning niche.

2.1.2. Gaussian Processes

Despite being an old topic and widely applied tool among the statistics literature, the Gaussian Process (GP) modeling framework (Rasmussen and Williams, 2006) has been gaining much attention over the past decade or so, particularly within the machine learning community. Their simplicity, ease of implementation, as well as their nonlinear and nonparametric abilities, make the GPs a powerful modeling tool in a wide range of regression and classification problems across numerous specialized fields. Unlike other well-known related techniques such as Support Vector Machines (SVM) (Vapnik, 1963) or Neural Networks (NN) (McCulloch and Pitts, 1943), the GP equally constitutes a supervised learning technique but provides a fully Bayesian approach, meaning that probabilities are used to quantify the uncertainty present within its predictions, essentially through Bayes' theorem (Bishop, 2006).

Within regression modeling settings, the first step is to identify the variables of interest for which we aim to establish some kind of functional relationship. Hence, an generic D-dimensional data set by $\mathcal{S} = \{(\mathbf{x}_i, y_i)|i = 1, \dots, n\}$, where $\mathbf{x} \in \mathbb{R}^D$ is an input vector, y the output (or target) variable, n is the number of observations, and D the size of the input feature space, must be firstly defined. Here, for the sake of simplification, we only consider a continuous input space. On the other hand, as we focus on regression problems, y must be continuous by definition. Aggregating the n input vectors yields the $(D \times n)$ design matrix X. Similarly for the n targets, we obtain $\mathbf{y}$. After these rearrangements, the data set can now be rewritten as $\mathcal{S} = (X, \mathbf{y}) \subseteq \mathbb{R}^{D\times(n+1)}$. Finally, assuming a certain functional dependency between the input and output variables, we want to infer the conditional distribution of $\mathbf{y}$ given the inputs X, $p(\mathbf{y}|X)$.

As Rasmussen and Williams (2006) puts it, a GP is a collection, possibly infinite, of stochastic variables $f = (f_t, t \in \mathcal{T})$, where any finite set of which jointly follows a multivariate Gaussian distribution. Here, $\mathcal{T}$ is a set of indexes. A strong point on the GP's favor lies on its simplicity when it comes to full characterization. Actually, a GP is completely defined by a single pair of functions, namely, the mean and covariance (or kernel), respectively denoted by $m_f(\mathbf{x})$ and $k_f(\mathbf{x}, \mathbf{x}')$, where $\mathbf{x}'$ is another input vector. Hence, a GP can be simply denoted as

$$\mathcal{GP}(m_f(\mathbf{x}), k_f(\mathbf{x}, \mathbf{x}')),$$

where

$$\begin{aligned} m_f(\mathbf{x}) &= \mathbb{E}[f(\mathbf{x})] \\ k_f(\mathbf{x}, \mathbf{x}') &= Cov(f(\mathbf{x}), f(\mathbf{x}')) \\ &= \mathbb{E}[(f(\mathbf{x}) - m_f(\mathbf{x}))(f(\mathbf{x}') - m_f(\mathbf{x}'))]. \end{aligned}$$

In this modeling context, the regression approach assumes a GP prior over functions, i.e., $y = f(\mathbf{x}) + \epsilon$, where $\epsilon \sim \mathcal{N}(0, \sigma^2)$ and $f(\mathbf{x}) \sim \mathcal{GP}(m_f(\mathbf{x}), k_f(\mathbf{x}, \mathbf{x}'))$. We view $f(\mathbf{x})$ as a random sample from a GP process.

Commonly for most applications, and also for simplicity reasons, we set $(\mathbf{x}) = 0$, since the mean of the posterior process is not thereby limited to zero. From this point, the prior over the latent function is then given by

$$p(\mathbf{f}|\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n) = \mathcal{N}(0, K_f),$$

where $\mathbf{f} = [f_1, f_2, \ldots, f_n]^\top$, $f_i \triangleq f(\mathbf{x}_i)$ and K_f is the covariance matrix, with entries $[K_f]_{ij} = k_f(\mathbf{x}_i, \mathbf{x}_j)$.

The covariance function, k_f, has a crucial role in the GP modeling, as it encapsulates the similarity between two any given data points and, consequently, its generalization capabilities. Besides these two input points, an arbitrary covariance function typically has a certain number of free parameters, θ, also called hyperparameter of the process, which can be estimated by optimizing, subjected to the training data, the marginal likelihood generically given by

$$p(\mathbf{y}|X) = \int \underbrace{p(\mathbf{y}|\mathbf{f}, X)}_{likelihood} \underbrace{p(\mathbf{f}|X)}_{prior} \, d\mathbf{f}.$$

After some calculations, it comes in the form of

$$\log p(\mathbf{y}|X,\theta) = -\frac{1}{2}\mathbf{y}^\top K_y^{-1}\mathbf{y} - \frac{1}{2}\log|K_y| - \frac{n}{2}\log(2\pi), \tag{2.1}$$

where $K_y = K_f + \sigma^2 I$ and K_f are the covariance matrices for the noisy targets y and noise-free latent variable f, respectively.

The Squared Exponential or Radial Basis Function is a common default choice for the GPs' kernel. It is generically defined as

$$k(\mathbf{x},\mathbf{x}') = \sigma_f^2 \exp\left(-\frac{1}{2}(\mathbf{x}-\mathbf{x}')^\top M(\mathbf{x}-\mathbf{x}')\right),$$

where σ_f^2 corresponds to the variance of the underlying signal function f, $M = diag(\boldsymbol{\sigma})^{-2}$, and $\boldsymbol{\sigma} = [\sigma_1, \sigma_2, \ldots, \sigma_D]^\top$ is a positive real-valued vector. The elements of $\boldsymbol{\sigma}$ are the characteristic length-scales associated with each of the D dimensions of the input feature space. The presence of these scales allow the hyperparameters optimization procedure to select the right weights for each input dimension according to their individual magnitude, additionally explaining why sometimes this function is called Squared-Exponential with Automatic Relevance Determination (SE-ARD). If the data is normalized, that is to say, if all the dimensions share a similar scale l^2, then the standard SE can be used directly. This version is obtained by setting $\sigma_1 = \sigma_2 = \ldots = \sigma_D = l^2$ thus yielding

$$k_f(\mathbf{x},\mathbf{x}') = \sigma_f^2 exp\left(-\frac{||\mathbf{x}-\mathbf{x}'||^2}{2l^2}\right),$$

where $||.||$ is the Euclidean norm.

After the hyperparameters are set, the posterior distribution for a single test point $\mathbf{x}_*$ is given by

$$f_*|X,\mathbf{y},\mathbf{x}_* \sim \mathcal{N}(\bar{\mathbf{f}}_*, Cov(\mathbf{f}_*)),$$

with

$$\begin{aligned}\bar{\mathbf{f}}_* &\triangleq \mathbb{E}[f_*|X,\mathbf{y},\mathbf{x}_*] = k_{f*}^\top [K_y]^{-1}\mathbf{y},\\ Cov(\mathbf{f}_*) &= \mathbb{V}[f_*] = k_{f**} - k_{f*}^\top [K_y]^{-1} k_{f*},\end{aligned}$$

where $k_{f*} = k_f(X,\mathbf{x}_*)$ and $k_{f**} = k_f(\mathbf{x}_*,\mathbf{x}_*)$.

We can now see that each single GP prediction is described by a Gaussian distribution with certain mean and variance values. Thus, each prediction corresponds to a whole range of values weighted by a well-defined probability density function. A $(1-\alpha)\%$ Confidence Interval for $\mathbf{y}^*$ is therefore given by

$$\bar{\mathbf{f}}_* \pm z_{\frac{\alpha}{2}} \sqrt{Cov(\mathbf{f}_*)},$$

where $z_{\frac{\alpha}{2}}$ is the $\frac{\alpha}{2}$th order quantile of $\mathcal{N}(0,1)$ and $\alpha \in (0,1)$. Within this setting, $\bar{\mathbf{f}}_*$ and $Cov(\mathbf{f}_*)$ are usually called the predictive mean and variance of the GP.

Through Bayesian formalism, the GP framework is able to characterize the variance within its own predictions. In fact, the predictive variance is a critical component in GP modeling, and it can be used to maximize information acquisition if regarded as a measure of uncertainty. Observe that any given GP prediction is deemed uncertain if its associated predictive variance is high. In other words, the higher the variance is, the more scattered the predictive distribution becomes around the mean value, consequently increasing the range of the most likely values to be predicted. Conversely, low variances imply narrower distributions concentrating most of the probability density in the vicinity of the expected value.

Notwithstanding, the cause for high predictive variance is two-fold, namely, inherent data variance and model uncertainty. Whereas the first relates to data variation caused by measurement errors, stochastic nature of the underlying process to be modeled or sampling scheme, among others, the second refers to the uncertainty of the prediction model in and of itself. Both types of variances can coincide, and, as a matter of fact, they should. In this particular situation, the predictive variance expressed by the model would approximately match the real variation inherently characterizing the observed data. Particularly in scenarios where a limited number of training samples are readily available for modeling and costly to generate, this can become quite a challenging task to fulfill. Efficient strategies must be adopted in order to reduce the involved computational burden while trying to aim at a model improvement at the same time.

In the following, we review a machine learning paradigm specially designed to jointly tackle the intertwined problems of model uncertainty reduction and performance improvement by actively acquiring the most informative data points within experimental settings where labeled data is challenging to obtain.

2.1.3. Active Learning

Active learning, also called query learning, consists of an iterative learning approach that aims to attain high prediction performance with as few training samples as possible. This is achieved by designing into the underlying learning process the ability to, according to certain criteria, actively select the data points from which it learns, proving to be particularly useful for modeling tasks where labeled data is difficult to obtain (Settles, 2010).

The general idea of active learning is to sequentially select the most informative data points that simultaneously boost the model training efficiency and improve its predictive performance. Hence, it is essentially focused on improving the overall model prediction performance, as well as on controlling the costs associated with acquiring new labeled instances. The entire learning process is guided by a label provider, also called the oracle, whose task is to provide labeled data instances which are then successively incorporated into the expanding training data set. Hence, a core assumption of active learning is that a label provider must be permanently accessible to be queried on-demand by the learning algorithm or model. The oracle is traditionally represented by a human annotator. Likewise, and depending on the experimental setting and application domain, its role can be played by a sensor, a simulator, a function, another algorithm, among others. The most important aspect of the oracle is that it successfully and consistently generates labeled instances from the ground truth function, inherently defining the process of interest.

As elegantly presented by Li and Sethi (2006) and Wang and Zhai (2016), although sightly modified to fit the purpose of the present work, an arbitrary active learning system encompasses five fundamental entities and can be summarized in the following quintuple

$$(\mathcal{L}, \mathcal{U}, \mathcal{M}, \mathcal{O}, \mathcal{Q}).$$

First, $\mathcal{L} = \{(\mathbf{x}_i, y_i) | \mathbf{x}_i \in \mathbb{R}^d, y_i \in L, 1 \leq i \leq l\}$ is the labeled training set, where d and l are positive constants, and L is the set containing the labels. On the other hand, the set of unlabeled data points is denoted by $\mathcal{U} = \{\mathbf{x}_i | \mathbf{x}_i \in \mathbb{R}^d, l+1 \leq i \leq l+n\}$. Generally we have that $n >> l$, i.e., the number of unlabeled points is significantly higher than the labeled ones, formally enclosing the earlier mentioned scenarios characterized by shortage of labeled data, as well as limited resources.

By its turn, $\mathcal{M}$ is the machine learning model, which, depending on the nature of the problem in question, can be a classification or a regression model. Within classification problems, L becomes discrete and finite (number of classes), whereas for regression approaches $y_i \in L \subseteq \mathbb{R}^p$, where p is the number of continuous target variables, and typically $p = 1$. Then, the oracle is represented by $\mathcal{O}$. Finally, $\mathcal{Q}$ is the query function that encodes the strategies and criteria for finding and selecting the most informative instances from $\mathcal{U}$ to be added to $\mathcal{L}$.

Different querying strategies can be encoded into function $\mathcal{Q}$, essentially representing different approaches to the underlying modeling problem. Settles (2010) lists several query frameworks, such as uncertainty sampling (Lewis and Gale, 1994), query-by-committee (Seung et al., 1992), expected model change (Settles et al., 2008) and error reduction (Roy and McCallum, 2001), variance reduction (Cohn et al., 1996) or density-weighted methods (Settles and Craven, 2008). In sum, most of these frameworks rely either on measures of informativeness or on the identification of uncertainty regions. Whereas the former addresses the querying problem by measuring the potential degree of information contained within an unobserved instance, the latter explicitly defines regions of uncertainty where the model is less confident with respect to its own predictions.

Closely related to the concept of the query function is the definition of stopping rules. Due to its inner iterative nature, active learning schemes must be stopped at a certain point in time. Although there is extensive research under the active learning paradigm, not much research has been dedicated to the study of stopping criteria (Vlachos, 2008; Wang et al., 2014), mainly focusing instead on the learning algorithm itself. A few suggestions can be found in the literature (Ghayoomi, 2010; Laws and Schätze, 2008; Vlachos, 2008; Wang et al., 2014; Zhu et al., 2010). Nevertheless, all the available criteria are fairly identical and generally based on an intrinsic measure of stability or confidence of the learning model (Settles, 2010) and, according to Wang and Zhai (2016), there is no stopping rule that is simultaneously suitable in all applications.

Regardless of the details of its definition, it is generally expected that the stated stopping criteria should take into account the constant trade-off between model performance and the cost of acquiring new labeled points. It is imperative to be able to identify those situations in which the model is reaching its theoretical performance threshold, that is to say when the addition of new training points will potentially have a negligible effect on its performance improvement. Here, the cost of requesting new labels, which can be assumed to be constant through the entire learning process, might prove to be highly disproportional in relation to its associated benefits.

Moreover, active learning systems can be further specialized depending on the availability of the unlabeled instances and on how they are presented to the model before they are sent out for query thereafter. According to (Settles, 2010; Wang and Zhai, 2016; Kumar and Gupta, 2020), three major working settings can occur, namely, i) membership query synthesis (Angluin, 1988), ii) stream-based (Zhu et al., 2007) and iii) pool-based (Lewis and Catlett, 1994).

In the first experimental setting, the model generates synthetic queries according to some membership hypotheses, especially near the decision boundary. On the stream-based case, the unlabeled instances are available in sequential order, one by one or in blocks. Here, the model does not directly request the oracle for new labels at will. Instead, it analyses a continuous stream of unlabeled instances and then decides whether or not to query them. Thus, the stream itself is independent of the model. Finally, the pool-based version corresponds to the most popular setting as it matches the requirements of most active learning applications. In this type of learning scheme, the entire space of unlabeled instances is available for on-demand queries at any given time and multiple times if required. Thus, the set of unlabeled data forms a pool from which the model chooses the points to be queried from the oracle, obviously following specific selection criteria. Conversely, the training set is renamed as the labeled pool, featuring a much lower cardinality than the latter, as previously seen.

Most of the existing active learning literature revolves around classification problems. Within the machine learning niche, developments and applications in regression problems are comparatively less abundant, as pointed out by Kumar and Gupta (2020). On the other hand, it has a long tradition in statistics, where it is commonly known as optimal experimental design (Settles, 2010; Fedorov, 1972).

In this thesis, we direct our focus to active learning applied within a regression setting where a simulation model plays the role of the oracle. Furthermore, a pool-based scheme is adopted since the whole range of possible instances are available for query, i.e., the simulator can be executed for any value combination within its input space.

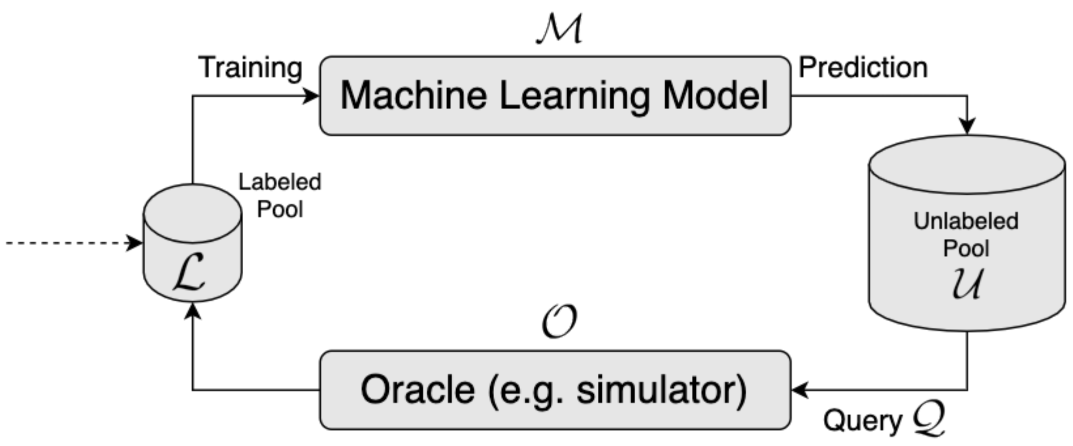


Figure 2.1: Illustration of the pool-based active learning approach adopted in this thesis. Albeit not explicitly depicted, any active learning system starts with an initial training set with which the machine learning model is firstly trained. Then, prediction is performed over the unlabeled pool, and model assessment is conducted. Afterward, according to certain criteria, the most informative points are selected to be labeled by the oracle. The resulting labeled data is finally added to the training set, thereby expanding it. The cyclic process is repeated until a given stopping rule is satisfied. In the particular context of this thesis, the unlabeled pool can be regarded as a population from which we can sample indefinitely.

This scheme can also be regarded as a population-based one since the entire range of possible input data values, that is to say, our population of input space points is always available for on-demand querying by default. In this context, a label encompasses any output value produced by a computer experiment, while the cost of this label acquisition corresponds to the computational burden of running the simulation model. Figure 2.1 illustrates this approach.

The core concept underlying active learning is that seeking out the most informative points and exploiting uncertainty regions of the input domain space reduces data redundancy and boosts model training efficiency, besides saving significant computational resources in the process. In this regard, the previously discussed GP framework proves to be an appropriate modeling tool that, due to its nonparametric and Bayesian properties, allows it to handle both informativeness and uncertainty in a relatively natural and intuitive way. In fact, active learning methodologies involving GPs are oftentimes associated with exploration-exploitation problems within Bayesian optimization contexts (Ling et al., 2016; Jones et al., 1998). Under these approaches, Bayesian optimization employs a reward function to more efficiently select the next unlabeled data point, according to a dynamic trade-off between high uncertainty (exploration) and high informative (exploitation) input domain regions (Schulz et al., 2017).

Information and uncertainty measurement is inherently related to the key concept of entropy (Cover and Thomas, 2006). Given a discrete random variable X with support S and probability mass function $p(x)$, its entropy is given by

$$H(X) = -\sum_{x \in S} p(x) log\big(p(x)\big).$$

Similarly, if S was continuous, the differential entropy, which is essentially the same concept of entropy but for continuous random variables, would be reformulated as

$$h(X) = -\int_S f(x) log\big(f(x)\big) dx.$$

Now, remember that the GP predictions come in the form of Gaussian distributions, and, as a consequence, we can compute the entropy associated with each of these predictions. Thus, the differential entropy of a variable $Y \sim \mathcal{N}(\mu, \sigma^2)$ is given by

$$h(Y) = \frac{1}{2} log\big(2\pi e \sigma^2\big),$$

where we can immediately observe that it is a function of the variance only, further supporting the idea that high predictive variance can serve as a proxy for a high degree of uncertainty in the context of the GP framework. This shows why the predictive variance has such a central role when designing active learning strategies that seek to maximize information gain by exploiting the model's prediction uncertainties with as few training examples as possible.

It is worthwhile to mention, however, that not all high variance regions are necessarily linked to potential information gains and uncertainty reductions when queried. Instead, such regions, which the model itself clearly describes, might be a consequence of the observed data's inherent variability. If the system under study is characterized by relatively high variability, the GP itself will try to reproduce that same behavior assigning high variance to its predictions. In these cases, where the predictive variance matches the data variability, less or no new information is potentially retrievable, as the GP is already closely describing the variance of the underlying process. Hence, active learning should also be able to recognize lower thresholds of uncertainty from which no further reduction is allowed, i.e., from which more information gains are no longer possible.

In the next section, we discuss how GPs and active learning can be jointly employed in the context of simulation analysis and their close relation to simulation metamodeling methodologies.

2.2. Simulation Models

Simulation models are conceptual mimic representations of a real-world phenomenon, as well as its associated processes and intertwined entities. The set of these processes of interest is often called a system, upon which several assumptions are made. These assumptions usually take the form of mathematical or logical relationships, which are then encoded into a conceptual framework, i.e., a formal model, in order to study the behavior of the corresponding system. When the relationships constituting this model are sufficiently simple and straightforward, it is often possible to apply mathematical methods, such as algebra or calculus, to obtain analytic solutions for the problems being studied. However, since most of the real-world systems are overwhelmingly complex to be approached by pure analytic methods, simulation models are used instead. In a simulation approach, a computer program is used to implement and numerically evaluate a model and to generate data from which the true model characteristics are estimated, typically via statistical inference (Law, 2015).

Within a simulation model's design, there are essentially two kinds of variables that control and describe its behavior, namely input variables, also called factors, and output or response variables. For the sake of parsimony, the terms 'inputs' and 'outputs' are often used instead. Moreover, some authors distinguish between inputs and parameters. Whereas the former is directly observable, the latter must be obtained through statistical inference (Kleijnen and Sargent, 2000). In this work, however, and without loss of generalization, we may use both terms interchangeably. Depending on the random nature of these variables, the simulation model can be either deterministic or stochastic. In deterministic models, there is no probabilistic component, meaning that the output is univocally determined once the input values are specified. Contrariwise, in the stochastic models, the same input values combination originates multiples output values. These combinations are also called scenarios or simply points (Kleijnen, 2009).

Following Friedman (2012), the primary goal of a simulation model and its subsequent experimentation trials is to achieve a deep understanding of the relationship between the input and output variables of the real-world system. This system can be actually functioning or a planned one. The mentioned relationship can be summarily formalized as

$$\mu = \phi(\mathbf{x}_1, \mathbf{x}_1, \ldots, \mathbf{x}_q),$$

where μ is the output variable and $\mathbf{x}_i$, $\forall i \in \{1, \ldots, q\}$, are the input variables. Note that, for the sake of simplicity, we only considered one output of interest. Here, μ is a proxy measure or indicator that characterizes the evolution of the real-world system over time, whereas the inputs $\mathbf{x}_i$, which can be controllable or environmental, determine the value of the system's behavior.

The relationship, ϕ, which is unknown as it is intrinsically defined by underlying processes governing the system, describes how the input variables affect the output. Thus, the aim of simulation is to approximate this function through the simulation model, denoted by f and formally defined as

$$y_i = f(\mathbf{x}_{i1}, \ldots, \mathbf{x}_{ik}, \mathbf{r}_{-i}),\ i = 1, \ldots, n.$$

Here, y_i represents the simulation output from the ith run, n is the total number of computer runs, k is the number of input variables actually considered, and x_{ij} is the jth input variable from the ith simulation run. Finally, $\mathbf{r}_{-i}$ encompasses the joint effect of the $q - k$ inputs from the real-world system that were not considered in the modeling. Alternatively, $\mathbf{r}_{-i}$ can be equivalently regarded as the simulation experimental error from the ith run, ϵ_i, that measures the discrepancy between y_i and the true value μ, $\epsilon_i = \mu - y_i$. The revisited expression comes as follows

$$y_i = f(x_{i1}, x_{i2}, \ldots, x_{ik}, \epsilon_i),\ i = 1, \ldots, n.$$

The minimization of ϵ actually corresponds to the calibration of the simulation model's parameters with respect to the reality. It should be sufficiently low so that $f \approx \phi$, according to the simulation's objectives. The error degree will depend upon the ability of $(\mathbf{x}_{i1}, \ldots, \mathbf{x}_{ik}, \mathbf{r}_{-i})$ to alone explain μ.

The process of a simulation study essentially involves four major sequential steps along with verification and validation feedback loops (Sanchez, 2007), as illustrated in Figure 2.2. The first step of this modeling process consists of the definition of

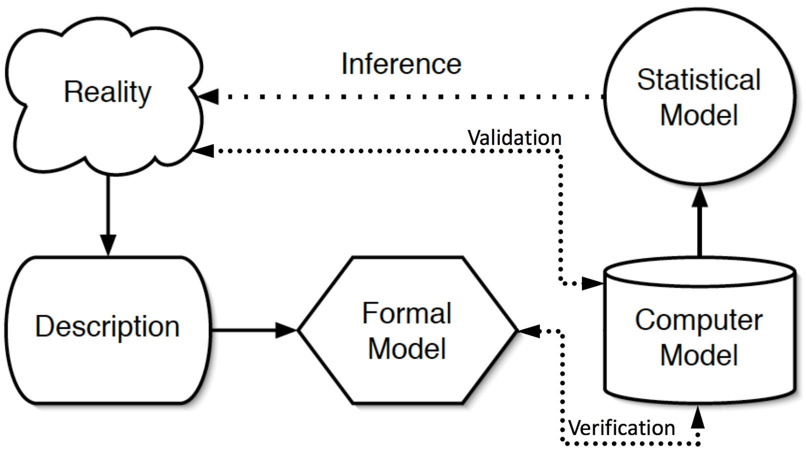


Figure 2.2: Simulation modeling process overview, including the verification and validation feedback loops. Adapted from Sanchez (2007).

the problem and the identification of the system of interest, as well as the scope and objectives of the simulation study. It also includes the selection of one or more system performance measures and the identification of the input variables. Eventually, we should end step 1 with a descriptive model of the real-world system of interest. Next, step 2 regards the conceptual description of the interactions between all the entities encompassed in the system and their interrelated behaviors. This is essentially attained via mathematical or logical relationships, such as, for example, causal relationships, differential equations systems, probability distributions, or optimization formulas. This step should culminate in the precise definition of the formal model, corresponding to an abstraction of the real-world system. Then, in step 3, the implementation of the simulation model itself is conducted, essentially comprising the conversion of the formal model into an executable computer program using an appropriate programming language. Finally, step 4 involves the development of a statistical model of the computer model for posterior inference.

We expect our simulation model to have a high degree of accordance with respect to the real-world system so that we can obtain appropriate insights and conclusions about the real problem under study. In the case of deterministic simulation models, their level of accordance with reality is essentially dependent on how well the formal model was defined. However, if the simulation uses stochastic schemes as part of its modeling process, the corresponding outputs are also stochastic and must therefore be treated only as an estimate of the real system output values.

As pointed out by Sanchez (2007), it is naive to think that running a stochastic model a single time is sufficient to achieve the solution to the problem, which is too often not true. Hence, the proper way to study the behavior of a stochastic system is through statistical inference over the data generated by the computer model across multiple trials. It is here that the verification and validation feedback loops play an essential role in the overall simulation process.

The verification feedback loop is meant to assess how reliable the computer model is with respect to the formal model. Whereas the formal model expresses the conceptual representation of the real-world system, the computer model represents its implementation in practice. Thus, the verification loop corresponds to the task of debugging the computer model in order to ensure that it computes precisely according to the rules of the formal model. On the other hand, the validation feedback loop guarantees that the computer model is a valid representation of the real-world system. Moreover, it is worthwhile to mention that the verification must be a part of the validation loop for obvious reasons. If the computer model does not correctly replicate our conceptual framework for the real problem, then it does not make sense to proceed to its subsequent validation.

One way to systematically and efficiently verify and validate a simulation model, and its associated output behavior, is by means of an experimental design. The statistical principles underlying the field of Design of Experiments (DOE) were initially developed during the 20's by Fisher (1937). The first experimental design methods were applied in the agriculture context and later expanded to the military and industry. These applications consisted of real-world experiments rather than simulated ones. Although the former can prove to be very different from virtual environments, classic experimental designs can be easily transposed to cover current simulation models. In this context, DOE generally aims to specify efficient sampling strategies, mainly through the identification of suitable combinations of input variables, with which simulation data should be systemically obtained in order to maximize information gain with minimal computational effort (Telford, 2007).

According to Sanchez (2005), there are three fundamental concepts in DOE, namely, control, replication, and randomization. The control highlights the systematic nature of the experiment, clearly distinguishing itself from ad hoc and trial-and-error approaches.

Replication provides a way to increase the sample size in order to achieve high statistical significance. Finally, randomization essentially protects against hidden bias sources that can compromise the quality of the attained results.

The research on DOEs is vast, and it has been around for quite some time now. However, a few types of designs are particularly recurrent in the simulation literature. Perhaps the most popular is the Latin Hypercube (LH) design (Iman and Conover, 1980; McKay et al., 2000). An arbitrarily experimental design can be represented in matrix notation and for that let us now consider the $p \times d$ matrix denoted by $\mathbf{X} = [\mathbf{x}_1 \ \mathbf{x}_2 \ \ldots \ \mathbf{x}_p]^\top$, where each column represents one of the d input variables. Conversely, each row $\mathbf{x}_i = [x_i^1 \ x_i^2 \ \ldots \ x_i^d]$ corresponds to one of the p samples. The matrix of an LH design is then constructed by assuring that each dimension d is divided into p equal levels containing a unique sample each. Note that here the sample refers to a particular point within the space of input variables, consequently corresponding to an input value combination that has not been simulated yet. Most importantly is that LH designs can be optimized without taking into account the statistical assumptions of the simulation model. Nevertheless, this type of optimization has already proven to be quite challenging as it is essentially a combinatorial optimization problem with order $p!^d$. As experimental designs are, at their core, developed to guide and optimize the simulation input space exploration and posterior analysis, general properties like uniformity, space-filling (coverage), and orthogonality are often expected from reasonably good one-shot experimental designs (Viana, 2016).

As mentioned earlier, the goal of any experimental design is to increase the efficiency of the simulation analysis process, that is to say, to search for the most informative input variables and their associated value combinations that have the most significant impact on the output system responses with the least possible number of simulations runs. Thus, at this point, the similarities between the DOE and the active learning paradigm, previously discussed in Section 2.1.3, should emerge quite obviously, explaining why the latter is also frequently called optimal experimental design.

There is no question that simulation is a powerful and reliable modeling tool to approach overwhelming problems and intricate systems. In fact, this is particularly true for studying transportation systems within urban environments (Fishburn et al., 1995; Kitamura and Kuwahara, 2006; Chung and Dumont, 2009), as these typically exhibit massive stochastic behavior involving a multitude of variables not always easy to identify and model, as well as human behavior. Such systems are usually not analytically tractable. Hence, insofar as simulated systems are concerned, numerical methods are by and large the only reliable approaches available in most of these cases.

As pointed out by Law (2015), among other advantages, simulation models allow the performance assessment of over different time frames and horizons, not only of currently existing systems but also of alternative system designs under a set of future operational and environmental conditions or interventions. Most of these evaluations would not be feasible to conduct in actual and functioning real-world environments. On the contrary, an obvious disadvantage is that the development of detailed simulation models, and their corresponding computer implementation, are traditionally disproportionately time-consuming tasks. Moreover, the possibility of developing simulation models that are too simplistic or poorly detailed to faithfully represent the real system may lead to misleading conclusions when the results are confronted with the initial intended simulation goals. Finally, given the stochastic nature of most approaches, a single simulation run merely produces a pointwise estimate for the actual underlying formal model. Therefore, several independent runs, which, as a whole, can end up being expensive in terms of computational time and resources, must be conducted across input variables and corresponding value ranges.

In the following section, we briefly discuss a methodology and a family of models that aim to address the main drawbacks of simulation modeling, especially when exhausting and systematic input space exploration is required.

2.3. Simulation Metamodels

The development and use of simulation metamodels (Friedman, 2012) can be traced back, at least, to the beginning of the 70s. Also known as emulators (McClarren, 2018), surrogates (Gramacy, 2020), response surfaces (Box et al., 1987), or sometimes more broadly as auxiliary models, these kinds of models aim, by definition, to approximate the function inherently defined by the simulation model, previously denoted as f in Section 2.2. Whereas a simulation model corresponds to an abstraction of reality, the simulation metamodel consists of an abstraction of the simulation model itself. In simple terms, a metamodel is a model of a model. The research on this topic is quite abundant and relatively sparse, in the sense that different application fields tend to use specific nomenclature and notations. Henceforth, for the sake of simplicity, we adopt the term 'metamodel,' as it better fits the scope of this thesis from a conceptual point of view.

Contrary to the simulation models, metamodels are explicitly defined by known and clear mathematical functions, which can be evaluated at any given point of their input space in a rather effortless manner. Mathematical simplicity, speed, and interpretability are characteristics typically attributed to metamodels. Consequently, the use of metamodels within simulation analysis provides an additional level of understanding of the underlying system, as well as of the relationships between the system's input and output variables, while maintaining a computationally straightforward and economical approach to the problem (Friedman and Pressman, 1988). The failure to meet these simplicity requirements can render the metamodeling approach unfeasible and even counter-productive.

In Figure 2.3, the general simulation metamodeling framework is presented, relating the problem entity under study, the simulation model, and the simulation metamodel itself. Placed on the second level of abstraction, the metamodel should be a valid representation of both the problem entity (e.g., some real-world system) and the simulation model itself. Nevertheless, the validity degree is closely related to the accuracy requirements, which eventually depend on the metamodeling goals.

Kleijnen and Sargent (2000) discusses four primary goals, namely, problem entity understanding, simulation output prediction, optimization, and verification/validation. The scope of this thesis concerns mostly with understanding the underlying real-world system and with the assessment of metamodel's prediction performance. In this work, our assumption is that the simulation model of interest is entirely validated, verified, optimized, and calibrated with respect to the problem under study. Thus, we are mostly concerned with the prediction of the simulation output.

As already mentioned, simulation metamodels are essentially input/output functions that are employed in such a way as to approximate the true, and usually much more complicated, function inherently defined by the simulation model itself. Whereas the function of the problem entity is intrinsically defined by "nature" (Kleijnen and Van Beers, 2004), the unknown function of the simulation model can be viewed as the

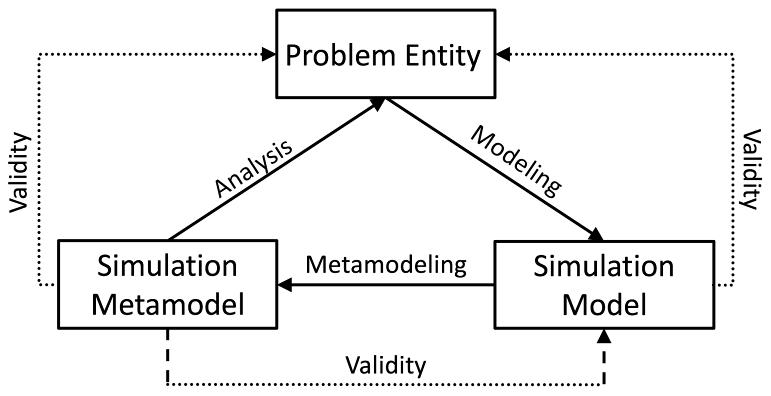


Figure 2.3: Relationship between problem entity, simulation model and simulation metamodel. Both the metamodel and the simulation model should be validated against the problem entity, which encompasses any real-world system being modeled. Then the metamodel, which, depending on its modeling goals, should be a sufficiently valid approximation of the simulation model, is used to effortlessly explore its inner structure, gaining insights and providing an auxiliary analysis tool for the problem under study. Adapted from Kleijnen and Sargent (2000).

emergent behavior resulting from the interactions of its variables and induced causality logic. Contrariwise, the metamodel's functional form is explicitly specified a priori by the user, with many of its inputs being commonly shared with those of the simulation model, although it is not entirely necessary. The metamodel's input variables can be regarded as independent but defined as functions of the original simulation inputs. If the relationship between these two sets of variables follows the identity function, then the inputs are shared between both frameworks. Otherwise, they are different by means of functional transformation.

Despite the apparent usefulness of metamodeling, it is unwise to consider that the simulation model can be modeled, as a whole, by the corresponding metamodel. This would be rather impractical, if not virtually impossible, for most simulation approaches with numerous input variables, as the number of required simulation runs would be too high, inevitably rendering the metamodeling itself unattractive and computationally unable to meet its goals. Instead, as suggested in Kleijnen and Sargent (2000), the domain of applicability, or experimental region, should be clearly specified. This encompasses the definition of the input domain or region for which the metamodel should be a valid approximation. It can be regarded as restricting the metamodel fitting to a particular and bounded sub-set contained within the much larger simulation input space and may not even consider all the input variables at once, which is too often the case. Wang and Shan (2006) actually discusses this problem by alerting for the fact that metamodels do too suffer from the well-known “curse of dimensionality”, especially in the context of large-scale modeling settings.

It should not be left unnoticed that the applicability domain is strongly related to the specifics of the underlying problem being addressed. In turn, the latter should be as clear as possible and be a constant presence during the entire metamodeling process, under penalty of failing to meet the initially planned metamodel’s goals. The modeling performance of the auxiliary model is inevitably defined by such goals on an everlasting effort to balance the trade-off between computational speed and simplicity, and accuracy with regards to the simulation model. Therefore, it proves to be equally important to specify the maximum or acceptable accuracy loss levels within the experimental region. The metamodel performance can be continually assessed via standard metrics such as absolute error, mean absolute error, or root mean square error.

Song et al. (2017) summarize the metamodeling methodology in three major steps, namely, 1) definition of the experimental design, 2) metamodel specification, and 3) metamodel learning/fitting. The first step consists of strategically sampling the input space to generate a data set for the metamodel training. This step is shared with the standard simulation procedures discussed in the previous section. Then, steps 2 and 3, which are usually regarded as a block, are conducted sequentially. While the former involves selecting a family of functional forms for the metamodel, the latter regards the fitting of the selected metamodel to the data set obtained from step 1. For a more in-depth description of the metamodeling approach and unfolding in ten detailed steps suggestions, please refer to (Kleijnen and Sargent, 2000).

When prediction is the intended goal, special attention should be given to the type of functional approximation of the metamodel since the accuracy required for this kind of task is often remarkably high. Given the modeling capabilities discussed earlier in Section 2.1.2, the GPs have been widely used for metamodeling tasks in several fields (Van Beers and Kleijnen, 2004; Kleijnen, 2009). It can also be referred to as Kriging (Matheron, 1963; Chilès and Desassis, 2018), named after his author, Danie Krige, who first developed this methodology in the 60s, within the field of geostatistics.

Initially, GP-based metamodels were mostly applied in deterministic simulations but rapidly moved to random ones Van Beers and Kleijnen (2003), showing that they had great potential in these kinds of applications too. Since then, several similar approaches have arisen (Kleijnen and Van Beers, 2005; Boukouvalas et al., 2009; Ankenman et al., 2010). Jones and Johnson (2009) generally discusses the use of GP as a reliable metamodel for the design and analysis of computer experiments.

Possibly the earliest applications of the metamodeling methodology involved simple queuing simulation systems (Kleijnen, 1975; Friedman and Pressman, 1988) and employed multiple linear regression metamodels during a time where computing power was evidently limited and resources were generally scarce and extraordinarily expensive. In this context, metamodels emerged as practical tools to overcome the computational hindrances posed by the most straightforward simulation systems.

We could be easily tempted to assume that, with the immense technology and computing power, as well as memory capacity, at our disposal nowadays, metamodels would somehow fall into disuse. After all, if their main purpose is to approximate simulation models so that systematic computer experiments are avoided to a certain extent, why would we care to use metamodels nowadays? The answer may reside in the fact that, although the technology has evolved exponentially ever since up to incomparable forms, it did too leverage the demand and the opportunities for modeling increasingly complex simulation models. In a sense, the increased computing power was canceled out by the requirements of more detailed computer models, as they both increased in the same direction and intensity, there explaining why metamodels are still used in current days. This can be regarded as an indication that the relative utility of metamodeling is not a function of the available computational resources per se but on the necessities of developing ever-increasing detailed simulation models.

It is somewhat expected that with more detailed and realistic models, the potential for prohibitive simulation running times is highly likely, which can render the analysis unfeasible, for example, in real-time applications or operational scenarios. In such situations, the use of metamodels should be considered (Kleijnen and Sargent, 2000). However, even if the computing times are not exceptionally high, but the simulation model presents a relatively large number of input variables, simulation metamodel can still play a significant role by providing a way of guiding the input space exploration process in a relatively efficient and parsimonious fashion, in lieu of more ad hoc hit-or-miss one.

Within the transportation literature, the application of simulation metamodels is relatively recent. According to Song et al. (2017), these applications can be essentially divided into two groups, namely, traffic prediction and network optimization. We briefly present some of these works in the following.

In 2013, Ciuffo et al. (2013) developed a methodology that applied a GP-based metamodel to conduct a sensitivity analysis using the mesoscopic traffic simulator AIMSUN as a case study. Using 512 different input combinations, the authors concluded that the metamodel estimated outputs and the real simulation outputs were significantly similar, thereby showing the potential of this type of approach in the field. In (Zhang et al., 2014) the authors developed a Bayesian stochastic Kriging metamodel that simultaneously optimizes travel behavior and dynamic traffic management using, as a case study, the real-world corridors of I-270 and MD355 in the state of Maryland, USA.

In a similar context, Chen et al. (2014) used a GP metamodel to approximate the response surface of a transportation simulation with expensive-to-evaluate objective functions and random noise. The goal was to minimize the network-wide average travel time by implementing optimal toll rates predicted by the metamodel. Similarly, Chen et al. (2015) developed a metamodel-based optimization framework to solve the bilevel Mixed Network Design Problem (MNDP). A mesoscopic Dynamic Traffic Assignment (DTA) simulator was used to evaluate the system response to several network design strategies. The authors showed that the optimal investment could reduce the network average travel time by approximately 18% during the morning period.

Finally, Song et al. (2017) presented a GP-based metamodeling framework that approximates Dynamic Network Loading (DNL) models. The authors show that the tested DNL metamodels attain high accuracy, providing prediction errors below 8%, and superior computational efficiency up to 455 times faster than the traditional DNL approaches.

It is our understanding that although metamodels per se have been used in the transportation field, mostly within simulation-optimization approaches, seldom applications can be found adopting active learning metamodeling strategies. In this sense, we see this work as a means to foster further discussion and applications by recognizing the challenges and looking for new opportunities not only in research but especially in more industry-oriented environments. We expand on several of these points later in Chapter 7.

Chapter 3

Efficient Transport Simulation with Active Learning

"We may at once admit that any inference from the particular to the general must be attended with some degree of uncertainty, but this is not the same as to admit that such inference cannot be absolutely rigorous, for the nature and degree of the uncertainty may itself be capable of rigorous expression."

Ronald A. Fisher, 1890 - 1962

Simulation modeling is a well-known and recurrent approach to studying the performance of urban systems. Taking into account the recent and continuous transformations within increasingly complex and multidimensional cities, the use of simulation tools is, in many cases, the only feasible and reliable approach to analyze such dynamic systems. However, simulation models can become very time-consuming when detailed input-space exploration is needed. To tackle this problem, simulation metamodels are often used to approximate the simulators' results.

In this chapter, we propose an active learning algorithm based on the Gaussian Processes (GP) framework that gathers the most informative simulation data points in batches, according to both their predictive variances and the relative distance between them. This allows us to explore the simulators' input space with fewer data points and in parallel, and thus in a more efficient way, while avoiding computationally expensive simulation runs in the process. We take advantage of the closeness notion encoded into the GP to select batches of points in such a way that they do not belong to the same high variance neighborhoods. In addition, we also suggest two simple and practical user-defined stopping criteria so that the iterative learning procedure can be fully automated.

We illustrate this methodology using three experimental settings. The results show that the proposed methodology is able to improve the exploration efficiency of the simulation input space in comparison with non-restricted batch-mode active learning procedures.

3.1. Introduction

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

To successfully overcome the problem of intractability, simulation approaches are often employed to virtually explore the behavior of such urban systems and to assess their performances. Urban dynamics require theoretical approaches and planning methodologies that are capable of modeling the subjacent spatio-temporal transformation processes from a multidimensional perspective. In fact, urban planning models usually encompass the possibility of predicting future scenarios of urban intervention focused on infrastructure improvement and service promotion. These models are simplified representations of the urban space, embedded into a computer-generated reality, considered as an experimental ground to understand the long-term performance of urban policy decisions and corresponding interventions (Marengo, 2014). Nevertheless, when detailed with enough realism, urban simulation models can become computationally expensive to run due to their overwhelming complexity. Moreover, if the simulator output space proves to have a complex functional structure, we might need to systematically explore the input domain with further detail, which often requires multiple and exhausting simulation experiments, turning the exploration process virtually intractable.

To address the problem of expensive simulation runs, i.e., simulations that require great computational workload and exhibit prohibitive runtimes, simulation metamodels are often used to approximate the simulation results and thus the simulation model itself. Furthermore, in experimental scenarios in which simulation data is computationally expensive to obtain, active learning also emerges as a powerful approach, as it aims to provide high prediction performance with fewer data points. Among many significant machine learning tools, the Gaussian Processes (GP) (Rasmussen and Williams, 2006), a fully Bayesian modeling approach, allow for an intuitive way to develop active learning algorithms by providing the posterior mean and variance, which in turn can be eventually used to search for the most informative data points.

In this chapter, we propose an active learning approach, based on GPs, that gathers the most informative simulation data points in batches, not only according to their variance but also taking into account the relative distance between them. This allows us to explore the simulation input space with fewer training points in a faster, more efficient, and parallel manner while avoiding computationally expensive simulation runs at the same time. Taking advantage of the closeness and similarity notions encoded into the GPs, mainly via the kernel function, our approach selects batches of points that do not simultaneously belong to the same high variance neighborhoods. In addition, we also suggest two simple and practical user-defined stopping criteria so that the iterative learning procedure can be fully automated. We illustrate our methodology using three independent settings within a controlled experimental environment. The first consists of a synthetic data set generated by a known function playing the role of an arbitrary simulation model. Then, we proceed to a one-dimension study of a Demand-Responsive Transportation (DRT) simulator. Finally, we explore the behavior of a road intersection implemented in a micro traffic simulation software, expanding our study to a two-dimensional input case. The obtained results show that the proposed batch-mode approach is able to improve the exploration efficiency of the simulation input space in comparison with non-restricted batch-mode active learning procedures.

The remainder of the chapter is organized as follows. In the next section, we provide a brief review of the main background topics and related literature. The proposed approach is detailed in Section 3.3, followed by the presentation of studied simulation data, experiments, and discussion (see Section 3.4). Then we proceed to the validation of the obtained results in Section 3.5. Finally, we end this chapter with some conclusions and possible lines of future work.

3.2. Background

Active learning is a special case of supervised machine learning consisting of an iterative sampling scheme that allows the algorithm to choose the data points from which it learns and an oracle, i.e., an instance label provider. It is particularly useful under scenarios where labeled data is expensive to obtain. Thus, the general idea of this learning paradigm is to actively select the most informative data points, as few as possible, in order to simultaneously boost the model training efficiency and its prediction performance (Settles, 2010). When the active learning paradigm was first proposed, the oracle was typically represented by a human annotator. Nowadays, however, due to the development of technology, the oracle's role can be taken by an algorithm, a sensor, a simulator, etc. Most importantly is that the oracle is able to provide labeled instances from the ground truth underlying function describing the process of interest.

Depending on how the unlabeled data is presented to the oracle, the active learning algorithms can be divided into two classes, namely, stream-based and pool-based. Whereas in the latter, the entire unlabeled data set is available for querying, in the former, each data point is presented individually or in successive blocks (Wang and Zhai, 2016). In addition, because of its intrinsic iterative nature, active learning procedures must be stopped at a certain time. Although there is vast research under the active learning paradigm, not many approaches suggest a stopping criteria (Vlachos, 2008; Wang et al., 2014), and, according to Wang and Zhai (2016), there is no best stopping rule that is suitable across all applications.

In most of the proposed active learning algorithms, the queries are presented sequentially, i.e., one at a time. This strategy can become quite inefficient for some heavy learning tasks. One way to address this issue is to select data points in batches in order to speed up the learning process by parallelizing it. However, to avoid redundancy within each batch, it should simultaneously account for diversity and informativeness among the selected data points (Wang and Zhai, 2016). Several batch-mode schemes have addressed this challenge, most of them within classification problems (Brinker, 2003; Hoi et al., 2006; Xu et al., 2007; Guo and Schuurmans, 2008).

Although active learning has been applied in many different fields, we are particularly interested in those of simulation metamodels (Friedman, 2012) applications. Their main purpose is to serve as surrogates for simulation models so that expensive simulations can be avoided. These models are essentially simple functions that approximate the true and more complex unknown function inherently defined by the simulation model itself (Kleijnen and Van Beers, 2004). They are fitted to the input-output data generated by computer experiments and then can be used for prediction purposes, among others (Kleijnen and Sargent, 2000). Hence, these simulation metamodels provide a practical framework to explore the behavior of complex and time-consuming simulation experiments in a rather less expensive way.

In our case, we assume that the simulation model is perfectly validated and calibrated with respect to the problem of interest. The metamodel, which should be a valid approximation of the simulation model, is then used to effortlessly explore its inner structure and, consequently, to provide yet another analysis tool for the problem under study. The Gaussian Processes (GP) framework has been widely used for simulation metamodeling (Boukouvalas, 2010; Chen et al., 2011; Conti and O'Hagan, 2010; Kleijnen, 2009), and other works combining GP metamodeling with active learning strategies have also been conducted (Christen and Sansó, 2011; Pfingsten, 2006). Its Bayesian formalism provides an easy way to develop algorithms that actively learn with uncertainty. Other machine learning techniques, such as Artificial Neural Networks (ANN), have also been employed (Cohn, 1996; Fonseca et al., 2003).

Within the transportation literature, the application of simulation metamodels is relatively recent. Only a handful of works is currently available, and, with respect to their application domain, they can be essentially divided into two groups, namely, traffic prediction and network optimization (Song et al., 2017). In 2013, Ciuffo et al. (2013) developed a methodology that applies a GP-based metamodel to conduct a sensitivity analysis using the mesoscopic traffic simulator AIMSUN as a case study. Using 512 different input combinations, the authors concluded that the metamodel estimated outputs and the real simulation outputs were significantly similar, thereby showing the strength and parsimony of their methodology. In (Zhang et al., 2014) the authors developed a Bayesian stochastic Kriging metamodel that simultaneously optimizes travel behavior and dynamic traffic management using, as a case study, the real-world corridors of I-270 and MD355 in the state of Maryland, USA. Using a similar case study, Chen et al. (2014) used a GP metamodel to approximate the response surface of a transportation simulation with expensive-to-evaluate objective functions and random noise. The goal was to minimize the network-wide average travel time by implementing the optimal toll rates predicted by the metamodel. Similarly, in (Chen et al., 2015) a metamodel-based optimization framework was developed to solve the

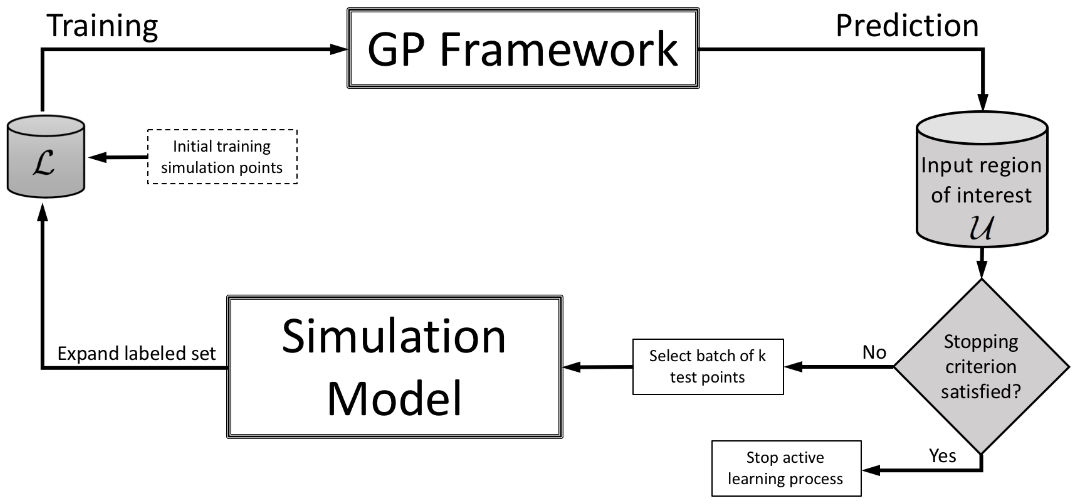

Training
GP Framework
Prediction
$\mathcal{L}$
Initial training simulation points
Input region of interest $\mathcal{U}$
Expand labeled set
Simulation Model
Select batch of k test points
No
Stopping criterion satisfied?
Yes
Stop active learning process

Figure 3.1: Pool-based batch-mode active-learning scheme with a simulation model serving as the oracle.

bilevel Mixed Network Design Problem (MNDP).

A mesoscopic Dynamic Traffic Assignment (DTA) simulator, DTALite, was used to evaluate the system response to several network design strategies. The authors showed that the optimal investment could reduce the network average travel time by approximately 18% during the morning period. Finally, and very recently, Song et al. (2017) presented a GP-based metamodeling framework that approximates Dynamic Network Loading (DNL) models. The authors show that the tested DNL metamodels attain high accuracy, providing prediction errors below 8%, and superior computational efficiency, up to 455 times faster than the traditional DNL approaches. Although these and other works constitute important applications of simulation metamodeling in transport problems, the research in this area is still scarce.

3.3. Approach

In this work, we adopt a pool-based batch-mode active learning approach in which a simulator plays the role of the oracle, as depicted in Figure 3.1. The machine learning model, used as a metamodel, is a GP, and the unlabeled pool, $\mathcal{U}$, is the input space where we want to explore the simulation behavior. The pool of labeled instances, $\mathcal{L}$, is formed by the results of the simulation runs already performed. From a regression perspective, we want to establish a functional relationship between the simulation inputs, $\mathbf{x} \in \mathbb{R}^D$, where D is the number of inputs, and its output, $y \in \mathbb{R}$, by assuming a GP prior over this relation, $y = f(\mathbf{x})$. Then, taking advantage of the Bayesian formalism from which the GP framework derives, we aim to infer the conditional distribution of the output given a set of unlabeled inputs. By doing so, we are not only bypassing the simulation's computational workload but also providing a way to effortlessly approximate and therefore analyze the simulator's behavioral structure. Notice that the simulation model is treated as a black box from which we aim to get better insights in terms of its functional behavior while avoiding as many simulation runs as possible.

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

3.3.1. Gaussian Processes

A GP (Rasmussen and Williams, 2006) is a stochastic process completely characterized by a mean and a covariance function, respectively denoted as $m_f(\mathbf{x})$ and $k_f(\mathbf{x}, \mathbf{x})$, with $\mathbf{x}$ and $\mathbf{x}'$ being two input data points and simply denoted by $\mathcal{GP}(m_f(\mathbf{x}), k_f(\mathbf{x}, \mathbf{x}'))$ where $m_f(\mathbf{x}) = \mathbb{E}[f(\mathbf{x})]$ and $k_f(\mathbf{x}, \mathbf{x}') = \mathbb{E}[(f(\mathbf{x}) - m_f(\mathbf{x}))(f(\mathbf{x}') - m_f(\mathbf{x}'))]$. More formally, the GP framework assumes a prior over functions, i.e., $y = f(\mathbf{x}) + \epsilon$, where $\epsilon \sim \mathcal{N}(0, \sigma^2)$ and $f(\mathbf{x}) \sim \mathcal{GP}(m_f(\mathbf{x}), k_f(\mathbf{x}, \mathbf{x}'))$. For simplicity, it is common practice to fix $m_f(\mathbf{x}) = 0$. Thus, the prior over the latent function is given by $p(\mathbf{f}|\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n) = \mathcal{N}(0, K_f)$ where $\mathbf{f} = [f_1, f_2, \ldots, f_n]^\top$, $f_i \triangleq f(\mathbf{x}_i)$ and K_f is the covariance matrix, with its elements given by $[K_f]_{ij} = k_f(\mathbf{x}_i, \mathbf{x}_j)$.

Most of the covariance functions, or kernels, have several free parameters which can be optimized to fit the training data by maximizing the marginal likelihood. After these parameters are obtained, the conditional distribution at a new test point $\mathbf{x}_*$ is given by $f_*|X, \mathbf{y}, \mathbf{x}_* \sim \mathcal{N}(k_{f*}^\top [K_y]^{-1}\mathbf{y}, k_{f**} - k_{f*}^\top [K_y]^{-1} k_{f*})$, where $k_{f*} = k_f(X, \mathbf{x}_*)$, $k_{f**} = k_f(\mathbf{x}_*, \mathbf{x}_*)$, $K_y = K_f + \sigma^2 I$, and $\mathbf{y}$ is the vector of the target values. Thus, we can naturally use the predicted Gaussian distribution at any given test point to guide the active learning process and therefore learn with its associated uncertainty.

3.3.2. Active Learning Procedure

The algorithm presented in Figure 3.2 refers to the general batch-mode active learning procedure used in this work, serving as the base for other algorithms forwardly presented and tested. Here, $\mathcal{L}$ represents the set of labeled training points, whereas $\mathcal{U}$

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

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

In each iteration, a new GP model is fitted to $\mathcal{L}$. Its hyper-parameters are obtained by maximizing the log-likelihood function conditional on this training set in a Leave-One-Out Cross-Validation (LOOCV) scheme. This constitutes a simple scheme to minimize potential overfitting problems during the training stage. Afterward, the trained GP is used to predict the simulation output values (labels) associated with the unlabeled points in $\mathcal{U}$, therefore avoiding many simulation runs. Then, several testing points are selected according to the approach described in Figure 3.2, their respective true labels are obtained via the oracle, i.e., the simulator, and finally $\mathcal{L}$ is expanded. This iterative process is repeated until the chosen stopping criterion is satisfied.

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

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

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

3.4. Experiments

In this section, we test the proposed methodology using three independent experimental settings. The first consists of a synthetic data set generated by a known

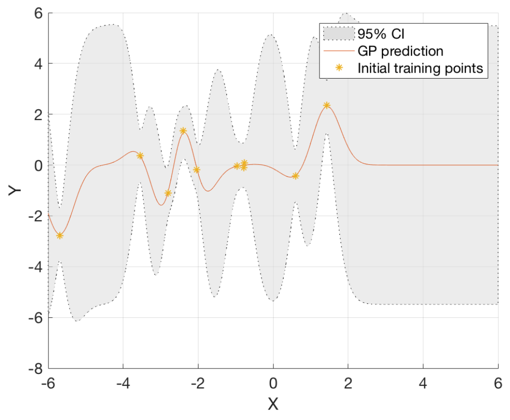
```
Input: $\alpha_1, \alpha_2, \beta \in [0,1], k \in \mathbb{N}, \mathcal{L}, \mathcal{U}$.
While $\frac{ITV-CTV}{ITV} < \alpha_1$ or $\frac{AVT_r}{AVT_s} < \alpha_2$ do
1: Train a GP with $\mathcal{L}$ and obtain the predicted labels for each point in $\mathcal{U}$, $k_{f*}^{\top}[K_y]^{-1}\mathbf{y}$,
   and their corresponding variances, $k_{f**} - k_{f*}^{\top}[K_y]^{-1}k_{f*}$.
2: Determine the top $k$ highest variance test points, $top_k$, so that the minimum dis-
   tance between them is $\beta \times 100\%$ of the diameter of $\mathcal{U}$.
3: Obtain the true labels for $top_k$, via simulator, and define $\mathcal{L}_+$ as the new labeled
   set.
4: Set $\mathcal{L} = \mathcal{L} \cup \mathcal{L}_+$.
5: Update stopping criterion.
end
```

Figure 3.2: General batch-mode active learning approach with spatial restriction.

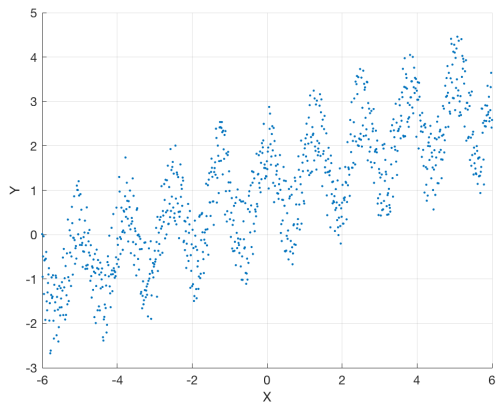
function playing the role of an arbitrary simulation model. Then, we proceed to a one-dimension study of a Demand-Responsive Transportation (DRT) simulator. Finally, we move to a two-dimensional study of a simple road intersection implemented in a free and open-source traffic micro-simulation software. We use the GP implementation from (Rasmussen and Williams, 2006) and select the Squared Exponential function as the GP kernel. Our study focuses on the analysis of the number of iterations required for each algorithm to satisfy the stopping criteria, rather than on the observation of the corresponding CPU times, despite its great importance from a practical point of view. This allows us to conduct a more meaningful and hardware-independent performance analysis, making it possible to compare the results originated from different simulation experiments with different hardware specifications.

3.4.1. Toy data set

For the first set of experiments, we used a toy data set generated by $f(x) = cos(5x) + \frac{1}{2}x + 2\,U(0,1)$, which plays the role of oracle, where $U(a,b)$ denotes the Uniform distribution in the interval $[a,b]$. Figure 3.3 shows the initial GP learning state and the ground truth data following f. The initial labeled pool, $\mathcal{L}$, is comprised of 10 randomly generated training points, $\mathcal{U}$ is formed by 10000 unlabeled test points uniformly spaced in $[-6,6]$. Figure 3.4 and 3.5 present the results.



(a)

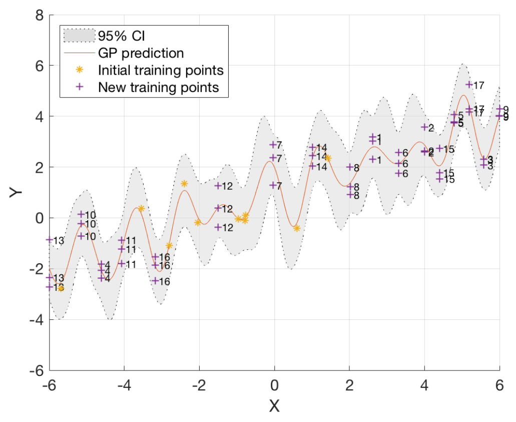


(b)

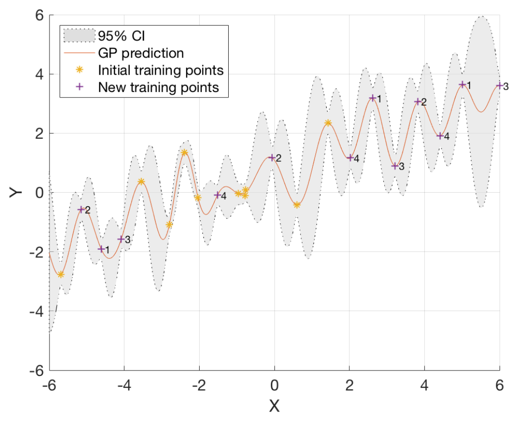
Figure 3.3: (a) Initial learning state for the toy data set, where the first 10 training points were randomly scattered in the input domain $[-6, 6]$. This corresponds to iteration 0, which is shared by the four algorithms in study. (b) Ground-truth data.

Comparing Algorithms 1 and 2, the superiority of the latter is clear from an efficiency point-of-view (see Figure 3.4). For the same criterion threshold ($\alpha_1 = 0.6$), Algorithm 2 is over four times faster than Algorithm 1 due to the space restriction induced by $\beta = 0.2$. This means that the testing points constituting the batches were not selected from the same high variance neighborhoods, scattering the input exploration process and thus making it more efficient. Moreover, it is important to note the number of iterations alone does not directly measure the real involved computational workload, which is intrinsically related to the simulation model. As we adopted a batch-mode active learning scheme, we should additionally take into account the size of the batch, k. Therefore, whereas Algorithm 1 requested the oracle $17 \times 3 = 51$ times, in Algorithm 2, this number decreased to $4 \times 3 = 12$.

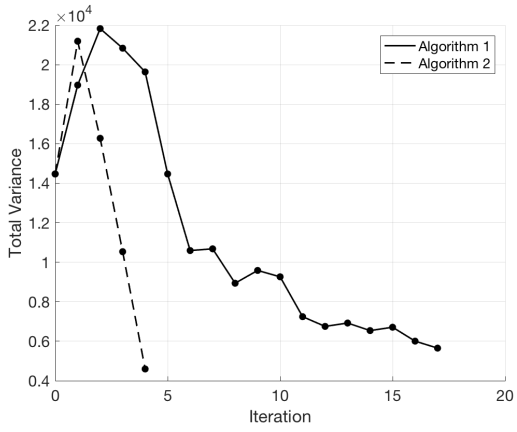
For Algorithms 3 and 4, a similar scenario has occurred. Using the same stopping rule, based on Criterion B with $\alpha_2 = 0.4$, Algorithm 4 required three fewer iterations than Algorithm 3. This represents a total difference of $3 \times 3 = 9$ simulation requests. However, note that, as previously mentioned in Section 3.3.2, Criterion B may be harder to satisfy, which explains the lesser performance in comparison in both Algorithms 1 and 2.



(a)



(b)

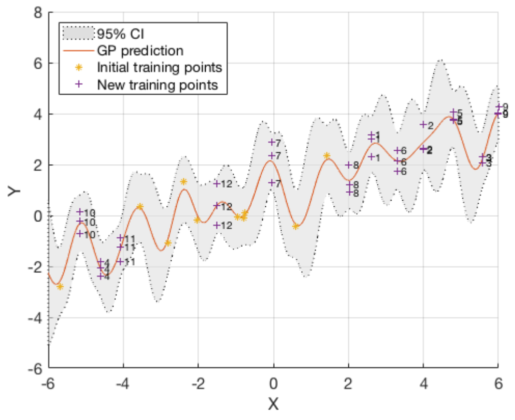


(c)

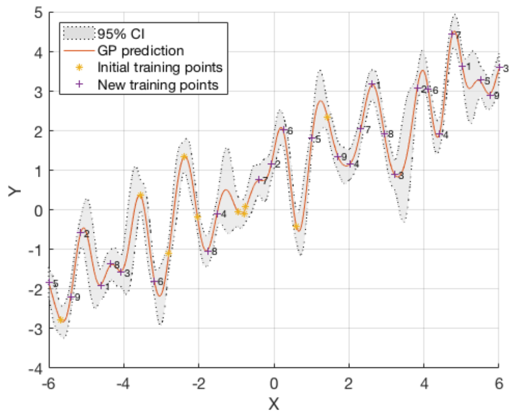
Figure 3.4: Final results for the toy data set using (a) Algorithm 1 with user-defined parameters $\alpha_1 = 0.6$, $\beta = 0$ and $k = 3$, and (b) Algorithm 2 with $\alpha_1 = 0.6$, $\beta = 0.2$ and $k = 3$. The active points are labeled with the number of the iteration in which they were selected. Panel (c) compares the total variance reduction between both algorithms, displaying it as a function of the iteration number.

3.4.2. DRT simulator

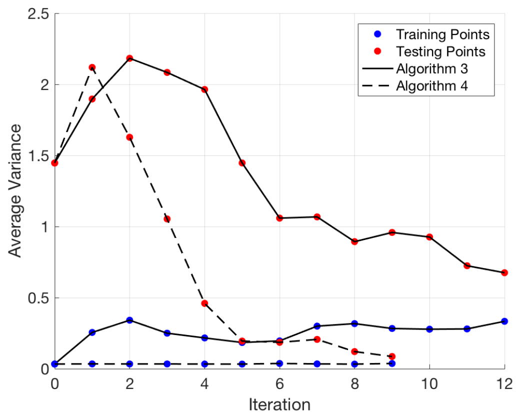
Demand Responsive Transportation (DRT) systems are a kind of hybrid transportation approach between the taxi and bus solutions that address the problems that emerge from the use of fixed routes and schedules, typically found in regular public road transportation. From the transport operators' point-of-view,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



(a)



(b)



(c)

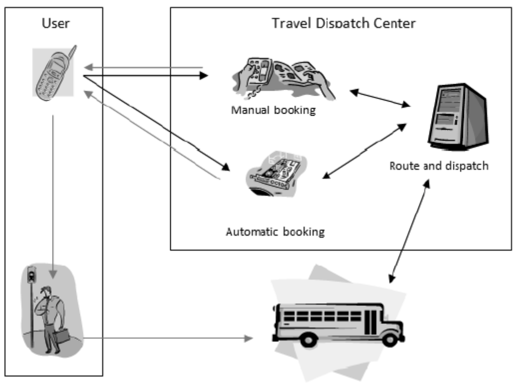
Figure 3.5: Final results for the toy data set using (a) Algorithm 3 with user-defined parameters $\alpha_2 = 0.4$, $\beta = 0$ and $k = 3$, and (b) Algorithm 4 with $\alpha_2 = 0.4$, $\beta = 0.2$ and $k = 3$. The active points are labeled with the number of the iteration in which they were selected. Panel (c) compares the training and test average variance between both algorithms, displaying them as a function of the iteration number.

tion solutions that are able to adapt, in real-time, their routes and frequencies to match the actually observed demand.

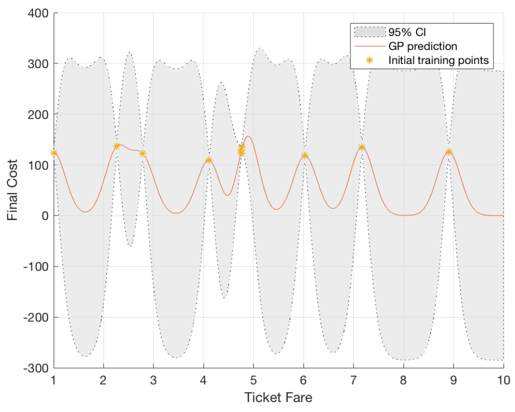
Service design is critical for the success of DRT systems, so decision-makers need to understand well how the different ways of operating the service affect its performance. The flexibility of DRTs may cause organizational problems such as, a) the number and type of requests may involve an exceedingly high number of vehicles, b) very sparse requests that are hard to combine efficiently, or c) the quality of the service in terms of pickup/delivery time and travel duration might not be guaranteed with the available resources or when unpredictable events occur. These effects are often studied through simulation, whose purpose is to obtain a better understanding of the behavior of a system under a given set of input conditions, even with uncertain events. The performance of these systems can be determined by observing what happens on the network, during simulation, for different input conditions. However, when dealing with real-world events (especially with a high degree of dynamism) and extremely complex road networks and demand, simulation models can become very time-consuming. An illustration of a standard DRT system is presented in Figure 3.6(a).

We now consider the problem of exploring the outcome of the DRT simulation model developed by Gomes et al. (2014). This simulation system integrates four submodels, covering the service area, trip requests (demand), vehicles, and real-time events. It has 22 inputs/parameters and, among a few outputs that measure the system's overall performance, we focus on the DRT's total operating cost. For a given demand structure, different input combinations will lead to different costs and service quality levels. We loaded the simulator with a real road network structure, within the metropolitan region of Porto (Portugal), with symbolic parameters values. For the sake of illustration, we then considered the Ticket Price as the input domain for which we aim to explore the outcome, Total Cost, of the simulator, maintaining the remaining inputs unchanged. From previous experiments, we concluded that, for our particular application, the simulation running times do not vary significantly from each other. Therefore, it does make sense to focus only on the number of iterations to assess the performances of the studied algorithms.

Similar to the previous experimental setting, we decided to explore the input domain contained in the interval $[1, 10]$ using 10000 uniformly spaced test points. This test set corresponds to $\mathcal{U}$, whereas $\mathcal{L}$ is constituted by ten random simulation data points within the same interval, as presented in Figure 3.6(b). The results for this simulation data are generally aligned with the results obtained for the toy set experiment. Again, and as depicted in Figure 3.7, Algorithm 2 is more than two times faster than Algorithm 1. For the same level of variance reduction, induced by $\alpha_1 = 0.60$ (about 40% of the initial total variance), the former requested the simulator $5 \times 3 = 15$ times, against $2 \times 3 = 6$ requests in the latter. It is obvious that the fewer runs the simulator executes, the better. Therefore, for the same stopping criteria and threshold, Algorithm 2 proved to be more efficient as it was able to scatter the learning points forming the batch along



(a)

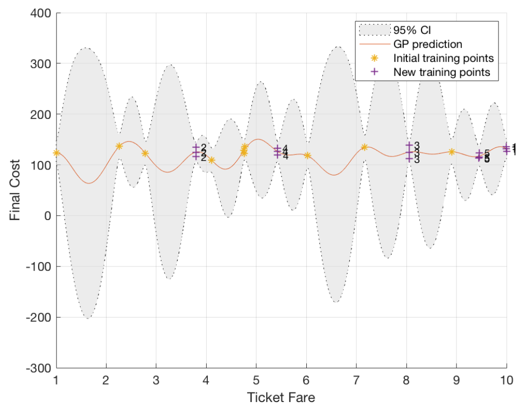


(b)

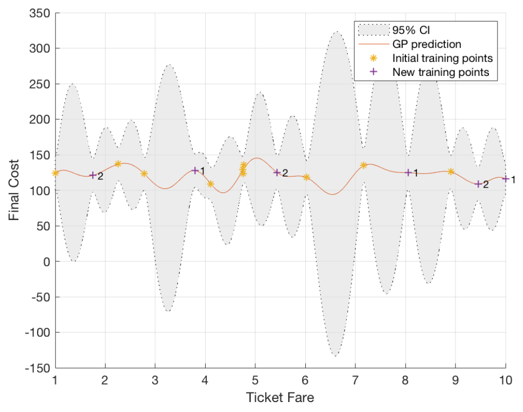
Figure 3.6: (a) Pictorial representation of a typical DRT system. Source (Gomes, 2012). (b) Initial learning state for the DRT simulation data set, where the first 10 training points were randomly scattered in the input domain [1, 10]. This corresponds to iteration 0, which is shared by the four algorithms in study.

with different high variance neighborhoods.

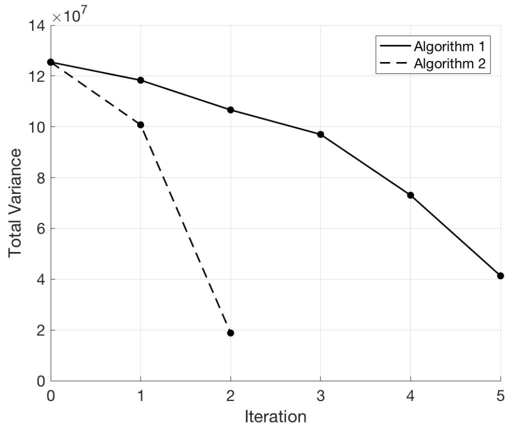
For Algorithms 3 and 4, we obtained an interesting result which highlights our concerns presented in Section 3.3.2 regarding Criterion B. In Figure 3.6 we can observe that for the first approximation (Iteration 0), the GP model assigned very little variance to the training points, meaning that, in the context of Criterion B, $AVTr \approx 0$. As the process iteratively evolves, i.e., as more simulation data points are added to the expanding data set, $AVTr$ seems to remain close to zero. On the other hand, $AVTs$ clearly shows a decreasing behavior towards zero. In several preliminary experiments, which we do not present due to space restrictions, we noticed that setting, for example, $\alpha_2 = 0.6$, as we did for α_1, was too ambitious for both Algorithms 3 and 4 to yield competitive performance. This means that these algorithms were taking too many iterations to converge to be fairly comparable to Algorithms 1 and 2. Therefore, after fixing $\alpha_2 = 0.1$, which is equivalent to say that the stopping criterion is satisfied when $AVTr$ is approximately 10% of $AVTs$, we concluded that both Algorithms 3 and 4 converged in reasonable and comparable running times. Despite these initial configuration problems, these algorithms attained similar results to those of Algorithms 1 and 2. It is worthwhile to mention that, once again, the restriction applied during the formation of the batches was a decisive factor in the reduction of the number of



(a)



(b)



(c)

Figure 3.7: Final results for the DRT simulation data using (a) Algorithm 1 with user-defined parameters $\alpha_1 = 0.6$, $\beta = 0$ and $k = 3$, and (b) Algorithm 2 with $\alpha_1 = 0.6$, $\beta = 0.2$ and $k = 3$. Each active point is labeled with the number of the iteration in which it was selected. Panel (c) compares the total variance reduction between both algorithms, displaying it as a function of the iteration number.

iterations.

3.4.3. Traffic simulator

In this section, we move to a microscopic traffic simulation example by exploring a road intersection implemented with the Simulation of Urban Mobility (SUMO) (Krajzewicz et al., 2012), in which the traffic flows in three directions only, North-South (NS), West-East (WE) and East-West (EW). The vertical axis is dedicated to important vehicles, whereas in the horizontal axis, we only have light passenger car traffic. Moreover, the model is designed to prioritize the NS traffic over the remaining flows.

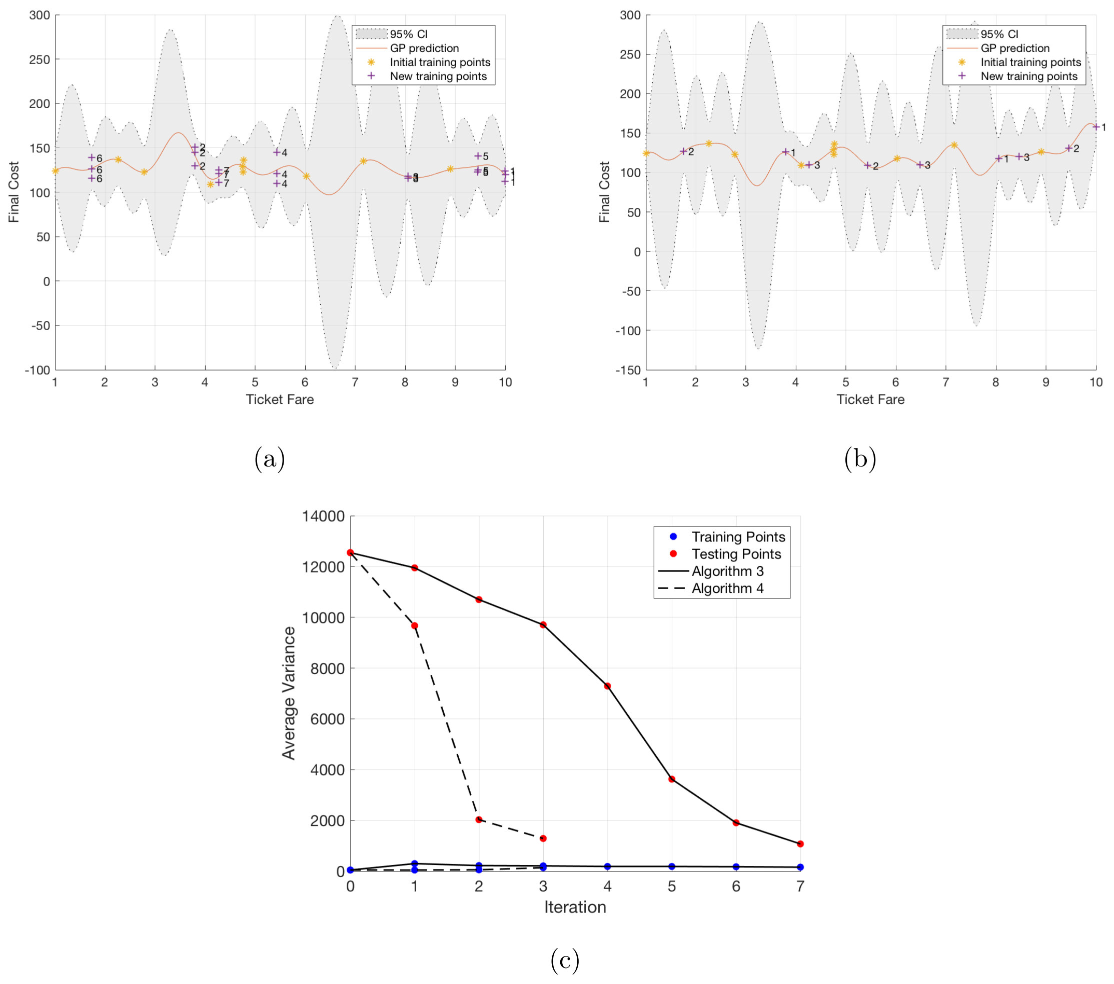


Figure 3.8: Final results for the DRT simulation data using (a) Algorithm 3 with user-defined parameters $\alpha_2 = 0.1$, $\beta = 0$ and $k = 3$, and (b) Algorithm 4 with $\alpha_2 = 0.1$, $\beta = 0.2$ and $k = 3$. Each active point is labeled with the number of the iteration in which it was selected. Panel (c) compares the training and test average variance between both algorithms, displaying them as a function of the iteration number.

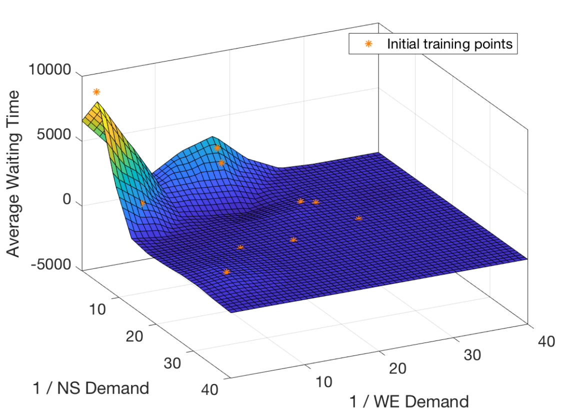
Therefore, it is expected that if this flow increases, the horizontal traffic flows will potentially form more and longer queues, consequently increasing the overall total waiting time.

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

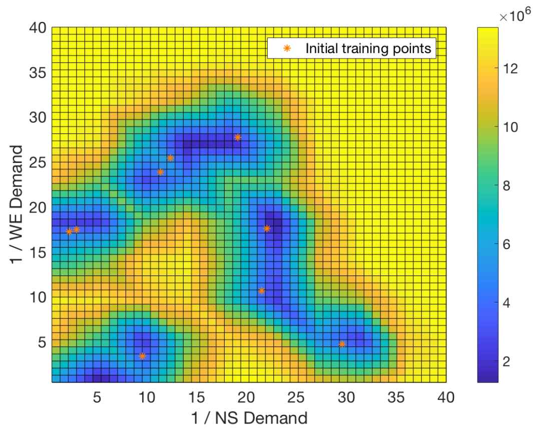
The simulated example encompasses three input parameters that have a direct influence on the intersection performance, namely, the NS, WE, and EW demands, each of which is associated with different Binomial parameters. Moreover, it is assumed that the three different traffic flows are mutually independent. To assess the performance of the simulated road networks, SUMO has a large number of different output measures. Raw vehicle positions, trip and route information, and simulation state statistics are just a few examples of possible outputs. For the sake of illustration of our methodology, we decided to focus on the total waiting time spent by all the vehicles crossing the intersection as our aggregated traffic performance measure. Our objective is to use the proposed active learning scheme to explore the simulation input space and to evaluate how it affects the total waiting time. Therefore, following a similar experimental design of the one-dimensional analyses presented in Sections 3.4.1 and 3.4.2, we now extend our study to a two-dimensional case where the traffic demands from NS and WE operational axes are considered as inputs, and the expected vehicular waiting time is our output performance of interest.

The new input region of interest ($\mathcal{U}$) is defined by the square $[0, 40] \times [0, 40]$, from which 10 random training points were selected, corresponding to the initial set of simulation runs ($\mathcal{L}$), as depicted in Figure 3.9(a). On the other hand, Figure 3.9(b) shows the variance across the entire test region, whereas panel (c) provides a graphical representation of the traffic intersection in question. As expected, the variance near the training points is lower than the variance associated with the test points. Starting from this initial learning stage, our approach is designed to actively search for the top k highest variance neighborhoods (yellow tones regions) that are not mutually within a radius of $\beta \times 100\%$ of the diameter of the input region. In any case, in this first approximation, we can already observe that the values of the average waiting time (z-axis) tend to increase when both the NS and WE demands increase, matching our initial guess regarding the simulation output behavior.

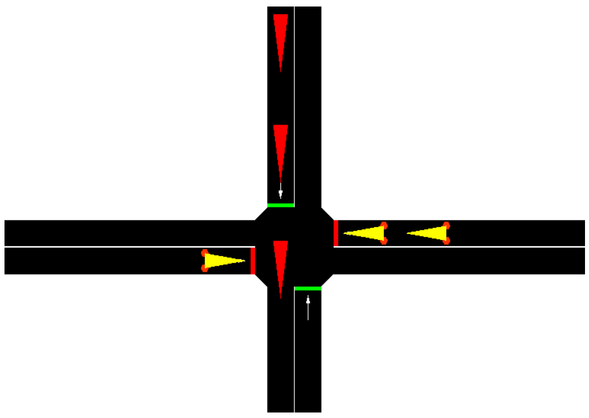
Figure 3.10 and 3.11 present the final results, showing fairly identical GP approximations across the four algorithms. We observe that Algorithm 1 took 11 iterations to achieve a total variance reduction of 95%, against seven iterations from Algorithm 2 (see Figure 3.10(c)). Both are based on Criterion A and on the same stopping threshold. However, due to the proposed space restriction (imposed by $\beta = 0.3$),



(a)



(b)



(c)

Figure 3.9: (a) Initial learning state for the traffic simulation data, where the first 10 training points were randomly scattered in the input space $[0, 40] \times [0, 40]$. This corresponds to iteration 0, which is shared by the four algorithms in study. (b) Variance behavior across the input region. (c) Graphical representation of the intersection under study.

the latter presents a more efficient performance than the former, with a difference of $(11 - 7) \times 3 = 12$ simulation runs. Finally, we can see from Figure 3.11(c) that Algorithm 4 took three iterations to stop, whereas Algorithm 3, whose batch formation is not restricted by β, required six to satisfy Criterion B with $\alpha_2 = 0.2$. Although these differences in the iteration numbers may seem to be of little significance for the current academic example, they can prove to be quite relevant in real-world and thus computationally heavy simulation models, which can take up to several days to finish.

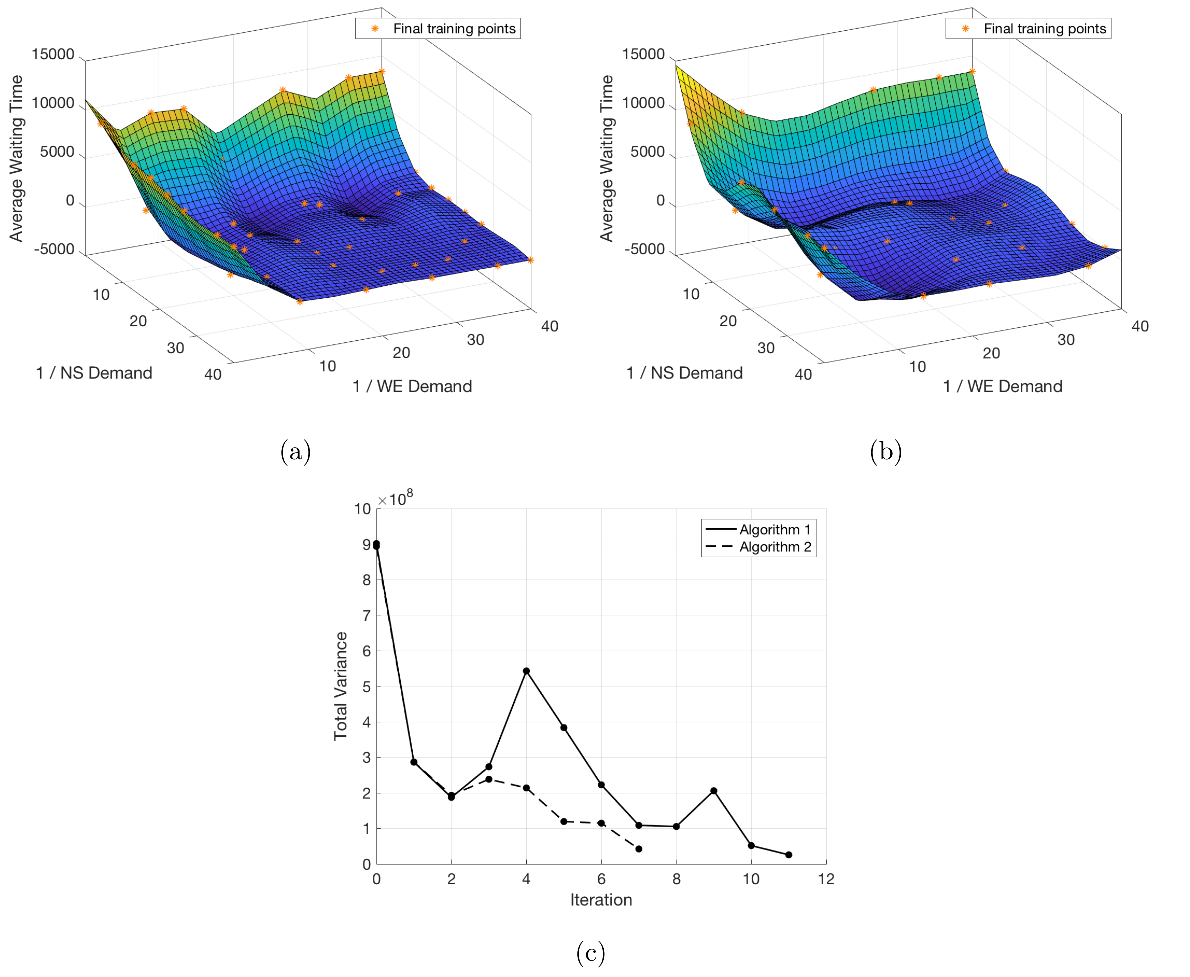
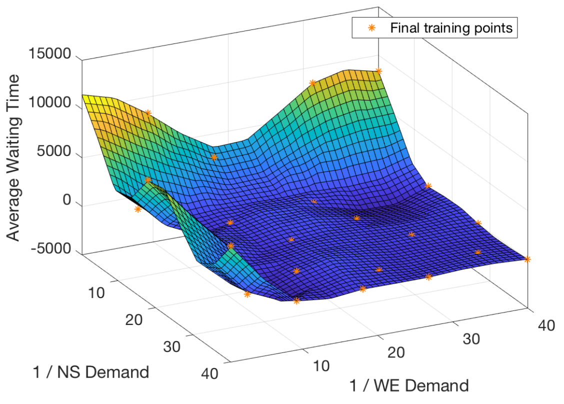


Figure 3.10: Final results for the traffic simulation data using (a) Algorithm 1 with user-defined parameters $\alpha_1 = 0.95$, $\beta = 0$ and $k = 3$, and (b) Algorithm 2 with $\alpha_1 = 0.95$, $\beta = 0.3$ and $k = 3$. Panel (c) compares the training and test average variance between both algorithms, displaying them as a function of the iteration number.

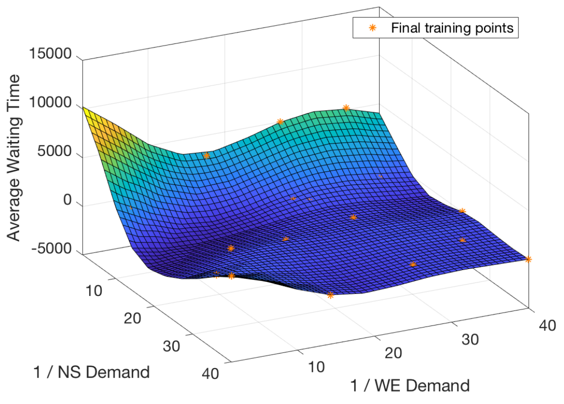
3.5. Validation

We now conduct a series of computer runs to not only compare the algorithms' results but also to assess the quality of the obtained GP approximations with respect to the underlying simulation functions under study. These results are summarized in Table 3.1.

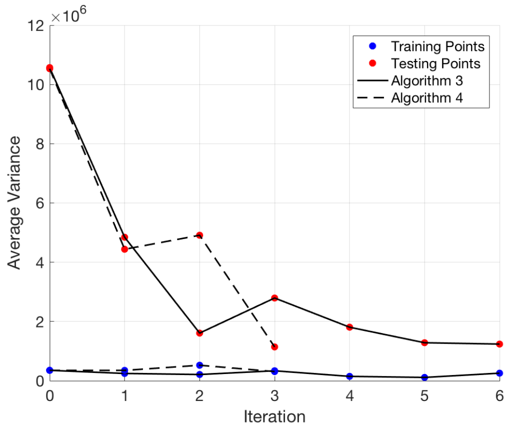
To evaluate the performance of the studied algorithms, we used the Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), Root Relative Squared Error (RRSE), and Pearson's linear correlation coefficient (Cor.). Within each experimental setting, these metrics were computed by comparing the predicted values provided by



(a)



(b)



(c)

Figure 3.11: Final results for the traffic simulation data using (a) Algorithm 3 with user-defined parameters $\alpha_2 = 0.2$, $\beta = 0$ and $k = 3$, and (b) Algorithm 4 with $\alpha_2 = 0.2$, $\beta = 0.2$ and $k = 3$. Panel (c) compares the training and test average variance between both algorithms, displaying them as a function of the iteration number.

the GP (in the last iteration of the algorithms) and the true simulation output values over a set of 20 test points scattered along the different studied input regions. We also obtained the average iteration difference between Algorithms 1 and 2 (Criterion A), and Algorithms 3 and 4 (Criterion B), respectively. Additionally, in order to evaluate the statistical significance of this difference, we conducted a paired sample t-test. Each computer run is determined by a different set of 10 random initial training points. The values for the parameters α_1, α_2, and β were maintained the same as in the previous section.

Overall, the results show that the GP was able to provide rather good approximations for the underlying simulation functions associated with the three experimental

Table 3.1: Average results obtained from 30 random computer runs using 20 test points for the different experimental settings and algorithms.

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

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

3.6. Conclusion and Future Work

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

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

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

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

Chapter 4

Active Learning Metamodeling for Policy Analysis

"How can it be that mathematics, being after all a product of human thought which is independent of experience, is so admirably appropriate to the objects of reality? Is human reason, then, without experience, merely by taking thought, able to fathom the properties of real things?"

Albert Einstein, 1879 - 1955

Simulation approaches constitute a well-established tool to model, understand, and predict the behavior of transportation systems and ultimately to assess the performance of transportation policies. Due to their multidimensional nature, such systems are not often approachable through conventional analytic methods, making simulation modeling the only reliable tool of study. Nevertheless, simulation models can turn out to be computationally expensive when embedded with enough detail. An immediate answer to this shortcoming is the use of simulation metamodels that are designed to approximate the simulators' results.

In this chapter, we propose a metamodeling approach based on active learning that seeks to improve the exploration process of the simulation input space and the associated output behavior. A Gaussian Process (GP) is used as a metamodel to approximate the function inherently defined by the simulation model itself. The GPs can nicely handle the uncertainty associated with their predictions, which eventually can be improved with active learning through simulation runs. This property provides a practical and efficient way to analyze the simulator's behavior and, therefore, to assess the performance of policies regarding the underlying real-world systems and services.

We illustrate the proposed methodology using an Emergency Medical Service (EMS) simulator. Two outputs are analyzed and compared, namely, the survival rate and response time averages. The medical emergency response time recommendation of eight minutes is explored as well as its relation with the survival rate. The results show that this methodology can identify regions in the simulation input space that might affect the performance and success of medical policies with regard to emergency vehicle dispatching services.

4.1. Introduction

Real-world transportation systems are characterized by their overwhelming complexity, not only due to the multitude of variables involved but also to their corresponding interrelationships. In order to model, understand, and then predict the behavior of such systems and to assess their performances, simulation modeling is often the only reliable tool available. Especially due to their high dynamism and dimensionality, as well as their intrinsic stochastic nature, these systems can hardly be described or studied analytically (Law, 2015).

Simulation models are virtual representations of the reality, often in a sufficiently simplified form, that are considered as experimental and virtual environments to test different system designs, and therefore to understand the impact of specific policies and interventions (Marengo, 2014). Simulation tools, due to their exploratory nature, prove to be highly advantageous for policy analysis (Bankes, 1992, 1993). However, simulation models can become computationally expensive to run, exhibiting prohibitive runtimes along with significant workloads. To address this shortcoming, simulation metamodels are usually employed as they are designed to approximate the simulation model itself and thus the function inherently defined by it.

Along with the consideration of simulation metamodels to decrease the burden of conducting expensive computer experiments, a machine learning paradigm called active learning (Settles, 2010), can also be taken into account. Active learning is particularly useful in situations in which data is difficult to obtain, as it aims to enhance the models' predictive performance with fewer training data points. Therefore, both active learning and simulation metamodeling approaches can be jointly used, on the one hand, to minimize the need for simulation runs, and on the other hand, to achieve and maintain a reasonably good approximation of the simulation model. The Gaussian Process (GP) framework (Rasmussen and Williams, 2006), also known as Kriging, is a popular modeling tool widely applied in numerous research and application fields. Due to its Bayesian formalism and highly non-linear properties, it constitutes an excellent option for designing active learning strategies within simulation metamodeling settings.

This work presents an active learning metamodeling methodology to address the problem of policy analysis within the context of computationally expensive simulation models. A GP is considered to approximate the simulator's behavior and then used to explore the simulation input space. The fitness of the GP is then iteratively improved with active learning via simulation runs by decreasing the associated variance of the given predictions over the simulation input region under study. This strategy provides an alternative way to perform policy analysis while avoiding, at the same time, a potentially large number of simulation runs.

The presented methodology is tested using an Emergency Medical Service (EMS) simulator. Two simulation outputs are studied, namely, the average survival rate and the average response time. Then, motivated by Pons and Markovchick (2002), the medical emergency response time recommendation of 8 minutes (480s) is analyzed, as well as its relation with the survival rate. The results show that this methodology is able to identify regions within the simulation input space that directly influence the successful application of medical policies with regard to emergency vehicle dispatching services.

4.2. Literature Review

The development and application of simulation metamodels (Kleijnen, 1975, 1979, 1987; Friedman and Pressman, 1988; Friedman, 2012) can be traced back to the 70's (Barton, 1998). Their main purpose is to serve as parsimonious approximations for simulation models so that expensive simulation runs can be avoided. Specific features such as mathematical simplicity, speed, and interpretability are usually attributed to metamodels. Consequently, the use of metamodels within simulation analysis provides an additional level of understanding of the underlying system, as well as of the relationships between the system input and output variables, while maintaining a computationally simple and economical approach to the problem (Friedman and Pressman, 1988). Simulation metamodels are often described by computationally fast and easy-to-implement functions that approximate the true but unknown function intrinsically defined by the simulation model itself. Commonly, many of these inputs are shared with those of the simulation model, although it is not entirely necessary, as Kleijnen and Sargent (2000) points out.

The earliest applications of metamodeling involved simple queuing simulation systems (Kleijnen, 1975) and used multiple linear regression metamodels. At the time, as the computational resources were low and scarce, when compared to the current days, the use of metamodels emerged as a practical tool to overcome the difficulties posed by even the most straightforward simulation systems. Although technology has evolved ever since, essentially providing more computational power, it also leveraged the demand and the opportunities for modeling increasingly complex systems. Both computational power and system model complexities increased in the same direction and with a similar intensity, therefore explaining the need for the use of metamodels nowadays. This shows that the application of metamodeling techniques is not only exclusively related to the available computing power but also to the demand for highly detailed models, which are eventually conditioned by it. Notice that with more realistic models usually come large or even prohibitive simulation running times, which cannot be used for real-time applications or practical simulation output behavior analysis. In these situations, the use of metamodels as an auxiliary tool is always preferred (Kleijnen and Sargent, 2000) and can eventually be used to explore the simulation behavior less expensively.

The GP framework is a well-known modeling approach widely used as a simulation metamodel (Boukouvalas, 2010; Chen et al., 2011; Conti and O'Hagan, 2010; Kleijnen, 2009), and when applied to regression problems, it is also called Kriging (Chilès and Desassis, 2018). Initially, GP-based metamodels had only been used in deterministic simulations. Nevertheless, Van Beers and Kleijnen (2003) started using GPs for random simulations, showing that they had great potential in these kinds of applications too. Since then, several similar approaches arose (Kleijnen and Van Beers, 2005; Boukouvalas et al., 2009; Ankenman et al., 2010). Jones and Johnson (2009) generally discusses the use of GPs as a reliable metamodel for the design and analysis of computer experiments. Boukouvalas (2010) constructs two GP representations and develops an experimental design to extend the metamodel framework to account for heteroscedasticity in the simulation output. In (Kleijnen and Van Beers, 2004), the authors developed a method based on application-driven sequential designs using a GP metamodel. Wang et al. (2005) shows that the GP, as a metamodeling technique, can accurately approximate the complex functions often associated with non-linear and non-monotonic probabilistic input/output functions. In (Bastos and O'Hagan, 2009), several diagnosis metrics were introduced to validate the GP framework as a simulation metamodel.

In the context where simulation data can be challenging to obtain (e.g., computationally expensive simulation experiments) in a systematic way, active learning can prove to be a powerful learning paradigm to enhance the application of simulation metamodels. As a subfield of supervised machine learning, active learning is essentially an iterative sampling strategy that allows any algorithm designed upon it to select the data points from which it learns actively. Thus, instead of selecting a large number of random points, the algorithm searches for the most informative ones sequentially so that both the model training efficiency and its prediction performance are improved with a few training data points as possible (Settles, 2010).

According to Wang and Zhai (2016), an arbitrary active learning strategy encompasses five essential elements enclosed in the following quintuple

$$(\mathcal{L}, \mathcal{U}, \mathcal{M}, \mathcal{O}, \mathcal{Q}).$$

First, $\mathcal{L}$ is the labeled training data set. Then, the set of the unlabeled data points is represented by $\mathcal{U}$. Generally, $\#\mathcal{U} >> \#\mathcal{L}$, i.e., the number of unlabeled data points is much higher than that of the labeled ones. The machine learning model is represented by $\mathcal{M}$. Depending on the nature of the problem being modeled, it can be a classification or a regression model, which in turn affects the nature of the labels in $\mathcal{L}$ of being discrete or continuous, respectively. $\mathcal{O}$ denotes the oracle whose role is to provide labeled instances from the underlying process under study. Finally, $\mathcal{Q}$ is the query function that encodes the strategies and criteria for finding and selecting the most informative instances of $\mathcal{U}$ to be added to $\mathcal{L}$. There have been many query strategy formulations that essentially translate into different perspectives to approach the problem in question. As mentioned by Settles (2010), depending on both the nature of the problem and the model being used, several query frameworks can be adopted, such as uncertainty sampling (Lewis and Gale, 1994), query-by-committee (Seung et al., 1992), expected model change (Settles et al., 2008) and error reduction (Roy and McCallum, 2001), variance reduction (Geman et al., 1992) or density-weighted methods (Settles and Craven, 2008).

Closely related to the concept of the query function is the definition of a stopping rule. Due to its iterative nature, techniques based on active learning must be stopped at a specific time, either manually by the user or automatically by a stopping criterion. In both cases, the stopping rule must take into account the trade-off between the overall prediction performance and generalization capacity of the machine learning model and the associated costs of acquiring new labeled data. Such costs could be, for example, the computational workload or running time associated with any simulation model.

Active learning has been used in a variety of research fields. However, this work focuses on those applications involving simulation metamodels, particularly the GPs, where the simulation model plays the role of the oracle. As mentioned earlier, by providing a fully non-linear Bayesian approach, the GPs allow for an intuitive way to develop active learning algorithms due to their ability to model the uncertainty present in the data explicitly. As high predictive variance can be associated with a high degree of uncertainty, both the posterior mean and variance, provided by the GPs for any given data point, can be directly used to explore the simulation input space and thereby guide the search for the most informative data points. Such active learning schemes involving GPs are usually associated with exploration-exploitation problems, often studied in Bayesian Optimization contexts (Ling et al., 2016; Jones et al., 1998).

According to Schulz et al. (2017), while a simple exploration approach aims to learn an unknown function of interest as accurately and fast as possible, in an exploration-exploitation setting, the goal is to find the input that maximizes the output of an unknown function in an equally fast manner. Under an active learning strategy, Bayesian optimization uses a reward function to more efficiently select the next unlabeled data point, according to an automatic trade-off between the domain regions where the objective function is very uncertain (exploration) and those of where the same function is expected to attain high value (exploitation), as seen in (Brochu et al., 2010).

Within the transportation literature, the application of simulation metamodels is still rare and relatively recent. The available research can be broadly categorized into traffic prediction and optimization of networks, as mentioned by Song et al. (2017). Some such works include, for example, metamodeling for mesoscopic simulation (Ciuffo et al., 2013; Chen et al., 2015) and for travel behavior and dynamic traffic optimization (Zhang et al., 2014). In (Antunes et al., 2018) the authors proposed a restricted batch-mode active learning strategy to address the problem of efficiency of the metamodeling process using a simple traffic simulation and a Demand-Responsive Transportation (DRT) system simulator.

4.3. Methodological Approach

In this work, a straightforward pool-based active learning strategy is adopted. The experimental design is depicted in Figure 4.1. Here the unlabeled data set $\mathcal{U}$ is entirely available for querying and represents the simulation input region in which the simulator's behavior is analyzed. The pool of labeled instances $\mathcal{L}$ is comprised of simulation results, i.e., input-output $(n+1)$-dimensional tuples. The machine learning model $\mathcal{M}$ is a GP, whereas the query function $\mathcal{Q}$ is based on the analysis of the predictive variance provided by the latter at each point in $\mathcal{U}$.

The general idea within this experimental setting is to assume that a GP describes the functional relationship between the simulation input vector $\mathbf{x}$ and the output y. After the GP is fitted to $\mathcal{L}$, the provided conditional distribution is used to predict the output values (or labels) for each unlabeled data comprised in $\mathcal{U}$. This feature makes it possible to bypass simulation runs and approximate the simulator's behavioral structure, making the exploration process more efficient. The predictive variance is used as a measure of fitness, which should decrease as the iterative process evolves. Eventually, the final trained GP model is used as a simulation metamodel to explore the behavior of the simulator and then to conduct policy analysis and assessment. Concerning the GP implementation, the Gaussian Processes for Machine Learning (GPML) tool is used. It constitutes a free, open-source, and well-established Matlab package provided and maintained by Rasmussen and Williams (2006). More details regarding the GP framework, as well as the adopted methodology, are presented in the following.

4.3.1. Gaussian Processes

Despite being quite an old topic in the field of probability and statistics, the application of GPs within machine learning tasks has emerged in the past decade. As

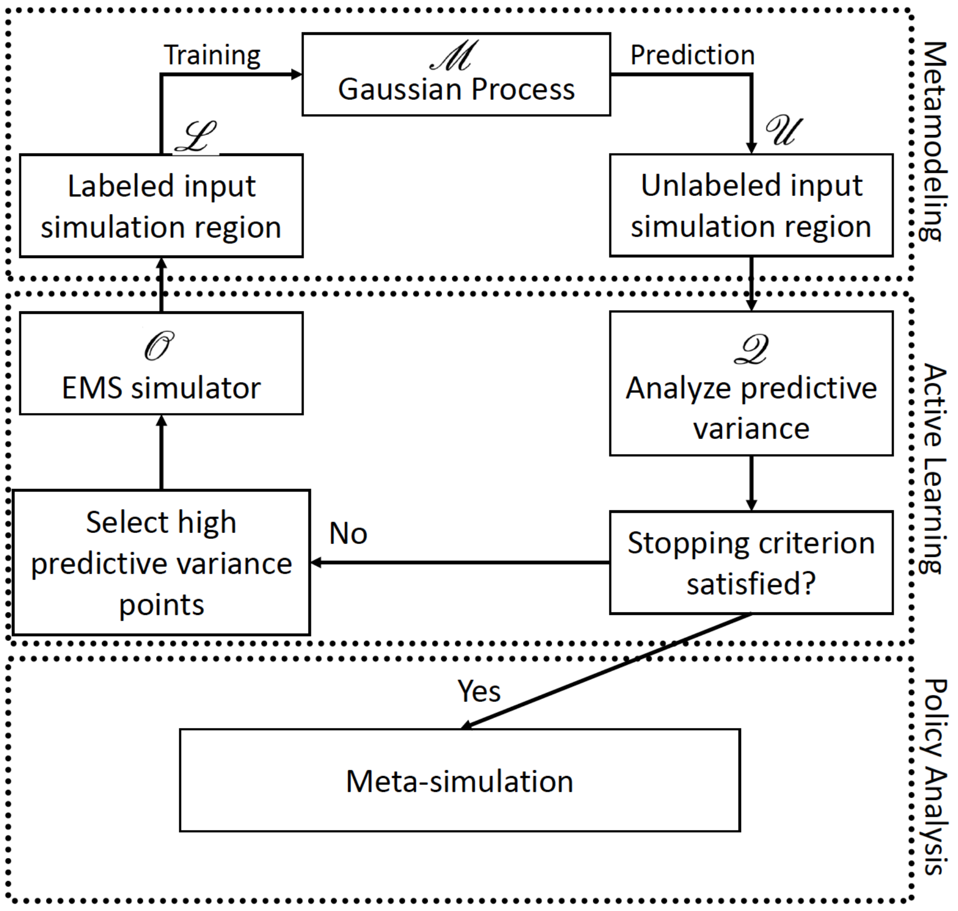


Figure 4.1: Active learning metamodeling experimental design. It is divided into three main blocks or steps. First, the simulation metamodeling comprises the approximation of the simulation model using a GP. Then, an active learning strategy is used to iteratively increase the fitting quality of the GP, in this case, by decreasing its total predictive variance across the unlabeled input simulation region. Finally, when the stopping criterion is verified, policy analysis using the provided meta-simulator is conducted.

described in (Rasmussen and Williams, 2006), a GP is a stochastic process from which each finite set of variables follows a multivariate Gaussian distribution. It is usually simply denoted as

$$\mathscr{GP}(m_f(\mathbf{x}), k_f(\mathbf{x}, \mathbf{x}'))$$

, where $m_f(\mathbf{x})$ and $k_f(\mathbf{x}, \mathbf{x}')$ are respectively a mean and a covariance function, with $\mathbf{x}$ and $\mathbf{x}'$ being two different input data points. Consequently, a GP is sufficiently characterized by these two functions. For the sake of simplicity, the mean function can be set to zero or any other constant value. This does not correspond to a significant restriction, as the mean of the process is not forced to be zero. On the other hand, the covariance function is of utmost importance for the modeling process, as it crucially encodes the notion of similarity between two given data points.

From a regression point-of-view, and as an intrinsic Bayesian approach, the GP modeling assumes a prior over functions, i.e., in the functional dependency established by $y = f(\mathbf{x}) + \epsilon$, where $\epsilon \sim \mathcal{N}(0, \sigma^2)$, it follows that $f(\mathbf{x}) \sim \mathcal{GP}(m_f(\mathbf{x}), k_f(\mathbf{x}, \mathbf{x}'))$. Most of the common mean and covariance functions typically have several free parameters, also called hyper-parameters of the GP, which can be optimized by marginal likelihood maximization subjected to the training data. The conditional distribution of a new unlabeled data point $\mathbf{x}_*$ is then given by

$$f_*|X, \mathbf{y}, \mathbf{x}_* \sim \mathcal{N}(k_{f*}^\top [K_y]^{-1}\mathbf{y}, k_{f**} - k_{f*}^\top [K_y]^{-1} k_{f*})$$

, with $k_{f*} = k_f(X, \mathbf{x}_*)$, $k_{f**} = k_f(\mathbf{x}_*, \mathbf{x}_*)$, K_y the covariance matrix, X the design matrix and $\mathbf{y}$ the set of training target values. Note that the prediction provided by the GP is a fully-defined Gaussian distribution, rather than a single point-wise estimate. This Bayesian property allows the GP to encapsulate the uncertainty not only of its owns predictions, but also that of the underlying signal being modeled. In turn, this uncertainty can be intuitively used to as an information criterion to design active learning strategies.

In this work, the standard Squared Exponential (or Radial Basis) covariance function, given by

$$k_f(\mathbf{x}, \mathbf{x}') = \sigma^2 exp\left(-\frac{||\mathbf{x} - \mathbf{x}'||^2}{2l^2}\right)$$

is used. Here, σ^2 and l^2 are respectively the signal variance and the characteristic length-scale. Note that this is a non-separable function with isotropic distance. For the GP mean, a constant function dependent on the average of output values of the training data set was used, $m_f(x) = k$.

4.3.2. Metamodelling Strategy

This work follows a modeling approach similar to that presented in (Antunes et al., 2018), and it is summarily depicted in Figure 4.2 in the form of a pseudo-algorithm.

The algorithm starts with the initialization data set, which is comprised of the first simulation input-output results. A GP is trained in this data set ($\mathcal{L}$) via likelihood maximization, and both the predictive mean and variance are obtained for each point in the input simulation region of interest ($\mathcal{U}$). Then, the algorithm selects the top five data points with the highest predictive variance values, top_5. This is meant to force an active selection of points rather than to select them at random, as the hypothesis is that the predicted values with higher predictive variance contain more information regarding the underlying process than those with lower variance. After this selection, the algorithm requests the simulator for the real output values (labels) corresponding to the unlabeled points in top_5. This is also called batch-mode active learning. It is particularly useful as it allows for the query processes to be parallelized. In this case, the points are being selected in batches of five, which are, eventually, added to $\mathcal{L}$. The process repeats until the stopping criterion is satisfied.

The stopping criterion is defined by the relation between the Initial Total Variance (ITV) and the Current Total Variance (CTV). Whereas the former is computed in the first iteration, the latter represents the current amount of variance summed over $\mathcal{U}$, and it is updated in each of the following iterations. As α lies in $(0, 1)$, the process stops if CTV reaches a fraction of ITV. In this work, $\alpha = 0.1$, which means that the stopping rule is satisfied when CTV is less than $(1 - \alpha)\%$, or 90% of ITV. Note that in the first iteration, $CTV = ITV$ holds. However, as the process evolves, CTV tends to decrease since the GP will inevitably acquire more training data points and thus greater knowledge about the simulation function. It is also worth noticing that the number of iterations required to satisfy this criterion is directly dependent on the size of the initial training set as well as on the value of α itself. If the initial set is both sized-up and representative enough, then the GP may be able to generalize well for the unobserved points present in $\mathcal{U}$. On the other hand, if α is near 1, then the criterion will be easily satisfied in a few iterations. Therefore, the right trade-off between these two elements must be carefully taken into account.

4.3.3. EMS simulator

In Emergency Medical Service (EMS) planning, evaluating strategic and tactical decisions in real-world systems, either via policies or operational solutions, is quite often physically unfeasible. Simulation and optimization approaches are thus preferable to mimic the systems' real conditions and to produce insightful assessments of decisions planned to be implemented. This is a synthetic process that can provide empirical

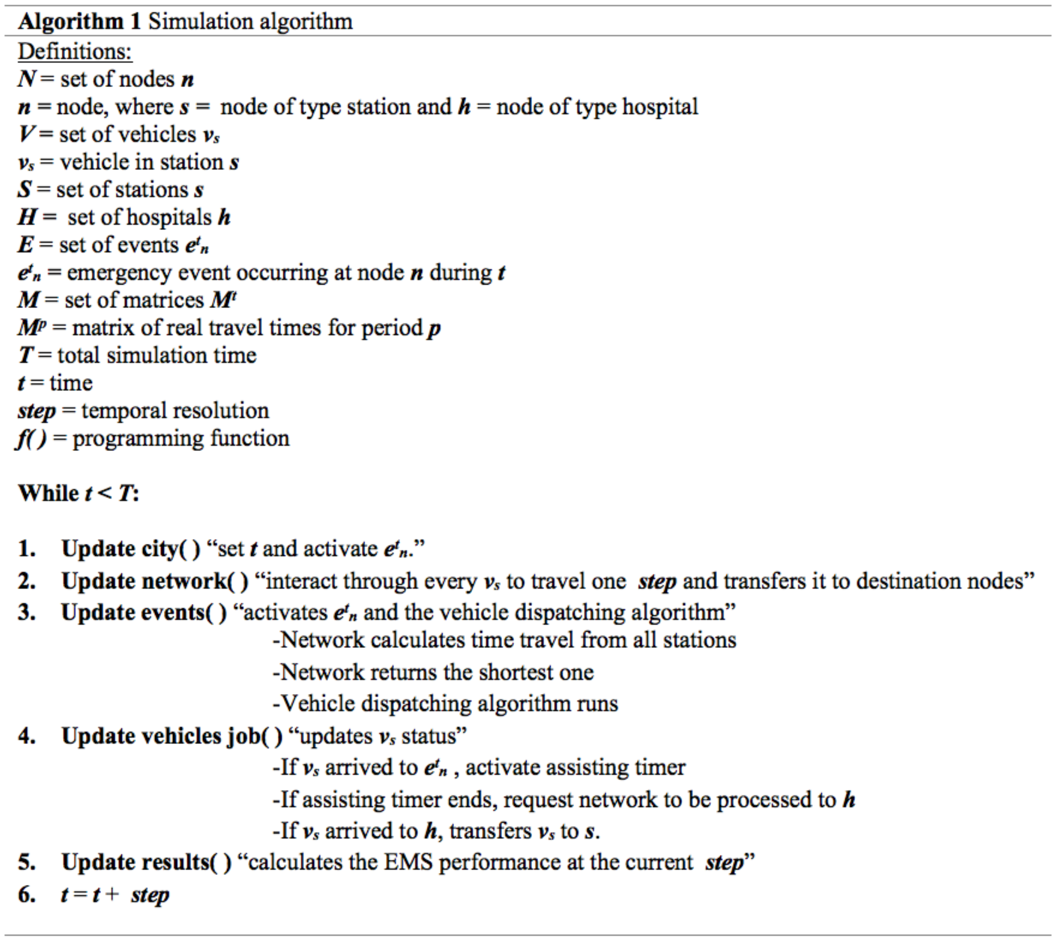
```
    Inputs: $\alpha \in (0,1), \mathcal{L}, \mathcal{U}$.
    While $CTV \geq (1-\alpha) \times ITV$ do
1:  Train the GP in $\mathcal{L}$ and predict the output values for each point in $\mathcal{U}$, $k_{f*}^{\top}[K_y]^{-1}\mathbf{y}$,
    and obtain their corresponding predictive variances, $k_{f**} - k_{f*}^{\top}[K_y]^{-1}k_{f*}$.
2:  Determine the top 5 highest predictive variance points in $\mathcal{U}$, $top_5$
3:  By simulation requests, obtain the true values for $top_5$, $\mathcal{L}_{+}^{top}$ as the new labeled set.
4:  Expand the labeled pool: $\mathcal{L} = \mathcal{L} \cup \mathcal{L}_{+}^{top_5}$.
5:  Update CTV.
    end
    Output: GP trained in $\mathcal{L}$ and associated predictions over $\mathcal{U}$.
```

Figure 4.2: Active learning metamodeling pseudo-algorithm.

evidence of how a certain choice or change might impact a certain targeted performance metric.

A simulation algorithm to numerically compute EMS solutions performance in terms of average response time and average survival is adapted from the work of Amorim et al. (2018), whose algorithm is summarized in Figures 4.3 and 4.4. This optimization model is used to assess the performance of station and vehicle locations in an urban area. It also provides a platform to measure the performance of dispatching policies and other tactical decisions. This work focuses on strategic and tactical decisions and their performance using two metrics: the classic average response time and a survival metric that tries to capture victims' outcomes.

The simulation model uses an agent-based model controlled by a city agent that takes the role of the emergency medical service. It allocates and dispatches vehicles using a closest dispatching rule (Haghani and Yang, 2007; Jagtenberg et al., 2017a; Yang et al., 2005). A network agent simulates stochastic traffic conditions and EMS vehicle movements by using nodes and arcs as an abstraction of the reality, pre-computing travel times for different periods of the day and the week using Google's Directions API. Events are agents that represent EMS calls and are triggered according to a historical database and a stochastic location change, denoted by w_1.



```
Algorithm 1 Simulation algorithm
Definitions:
$N$ = set of nodes $n$
$n$ = node, where $s$ = node of type station and $h$ = node of type hospital
$V$ = set of vehicles $v_s$
$v_s$ = vehicle in station $s$
$S$ = set of stations $s$
$H$ = set of hospitals $h$
$E$ = set of events $e^t_n$
$e^t_n$ = emergency event occurring at node $n$ during $t$
$M$ = set of matrices $M^t$
$M^p$ = matrix of real travel times for period $p$
$T$ = total simulation time
$t$ = time
step = temporal resolution
f( ) = programming function

While $t < T$:

1. Update city( ) "set $t$ and activate $e^t_n$."
2. Update network( ) "interact through every $v_s$ to travel one step and transfers it to destination nodes"
3. Update events( ) "activates $e^t_n$ and the vehicle dispatching algorithm"
        -Network calculates time travel from all stations
        -Network returns the shortest one
        -Vehicle dispatching algorithm runs
4. Update vehicles job( ) "updates $v_s$ status"
        -If $v_s$ arrived to $e^t_n$ , activate assisting timer
        -If assisting timer ends, request network to be processed to $h$
        -If $v_s$ arrived to $h$, transfers $v_s$ to $s$.
5. Update results( ) "calculates the EMS performance at the current step"
6. $t = t +$ step
```

Figure 4.3: Pseudo-algorithm for the studied EMS simulation model.

The model runs in short fixed time steps, and every time a step is made, an update process is triggered, as depicted in Figure 4.4. This trigger initiates the city agent tasks. Firstly, the city agent contacts the event agent to check the status of the current events. The event agent updates its list and reports back to the city agent. If a new event exists, it will request the latter for assistance. If all events' statuses are up-to-date, then the time will advance. At the same time, the city agent communicates with the network to keep track of the vehicles' location and status. Likewise, the network reports such status to the city agent and also informs the event agent not only when a victim has been assisted (storing the vehicle arrival time in the database for performance assessment) but also when the victim arrives at the hospital. The assisting time is a stochastic variable that assumes a gamma distribution with a mean of 10 minutes and a standard deviation of 5 minutes, according to the observations made by Pons and Markovchick (2002). In case a new event is activated during the

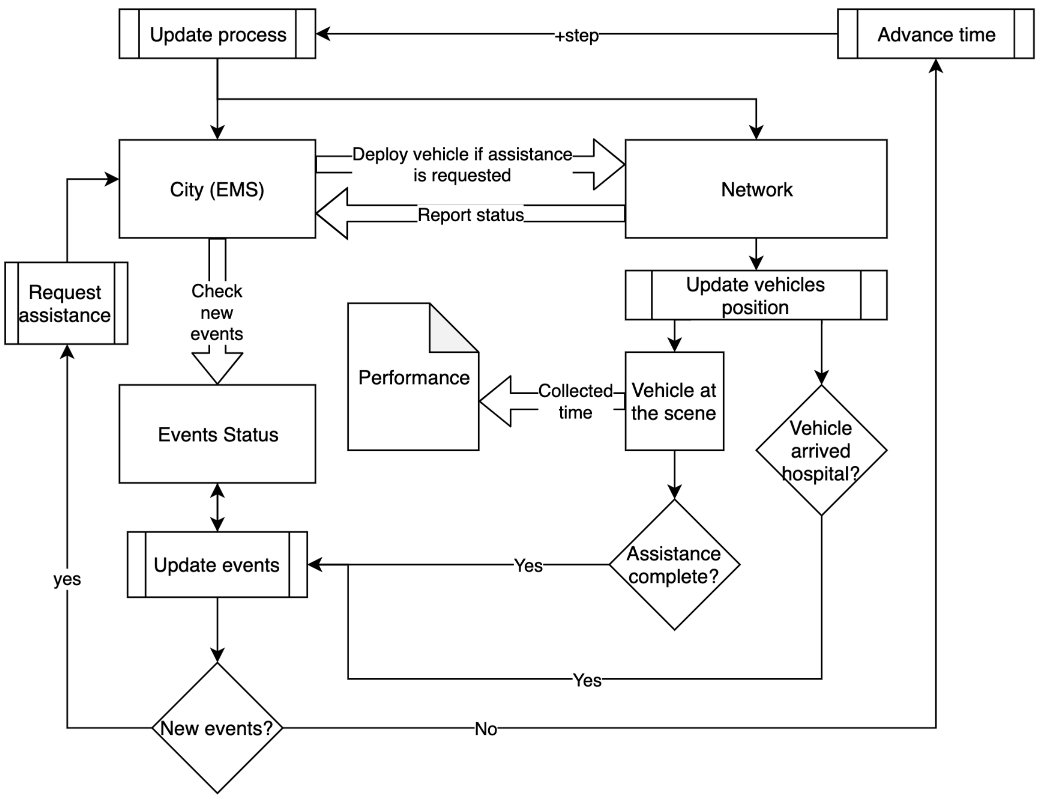


Figure 4.4: Flowchart describing the essential steps implemented in the studied EMS simulation model.

events update, a request for assistance is sent to the city agent, which, after receiving the network report status, decides on the response strategy by selecting the proper vehicle to be allocated to the new event.

The stochastic location change captures a possible randomness in the demand by re-allocating a medical emergency event e_p at node p to node p', with a probability of w_1, according to $e_{p'} = p(g(E_s), e_p, w_1, w_2)$. Here, $g(E_s)$ is a function that chooses a location from a set of possible locations weighted by the observed demand at period s, and $p(e_1, e_2, w_1, w_2)$ is a function that picks either e_1 or e_2 with probability w_1 and w_2, respectively, where $w_1 + w_2 = 1$ must hold.

On the other hand, traffic stochasticity is controlled by parameter ε. It introduces an error term in any observed travel time $r_{s,l,p}$, which measures the time it takes to travel from point l to point p during s, according to $r'_{s,l,p} = r_{s,l,p} + r_{s,l,p} \times \mathcal{N}\left(\mu, \frac{\varepsilon^2}{4}\right)$. When the mean is set to zero and the standard deviation to $\varepsilon/2$, a confidence of 95% exists that the travel time has at most an error of $\varepsilon \in [0, 1]$.

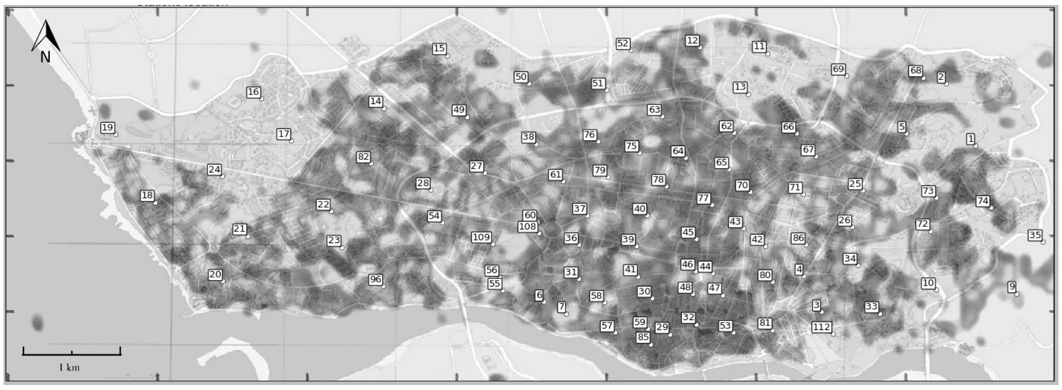


Figure 4.5: Locations of the 90 emergency vehicle stations in Porto, Portugal.

4.4. Results

In this section, the presented methodology is applied to the EMS simulator developed in (Amorim et al., 2018). As previously seen, this simulator is designed in terms of three kinds of input dimensions, namely, the location change probability, traffic error, and vehicle station locations. In terms of value ranges, the first two lie in the interval $[0, 1]$, whereas the locations assume discrete positive values, $[0, 1, 2, 3, ...)$, representing the number of vehicles allocated to each position. Figure 4.5 shows the location of these stations, 90 in total, scattered in the city of Porto, Portugal. Note that not all the depicted labels follow a sequential order. The last four locations are respectively labeled with 96, 108, 109, and 112.

In Table 4.1, a sample of the data used in this work is presented. The dimension, or feature, denoted by x_1 is the location change probability, whereas x_2 is the traffic error. The locations and the associated number of vehicles are coded in features x_3 to x_{92}. For the simulation outputs, y_1 and y_2 are the survival rate and response time, respectively. Due to space limitations, only the results for x_1, x_2 and x_3 are analyzed in this chapter.

Each output was modeled independently, i.e., a different GP was used to approximate the simulation results. Figures 4.6-4.9 present the obtained results. Similar conclusions can be drawn for both simulation output variables. Notice that panels (a) and (b) of Figures 4.6 and 4.8 do not depict any kind of time series per se, but rather a straightforward way to simply represent the prediction originated from high-dimensional data and their corresponding confidence intervals. Therefore, particularly in these representations, there is no neighborhood notion between observations, nor

Table 4.1: A data sample showing the dimensional structure with 92 features and two output variables: x_1 and x_2 lie in the interval $[0, 1]$, x_3 to x_{92} assume discrete values in $[0, 1, 2, ...)$, and y_1 and y_2 are a real-valued variables.

x_1	x_2	x_3	x_4	...	x_{90}	x_{91}	x_{92}	y_1	y_2
0.0445	0.2921	0	0	...	0	0	0	0.3547	379.0279
0.4923	0.7756	0	0	...	2	1	0	0.1953	521.3361
0.0708	0.3740	1	0	...	0	0	0	0.3800	331.0639
0.2199	0.6818	2	1	...	0	1	0	0.2252	515.0807

the sequential order of each observation is of any relevance.

In both cases, the active learning metamodeling process started with the GP training stage using 100 random simulation points. Then, the prediction was conducted over 856 points scattered throughout the simulation input space. Take, for instance, the case of y_2, the average response time. Figure 4.8(a) shows the first GP approximation along with a confidence interval of 95%. As expected for this first iteration, the uncertainty of the model, which is encoded into the variance of each prediction, is relatively high. It can be observed that the GP is approximating the simulation output by values fairly around 450 (minutes).

Moreover, these predictions are associated with large confident intervals, which is not desirable in real-world practice, especially for emergency-related operations decisions along which rapid responses ought to be undertaken. The width of these intervals can be viewed as a proxy for the uncertainty present not only within the data but also within the GP model itself. In the end, after 59 iterations, this variance decreased considerably, as depicted in Figures 4.8(b) and (c).

Similar behavior can be observed for y_1. However, contrary to the case of y_2, this time, the active learning procedure required 16 extra iterations to achieve the same reduction of 90% of the total initial variance (see Figure 4.6(c)). It is worthwhile mentioning that, in both cases, the corresponding final GP approximations present a wider amplitude of predicted output values when compared to the results regarding the first iterations. In Figures 4.6(a) and 4.8(a), the GP is predicting inside narrow horizontal bands centered around 0.27 and 450, respectively, under a relatively large margin of uncertainty. Then, when the final iterations were reached, not only the associated predictive variance decreased, as mentioned before, but also the amplitude of the predicted values changed considerably. This translates into an improved prediction performance for both output variables, as the GP is now closer to the real underlying unknown function described by the simulation model.

In (Pons and Markovchick, 2002), an empirical study involving 3576 patients transported to a single Level I trauma center was conducted in order to assess the 8-minute guideline for ambulance responses. The authors concluded that there was no significant difference in the survival rate due to traumatic injuries between the patients who were assisted within and above the established response time policy, respectively. In the same work, it is mentioned that the mortality odds ratio is approximately 0.81 for response times higher than 480s. Conversely, the odds for the patients' survival are circa 0.19. By taking these two elements into account, a meta-simulation analysis was carried out in terms of the two outputs provided by the studied EMS simulator.

Figure 4.7(a) shows that the obtained survival rates averages were conditioned by both the traffic error and location change probability inputs. From traffic errors in the order of 60%, it is visible that emergency cases associated with lower rates of survival start to emerge. On the other hand, as the probability of location change increases, the rate of survival also decreases. Most of these low survival observations are concentrated in the upper right corner, slightly within $[0.5, 1.0] \times [0.6, 1.0]$. Similar conclusions can also be derived from the observation of Figures 4.7(b) and (c). Again, and this time concerning the number of vehicles in location 1, which corresponds to variable x_3, most of the mortal occurrences are associated with higher values of location change probability and traffic error. This is particularly more evident for the latter simulation input.

Regarding the traffic error, and taking into account the earlier presented details of the used EMS simulator, the obtained results meet the initial expectations. This simulation input represents the error associated with the traffic prediction in comparison with the real traffic immediately before the vehicles dispatching. Therefore, it makes sense that higher traffic errors may lead to poor operational decisions, resulting in higher response times and consequently in more fatal occurrences. With respect to the location change probability, the results also confirm the intuition. Increasing this probability will lead to higher variability in the location of the life-threatening events or emergency calls in comparison with historical data.

For the average response time, similar results were attained. In Figure 4.9(a), it can be concluded that the observations associated with response times greater than or equal to 480s are concentrated in $[0.0, 1.0] \times [0.6, 1.0]$. Contrary to what one would expect, not so many of these observations had led to survival rates below 0.19. This matches with the conclusions made by (Pons and Markovchick, 2002), i.e., that average response times greater than 8 minutes do not necessarily lead to high mortality rates. Nevertheless, high response times, combined with a high probability of call location change, seem to lead to higher values of this output performance measure. In the end, it is true that when the traffic error increases, the delays in the vehicles' arrivals increase accordingly.

It is crucial to take into account the kind of variation encoded into these two simulations inputs. Note that the traffic error does not have a direct implication on the vehicles' travel times. Instead, what is being measured by the EMS simulator is to which extent traffic errors lead to bad choices of dispatching the correct vehicles, both in terms of number and station locations. Given a specific emergency call, and ideally assuming no traffic congestion, the ideal vehicle to dispatch would be the closest one. However, when traffic congestion exists and, besides that, the real traffic information is only available with a certain degree of error, the dispatching solution is not so obvious.

Furthermore, the distance is no longer measured in space units, but rather in time ones, due to the existence of traffic congestion, which in turn makes the problem more challenging. Take, for instance, the following example where station 1 is 10 minutes from the emergency event location (E), whereas station 2 is 20 minutes away from the same event. Given an error of $\pm 20\%$, this means that the emergency service operator may assume that stations 1 and 2 are, respectively, 12 and 16 minutes from E. In practice, especially in these kinds of life-threatening situations, this represents a significant error with potentially dramatic consequences. For this case, the decision result will be the same (i.e., a vehicle will be dispatched from location 1), which turns out to be the best decision. However, consider an alternate configuration, where both stations 1 and 2 are, respectively, 10 and 11 minutes from E. Here, the same traffic error could lead the operator to take a poor decision based on the assumption that station 1 is distanced in 12 minutes and station 2 in 9 minutes, which, in reality, is not

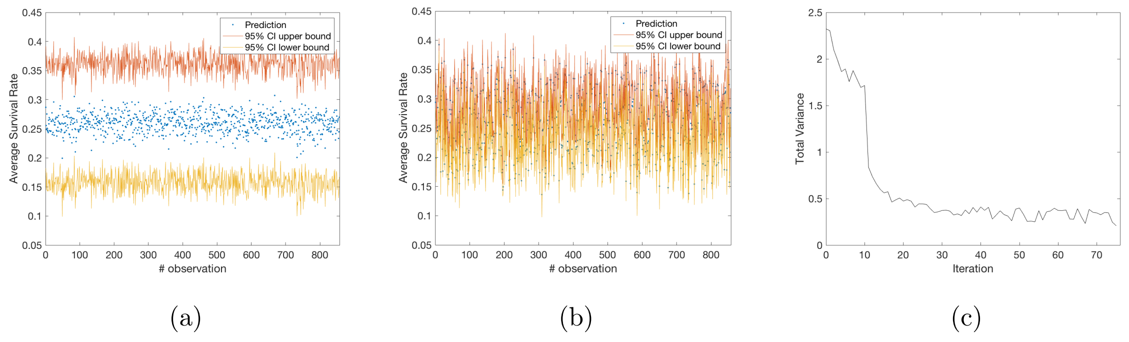


Figure 4.6: Results for the average survival rate output: (a) depicts the initial GP predictions over 856 simulation points, along with the corresponding 95% bounds, whereas (b) the final GP approximations. 100 random input points were used as the initial training set. Panel (c) shows the required number of iterations to satisfy the stopping criterion with $\alpha = 0.1$. In each iteration, the top five predictive variance points were selected to be added to the training set.

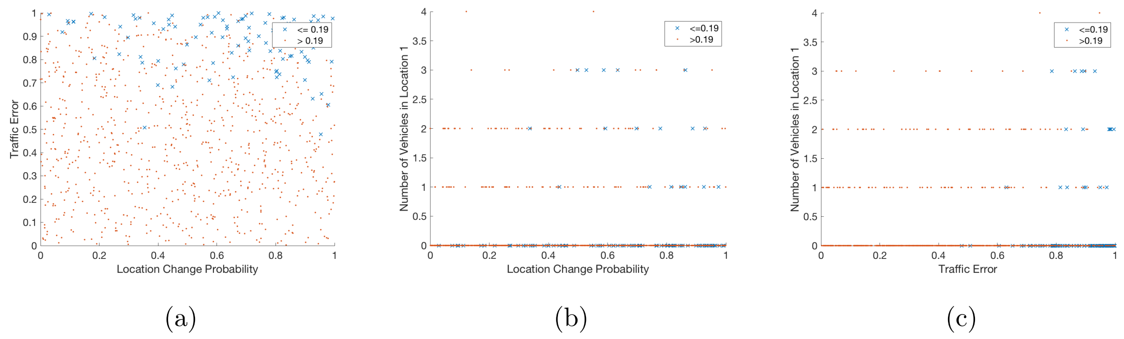


Figure 4.7: Comparison of the obtained predicted values for different input dimension with an average survival rate threshold of 0.19: (a) Location Change Probability versus Traffic Error, (b) Location Change Probability versus Location 1 and (c) Traffic Error versus Location 1.

valid. In such a situation, this unfortunate decision would be, for example, to send a vehicle from location 2, which is, in fact, 1 minute further than the other. Therefore, the same error can lead to different decisions and different outcomes.

On the other hand, the probability change location is especially important to induce a stochastic behavior to the emergency calls' locations so that the conclusions are not drawn exclusively from historical data. The simulation model must take into account the inherent dynamics of emergency events and should be able to respond accordingly.

Finally, in order to validate the generalization capacity of the obtained GP approximations for each simulation output, 30 random runs using 200 test points were conducted in a 10-fold cross-validation scheme. These points were randomly sampled

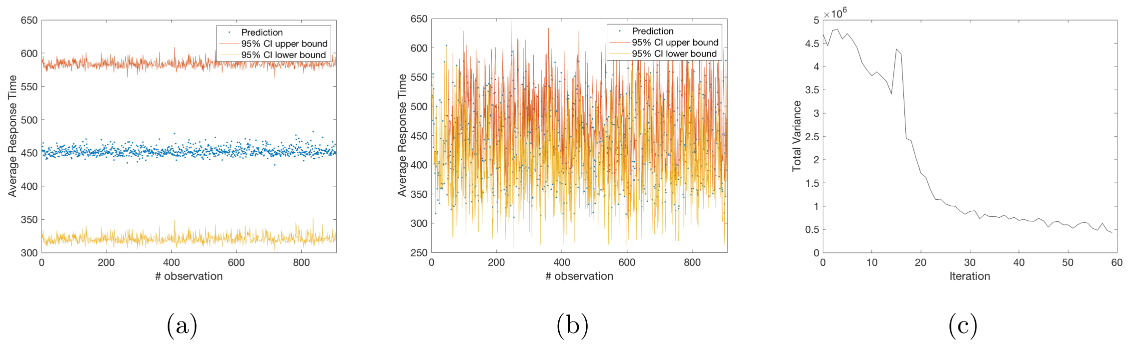


Figure 4.8: Results for the average response time output: (a) depicts the initial GP predictions over 856 simulation points, along with the corresponding 95% bounds, whereas (b) the final GP approximations. 100 random input points were used as the initial training set. Panel (c) shows the required number of iterations to satisfy the stopping criterion with $\alpha = 0.1$. In each iteration, the top five predictive variance points were selected to be added to the training set.

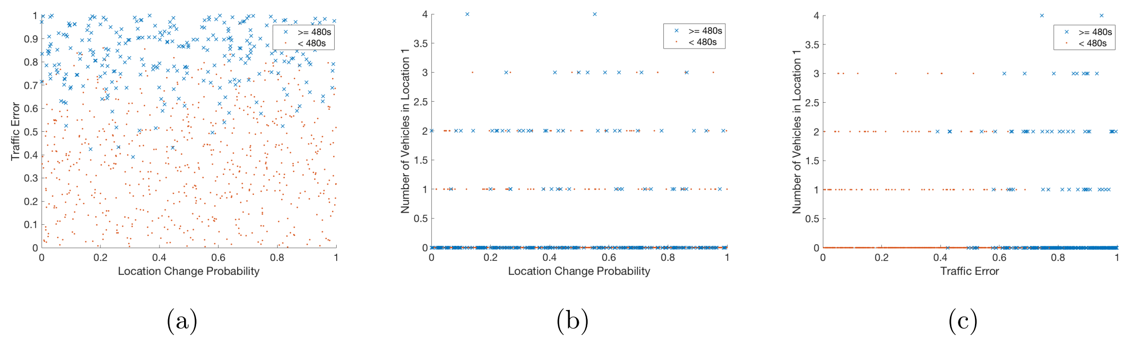


Figure 4.9: Comparison of the obtained predicted values for different input dimension with an average response time threshold of 480 seconds (8 minutes): (a) Location Change Probability versus Traffic Error, (b) Location Change Probability versus Location 1 and (c) Traffic Error versus Location 1.

from the simulation input space. Note that, for each output variable, only the GP obtained in the last iteration was used for validation purposes. Five well-known metrics were used, namely, the Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), Root Relative Squared Error (RRSE), and the Pearson's correlation coefficient (Corr.), in order to evaluate the predicted performance of the these GPs. The results are summarized in Table 4.2. These metrics were computed by comparing the predicted values given by the GP against the real known output values obtained from simulation sampling. The GP yields slightly better performance for the survival rate output (see RAE, RRSE, and Corr. columns), despite requiring more iterations to satisfy the ac-

Table 4.2: Average results, and associated standard deviation values, obtained from 30 random computer runs using 200 test points and a 10-fold cross-validation scheme, for the two studied simulation outputs, average survival rate (y_1) and average response time (y_2).

Output	**RMSE**	**MAE**	**RAE**	**RRSE**	**Corr.**
y_1	0.0298 ± 0.0020	0.0237 ± 0.0017	0.4970 ± 0.0436	0.5287 ± 0.0397	0.8497 ± 0.0231
y_2	40.4023 ± 2.5809	32.1174 ± 1.9981	0.5335 ± 0.0353	0.5681 ± 0.0326	0.8238 ± 0.0215

tive learning stopping criterion (see Figures 4.6(c) and 4.9(c)). The standard deviation values provide a systematic assessment of the variability around the averaged results across the multiple cross-validation random runs. It can be concluded that, in both cases, the GP presents good generalization and prediction performances.

4.5. Conclusion & Future Work

This chapter presented a methodology based on active learning metamodeling to address the problem of exploring the behavior of simulators developed in the context of transportation simulation and policy analysis. Particularly, an Emergency Medical Service (EMS) was used for illustration, and two important output thresholds were considered: 0.19 for the average survival rate and 8 minutes (480s) for the average response time.

The presented work shows that the proposed methodology can help in the identification of important regions of the simulation input space that have a direct impact on the performance or implementation of specific policies while avoiding several simulation runs at the same time. Moreover, if the simulation input space proves to be sufficiently high-dimensional or if each simulation run shows prohibitive computational workload and runtimes, an exhaustive exploration process is virtually impossible. Therefore, the joint use of active learning strategies and simulation metamodels designed to identify policy-relevant regions within such space has great potential in practice, especially for decision-making processes.

In the future, several research directions can be undertaken in order to extend and improve the presented work. Firstly, the methodology should improve in terms of graphical representation, although working with high-dimensional data constitutes a significant challenge in that aspect. The graphical results (policy-relevant simulation input regions) should be provided in such a way that they make it easier for policies to be analyzed. Moreover, the presented methodology can also be improved through clustering techniques applied to the results so that such important input regions are more easily identifiable.

This work did not take into account any specific design for computer experiments (such as the well-known Latin hypercube scheme) to select the initial training set. These designs provide sampling strategies that lead to a good and systematic understanding of the underlying properties of simulation models. This eventually improves the statistical significance as well as the prediction performance of simulation metamodels. Instead, simple random sampling was used. There are plans to apply such sampling schemes in combination with new active learning strategies soon and use the current work as a baseline benchmark.

In order to further validate its potential, this work should be replicated not only using other kinds of transport simulation models but also considering more complex policy analysis, possibly taking into account a combination of policies. Additionally, it would also be interesting to provide a certain degree of interactivity with the user. For instance, instead of letting the active learning independently decide which data points to request the simulator to label, the user could be allowed to engage in this process too. This could represent an interesting feature for decision-making practitioners, as it would directly involve them and their expertise in the modeling process itself.

Further numerical experiments will be conducted in the future, especially concerning the tuning of the involved parameters, namely, the GP hyperparameters and the methodology-related parameters, i.e., α, size of the initial training set, and the number of active learning data points. This should improve both the fitting and generalization capacity of the obtained GP metamodels. Furthermore, sensitivity analysis of these parameters will also be conducted.

Due to its close relationship with simulation metamodeling, Bayesian optimization techniques will receive particular attention in future works. If the goal is to seek for those set of points that maximize (or minimize) the expensive and black-box function inherently defined by the simulation model, then such derivative-free optimization approaches should be considered.

Lastly, but not least, another important issue is the possible correlation between the simulation outputs, which was not considered in this work, as each output was modeled independently. Multiple output regression, such as the multi-output GP framework, can improve its prediction performance by considering relationships across the outputs. The integration of such models in the proposed methodology would also be an interesting line for future research.

Chapter 5

Integrating Simulation Metamodeling with Optimization

"It is not knowledge, but the act of learning, not possession but the act of getting there, which grants the greatest enjoyment."

Carl Friedrich Gauss, 1777 - 1855

Choosing locations for emergency medical service stations and vehicles has been thoroughly investigated. However, the formulations presented to solve this question are not always done in a way that can be applied in practice, as they are based on over-simplified mathematical functions, which makes them rather unrealistic. The problem persists as integrated strategic and tactical approaches require an analytical complexity that often invalidates the exact solution.

The work developed within this chapter proposes an integrated strategic and tactical planning decision methodology that complements an optimization model with a local search using a metamodel as a proxy of the real system. This allows empirical evidence to be inferred for vehicle location solutions that improve performance in the real system. The methodology is applied to the city of Porto, and two dilemmas are tested to show proof of application. First, the debate between the integrated versus non-integrated approach is analyzed. Second, an assessment of the advantages of adding a vehicle or a station to the planning budget is analyzed.

The conclusions of this research support the advantages of an integrated approach indicated by other studies. The results also show that adding a new vehicle to the system is more advantageous than adding a new station when it comes to victims' survival. These two application examples provide proof of the methodology's applicability and open doors for future research on the subject.

5.1. Introduction

Stations, distribution centers, and other facilities that have a radius of action, and need to meet certain demands within that radius, are often in operation for many years, and therefore subject to physical and temporal changes in the environment in which they operate. Classic facility and vehicle location problems are usually faced with highly uncertain costs, variable demand, and ambiguous travel times, as well as other measures that are hard to correlate. Consequently, these types of problems require decision-making tools to deal with the underlying uncertainties and avoid the risk of underestimating or overestimating their design, which in return results in lower system performance or more expensive solutions.

Emergency Medical Service (EMS) strategic and tactical decisions are two of the classic facilities and vehicle location problems. In general, designing an EMS response system, from a transportation perspective, covers the location of vehicle facilities, allocation of vehicles to facilities, and response policies to outline rules that help to decide which vehicle is to be dispatched to a certain medical emergency. The design of the EMS response plan can be divided into two planning stages: strategic and tactical decisions.

At the strategic level, long-term decisions are usually made concerning the location of emergency vehicle stations, and a long-lasting infrastructure is established for the EMS response (Amorim et al., 2017; He et al., 2018; Tufuor et al., 2018). These decisions result in building or renting warehouse structures, which should last for many years to come and must accommodate the required resources, e.g., response vehicles. Tactical decisions, in contrast, define mid or short-term decisions (Amorim et al., 2018; Van Essen et al., 2013), such as the allocation of response vehicles to EMS stations. Hence, there is no permanent commitment to the chosen solution.

These two planning stages have been studied in-depth, especially focusing on the mathematical problem, the algorithm itself, and problem-solving techniques, mainly heuristics. Nevertheless, mathematical models are an abstraction of reality and often require simplifications that lead to one important question: what is being improved? What looks optimal on paper might not correctly translate into practice. Recently, new research directions have pointed towards practice-ready decision tools, such as simulation or scenario-based optimization. These techniques are able to integrate the system characteristics successfully and can elaborate and solve more complex problems having a practical focus.

The present study follows the aforementioned new direction and studies a methodology for integrated strategic and tactical planning decisions. As stated before, strategic decisions have a permanent commitment, and a wrong decision can affect the performance of the forthcoming tactical decisions. We believe that separating these two decisions can lead to sub-optimal solutions. This fact has been shown by Van Essen et al. (2013). However, they point out that a combined approach leads to an increase in the problem size, and extra simplifications might be necessary so that a solution can be found.

A two-stage methodology is proposed for solution exploration with empirical evidence. In the first stage, a scenario-based optimization model provides integrated strategic and tactical planning with the necessary simplifications to deliver a solution within a reasonable time. The second stage focuses on exploring the neighborhood of the optimized solution for alternative solutions that might perform better using empirical evidence. A simulation model is an alternative for empirical proof inference because experimenting in the real system is prohibitive. However, simulation models require excessive computing resources and are time-consuming. Evaluating a unique solution might take several minutes or even hours. For this reason, a metamodel-based simulation is preferred because it allows a much quicker evaluation of alternative solutions with a minor accuracy loss.

This work contributes to the state of the art by 1) providing a methodology to integrate strategic and tactical EMS planning decisions with a practical focus, 2) supporting the claim that separate strategic and tactical decisions lead to sub-optimal solutions, and 3) using a metamodel-based simulation to complement mathematical optimization derived solutions with empirical evidence and applications to a case study.

5.1.1. EMS station and vehicle location

Emergency medical service location problems are one of the classic problems in operational research (OR) and mathematical programming. The two most important studies in the field of EMS and station location are the works of Toregas et al. (1971) and Church and ReVelle (1974). The former presents a solution to solve the location set covering problem (LSCP), making sure all demands are covered within a maximum time or distance radius, whereas the latter focuses on maximizing the coverage. Nonetheless, full coverage is hard to reach, especially when resources are limited, which is the case of practical applications. Li et al. (2011) reviewed covering models for EMS and highlighted many of them that relax some of the assumptions made in (Toregas et al., 1971).

The follow-up model formulations tried to take into consideration the problem of facility or vehicle availability. From the hierarchical approach proposed by Daskin and Stern (1981), which firstly integrated multiple coverage standards, Gendreau et al. (1997) formulated the famous Double Standard Model (DSM). This model has been widely used as a solution to account for vehicle or facility unavailability, ensuring that an alternative facility or vehicle is available within a second standard distance. ReVelle and Hogan (1989) tackled the problem with a different approach and formulated a probabilistic version of the LSCP with the requirement that all the demand points must be covered with a reliability level α.

Another important point to highlight is that demand and travel times are not evenly distributed temporally and spatially, and thus the busy fraction of a vehicle varies from facility to facility. This problem was investigated, and a maximal expected coverage location model with time variation (TIMEXCLP), where varying demands are incorporated over time, is proposed by Repede and Bernardo (1994).

Once more, the review carried out by Li et al. (2011) indicates several other extensions of the maximal expected coverage location model (MEXCLP). Fujiwara et al. (1987) applied simulation modeling to make further analysis on the optimality of an EMS location problem in Bangkok using MEXCLP, whereas more recently, Schmid and Doerner (2010) developed a multi-period version of the DSM which takes into account time-varying coverage to optimize coverage at various points in time simultaneously. The latter work concluded that it is vital for EMS location problems to consider time-dependent variations in travel times and coverage, respectively. As a follow-up, Dibene et al. (2017) used a similar approach in the city of Tijuana and concludes that demand coverage and response times in Tijuana can be enhanced by relocating the current stations without needing additional resources.

However, the above-mentioned authors have focused their models on operational performance metrics such as response time or coverage. These are the common metrics used both in practice and research (Bélanger et al., 2016; Ünlüyurt and Tunçer, 2016). One of the greatest impacts of planning EMS is the medical response time and how it can change patients' survival. Accordingly, Erkut et al. (2008) developed the Maximum Survival Location Problem (MSLP), which incorporates a survival function into the covering model to maximize patients' survival.

A second issue of the former models is the fact that using analytic formulations is still oversimplifying the real system. Notwithstanding, using robust solutions such as the scenario-based approach, which already captures part of the stochasticity existing in daily emergency medical services, is still a high simplification of the reality. Simulation models come as a response to this problem because they attempt to provide a tool that can deliver an empirical platform that translates the real system.

Using simulation models allow researchers to formulate more realistic and complex problems, usually to assess solutions or to support optimization models (Bélanger et al., 2016; Iannoni et al., 2009; Maxwell et al., 2010; Restrepo et al., 2009; Su and Shih, 2003; Ünlüyurt and Tunçer, 2016; Yue et al., 2012). McCormack and Coates (2015) investigates how simulation can enhance the level of realism in EMS models, making it applicable to complex real-life systems if proper data exists. Yet, a more detailed and complex model comes with the cost of higher computing power and time. For a deeper review on simulation applied to EMS problems, the reader is forwarded to the review made by Aboueljinane et al. (2013).

One can assess a short set of solutions by using simulations. However, when it comes to a local search, where thousands or millions of alternative solutions might exist, using simulations becomes unfeasible. When the search for the solution requires extensive experimentation, simulating each instance becomes time-consuming. Thus, simpler estimations are preferred. These are models of the model, which have been introduced as metamodels by Blanning (1974) and further statistically developed by Kleijnen (1975). In the transport research field, metamodels have only recently been applied, for example, in traffic problems (Ciuffo et al., 2013), for network optimization (Song et al., 2017) or in demand-responsive transportation (Antunes et al., 2018). Furthermore, Barton and Meckesheimer (2006) discuss and show the usability of metamodels in optimization problems to explore local solutions that can better fit the real system characteristics.

The idea of complementing optimization with a metamodel to evaluate local solutions is shown to be valid, and, as far as the authors know, there are no relevant studies in this regard within the scope of EMS stations and vehicle locations.

5.2. Methodology

A two-stage methodology is proposed to integrate EMS response strategic and tactical decisions in a unique plan design. In the first stage, an optimization model looks for a conceptual solution that performs optimally on paper, and in the second stage, a metamodel-based simulation is used to explore alternative solutions for empirical evidence and to choose the best one. The core of the methodology is a multi-period scenario-based optimization model that mathematically finds a solution for the EMS location problem. Afterward, a metamodel trained to use a Gaussian Process is applied to assess alternative local solutions that are generated by a combinatorial algorithm. This process leads to possible alternatives that will empirically perform better in the real system.

The two fundamental stages are achieved using three tools, namely, 1) an optimization model, 2) a local search supported by a metamodel-based simulation, and 3) a simulation model to train the metamodel. A diagram of these stages and the necessary modeling processes are systematized in Figure 5.1.

The optimization model locates medical emergency vehicles and stations while maximizing patients' survival. This model considers the station's "busy-fraction" to define the number of vehicles, uses "discrete time" to define scenarios, and has a set of parameters that allow for tailored solutions pending operational constraints. The main goal is to provide optimized station and vehicle locations.

A Gaussian Process-based metamodel is trained using data points obtained from a simulation model to serve as empirical evidence of the solution performance within a stochastic environment that mimics the real system. A metamodel-based simulation is preferred because, with a minor loss in accuracy, it overcomes the existing simulation models because they use many resources (computational and human) and time, which would lead to a drastic reduction in the number of solutions that could be tested. The only time-consuming tasks are the metamodel parameter calibration and generation, via simulation, of the associated training.

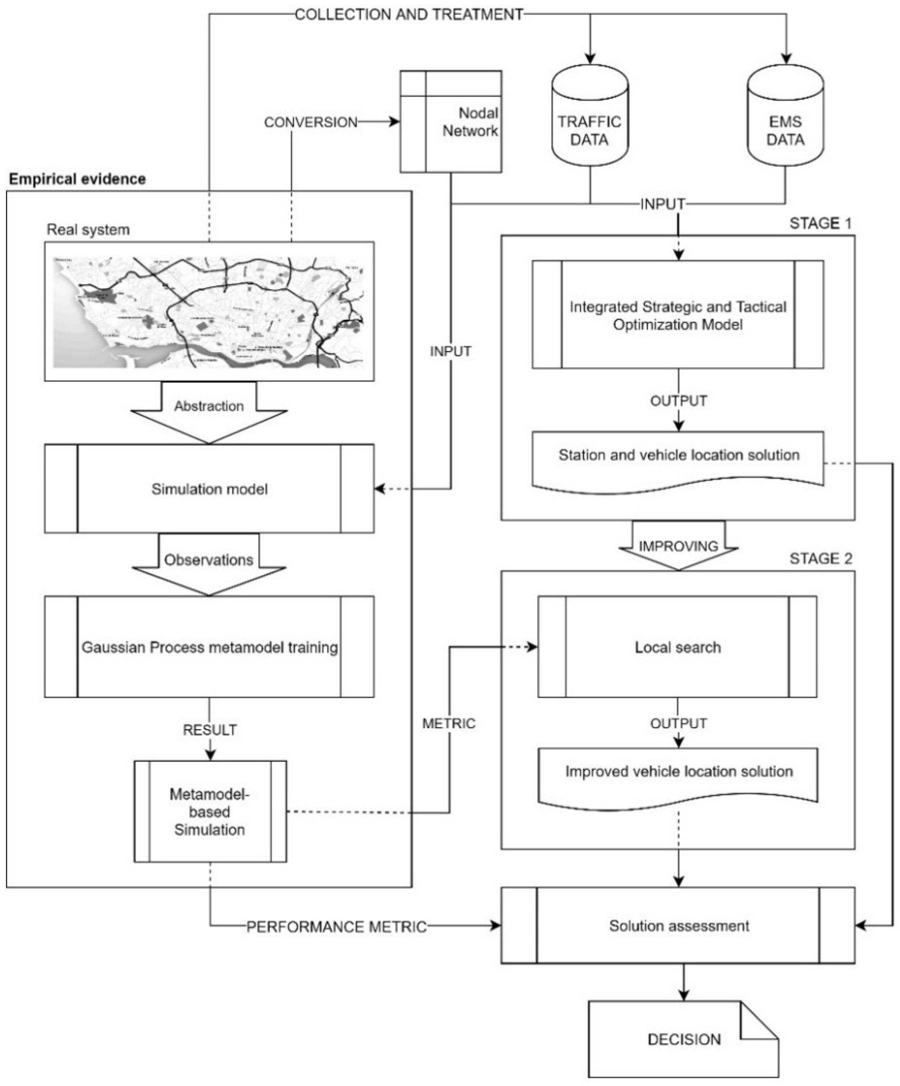


Figure 5.1: Diagram of the integrated optimization and simulation metamodeling framework.

The next sections will further detail the optimization model, the simulation model to train the metamodel, and the metamodel formulation with the respective algorithm for the local search.

5.2.1. Strategic and Tactical Decisions - Integrated Optimization Model

The objective of the Strategic and Tactical Integrated Model (STIM) is to provide a tool to combine the strategic and tactical planning of emergency vehicle and station locations and offer the first analytical solution.

The first important step in the solution search is to define the performance metric to be optimized. As was previously discussed, there is a wide range of performance metrics that can be integrated into the objective function. These can focus on the operational performance or on the victims' outcomes. This work follows a survival approach, firstly introduced by Erkut et al. (2008), and afterward adopted by many other researchers (Amorim et al., 2018; Bandara et al., 2014; Knight et al., 2012; Mayorga et al., 2013; McCormack and Coates, 2015). This approach uses a survival function that, when combined with victims' heterogeneity, takes the form of

$$S_e = [1 + exp(C^e + m^e \times t_e)]^{-1}, \tag{5.1}$$

and the system performance, P^s, is computed via

$$P^s = \sum_{e \in E} S_e, \tag{5.2}$$

where e is the index of a type of event from the set of events type E, e.g. cardiac arrest or road injury, t is the response time to event e, and C^e and m^e are the survival function parameters for medical emergency type e.

To capture traffic and demand spatial and temporal heterogeneity, a multi-period scenario-based optimization is preferred, as seen in (Dibene et al., 2017). The proposed formulation derives directly from the strategic planning model for emergency vehicle station locations formulated by Amorim et al. (2016, 2019c) and is adapted to vehicle locations using the notion of station busy fraction; firstly introduced by Daskin (1983) and adapted in the works of Snyder and Daskin (2005) and Berg and Essen (2019). The formulation of the model uses $s \in S = [1, 2, \ldots, s]$ scenarios that discretize the traffic and demand continuous changes. Moreover, and for the sake of notation simplicity, allow the availability set to be defined as $A = \{(s, l, p) \mid r_{s,l,p} \leq R_{max}, \forall s \in S, \forall l \in L, \forall p \in P\}$, and the following subsets as $A^{s,p} = \{(s, p) \mid \forall (s, l, p) \in A\}$, $A^{s,l} = \{(s, l) \mid \forall (s, l, p) \in A\}$, and $A^l = \{l \mid \forall (s, l, p) \in A\}$. Now, adopting performance metric P^s as the objective to maximize and B as the maximum busy fraction of the first order of a station, the new scenario-based Strategic and Tactical decisions Integrated optimization Model (STIM) is formulated as follows:

$$\max \sum_{s \in S} \sum_{e \in E} \sum_{l \in L} \sum_{p \in P} y_{s,l,p} \times d_{s,p} \times [1 + exp(C^e + m^e \times t_{s,l,p})]^{-1} \tag{5.3}$$

subjected to

$$\sum_{l \in L} y_{s,l,p} = 1, \forall (s,p) \in A^{s,p} \tag{5.4}$$

$$y_{s,l,p} \leq x_l, \forall (s,l,p) \in A \tag{5.5}$$

$$\sum_{l \in L} x_l \leq L_{max} \tag{5.6}$$

$$\sum_{p \in P} y_{s,l,p} \times d_{s,p} \times N_s \leq B \times \frac{z_l}{b}, \forall (s,l) \in A^{s,l} \tag{5.7}$$

$$\sum_{l \in L} z_l \leq V_{max} \tag{5.8}$$

$$z_l \leq M_{max}, \forall l \in A^l \tag{5.9}$$

$$x_l \in \{0,1\}, \forall l \in A^l \tag{5.10}$$

$$y_{s,l,p} \in \{0,1\}, \forall (s,l,p) \in A \tag{5.11}$$

$$z_l \in \mathbb{N}_0, \forall l \in L \tag{5.12}$$

where the sets are

- $s \in S$ is the set of scenarios for the multi-period approach;
- $e \in E$ is the type of emergency;
- $l \in L$ is the set of possible station locations;
- $p \in P$ is the set of demand points;
- $a \in A$ is the availability set which consists of a set of tuples (s,l,p) that satisfy the maximum response time constrain, e.g. $r_{s,l,p} \leq R_{max}$. Set A can reduce the number of decision variables and constrains, hence reducing the problem size;

and the decision variables are

- $y_{s,l,p} = 1$ if a vehicle from station l serves node p during scenario period s, 0 otherwise;
- $x_l = 1$ if a station is located at possible station locations l, 0 otherwise;
- z_l is the number of vehicles deployed at station l.

Furthermore, the parameters are

- $d_{s,p}$ is the estimated demand during period s at demand node p;
- $r_{s,l,p}$ is the travel time from station location l to demand node p during scenario period s;
- L_{max} is the maximum number of open stations;
- N_s is the demand peak factor during period s;
- B is the busy fraction control parameters;
- b is the average time a vehicle is busy when responding to an emergency (a sum of the response and assistance time, travel time to the emergency service location and corresponding victim drop off time, return time to station and be ready for the next call);
- V_{max} is the maximum number of vehicles per station;
- M_{max} is the maximum number of emergency vehicles;
- R_{max} is the maximum response time allowed.

Function (5.3) maximizes the overall survival considering the demand at each node and at each period. (5.10)-(5.12) represent the decision variables and their corresponding domains. For logistic purposes, the constraints represented by inequations 5.6, 5.8 and 5.9 limit the total number of open stations, the total number of deployed vehicles, and the maximum number of vehicles at one station, respectively. Notice that, as a direct consequence of (5.9), $z_l \in \{0, 1, 2, \ldots, M_{max}\}, \forall l \in A^l$.

Equation 5.4 ensures that for each scenario period, only one station is allocated to each demand point and inequation 5.5 ensures that there is a vehicle deployed at that station. Additionally, remember that A represents the availability set, whose use drastically reduces the size of the problem by considering only the pairs (period scenario, station location, demand node) that respect the constraint $r_{s,l,p} \leq R_{max}$.

The parameter B varies between $[0, 1]$, reflecting the percentage of time, within a scenario, during which a certain station is busy and can no longer respond to any other emergency call, i.e., it sets the working load of a station and its vehicles. Therefore, inequation (5.7) is the contribution to the station location formulation proposed in (Amorim et al., 2016) and it adopts the concept of a busy fraction to control the number of vehicles deployed at each station to limit the busy fraction B of the service stations during a scenario period demand peak characterized by N_s. It allows the user to control the station or vehicle busy fraction and to have solutions tailored for specific failure rates, i.e., a solution that will suffice for a certain fraction of the analyzed period.

5.2.2. Simulation Model

A simulation model is retrieved from (Amorim et al., 2018) to generate data points for training the metamodel that will be used for alternative solution exploration. The adopted simulation model is an agent-based simulation that can be quickly adapted for the objective of this work.

The model translates each EMS stakeholder into independent agents that interact within a stochastic environment. The main agent representing the city acts as the EMS provider by answering emergency calls and assigning a response vehicle to them. Each medical emergency is formulated as an agent with certain characteristics that define its type, location, and timestamp. Each vehicle is also an agent that answers to the city agent and has the autonomy to travel to the city, assist and pick up patients and transport them to the desired hospital.

The environment abstracts the complexity of the road network by considering a nodal network. Nodes are represented by a coordinate system, and each pair of nodes has a directional multi-period travel time associated. This means that traveling from A to B is not the same as traveling from B to A, and traveling from A to B during period s is not the same as traveling from A to B during period s'.

The advantage of the simulation is to have a stochastic representation that resembles the real system it tries to mimic. To meet the objective of this study, two uncertainties are studied: 1) traffic behavior and how it affects travel times, and 2) the EMS demand and how it fluctuates during the day. Two parameters are implemented to control these two behaviors. The first parameter ε_T captures the traffic stochasticity by introducing an "error term", when dispatching a vehicle, in the observed travel time $r_{s,l,p}$ according to

$$r'_{s,l,p} = r_{s,l,p} + r_{s,l,p} \times f\left[\mathcal{N}\left(\nu, \frac{\varepsilon_T^2}{4}\right)\right],$$

where $f\left(\mathcal{N}(\nu,\sigma^2)\right)$ is a function that returns a random value from a Gaussian distribution with mean ν and standard deviation σ. Here, the mean is set to zero, and the standard deviation is set to $\frac{\varepsilon_T}{2}$, meaning that there is a confidence of 95% that the estimated travel time has at most an error of $\varepsilon \in [0,1]$.

The second parameter, w_1, captures the possible randomness in the demand by reallocating the medical emergency event locations, e_p, through the network with a probability of w_1, described by

$$e'_p = p(g(E_s), e_p, w_1, w_2),$$

where $g(E_s)$ is a function that randomly picks a location from a bag of possible locations, E_s, weighted according to period s, and $p(e_1, e_2, w_1, w_2)$ is a function that selects either e_1 or e_2 with a probability of, respectively, w_1 and w_2, where $w_1 + w_2 = 1$.

These two parameters, ε_T and w_1, allow a controlled integration of the randomness that is observed in a real environment by manipulating the analytical data fed to the simulation model. To control the simulation output, two hyperparameters are also implemented: the number of runs, n^r, which controls the number of times the same input and parameters are evaluated by the simulation model, and the number of simulated days, nd, which controls the number of days the model will simulate to produce each output-solution performance. The simulation output uses equation (5.2) to calculate the average system performance during all runs n^r, additionally returning the average response time for all the calls served.

5.2.3. Metamodel-based Simulation and Local Search

The time-consuming problem of a simulation model to produce relevant outputs invalidates the possibility to test big batches of inputs and different combinations of parameters. A metamodel can overcome this drawback and can be used to quickly explore alternative solutions during strategic and tactical EMS decision-making processes.

A metamodel-based simulation using Gaussian Processes is proposed. By definition, and following Rasmussen and Williams (2006), a Gaussian Process (GP) is a stochastic process in which any finite number of random variables forms a multivariate Gaussian distribution. Each GP is solely defined by a mean function and a covariance function (or kernel), respectively denoted as $m_f(x)$ and $k_f(x, x')$, where x and x'

are two different input observations. The *GP* framework is a well-known and widely applied machine learning tool in a variety of research areas, being a recurrent tool for both regression and classification problems, especially within the machine learning community (Christopher, 2016; Li and Chen, 2016; Lifshits, 2012; Robert, 2014).

In terms of notation, a GP is commonly denoted by $\mathcal{GP}(m_f(x), k_f(x,x'))$. Within the standard regression problem $y = f(x)$, where $\varepsilon \sim \mathcal{N}(0, \sigma^2)$, this framework places a prior over $f(x)$, i.e.

$$f(x) \sim \mathcal{GP}(m_f(x), k_f(x,x')).$$

It is usual for the mean and covariance functions to have a certain number of free parameters. These parameters, also called hyperparameter of the GP, can be estimated by the maximization of the marginal likelihood. After this fitting procedure, the conditional distribution for an unobserved data point x^* is given by

$$f_*|X, \mathbf{y}, \mathbf{x}_* \sim \mathcal{N}(k_{f*}^\top [K_y]^{-1}\mathbf{y}, k_{f**} - k_{f*}^\top [K_y]^{-1} k_{f*}),$$

where $k_{f*} = k_f(X, \mathbf{x}_*)$, $k_{f**} = k_f(\mathbf{x}_*, \mathbf{x}_*)$, X and K_y are the design and covariance matrices respectively, and $\mathbf{y}$ is the vector of the target values. Hence, notice that the GP makes predictions in the form of Gaussian distributions.

The local search is then accomplished through an algorithm that generates all possible combinations C_{k^n} where n is the possible vehicle locations (the maximum number of vehicles per each available station), and k is the total number of vehicles. The exploration space is generated using *Python itertools* library and tested using the aforementioned metamodel-based simulation, and the solution with the maximum survival output is chosen.

5.3. Case Study Application

The proposed methodology and claims of this work are validated using real data from the city of Porto, collected from the INEM (National Medical Emergency Institute) database, consisting of 35k emergency medical calls between 2012 and 2013 (see Figure 5.2).

The data was processed using Google maps geocoding API to translate addresses into coordinates and stored in a SQL database. The national census database subsection division was used to build a 90-node network. Using a clustering algorithm

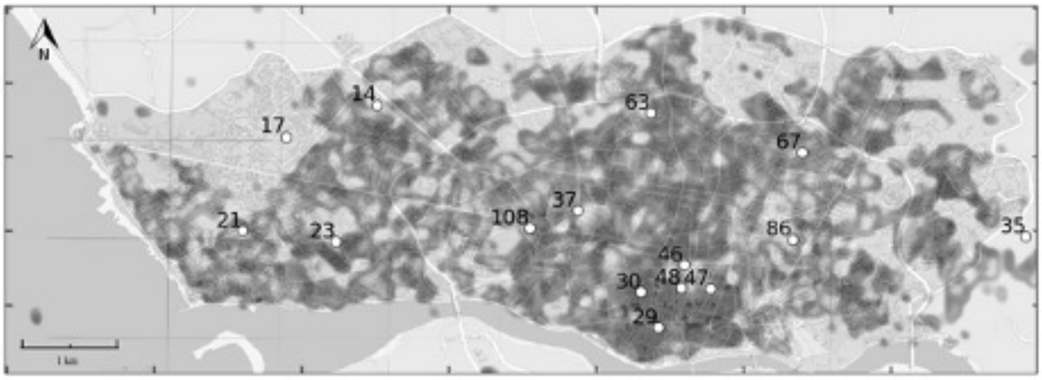


Figure 5.2: City of Porto, Portugal, and possible station locations with corresponding label. Darker areas represent a higher number of emergency calls.

along with a Euclidean distance metric, the medical emergency calls were allocated to the network nodes. From the 90 network nodes, 15 were chosen as possible station locations.

Five scenarios are set corresponding to different periods of the day and days of the week. The working days were separated from weekend days. For the working days, the day was divided into three periods: from 6:00 am to 2:00 pm, from 2:00 pm to 10:00 pm, and from 10:00 pm to 6:00 am. The weekend days were divided into two periods: from 6:00 am to 10:00 pm and from 10:00 pm to 6:00 am. These periods were defined after carefully studying the traffic and demand temporal and spatial fluctuations. Figure 5.3 shows an overview of the traffic and demand study where it is possible to observe the different behaviors of traffic and demand during the morning, afternoon, and night periods. During the weekend, the morning and afternoon period differences were less visible. Thus, these two periods were merged together.

Part of the optimization model, the parameters were calculated according to the resulting database, which are the demand parameters $d_{s,p}$, the travel time matrix with entries $r_{s,l,p}$, and the demand peak factors N_s.

The maximum number of stations, maximum number of emergency vehicles and maximum number of vehicles per station values were set to $L_{max} = 7$, $M_{max} = 14$, $S_{max} = 3$, respectively. From the authors' experience (Amorim et al., 2019c), these are the values that allow for a service with no significant delays, i.e., there is always an available vehicle to respond to a call. To reduce the problem size, the maximum response time was set to $R_{max} = 12$ min.

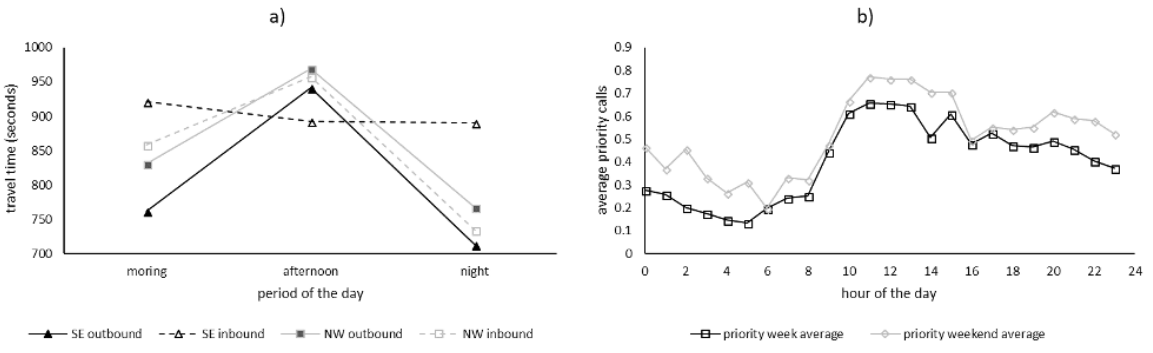


Figure 5.3: Dynamism in travel time and demand during the day.

Finally, the average time a vehicle is busy was estimated as $b = 120$ min, and the busy fraction parameter was set to $B = 1$. This is the limit of feasibility since it tolerates a 100% rate of vehicle usage during the peak period characterized by N_s. With the optimization model solution, the local search algorithm is applied to find an empirically better solution.

5.4. Metamodel-based Simulation Training

To train the metamodel, a batch of 800 simulation runs is used as observations. These observations consist of an input vector with the position of emergency medical vehicles and the stochastic parameters ε and w_1, and the corresponding average survival output.

The set of input vectors was generated by constraining the number of stations between 7 and 10, the total number of vehicles between 11 and 15, and the maximum number of vehicles per station to 3. After several optimization models tested in previous studies, 15 possible station locations were considered to generate the simulation input.

An algorithm randomizes a feasible solution according to the above restrictions, then randomly picks a value for the parameters ε and w_1, and inputs it in the simulation model with hyperparameters $n^r = 20$ and $n^d = 15$. The input vector and the outputs are stored in a database to be used to train the Gaussian Process metamodel.

As mentioned earlier, any GP is fully characterized by its mean and covariance functions. For this study, the constant mean function defined over the average of the simulation output variables values was set. Consider the training set, (X, y), where X is the design matrix composed of the independent variables (regressors), and y is the

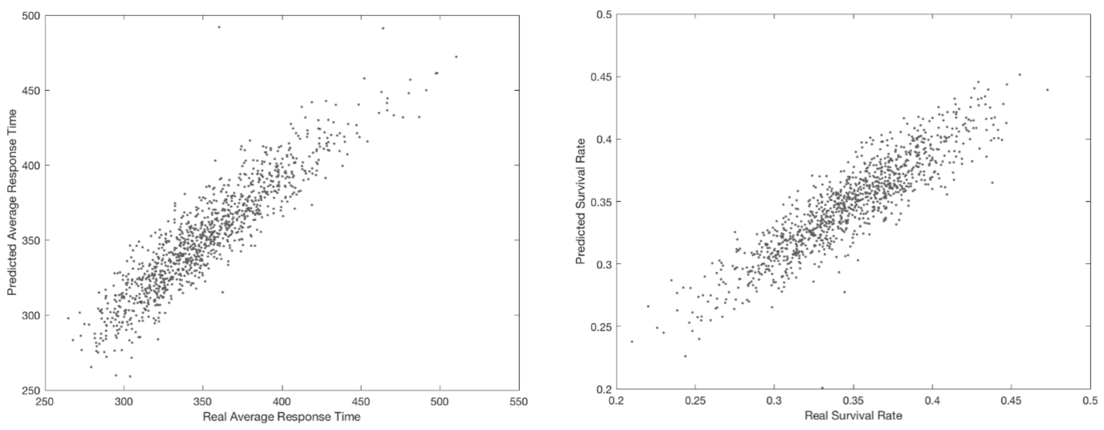


Figure 5.4: Predicted values using the metamodel versus real values for the average response time (left) and the survival (right) outputs.

vector containing the values of the dependent variable. Thus, the mean function comes as $m_f(x) = \text{average}(y)$. On the other hand, the well-known Squared Exponential, defined by $k_f(x, x') = \sigma^2 exp\left(-\frac{||x-x'||^2}{2l^2}\right)$ was chosen to serve as the GP's kernel. Here, σ^2 and l^2 are the signal variance and length-scale, respectively.

The GP was trained using a data set comprised of 1,098 random simulation results. Each observation has 17 dimensions in total, namely, the location change probability w_1, traffic error ε_T, and 15 locations. The training process was applied for both simulation outputs, average response time, and average survival rate independently. The hyperparameters were obtained using marginal likelihood optimization over this training data set. The predictive performance of the metamodel is depicted in Figure 5.4, where a strong linear correlation between the predicted and real values can be graphically observed. The validation of the metamodel generalization capacity was conducted using a 10-fold cross-validation scheme across 30 randomized runs for statistical significance purposes. The obtained results were averaged and summarized in Table 5.1. Several well-known performance metrics were used, namely, Pearson's linear correlation coefficient (Cor), Root-Mean-Square Error (RMSE), Mean Absolute Error (MAE), Relative Absolute Error (RAE), and finally Root Relative Squared Error (RRSE).

The metamodel achieved a linear correlation (Cor) of 0.912 for the average response time and 0.898 for the survival rate. This complements the information already provided in Figure 5.4. In terms of RMSE, the GP achieved values of 16.05 and 0.018,

Table 5.1: Average performance results of the metamodel resulting from a 10-fold cross-validation across 30 randomized experiments.

Output	Cor	RMSE	MAE	RAE	RRSE
response time	0.912	16.050	12.398	0.397	0.409
survival rate	0.898	0.018	0.014	0.426	0.441

whereas the MAE slightly decreased to 12.398 and 0.014 for the same output variables, respectively. In line with the correlation values, the metamodel attained too slightly better results with respect to RAE and RRSE for the response time output. Note that only Cor, RAE, and RRSE are directly comparable as they constitute normalized measures. On the other hand, RMSE and MAE reflect the magnitude of the involved values. All in all, these results show a high predictive and generalization performance of the trained model.

5.5. Results and Discussion

5.5.1. Strategic and Tactical Planning: Integrated vs. Non-Integrated

A discussion on planning strategic and tactical decisions as a combined problem is led in (Van Essen et al., 2013). To contribute to this question, the methodology of this study is applied to define an integrated solution and afterward to compare it with a non-integrated solution.

First, the solution achieved solely through analytical optimization is analyzed. Using the model proposed in (Amorim et al., 2016), a station location solution was obtained. Then, this solution was used to restrict the station location in the current optimization model, and another solution for the vehicle location was calculated. This two-phased optimization process results in a non-integrated solution. For the integrated solution, the STIM optimization model was applied to produce a solution that selects station and vehicle locations simultaneously.

Afterward, the local search algorithm was applied to each of the initial solutions resulting in two new solutions (with local search). The metamodel was used for assessing the various solution performance in terms of survival. For reference, and as a collateral consequence of station and vehicle locations, the average response time of

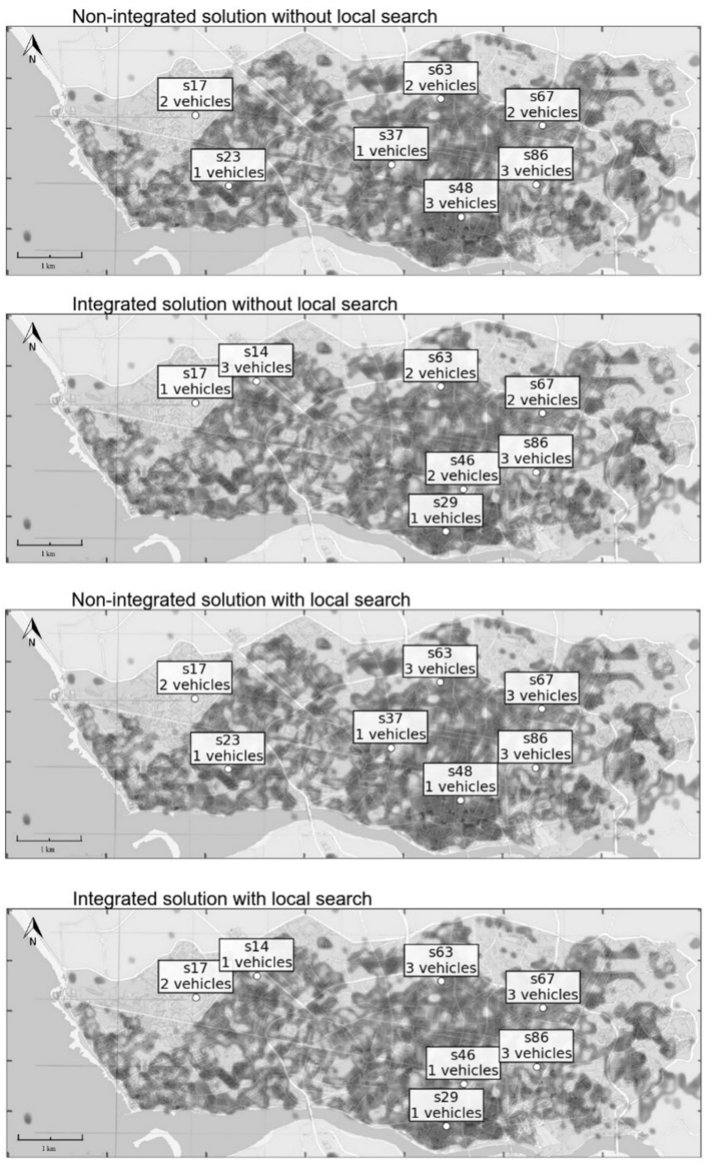


Figure 5.5: Visualization of the non-integrated and integrated solutions with and without local search.

each solution is calculated.

The solutions are represented in Figure 5.5 and the performance result for different values of the stochastic coefficient (with $\varepsilon_t = w_1$) are plotted in Figure 5.6.

It is obvious that the integrated solution outperforms the non-integrated solution. Nevertheless, higher survival comes at the cost of higher average response times. It is expected that in order to better respond to life-threatening events, emergency vehicles must be allocated closer to the nodes where these types of events dominate. Therefore, on the one hand, minimizing the average response time forces vehicle locations to be closer to nodes with high demand. On the other hand, maximizing survival pulls

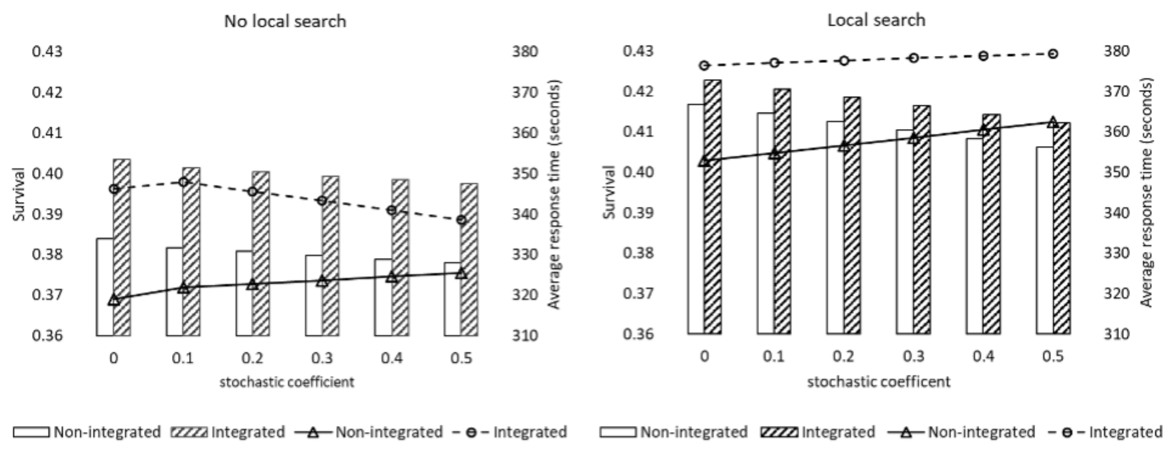


Figure 5.6: Performance of the integrated and non-integrated solutions with and without local search. Lines represent the average response time and the bars represent the survival metric.

vehicle location towards nodes where a higher likelihood of life-threatening emergency calls is present. The results show that increasing survival (which consists of reducing the response to life-threatening events) has a negative impact on the average response time (the response time for non-life-threatening events increases) because the number of non-life-threatening events dominates over life-threatening events. This shows that demand is not homogeneous (different types of medical emergencies have different temporal and spatial patterns) and that the use of survival functions as a performance metric allows decision-makers to optimize the outcome of a medical emergency rather than optimize the system operation.

The local search allows for solutions with better empirical performance as expected. When the stochastic coefficients increase, an obvious decrease in performance is visible, although this is not always true when it comes to the response time. It is worth noting that the integrated solution has a lower performance variation when the stochastic coefficients increase. This might indicate that this solution is more robust, thus adapting better in uncertain environments.

5.5.2. Improving Solution And Scenario Analysis

This section briefly explores the capabilities of the methodology. The dilemma of adding more vehicles at the cost of reducing the number of stations, and the inverse, is analyzed.

A set of five solutions are considered:

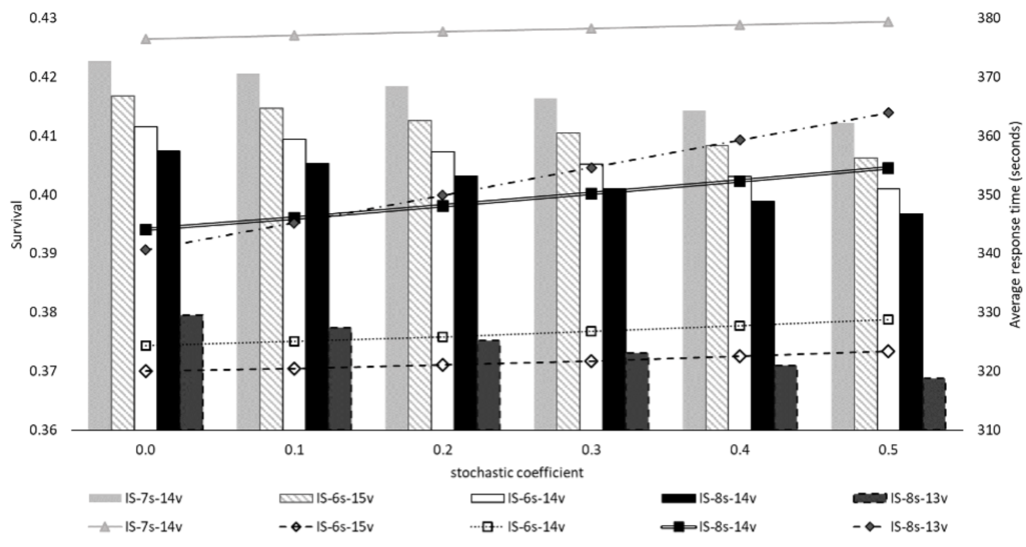


Figure 5.7: Performance of the different tested solutions. The darker bars represent the most expensive solutions. Lines represent the average response time and the bars represent the survival metric.

- IS-7s-14v, Integrated solution with 7 stations and 14 vehicles
- IS-6s-14v, Integrated solution with 6 stations and 14 vehicles
- IS-6s-15v, Integrated solution with 6 stations and 15 vehicles
- IS-8s-14v, Integrated solution with 8 stations and 14 vehicles
- IS-8s-13v, Integrated solution with 8 stations and 13 vehicles

The same previous parameters are used, and the full methodology is applied, i.e., optimization solution followed by a local search followed by a performance assessment. However, for the solution IS-8s-13v the busy fraction parameter B had to be raised to 1.1 so that the optimization model could find a feasible solution. The results are compiled in Figure 5.7.

From the tested solutions, it can be concluded that having seven stations is more advantageous than one less or one more station. When there is an additional vehicle available in the solution with six stations, the solution with seven stations is preferable. Curiously, adding a new station does not lead to higher performances.

In terms of the average response time performance, most of the solutions seem to have smaller variances as the stochastic coefficients increase. Nevertheless, it is interesting to note that the solutions with eight stations are less stable. Having more stations, it is most likely that when dispatching a vehicle, an error in travel time measures leads to the selection of a non-optimal vehicle. The dispersion of stations makes the solution more susceptible to traffic error. If vehicle cost<station cost, then in terms of economical preference IS-6s-14v>IS-6s-15v>IS-7s-14v>IS-8s-13v> IS-8s-14v, it can be concluded that the solutions with six and seven stations are preferable.

5.6. Conclusions

The study of strategic and tactical planning for station and vehicle locations in EMS has attracted much attention in recent years. With the availability of big data sets and high computer resources, the classical approach for EMS planning becomes obsolete as analytical formulations fail to capture important components of such systems, e.g., traffic changes, demand variability.

This study proposed a framework to tackle strategic and tactical decisions in a way that allows the use of stochastic variables and incorporates the complexity of an EMS system and the urban area it serves.

An optimization model was proposed that can be solved within a reasonable time and produces solutions adapted to the continuous demand and traffic changes during the day. To complement the optimization model, a metamodel-based simulation was also proposed. It allows the evaluation of solutions based on a simulation model that mimics a real system and the environment it is implemented in. These two tools allow for a chained methodology to plan EMS decisions with robustness and flexibility, ensuring decision-makers of the expected performance.

The methodology applied in the city of Porto contributed to the discussion between integrated and non-integrated strategic and tactical decision planning. It was shown, through the metamodel, that the solution performance of a non-integrated approach leads to lower system performance when compared to an integrated solution produced by the proposed methodology.

Furthermore, the methodology was also applied to a station versus vehicle dilemma and was shown to be a practical tool to support decisions or to explore alternatives in uncertain scenarios. The analytical formulation of the metamodel presents a simple yet robust performance function that can be easily used by decision-makers or by designers to quickly prototype system improvements. The results show that, in practice, an increase in vehicles is preferable to an increase in stations.

As both studies illustrate, better average response times do not always lead to higher survival rates. This is an important finding supporting survival metrics over classical operational ones.

For future research, the study of metaheuristics or mathematical formulation is suggested to allow direct use of the metamodel function as the objective function of an optimization model for EMS. Implementing robustness measures, such as the variance obtained during the simulation runs, is also worth investigating as this will provide more robust metrics when exploring or evaluating solutions.

Moreover, there are important points to highlight when measuring and addressing real systems or when supporting decision-makers. In a real system, the available resources might not always be enough to respond to every active call. In these cases, a triage takes place in order to select which calls will be answered first. Queuing rules should be developed and implemented in the simulation model. Furthermore, balancing the mix of patients' acuity and ensuring that all patients have acceptable levels of service is a key challenge in EMS. It is important to develop robust rules, which will consider survival and will balance service level. Some initial work regarding this challenge is present in (Amorim et al., 2019c; Jagtenberg et al., 2017b; Knoops and Lundgren, 2016).

Two EMS-specific events should be considered during solution assessment. These are the EMS hospital department overcrowding which will divert EMS vehicles to different hospitals, thus increasing the vehicle mission time, and no-service calls which corresponded to responded emergency calls but that at the end, the emergency vehicle is unable to find a patient on arrival at the incident location (no-service).

As a final remark, it is important that multidisciplinary teams (e.g., researchers in the area of emergency medicine and operational research) work together to develop more robust performance metrics that can provide decision-makers with a clear picture of the improvements in new computational models (e.g., like the one presented in this work) can bring when implemented in practice.

Chapter 6

Directional Active Learning Metamodeling

"Whenever a theory appears to you as the only possible one, take this as a sign that you have neither understood the theory nor the problem which it was intended to solve."

Karl Popper, 1902 - 1994

Emergency Medical Services (EMS) constitute a crucial pillar of today's cities by providing urgent medical responses to their citizens. Their study is often conducted via simulation modeling, as the assessment of planning and policy solutions are generally not feasible in the real world. However, these models can become computationally expensive to run. Thus simulation metamodels can be used to approximate the simulation results.

This chapter presents a metamodeling strategy supported by a directional active learning scheme to explore an EMS simulator's survival outcome. Using a series of training grids, the algorithm guides the exploration process towards simulation input regions whose output results match a specific search value defined a priori, providing a computationally efficient way of exploring the associated input space. The results show that this approach can identify such relevant simulation input regions and that it can be easily expanded to multiple search value approaches.

6.1. Introduction

In 2018, more than 55% of the world's population was estimated to live in cities or urban areas. The following decade's projection is that this number reaches 60% and that one-third of the people are likely to live in cities with around five thousand citizens or more (United Nations, 2018). The exponential growth of the urbanized areas worldwide will generate increased pressure on the urban and transportation infrastructure and the need for new ones. When not planned, it can lead to undesirable transformations. The potentially negative consequences of these changes can be minimized and handled if proper technical and policy decisions are taken, in the present, with a strategic planning view for the inevitable steady urban growth (Martine et al., 2007; Weber and Puissant, 2003; Levy, 2016).

As the population density increases within urban settlements, so does too the demand for mobility. Transportation systems play a particularly vital role: they are the backbone that supports sophisticated social and economic advancements. A healthy economic activity inevitably requires the movement of people, goods, and services (Moore and Pulidindi, 2013). Urbanization generally follows the evolution of transportation (Rodrigue et al., 2016). Moreover, with this intense urbanization and agglomeration of people, the concerns for health and safety conditions increase accordingly (Vlahov and Galea, 2002). To this end, Emergency Medical Services (EMS) constitute a crucial pillar of today's cities by providing urgent medical responses to their citizens, particularly those in serious life-threatening situations. In (Bélanger et al., 2019), a review of the last ten years of EMS research is provided, focusing on the current state-of-the-art modeling tools and projecting future trends whereas Andersson et al. (2020) presents three concrete managerial cases of EMS in the scope of strategic decision support systems. On the other hand, Poulton et al. (2019) addresses the problem of optimizing ambulance routing to decrease response times to life-threatening events under blue light conditions.

EMS performance is highly dependent on the characteristics and idiosyncrasies of the urban environment in which they are integrated, as well as on the information available to the service operator. Especially in highly populated areas, such systems' emergency response is inevitably constrained by multidimensional urban dynamics. Heavy traffic and population movements have a particularly relevant and direct impact on the EMS's outcome (Amorim et al., 2019b). To address the potential hindrances in the implementation and deployment of emergency services, strategic, tactical, and operational planning must be undertaken to ensure the proper emergency response of people in need of medical assistance.

The study of EMSs is often conducted via simulation-based modeling, as the assessment of planning decisions, as well as policy solutions, is generally not feasible in the actual real-world systems. However, despite their obvious advantages, simulation models can become computationally expensive to run whenever a high degree of realism and detail is required (Law, 2015). To address this shortcoming, simulation metamodels (Friedman, 2012) can be employed to approximate and explore simulation results, thereby mimicking the simulator itself (Kleijnen, 1979, 1987). To adequately serve their purpose, simulation metamodels should be defined as fast and straightforward functions whose input/output domains match that of the original simulation. Kleijnen and Sargent (2000) mentions four main goals within simulation metamodeling, namely, real-world problem understanding, simulation output prediction, optimization, and verification/validation. In this work, we mostly focus on the first two objectives.

Consequently, it is assumed that the simulation model is perfectly calibrated concerning the entity problem under study. Within this context of expensive simulation models, in which labeled data is difficult to obtain, active learning emerges as a particularly relevant modeling approach that aims to increase a model's predictive performance with as few training points as possible. This is essentially accomplished by designing algorithms with the ability to choose the training points from which they learn in an iterative manner (Settles, 2010; Wang and Zhai, 2016).

Closely related to both concepts of simulation metamodeling and active learning is the Gaussian Process (GP) framework. Due to its Bayesian properties, it provides predictions in the form of Gaussian probability distributions, with specified mean and variance. This predictive variance, as an information criterion, can be easily used to develop active learning strategies to seek the most informative points within a simulation input space. On the other hand, GPs have a long tradition of being used as metamodels (Kleijnen, 2009; Chilès and Desassis, 2018), due to their relatively simple formulation, high non-linear modeling capabilities, and general modeling versatility. The GP-based metamodeling framework is oftentimes called Kriging.

In this work, by combining elements of both simulation metamodeling and active learning, a strategy is proposed to study the survival outcome of an EMS simulator by exploring its input space. Employing a Gaussian Process (GP) as a metamodel, the methodology guides the exploration process in the direction of simulation input regions whose output results match a specific search value defined by the user in advance.

This is achieved by a series of training grids and sub-grids that get iteratively closer to the input regions of interest. The results show that this approach can identify such relevant simulation input regions while minimizing the computational workload at the same time, making the exploration process more efficient. From the discussion of these results, we draw several practical implications for the field of EMS and possible lines of improvements in a near-future work.

6.2. Methodological Approach

6.2.1. Gaussian Processes

A GP is a stochastic process in which any finite number of random variables forms a multivariate Gaussian distribution (Rasmussen and Williams, 2006). Any GP is completely defined by a pair of functions, namely, the mean, $m_f(\mathbf{x})$, and the covariance, $k_f(\mathbf{x}, \mathbf{x}')$, where $\mathbf{x}$ and $\mathbf{x}'$ are two different input observations, and is commonly denoted by $\mathcal{GP}(m_f(\mathbf{x}), k_f(\mathbf{x}, \mathbf{x}'))$. Within a standard regression problem $y = f(\mathbf{x}) + \epsilon$, where $\epsilon \sim \mathcal{N}(0, \sigma^2)$, the GP places a prior over $f(\mathbf{x})$, i.e., $f(\mathbf{x}) \sim \mathcal{GP}(m_f(\mathbf{x}), k_f(\mathbf{x}, \mathbf{x}'))$.

Is it usual for the mean and covariance functions to have a certain number of free parameters which can be optimized via marginal likelihood maximization. The conditional distribution for an unobserved data point $\mathbf{x}_*$ is given by $f_*|X, \mathbf{y}, \mathbf{x}_* \sim \mathcal{N}(k_f(X, \mathbf{x}_*)^\top [K_y]^{-1}\mathbf{y}, k_f(\mathbf{x}_*, \mathbf{x}_*) - k_f(X, \mathbf{x}_*)^\top [K_y]^{-1} k_f(X, \mathbf{x}_*))$, where K_y is the covariance matrix, X is the design matrix composed by the independent variables and $\mathbf{y}$ is the vector containing the values of the dependent variable. Each GP prediction is associated with a well-defined Gaussian distribution. Hence, instead of a single point-wise estimate, the GP provides an entire range of possible values, weighed by a density function. This is an intrinsic Bayesian property which allows the GP to encode the uncertainty associated not only to its estimates but also to the underlying process being studied. The predictive variance $k_f(\mathbf{x}_*, \mathbf{x}_*) - k_f(X, \mathbf{x}_*)^\top [K_y]^{-1} k_f(X, \mathbf{x}_*)$ can be viewed as a proxy for both data and prediction uncertainties. Higher variance translates into more dispersion, which in turn implies more variability around the mean. This property allows straightforward implementations of active learning schemes, by setting, for example, the predictive variance as an information criterion.

6.2.2. Active Learning Strategy

The strategy followed in this work combines crucial elements of both active learning and simulation metamodeling, similar to that of seen in (Antunes et al., 2018) and (Antunes et al., 2019). Figure 6.1 briefly depicts the proposed methodology, whose general objective is to approximate the behavior of the simulation model under study around pre-defined output values of interest and corresponding triggering input region. This is achieved by using a GP that, while modeling the functional relationship between the simulation input and output spaces, ultimately guides the training process itself towards such simulation output value.

Three essential ingredients must be defined a priori to starting the algorithm. These are the pre-defined simulation output search value (sV) of interest, the initial training grid ($trGrid$), and the search grid ($sGrid$). The algorithm is designed to provide the user with a directional search towards the sets of input values or input regions that trigger, via simulation, output values that lie close to sV, by successive predictions over $sGrid$. It is worthwhile to mention that, contrary to sV and $sGrid$, which are both fixed structures, $trGrid$ evolves as part of the active learning scheme.

For the sake of simplicity, this work focuses on the two-dimensional case, although the proposed methodology also applies to high-dimensional spaces. The method is deemed as directional since the GP effectively directs the iterative metamodeling process towards the input regions that are more likely to assume output values closer to sV. This directionality is accomplished by considering a series of iterative training grids and corresponding sub-grids that approximate the simulation results in the vicinity of sV. These grids are sequentially added to the increasing training grid. The GP, used as a metamodel, guides the expansion process of $trGrid$ towards the identification of simulation input regions whose simulation results are close to the pre-defined output value of interest. This allows the user to concentrate the computational workload (associated with each simulation run) specifically in these input regions, turning the exploration process not only faster but more efficient.

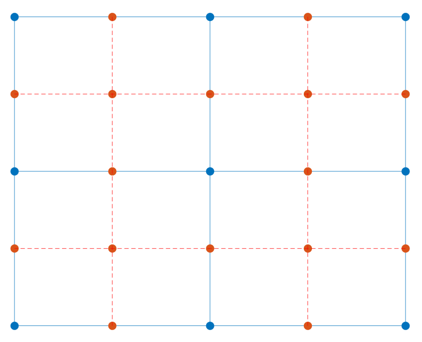
Figure 6.1(a) depicts an example of a training unit grid and its associated sub-grids. The unit grid is composed of four contiguous sub-grids, defined by a set of nine points arranged in a square shape. The same principles apply to any rectangular shape. Each vertex represents a Cartesian pair of values within the simulation input space whose corresponding simulation results (output values), *simR*, were previously computed. The training set denoted by $(trGrid, simR)$ is then a labeled data set. Furthermore, observe that the unit grid and sub-grid have several points in common, which could imply repeated simulation runs. Although this might prove to be useful for simulation models that exhibit a high degree of stochasticity, such scenarios were not considered in this particular work. As the iterative process advances, the sub-division of the successive grids continues recursively.

On the other hand, the search grid *sGrid*, besides being characterized by a static structure, has a different nature and purpose. It is only comprised of input data points for which the associated simulation results are not known. Thus, it is comprised of what is often called unlabeled simulation instances or observations, and it is exclusively used for prediction purposes. It should be highly dense so that the GP can predict over *sGrid* with increased great detail. Contrary to the training stage, predicting does not require much computational effort, and it is often a relatively fast procedure. This is valid not only for the GP framework but also for the vast majority of the machine learning tools.

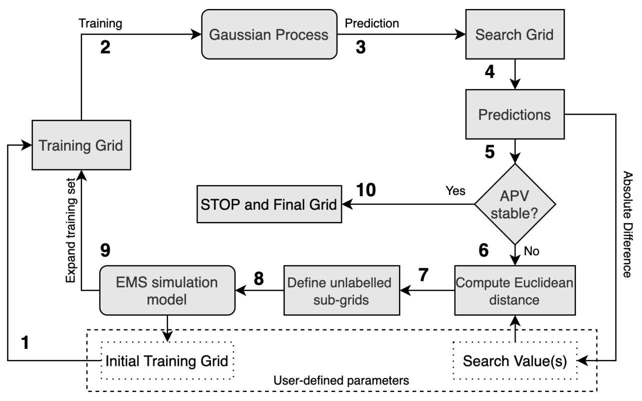
In practice, *sGrid* is the simulation input region in which the user aims to explore the simulation output behavior, in particular, to eventually locate the set of input values that trigger sV. Although not completely required, some prior knowledge regarding the simulation model is desirable when defining this grid. There is no point in defining a search grid in which the search value is never triggered. Hence, expert opinion and experience are of utmost importance in this stage so that *sGrid* encompasses some prior knowledge as to where sV is most likely to be triggered within the simulation input space.

Following Figure 6.1(b), the algorithm starts with the labeling process of the points contained in the initial training set. For each input point in this grid, a simulation experiment is run so that the corresponding simulation output result is obtained. The vector of all these output values is denoted by *simR*, and the associated data set by $(trGrid, simR)$. Notice that in this first iteration, *trGrid* matches the training unit depicted in blue previously seen in Figure 6.1(a).

Afterward, a GP is fitted to $(trGrid, simR)$, and predictions are made over the unlabeled set *sGrid*, effectively serving as a simulation metamodel. These predictions



(a)



(b)

Figure 6.1: (a) Grid-based training unit (blue) with corresponding sub-grids (red), and (b) proposed active learning methodology with 1-10 representing the iterative flows in chronological order. Notice that steps 2-9 are cyclic while the stopping criterion is not satisfied.

are estimates for the simulation output results associated with the points within the search grid. As mentioned in Section 6.2.1, the GP predicts in the form of a probability distribution, particularly providing the predictive variance associated with each prediction. This variance is then used to compute the Average Predictive Variance (APV) over $sGrid$, and its stability is accessed. For obvious reasons, this step is not applicable during the first iteration, as there are no previous iterations whose APV values can be compared to. Finally, sub-grids are defined, and some are selected to incorporate $trGrid$, naturally expanding it. This selection is conducted according to the predictions provided by the GP by computing the point-wise Euclidean distance between the GP surface and the nearest point comprising the horizontal plane defined by $z = sV$. Hence, only those sub-grids whose predicted values are closer to sV are added to the training grid.

Hereupon, the entire active learning process is iteratively repeated until APV shows no signs of significant variation between iterations. Observe that APV is viewed as a measure of the GP's overall prediction performance. As the training set is expanded by successive finer grids whose simulation results lie closer to sV, APV is expected to decrease from iteration to iteration. This decrease should reach a lower bound, representing the point from which the GP is not capable of further improving its predictions. At this point, the simple addition of new points to the training set is a waste of time and computational resources. In the end, this methodology is designed to

```
   INPUTS: $sV$, $trGrid$, $sGrid$
   WHILE $APV$ not stable do
1:     Train a GP using $trGrid$ and obtain predictions over $sGrid$.
2:     Select unlabeled sub-grid whose prediction points are closest to $sV$.
3:     Run the simulator to obtain the output results for the selected sub-grid.
4:     Expand $trGrid$ with the newly obtained simulation results.
5:     Update $APV$.
   END
   OUTPUTS: Trained GP, $trGrid$.
```

Figure 6.2: Active learning strategy pseudo-algorithm.

provide a final and fine mesh grid that encloses the region contained in the simulation input space that specifically triggers the search value of interest. This approach is summarized in the form of a pseudo-algorithm in Figure 6.2.

6.3. Experimental Setting

6.3.1. EMS simulator

In this work, the EMS simulator developed by Amorim et al. (2018) was used as a case study. This simulation model implements the allocation and dispatching of emergency vehicles according to the closest idle vehicle rule (Haghani and Yang, 2007; Jagtenberg et al., 2017a; Yang et al., 2005). The underlying model relies on an agent-based simulation design where a city agent serves as the operator of the emergency medical service. After answering emergency calls, it allocates and dispatches idle vehicles to emergency locations. Each event is represented by an agent that encloses several characteristics that describe the type of medical emergency, location, and timestamp. Within the simulation, these events are set off according to a historical emergency call database and, as well as to the probability of location change, forwardly discussed in more detail. The vehicles are also formulated as agents that immediately react to the city agent requests.

Most EMS are still developed and managed by rudimentary and outdated vehicle dispatching and reallocation rules. Essential factors such as traffic conditions, roadblocks, vehicles' exact positions, and emergency demand predictions are not entirely taken into account during EMS planning. These and other time-dependent elements do have a fundamental impact on the emergency system's response (Schmid, 2012). However, since the beginning of the 21st century, the proliferation of ubiquitous intelligent technologies (Kindberg and Fox, 2002) and the development of the Internet of Things (IoT) (Li et al., 2015), has culminated in unprecedented data growth. With this information technology revolution, EMS staff, practitioners, and researchers, in addition to the traditional historical data, are now able to access, collect and process large amounts of heterogeneous real-time information, such as traffic congestion, satellite-based navigation, and emergency alerts.

Inputs

The mentioned EMS simulation model features three kinds of inputs, particularly the location change probability, traffic error, and station locations. Whereas the former two assume real values in the interval $[0, 1]$, the latter assumes discrete positive values, $[0, 1, 2, ...)$, additionally breaking down into 90 independent, although related, inputs dimensions. These inputs correspond to a 90-node emergency response network for possible station locations. For this work, 15 of these 90 possible locations were assigned to hold emergency stations. Figure 6.3 depicts the spatial distribution of the selected locations within the city of Porto. Here, darker areas correspond to higher values of emergency demand. This historical data consists of 35k emergency call records collected from the Portuguese Institute of Medical Emergencies (INEM) (Gomes et al., 2004) between 2012 and 2013. Table 6.1 summarizes the locations and the number of vehicles per station.

On the other hand, the location change probability and the traffic error inputs are designed to reflect the operating conditions of the EMS somehow, as well as to induce a degree of variability often present in real-world systems. The former is particularly useful to emulate the expected stochasticity of the emergency events' spatial distribution. High values of probability change mean that the emergency locations present in the historical data are more likely to change. This introduces unforeseen dynamics in the spatio-temporal distribution of the emergency events, forcing the simulated EMS to respond accordingly.

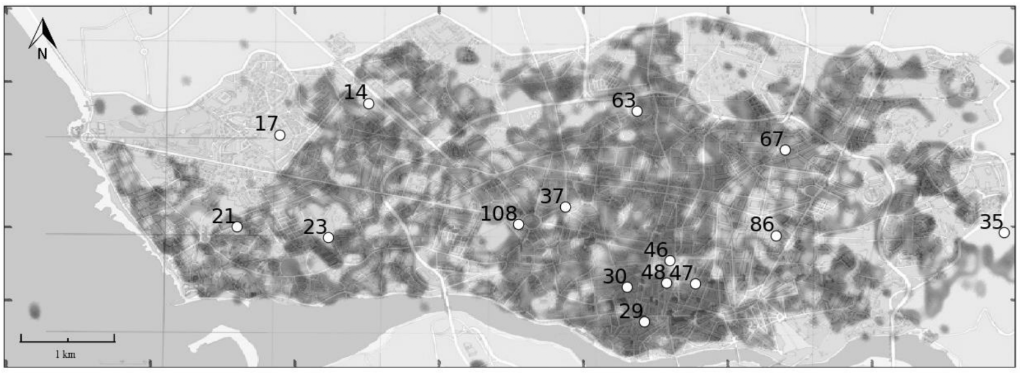


Figure 6.3: Locations of the 15 emergency vehicle stations in Porto, Portugal. Darker regions represent higher emergency risk.

With no surprise, the traffic error input relates to the road network conditions, although it does not directly affect the vehicle's travel times. Instead, it represents how traffic errors lead to bad operational decisions when dispatching vehicles. Consider the scenario where there is no traffic congestion, roadblocks, or any other factor that can increase travel times. For a given emergency event, the optimal dispatching decision would be to assign the closest idle vehicle. However, when heavy traffic is present (e.g., peak hours), the best dispatching decision is not trivial. Moreover, even with real traffic information, errors might occur, such as communication delays or data transmission problems.

The consideration of traffic errors instead of the traffic information itself allows measuring distances by travel times rather than in length units. In practice, the time it takes for a vehicle V to arrive at an emergency location E is far more crucial than the distance between its dispatching station location and E. Especially under severe traffic conditions, smaller distances do not imply reduced travel times. Therefore, traffic errors are considered in terms of time units and are taken into account to reflect the intrinsic dynamics of traffic congestion and other road network conditions, ultimately leading to more robust dispatching decisions.

Outputs

To evaluate the performance of the underlying EMS, two output metrics are considered in this simulation model, particularly the response time and survival rate. The vehicle response time measures how much time it takes for the medical staff to start

Table 6.1: Stations used and number of emergency vehicles per station.

location	14	17	21	23	29	30	35	7	46	47	48	63	67	86	108
vehicles	2	2	3	2	5	4	2	2	3	4	4	1	4	1	3

assisting the victim. As a single-point quality measure for the entire dispatching operation, the sum of all emergency events or their corresponding average can be considered.

As for the survival rate, it represents the victims' likelihood of survival, which will depend simultaneously on the response times and severity of the emergencies. Again, the sum or average over all the medical occurrences can be considered to assess the overall performance of the EMS. In this work, only the survival output is taken into consideration for modeling purposes.

6.3.2. Results

From a regression point-of-view, a GP is used to model the underlying functional relationship between the inputs, location change probability and traffic error, and the output survival rate. Two sets of experiments were considered. Whereas the first considers only one search value, the second illustrates the methodology for two search values. In both experiments, we considered the mean of the GP to be a function of the average of the training set, whereas the covariance was set to the widely known general-purpose Squared-Exponential with Automatic Relevance Determination (SE-ARD). We also used the GP implementation available from (Rasmussen and Williams, 2006). All the experiences were conducted in Matlab R2019b on a 2.7 GHz Quad-core Intel i7 processor with 16GB of RAM.

The goal of our methodology is to efficiently search for regions in the input simulation space that trigger a particular output value, or values, of interest. In the particular case of the mentioned EMS simulator, the proposed approach tries to identify sets of values defined within the space defined by the ranges of the probability change and traffic error inputs, $[0, 1] \times [0, 1]$, that trigger, via simulation, a user-specified output value lying in survival rate's range, $[0, 1]$. An active learning and metamodeling approach is adopted, so that exhausting and systematic computer experiments are avoided, thereby turning the exploration process of the EMS simulator's input space not only faster but more efficient.

As highlighted in Section 6.2.2, the proposed algorithm requires three components to be defined in advance, namely, sV, $trGrid$ and $sGrid$. The latter two are shared across the two sets of experiments. The initial training grid is set as the nine-point training unit depicted in blue in Figure 6.1(a), whereas the search grid is defined as a mesh grid of 10,000 unlabeled points uniformly distributed in $[0, 1]^2$. The edges of both squared grids match those of the input domain being explored. The search values varied according to the experiment and were arbitrarily set for the sake of demonstration.

Experiment 1

In this experiment, the active learning strategy was applied to identify the input region that is mostly associated with survival rate values close to 50%. Thus, 0.50 was set as the search value of interest, i.e., $sV = 0.50$. The results are summarily depicted in Figure 6.4 and 6.5.

As the initial training grid is comprised of a set of labeled data points, that is, input-output tuples resulting from a series of computer experiments, a GP is fitted to these simulation results. This first approximation, represented by a three-dimensional surface, can be observed in Figure 6.4(a), and it corresponds to the GP predictions made over $sGrid$. The same panel does show not only the surface contours but also the input points constituting $trGrid$. Hereafter, using the GP predictions, the algorithm searches for the best-suited sub-grids (recall the red grids in panel (a) from Figure 6.1) to advance with the learning process. The best candidates are associated with those GP predictions whose values are closer to 0.50. This selection requires the computation of the Euclidean distance between the GP approximation and the plane $z = 0.5$. Observing Figure 6.4(b), the sub-grid delimited by $[0.0, 0.5] \times [0.0, 0.5]$ was selected as the best candidate to be added to $trGrid$ for the following iteration. In other words, this means that it is most likely to contain simulation input values that trigger the search value 0.50.

The predictive variances related to each GP prediction are then averaged out over $sGrid$, and thus APV is computed. Finally, the selected sub-grid is labeled by the simulator. This is attained by running the simulation model with the input values associated with the points comprising the selected sub-grid. Naturally, each two-dimensional data point corresponds to a single simulation run.

The algorithm proceeds in an iterative manner, alternating between the GP training/fitting stage, the simulation runs, and the expansion of the training data. Figures 6.4(d)-(f) show the results from the last three iterations, as well as the final configuration stages of *trGrid*. The algorithm was able to detect that the series of sub-grid defined within $[0.0, 1.0] \times [0.625, 0.750]$ specifically trigger simulation output values close to 0.50. This input region approximately matches the intersection area between the plane z and the GP surface. On the other hand, Figures 6.5(a)-(h) show the sequence of the distances between the search value and surface provided by the GP. Dark color tones imply small distances. Here we can observe that the search procedure does not evolve linearly, not at least in the first iterations. As more points are added to the training grid, the GP reacts with different prediction behaviors until it eventually stabilizes. The obtained simulation input region of interest is visible in Figure 6.5(h), here depicted as a dark narrow horizontal band.

The GP's fitting performance within the input region identified by the algorithm is supported by the finer grid of training points enclosing it. Note that the sub-grids gradually added to the training grid get narrower as the process advances. Thus, *sGrid* expands towards the input regions of interest defined by sequentially and increasingly finer grids. This is not only expected but also the ultimate goal of the proposed methodology. The algorithm stops when the addition of a new training point does not improve the fitting of the GP, i.e. when APV starts not to vary significantly from iteration to iteration. Figure 6.5(i) shows the evolution of APV. The algorithm required 17 iterations to stop.

It is worthwhile to mention that, in this work, the main objective of the GP is not to serve as the perfect simulation metamodel. Although it should be, of course, a reasonably good approximation for the underlying simulation model, most importantly, it should guide the active learning algorithm towards the identification of the most informative input points with regards to the simulation output search value. These points, arranged in grids, lead the algorithm in the direction of relevant input regions while minimizing the computational workload. This ultimately turns the exploration process of such regions faster, as well as more efficient.

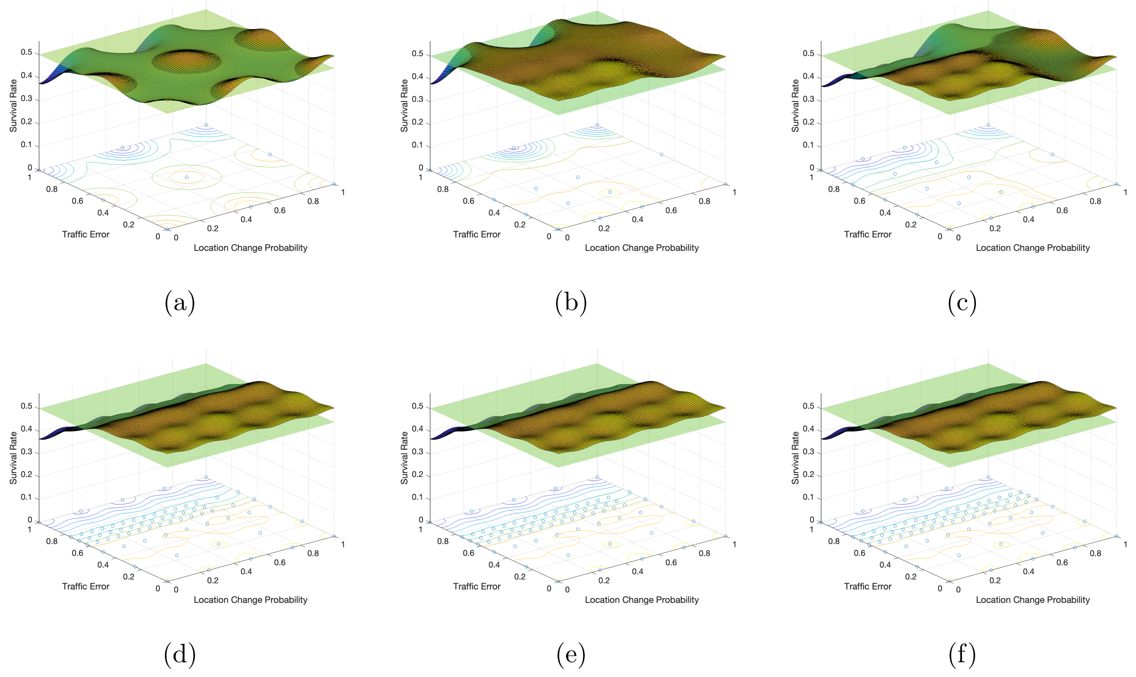


Figure 6.4: Iterative GP surface approximations and associated contour plots with sequential training grids and $sV = 0.50$. Panels (a)-(c) and (d)-(f) correspond to iterations 1-3 and 15-17, respectively. The flat horizontal surface is located a $z = 0.50$ (search value).

Experiment 2

In this experiment, the algorithm was extended to encompass two search values, namely, $sV_1 = 0.40$ and $sV_2 = 0.55$. Again, these values are arbitrarily set for the sake of illustration. Figures 6.6 and 6.7 summarize the results. The main goal of this experiment was to show that the proposed algorithm can be easily adapted to a multi-directional nature. As expected, the problem gets more challenging as the number of search values, input domain range, and dimensionality increase.

Following the same strategy, the algorithm starts from the simple nine-point grid (see Figure 6.6(a)), ending up with a mesh grid of training points, as seen in Figure 6.6(f). Within this final grid, two series of finer sub-grids clearly stand out from the rest, in particular the grids defined by $[0.0, 1.0] \times [0.875, 1.0]$ and $[0.0, 1.0] \times [0.375, 0.50]$, corresponding to the input regions that trigger sV_1 and sV_2, respectively.

In Figure 6.7, panels (a)-(h) show several non-consecutive steps in the evolution of the distance (or absolute difference) between the GP surface and each of the search values. Here, notice that, contrary to the input region associated with sV_1, which

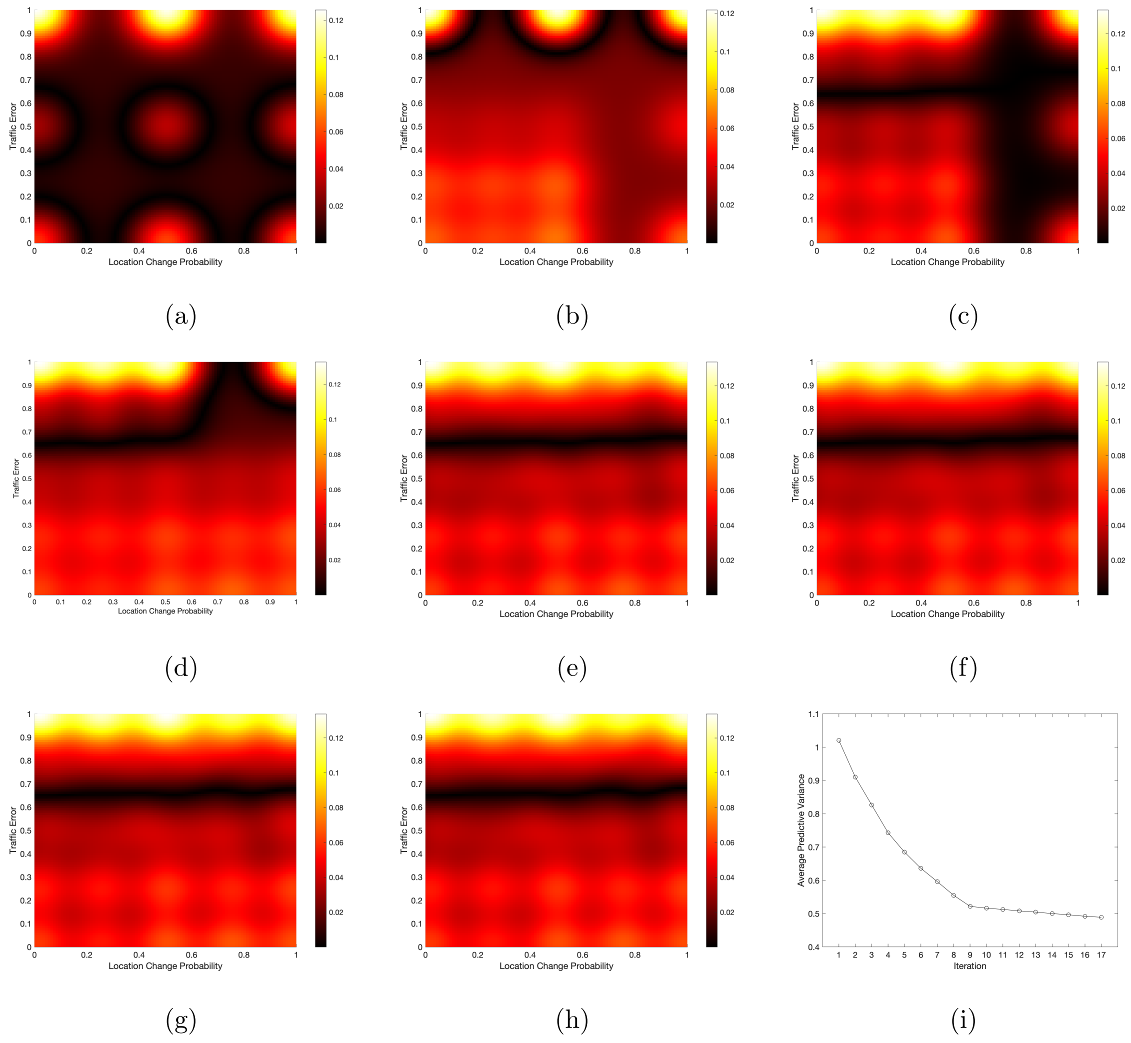


Figure 6.5: Absolute difference between the obtained GP surface approximations and $sV = 0.50$. Panels (a)-(d) and (e)-(h) correspond to iterations 1-4 and 14-17, respectively. Panel (i) depicts the evolution of the Average Predicted Variance.

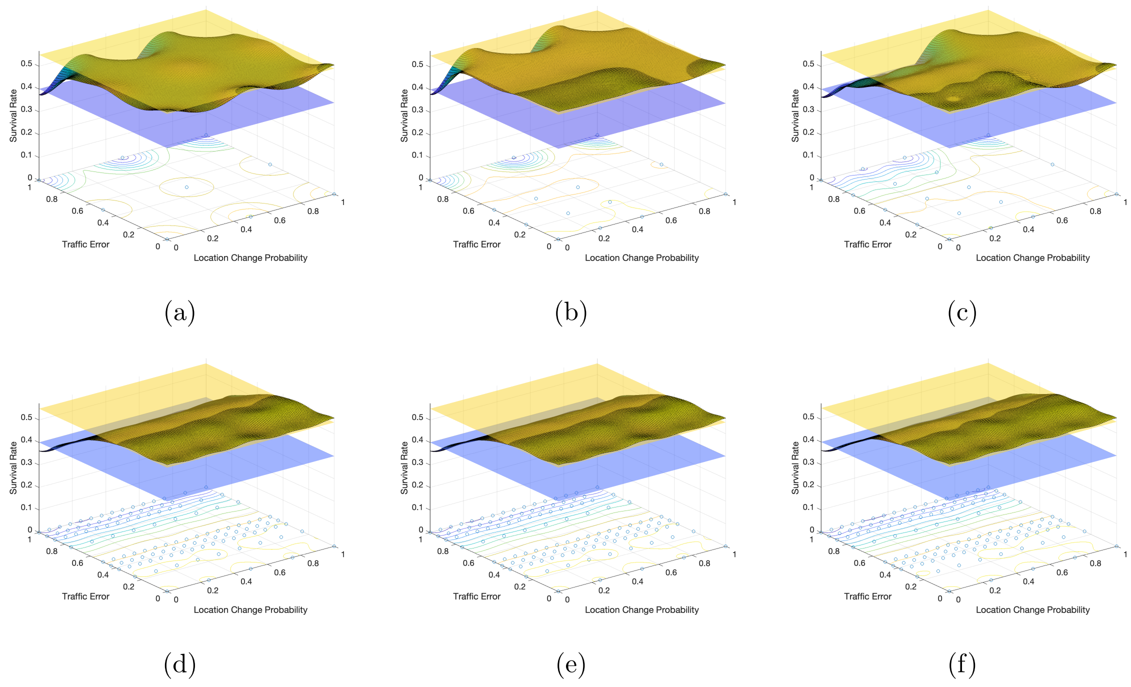


Figure 6.6: Iterative GP surface approximations and associated countour plots with sequential training grids with $sV_1 = 0.40$ and $sV_2 = 0.55$. Panels (a)-(c) and (d)-(f) correspond to iterations 1-3 and 27-29, respectively. The flat horizontal surfaces are located a $z_1 = 0.40$ and $z_2 = 0.55$, representing the two search values.

is highlighted by the dark tone horizontal strip seen in panel (g), the same does not hold for sV_2 (see panel (h)). In terms of distances, we observe that instead of a single narrow region, a wider band is also visible, roughly defined within $[0.0, 1.0] \times [0.0, 0.5]$. Although the algorithm was able to identify the input region that effectively triggers the output value of 0.55, it does not pay any particular attention to the surroundings of that value. Consider the scenario where the GP partially exhibits a near-flat surface parallel to the search value plane ($z_2 = 0.55$ in this case) and that the distance between the two is sufficiently low. Here, the algorithm will ignore most of the associated GP predictions, to the detriment of those near the intersection zone between both surfaces, despite the closeness to sV_2. This is what has occurred in this experiment, and it is observable, for instance, in Figure 6.6(f). On the other hand, the same did not happen to sV_1 since the GP surface does not present a flat shape parallel to z_1 in the vicinity of this search value, as depicted in the same figure. An improvement to address this kind of situation should be considered in the future. Figure 6.7(i) shows that this time, the algorithm took 29 iterations to stop, as well as the expected decrease of APV.

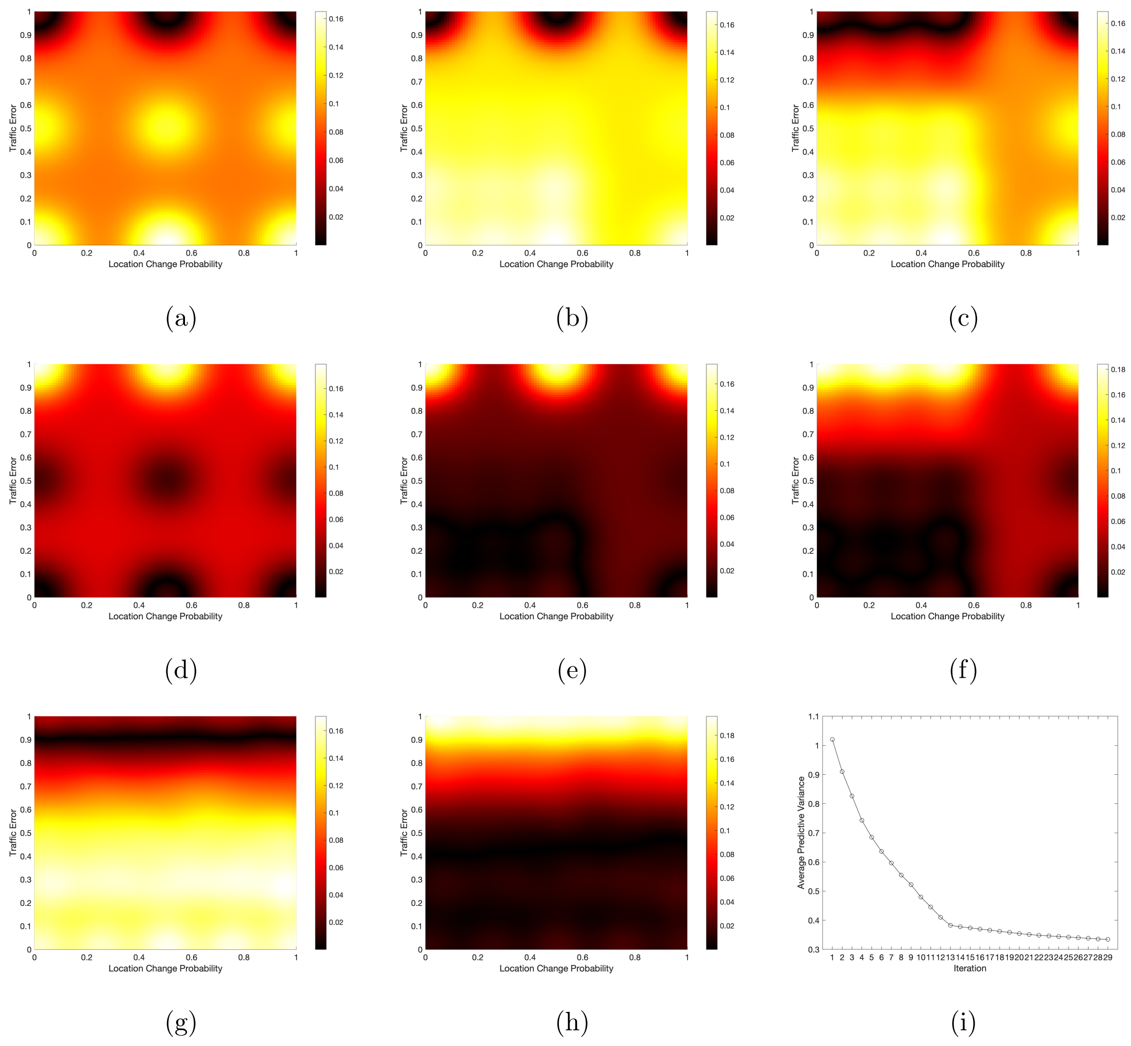


Figure 6.7: Absolute difference between the obtained GP surface approximations and the two search values. Iterations 1-3 are depicted in panels (a)-(c) and (d)-(f), respectively for $sV_1 = 0.40$ and $sV_2 = 0.55$. Panels (g) and (h) represent the two last iterations for each corresponding search value. Panel (i) depicts the evolution of the Average Predicted Variance.

Table 6.2: Averaged performance results from 30 random runs for the two experiments.

Exp	Grid	sV	RMSE	MAE	RAE	RRSE
1	inside	0.50	0.0049	0.0039	0.3658	0.3917
1	outside	0.50	0.0428	0.0360	0.6441	0.6677
2	inside	0.55	0.0033	0.0026	0.5255	0.5603
2	inside	0.40	0.0054	0.0045	0.3690	0.3805
2	outside	0.55 & 0.40	0.0302	0.0231	0.5730	0.6451

6.3.3. Validation

In this section, we present the results of a series of computer runs in order to assess the general predictive performance of the GP-based metamodel within the obtained final grids. We summarize these results in Table 6.2.

Several error-based metrics were used, namely Root Mean Squared Error (RMSE), Mean Absolute Error (MAE), and Root Relative Squared Error (RRSE). The reported average values were obtained from 30 random computer runs and 10 test points. Each metric was computed by comparing the predictions provided by the GP from the last iterations against the corresponding simulation outputs. Within each experiment, we compared the GP's predictive performance inside and outside the obtained training grids. Here, the grids used for validation correspond to the finer grids that ultimately lead to the identification of the input region of interest, particularly $[0.0, 1.0] \times [0.625, 0.750]$, and $[0.0, 1.0] \times [0.375, 0.50] \cup [0.0, 1.0] \times [0.875, 1.0]$, respectively for experiments 1 and 2.

We can see in Table 6.2 that the GP consistently performs better inside each obtained grid as opposed to the outside. As more points are acquired during the active learning procedure, particularly towards the unknown input regions of interest, the density of the training grid increases locally. Thus, it is expected that the GP yields increased predictive performance within the finer obtained grids. This behavior is particularly highlighted within RMSE and MAE columns.

6.4. Practical Implications

The proposed method can be used, as stated, to explore the input space that leads to pre-determined outputs. This process opens doors to a reverse engineering methodology, benefiting from existing transport system models, leading to a novel approach for traffic and mobility planning. As of now, city planners, transport providers, and researchers build transport system models (or use existing ones) when faced with the necessity of improving a specific performance indicator of their system. With these models, they are able to test and assess possible measures, changes, or additions that may or may not lead to an improvement of the specified indicator. This process requires the definition of potential scenarios, and their respective analyses within the defined models, through an interactive process of trial-and-error in which the user continuously fine-tunes them or eventually proposes new ones until they reach a satisfactory output performance. Such processes of scenario design and planning, which must be guided by specialized domain knowledge, traditionally require systematic simulation experimentation and exploration.

With the presented approach and the results obtained herein, there is the possibility to inverse the process by defining the target output of interest, or even a rough collection of possible output values or even regions, and explore the respective input domain that triggers them. This methodology would allow for an efficient and expeditious analysis of transport system requirements or changes, which would improve the decision mechanism and would offer a better overview of the possible solutions. Hence, the proposed approach provides EMS managers and planners a method to identify input domain regions that most influences target performance of interest. In this case, the input domain is the location of vehicles and the uncertainty of traffic and demand. This allows them to quickly identify where to reinforce resources and where, for example, traffic uncertainty will have a higher impact. In practice, the approach can provide practitioners with a solution matrix with the station configurations that, for a certain condition, would improve the target metric such as the survival rate discussed in this work.

This work was highly motivated by recent early work. In the same context of EMS and emergency response systems, Amorim et al. (2019a) has proposed an integrated strategic and tactical planning decision methodology to address the problem of optimal ambulance location, particularly relying on a GP-based metamodel to mimic the real-world system state as a complement to a location optimization model. Using such a metamodel, a local search for feasible and reasonable solutions was employed with successful results. In most modeling situations, evaluating a single solution might require several minutes or even hours. Therefore, a metamodel-based simulation was preferred since it allows a rather fast evaluation of alternative solutions with a minor and controlled accuracy loss. In this particular setting, the metamodel took the role of the simulator, effectively proposing a heuristic approach based on simulation metamodeling to help to solve the underlying hard optimization problem. On the other hand, it is worthwhile mentioning that a pre-trained GP was used during the entire optimization process. Although this heuristic proved to attain good results, we believe that it can be further improved if active learning is added to the equation. Instead of using a pre-trained GP, a more dynamic learning approach can be easily adopted by integrating the three main entities, namely, the simulation model, the optimization model, and the active learning metamodeling strategy as the link between them. From the optimization problem perspective, both the simulation model and metamodel are treated as black-boxes that provide alternative empirical evidence with respect to the real world since experimenting in the latter can prove to be prohibitively or impractical. Therefore, active learning can be employed to effectively balance the challenging trade-off between computational parsimony and accuracy loss. More training points can be added by request as the optimization process evolves in order to not only steer the learning towards important input regions but also, and consequently, to increase the prediction performance of the simulation metamodel.

To the best of our knowledge, the recent works of Amorim et al. (2019a) and Antunes et al. (2019) constitute the first applications of simulation metamodeling to an EMS simulator within a simulation-optimization setting. As of now, most emergency response studies continue to rely on systematic and exhausting simulation experiments on a trial-and-error basis. This often poses a considerable hindrance to a fast modeling and decision-making process. Hence, we are certain that the conjoint use of active learning and simulation metamodeling represents important complementary tools for researchers and practitioners in the field.

The obvious burden of this methodology falls on the data acquisition to train the metamodel, inevitably requiring the use of the simulation model. This drawback is also the main motivation for our directional approach, which steers the metamodel training phases to the most important simulation input regions from a modeling point of view. The proposed approach will always be dependent on the simulation model itself. The idea is that we extract as few training points as possible, especially avoiding redundancy, in the most informative way possible. Nevertheless, despite the complexity of the underlying problem and simulation model, we can eventually bypass the simulation burden of an in-depth exploration of scenarios. Additionally, we are even able to target specific simulation outcomes to understand the input regions which are directly responsible for influencing them.

6.5. Conclusions and Future Work

This chapter presents a methodology combining active learning and simulation metamodeling, which identifies relevant regions within the simulation input space that specifically triggers an output value previously set by the user. The survival rate outcome of an EMS simulation model was used as a case study. The results show that the presented approach can effectively guide the exploration process of the input space of a simulation model towards specific input regions whose output values are close to pre-defined search values of interest while minimizing computational workload at the same time. This makes the exploration process not only faster but also more efficient.

The presented approach can be improved in several directions. The possible correlation between the simulation outputs can prove to be useful to improve the modeling process. Therefore, a multi-output regression, namely the multi-output GP framework, will be taken into account in the future. Adopting popular designs for computer experiments, such as the Latin-hypercube, and combining them with the proposed sub-grid sequence, should further improve the statistical significance of the proposed approach. Additionally, the grid-based search should also be revisited. The generalization for more appropriate geometric arrangements, rather than the squared grid, should be taken into account, depending on the properties of the simulation input space under analysis, as well as on the particular goals of the metamodeling approach. Ultimately, we aim to compare the current approach with similar alternative methods, including different active learning schemes and stopping criteria. We also plan to improve the presented method by encompassing the development of heuristics (e.g., local search) within simulation-optimization approaches, which are commonly used tools in the research field of EMS planning and related decision support systems.

Chapter 7

Conclusion and Future Work

"Tudo em mim é a tendência para ser a seguir outra coisa; uma impaciência da alma consigo mesma, como uma criança inoportuna; um desassossego sempre crescente e sempre igual. Tudo me interessa e nada me prende. Atendo a tudo sonhando sempre."

Fernando Pessoa, 1888 - 1935

Simulation modeling is a powerful and, in many cases, the only reliable approach to address complex and intricate systems that otherwise would not be tractable via closed-formula analytic methods. It constitutes a well-known and perfectly established approach to study real-world urban transportation systems or to assess the performance of projected ones. Nevertheless, simulation models can become computationally expensive to execute when designed with enough detail and realism, especially when systematic experimentation is required. Within the transportation community, many simulation-based studies focused on predicting future states of reality typically rely on the manual exploration of the simulation model's behavior to retrieve outputs that ultimately lead to meaningful insights regarding the underlying real-world system they are meant to replicate. Scenario planning and what-if approaches are ultimately employed in an effort to reduce and discretize uncertainty into a finite set of possible futures. As expected, this kind of exploratory process can prove to be a computationally exhausting task whenever supported by the analysis of heavy simulation models.

In this thesis, we explored a methodological approach to reduce the computational hindrances emerging from exhausting and systematic computer-assisted analyses in the context of transport simulation modeling. Furthermore, we investigated solutions for automatizing the typical manual workflows, often associated with such exploratory processes. The presented methodology essentially relies on an integrated approach combining simulation metamodeling and active learning. On the one hand, metamodels are employed to approximate the input-output mapping inherently defined by simulation models and are typically recognized by their functional simplicity and speed, thereby providing parsimonious alternatives to study the underlying system under study. On the other hand, active learning is a modeling paradigm that aims to achieve higher predictive performance with as few training points as possible. This goal is attained by actively selecting the most informative points which potentially provide the highest modeling gains.

To the best of our knowledge, despite the advantages of both simulation metamodeling and active learning, these techniques have seldom been applied in an integrated manner within the transport simulation community. In this work, we investigated the feasibly of such an integrated approach in the context of transport simulation problems. The results obtained from different experimental settings show the potential of such kind of approach in minimizing the computational impacts of simulation-based studies. Moreover, the exploration process is effectively automatized, clearly distinguishing itself from other more manual-intensive approaches. Nevertheless, this does not mean that proper supervision is not required. On the contrary, the entire process should be overseen by expert knowledge to ensure that the obtained results generally follow the domain-specific premises and expected behaviors. As expected, minor adjustments might be required, depending on the problem under study and the specific goals the methodology is aiming at.

Another important aspect to bear in mind across the entire research is the approximate nature of the resulting simulation metamodels. We should not expect any metamodel to completely replace the underlying simulation model without, at least, a minor accuracy loss. Despite the nonlinear and nonparametric modeling properties of GPs, their generalization capabilities still heavily rely on the data points comprising the training set and their corresponding spatial neighborhoods. Whenever the exploration process begins to visit input regions where the similarities between the training points and the unlabeled ones, measured by the kernel, are difficult to establish, the GP's prediction uncertainty increases dramatically. This can be regarded as a limitation but also as a feature to exploit. Actually, it is not our intent to dismiss the simulation model entirely after the metamodel is obtained. Instead, our ultimate goal is to deploy the active learning metamodel along with the simulator itself as a bundled modeling framework, as both models play a crucial and complementary role within this integrated approach. While the metamodel aims at reducing the exploratory redundancy by trying to seek the most informative and distinct input data points, the simulation model ensures that this exploration process is maintained on the right track.

Besides encompassing a parsimonious auxiliary model that strives to mimic the simulation model's behavior, the metamodel is particularly useful to discover or suggest exploration directions in which the information gains are potentially higher. If we can avoid redundancy within the training set, we can certainly reduce the number of noninformative simulation runs, i.e., computer experiments whose results do not add relevant insights from a modeling perspective. The metamodel should discover those input regions that it has never "seen" before, thereby extracting the most out of each simulation run. Differently, the simulation model, by providing ground truth data on-demand, forces the metamodel to be constantly connected to the underlying process it aims to approximate, especially within high uncertainty regions, where its generalization tends to underperform.

Most of the GP's generalization performance is encapsulated in its kernel, as it is through this covariance function the similarities between the training points and the unlabeled ones are computed. During the metamodeling process, fitting a GP to simulation data encompasses optimizing the kernel's parameters, typically via likelihood maximization. As mentioned earlier, whereas the GP may perform outstandingly inside the simulation input region designed for training, the same often does not hold outside it. This limitation shows that, up to a certain extent, the GP essentially tries to find patterns based on correlations and other statistical properties within the observed data. Whenever predicting outside the training region where spatial similarities are difficult to find, the GP's performance drops, as correlations are likewise hard to establish or virtually nonexistent. Notice that correlations are essentially normalized covariances.

From the above aforementioned generic conclusions, we reintroduce our scientific contributions, which could address the research questions and help us attain the goals initially defined in this thesis. We summarize such contributions in the following.

- The feasibility of active learning metamodeling strategies in transport-related simulation problems was shown through several experimental settings. We used a

Demand Responsive Transportation simulator, a Traffic simulator, and an Emergency Medical System simulation model.

- Through the definition of concrete predictive variance-based simulation queries and stopping criteria, the metamodeling process was automatized. In turn, this has enabled us to reduce, to a certain degree, the need for manually exhausting simulation runs, thereby decreasing its associated computational drawbacks and accelerating the transport simulation analysis.
- We showed the potential of the presented integrated active learning metamodeling methodology for applications of policy analysis and decision-making processes in the context of a simulated emergency response system. Also, in a similar setting, but within a simulation-optimization problem, we showed the applicability of simulation metamodeling to study strategic and tactical planning solutions via metamodel-based heuristics.
- The explored methodology provided an opportunity to establish different communication platforms and research collaborations between different areas of expertise, especially bridging the machine learning and the transport simulation modeling fields.

The earlier discussed limitations of the GP modeling are not exclusive to this framework but rather a common characteristic of most machine learning approaches. Hence, if, on the one hand, these kinds of methods, when employed as metamodels, are successfully capable of approximate the simulation results with generally good performance, on the other hand, as correlation does not imply causality, they fail to capture causal relationships between the variables composing the simulator. Motivated by this idea of extracting causality (Pearl, 2009) from simulation models, the development of active learning causal metamodels might constitute an interesting challenge for future work. These models should be able to outperform the traditional simulation metamodels solely based on statistical patterns, especially in terms of generalization power outside the training regions. Additionally, this would also present an opportunity to diverge from the typical black-box metamodeling approaches to more explainable ones.

Another research direction to be explored in future work is related to developing active learning metamodeling strategies based on heteroscedastic metamodels. In this research, only the homoscedastic variant of the GP framework was used. Here, the assumption is that the signal variance is constant throughout the input space. Contrariwise, heteroscedastic implementations (Le et al., 2005; Kersting et al., 2007; Titsias and Lázaro-Gredilla, 2011; Wang and Neal, 2012) assume a variable disturbance of the output data as a function of the input. The assumption of homoscedasticity is fairly sufficient for most applications. However, in the absence of prior knowledge regarding the simulation output variance, our approach can be further improved if heteroscedasticity is taken into proper account. As our active learning strategy deeply relies on using the GP's predictive variance as a proxy for potential information gains, characterizing the variance as a function of the input space should make the exploration process more reliable and efficient.

It is worth remembering that no discrete input variables were used as part of the active learning strategies developed in this research. In discrete spaces, the notion of a neighborhood, which is crucial in our approach, degenerates completely, posing new challenges to the sampling process, mainly when both continuous and discrete are jointly used. In the end, performing exploration within discrete space has clear resemblances with the standard combinatorial problems. Therefore, the active learning sampling schemes must be revisited accordingly, constituting a possible line of future work.

Following the results from Chapter 6, we also would like to explore the idea of "reversed metamodeling" in the context of active learning. In other words, we would like to use a metamodel to approximate the output-input mapping rather than the opposite conventional direction. This approach would enable us to make predictions within the space of inputs while using the output variables as regressors. For example, given a pre-defined output region of interest, in which a particular transport policy tends to fail or perform poorly, active learning could be employed to guide the exploration process towards those input points that trigger, via simulation, the output values contained in those regions. Furthermore, the metamodel and the reversed one could be combined in an integrated active learning loop in which one feeds, or suggests, the other with new training points, each of which is within their own corresponding input spaces. This approach can be implemented via multi-output modeling (Alvarez and Lawrence, 2011; Moreno-Muñoz et al., 2018; Liu et al., 2018). Some of these ideas are, in fact, part of ongoing exploratory work.

One last future line of work would be to develop a software package, possibly with a graphical user interface (GUI), in which the present or similar approaches could be easily accessible to the average practitioner and simulation modeler. Nevertheless, many hindrances are likely to be ahead of this plan. As discussed during this research, a data link between the simulation model and the metamodeling algorithm is constantly required for active learning to work correctly. Moreover, to ensure smooth communication between them, both must feed on similar input data structures. Whereas in most machine learning techniques, the input data is conventionally organized in matrix structures, where typically each column corresponds to an input variable and each line to a single observation, the same might not be so straightforward in the case of some simulators. Depending on the simulation platform, language, or even implementation design itself, data might be organized in fundamentally different ways. Additionally, the way the data is stored, either within the simulation implementation via simple text-based files or externally using dedicated databases, presents another critical detail to be taken into account from a rather practical point of view. Hence, the idea of developing a universal software package that works with an arbitrary simulation model is a very ambitious one. The experience resulting from this research tells us that significant changes are usually necessary according to the simulator's implementation nature and inherent design. A plug-and-play solution might be more feasible if conditional on the type of simulation model.

Some of these plans will be totally or partially put into practice in the immediate future as part of the EU-funded NOSTROMO research project mentioned in Chapter 1. In the context of Air Traffic Management (AMT) simulation analysis, we will have the opportunity to further explore and improve the presented methodology in the context of transport-related problems. More complex simulation models will be studied, and thus many unpredictable challenges are likely to be faced. In any case, despite their limitations, we are confident that integrated active learning metamodeling methodologies have their rightful within the transport simulation community and, if employed wisely and carefully, constitute a robust auxiliary framework that further enhances and complements the traditional simulation modeling approaches.

Bibliography

Aboueljinane, L., Sahin, E., and Jemai, Z. (2013). A review on simulation models applied to emergency medical service operations. *Computers & Industrial Engineering*, 66(4):734–750.

Alpaydin, E. (2020). *Introduction to machine learning.* MIT press.

Alvarez, M. A. and Lawrence, N. D. (2011). Computationally efficient convolved multiple output gaussian processes. *The Journal of Machine Learning Research*, 12:1459–1500.

Amorim, M., Antunes, F., Ferreira, S., and Couto, A. (2019a). An integrated approach for strategic and tactical decisions for the emergency medical service: Exploring optimization and metamodel-based simulation for vehicle location. *Computers & Industrial Engineering*, 137:106057.

Amorim, M., Ferreira, S., and Couto, A. (2017). Road safety and the urban emergency medical service (uems): Strategy station location. *Journal of Transport & Health*, 6:60–72.

Amorim, M., Ferreira, S., and Couto, A. (2018). Emergency medical service response: analyzing vehicle dispatching rules. *Transportation Research Record*, 2672(32):10–21.

Amorim, M., Ferreira, S., and Couto, A. (2019b). Corrigendum to how do traffic and demand daily changes define urban emergency medical service (uems) strategic decisions?: A multi-period survival approach [jth 12 (2019) 60-74]. *Journal of Transport & Health*, page 100570.

Amorim, M., Ferreira, S., and Couto, A. (2019c). How do traffic and demand daily changes define urban emergency medical service (uems) strategic decisions?: A robust survival model. *Journal of Transport & Health*, 12:60–74.

Amorim, M., Ferreira, S., and Couto, A. F. (2016). Urban emergency medical service: dynamic model for dynamic cities. *Technical Report*.

Andersson, H., Granberg, T. A., Christiansen, M., Aartun, E. S., and Leknes, H. (2020). Using optimization to provide decision support for strategic emergency medical service planning – three case studies. *International Journal of Medical Informatics*, 133:103975.

Angluin, D. (1988). Queries and concept learning. *Machine learning*, 2(4):319–342.

Ankenman, B., Nelson, B. L., and Staum, J. (2010). Stochastic kriging for simulation metamodeling. *Operations research*, 58(2):371–382.

Antunes, F., Amorim, M., Pereira, F., and Ribeiro, B. (2019). Active learning metamodeling for policy analysis: application to an emergency medical service simulator. *Simulation Modelling Practice and Theory*, page 101947.

Antunes, F., O'Sullivan, A., Rodrigues, F., and Pereira, F. (2017). A review of heteroscedasticity treatment with gaussian processes and quantile regression metamodels. In *Seeing Cities Through Big Data*, pages 141–160. Springer.

Antunes, F., Ribeiro, B., Pereira, F., and Gomes, R. (2018). Efficient transport simulation with restricted batch-mode active learning. *Transactions on Intelligent Transportation Systems*, PP:1–10.

Bandara, D., Mayorga, M. E., and McLay, L. A. (2014). Priority dispatching strategies for ems systems. *Journal of the Operational Research Society*, 65(4):572–587.

Bankes, S. (1993). Exploratory modeling for policy analysis. *Operations research*, 41(3):435–449.

Bankes, S. C. (1992). Exploratory modeling and the use of simulation for policy analysis. Technical report, RAND, Santa Monica, CA.

Barton, R. R. (1998). Simulation metamodels. In *Simulation Conference Proceedings, 1998. Winter*, volume 1, pages 167–174. IEEE.

Barton, R. R. and Meckesheimer, M. (2006). Metamodel-based simulation optimization. *Handbooks in operations research and management science*, 13:535–574.

Bastos, L. S. and O'Hagan, A. (2009). Diagnostics for gaussian process emulators. *Technometrics*, 51(4):425–438.

Bélanger, V., Kergosien, Y., Ruiz, A., and Soriano, P. (2016). An empirical comparison of relocation strategies in real-time ambulance fleet management. *Computers & Industrial Engineering*, 94:216–229.

Berg, P. L. V. D. and Essen, J. T. V. (2019). Comparison of static ambulance location models. *International Journal of Logistics Systems and Management*, 32(3-4):292–321.

Bishop, C. M. (2006). *Pattern recognition and machine learning.* Springer.

Blanning, R. W. (1974). The sources and uses of sensitivity information. *Interfaces*, 4(4):32–38.

Boukouvalas, A. (2010). *Emulation of random output simulators.* PhD thesis, Aston University.

Boukouvalas, A., Cornford, D., and Singer, A. (2009). Managing uncertainty in complex stochastic models: Design and emulation of a rabies model. In *6th St. Petersburg Workshop on Simulation*, pages 839–841.

Box, G. E., Draper, N. R., et al. (1987). *Empirical model-building and response surfaces*, volume 424. Wiley New York.

Box, G. E. and Tiao, G. C. (2011). *Bayesian inference in statistical analysis*, volume 40. John Wiley & Sons.

Brinker, K. (2003). Incorporating diversity in active learning with support vector machines. In *ICML*, volume 3, pages 59–66.

Brochu, E., Cora, V. M., and De Freitas, N. (2010). A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *arXiv preprint arXiv:1012.2599.*

Bélanger, V., Ruiz, A., and Soriano, P. (2019). Recent optimization models and trends in location, relocation, and dispatching of emergency medical vehicles. *European Journal of Operational Research*, 272(1):1 – 23.

Carbonell, J. G., Michalski, R. S., and Mitchell, T. M. (1983). An overview of machine learning. *Machine learning*, pages 3–23.

Chen, T., Hadinoto, K., Yan, W., and Ma, Y. (2011). Efficient meta-modelling of complex process simulations with time–space-dependent outputs. *Computers & chemical engineering*, 35(3):502–509.

Chen, X., Zhu, Z., He, X., and Zhang, L. (2015). Surrogate-based optimization for solving a mixed integer network design problem. *Transportation Research Record: Journal of the Transportation Research Board*, 2497(2497):124–134.

Chen, X. M., Zhang, L., He, X., Xiong, C., and Li, Z. (2014). Surrogate-based optimization of expensive-to-evaluate objective for optimal highway toll charges in transportation network. *Computer-Aided Civil and Infrastructure Engineering*, 29(5):359–381.

Chilès, J.-P. and Desassis, N. (2018). *Fifty Years of Kriging, in Handbook of Mathematical Geosciences: Fifty Years of IAMG*, pages 589–612. Springer International Publishing.

Christen, J. A. and Sansó, B. (2011). Advances in the sequential design of computer experiments based on active learning. *Communications in Statistics-Theory and Methods*, 40(24):4467–4483.

Christopher, M. B. (2016). *Pattern Recognition and Machine Learning*. Springer-Verlag New York.

Chung, E. and Dumont, A.-G. (2009). *Transport Simulation: Beyond Traditional Approaches*. EPFL press.

Church, R. and ReVelle, C. (1974). The maximal covering location problem. In *Papers of the regional science association*, volume 32, pages 101–118. Springer-Verlag.

Ciuffo, B., Casas, J., Montanino, M., Perarnau, J., and Punzo, V. (2013). Gaussian process metamodels for sensitivity analysis of traffic simulation models: Case study of aimsun mesoscopic model. *Transportation Research Record: Journal of the Transportation Research Board*, 2390(2390):87–98.

Cohn, D. A. (1996). Neural network exploration using optimal experiment design. *Neural networks*, 9(6):1071–1083.

Cohn, D. A., Ghahramani, Z., and Jordan, M. I. (1996). Active learning with statistical models. *Journal of artificial intelligence research*, 4:129–145.

Conti, S. and O'Hagan, A. (2010). Bayesian emulation of complex multi-output and dynamic computer models. *Journal of statistical planning and inference*, 140(3):640–651.

Cover, T. M. and Thomas, J. A. (2006). *Elements of Information Theory*. John Wiley & Sons.

Daskin, M. S. (1983). A maximum expected covering location model: formulation, properties and heuristic solution. *Transportation science*, 17(1):48–70.

Daskin, M. S. and Stern, E. H. (1981). A hierarchical objective set covering model for emergency medical service vehicle deployment. *Transportation Science*, 15(2):137–152.

Dibene, J. C., Maldonado, Y., Vera, C., de Oliveira, M., Trujillo, L., and Schütze, O. (2017). Optimizing the location of ambulances in tijuana, mexico. *Computers in biology and medicine*, 80:107–115.

Domingos, P. (2015). *The master algorithm: How the quest for the ultimate learning machine will remake our world*. Penguin Books.

Erkut, E., Ingolfsson, A., and Erdoğan, G. (2008). Ambulance location for maximum survival. *Naval Research Logistics (NRL)*, 55(1):42–58.

Fedorov, V. V. (1972). *Theory of optimal experiments*. Academic Press.

Fishburn, P. T., Golkar, J., and Taaffe, K. M. (1995). Simulation of transportation systems. In *Proceedings of the 27th conference on Winter simulation*, pages 51–54.

Fisher, R. A. (1937). *The Design of Experiments.* Oliver And Boyd; Edinburgh; London.

Fonseca, D. J., Navaresse, D. O., and Moynihan, G. P. (2003). Simulation metamodeling through artificial neural networks. *Engineering Applications of Artificial Intelligence*, 16(3):177–183.

French, R. M. (2000). The turing test: the first 50 years. *Trends in cognitive sciences*, 4(3):115–122.

Friedman, L. W. (2012). *The simulation metamodel.* Springer Science & Business Media.

Friedman, L. W. and Pressman, I. (1988). The metamodel in simulation analysis: Can it be trusted? *Journal of the Operational Research Society*, 39(10):939–948.

Fujiwara, O., Makjamroen, T., and Gupta, K. K. (1987). Ambulance deployment analysis: A case study of bangkok. *European Journal of Operational Research*, 31(1):9–18.

Geman, S., Bienenstock, E., and Doursat, R. (1992). Neural networks and the bias/variance dilemma. *Neural computation*, 4(1):1–58.

Gendreau, M., Laporte, G., and Semet, F. (1997). Solving an ambulance location model by tabu search. *Location science*, 5(2):75–88.

Ghayoomi, M. (2010). Using variance as a stopping criterion for active learning of frame assignment. In *Proceedings of the NAACL HLT 2010 Workshop on Active Learning for Natural Language Processing*, pages 1–9. Association for Computational Linguistics.

Gomes, E., Araújo, R., Soares-Oliveira, M., and Pereira, N. (2004). International ems systems: Portugal. *Resuscitation*, 62(3):257–260.

Gomes, R., de Sousa, J. P., and Galvão, T. (2014). An integrated approach for the design of demand responsive transportation services. In *Computer-based Modelling and Optimization in Transportation*, pages 223–235. Springer.

Gomes, R. J. R. (2012). *Dynamic vehicle routing for demand responsive transportation systems.* PhD thesis, University of Porto.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press. http://www.deeplearningbook.org.

Gramacy, R. B. (2020). *Surrogates: Gaussian process modeling, design, and optimization for the applied sciences*. CRC Press.

Guo, Y. and Schuurmans, D. (2008). Discriminative batch mode active learning. In *Advances in neural information processing systems*, pages 593–600.

Haghani, A. and Yang, S. (2007). Real-time emergency response fleet deployment: Concepts, systems, simulation & case studies. In *Dynamic fleet management*, pages 133–162. Springer.

He, Z., Qin, X., Xie, Y., and Guo, J. (2018). Service location optimization model for improving rural emergency medical services. *Transportation Research Record*, 2672(32):83–93.

Hoi, S. C., Jin, R., and Lyu, M. R. (2006). Large-scale text categorization by batch mode active learning. In *Proceedings of the 15th international conference on World Wide Web*, pages 633–642. ACM.

Iannoni, A. P., Morabito, R., and Saydam, C. (2009). An optimization approach for ambulance location and the districting of the response segments on highways. *European Journal of Operational Research*, 195(2):528–542.

Iman, R. L. and Conover, W. (1980). Small sample sensitivity analysis techniques for computer models. with an application to risk assessment. *Communications in statistics-theory and methods*, 9(17):1749–1842.

Jagtenberg, C., van den Berg, P., and van der Mei, R. (2017a). Benchmarking online dispatch algorithms for emergency medical services. *European Journal of Operational Research*, 258(2):715 – 725.

Jagtenberg, C. J., Bhulai, S., and van der Mei, R. D. (2017b). Dynamic ambulance dispatching: is the closest-idle policy always optimal? *Health care management science*, 20(4):517–531.

Jones, B. and Johnson, R. T. (2009). Design and analysis for the gaussian process model. *Quality and Reliability Engineering International*, 25(5):515–524.

Jones, D. R., Schonlau, M., and Welch, W. J. (1998). Efficient global optimization of expensive black-box functions. *Journal of Global optimization*, 13(4):455–492.

Kersting, K., Plagemann, C., Pfaff, P., and Burgard, W. (2007). Most likely heteroscedastic gaussian process regression. In *Proceedings of the 24th international conference on Machine learning*, pages 393–400. ACM.

Kindberg, T. and Fox, A. (2002). System software for ubiquitous computing. *IEEE pervasive computing*, 1(1):70–81.

Kitamura, R. and Kuwahara, M. (2006). *Simulation Approaches in Transportation Analysis: Recent Advances and Challenges*. Springer Science & Business Media.

Kleijnen, J. (1987). Model behaviour: Regression metamodel summarization. *Encyclopedia of Systems and Control*, 5:3024–3030.

Kleijnen, J. P. (1975). A comment on blanning's "metamodel for sensitivity analysis: the regression metamodel in simulation". *Interfaces*, 5(3):21–23.

Kleijnen, J. P. (1979). Regression metamodels for generalizing simulation results. *IEEE transactions on systems, man, and cybernetics*, 9:93–96.

Kleijnen, J. P. (2009). Kriging metamodeling in simulation: A review. *European journal of operational research*, 192(3):707–716.

Kleijnen, J. P. and Sargent, R. G. (2000). A methodology for fitting and validating metamodels in simulation. *European Journal of Operational Research*, 120(1):14–29.

Kleijnen, J. P. and Van Beers, W. C. (2004). Application-driven sequential designs for simulation experiments: Kriging metamodelling. *Journal of the Operational Research Society*, 55(8):876–883.

Kleijnen, J. P. and Van Beers, W. C. (2005). Robustness of kriging when interpolating in random simulation with heterogeneous variances: Some experiments. *European Journal of Operational Research*, 165(3):826–834.

Knight, V. A., Harper, P. R., and Smith, L. (2012). Ambulance allocation for maximal survival with heterogeneous outcome measures. *Omega*, 40(6):918–926.

Knoops, L. and Lundgren, T. (2016). Modeling ambulance dispatching rules for ems-systems.

Krajzewicz, D., Erdmann, J., Behrisch, M., and Bieker, L. (2012). Recent development and applications of sumo-simulation of urban mobility. *International Journal On Advances in Systems and Measurements*, 5(3&4):128–138.

Kumar, P. and Gupta, A. (2020). Active learning query strategies for classification, regression, and clustering: A survey. *Journal of Computer Science and Technology*, 35(4):913–945.

Law, A. M. (2015). *Simulation Modeling and Analysis*. McGraw-Hill Higher Education, 5th edition.

Laws, F. and Schätze, H. (2008). Stopping criteria for active learning of named entity recognition. In *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*, pages 465–472. Association for Computational Linguistics.

Le, Q. V., Smola, A. J., and Canu, S. (2005). Heteroscedastic gaussian process regression. In *Proceedings of the 22nd international conference on Machine learning*, pages 489–496. ACM.

Levy, J. M. (2016). *Contemporary urban planning*. Taylor & Francis.

Lewis, D. D. and Catlett, J. (1994). Heterogeneous uncertainty sampling for supervised learning. In *Machine learning proceedings 1994*, pages 148–156. Elsevier.

Lewis, D. D. and Gale, W. A. (1994). A sequential algorithm for training text classifiers. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 3–12. Springer-Verlag New York, Inc.

Li, M. and Sethi, I. K. (2006). Confidence-based active learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(8):1251–1261.

Li, P. and Chen, S. (2016). A review on gaussian process latent variable models. *CAAI Transactions on Intelligence Technology*, 1(4):366–376.

Li, S., Da Xu, L., and Zhao, S. (2015). The internet of things: a survey. *Information Systems Frontiers*, 17(2):243–259.

Li, X., Zhao, Z., Zhu, X., and Wyatt, T. (2011). Covering models and optimization techniques for emergency response facility location and planning: a review. *Mathematical Methods of Operations Research*, 74(3):281–310.

Lifshits, M. (2012). Lectures on gaussian processes. In *Lectures on Gaussian Processes*, pages 1–117. Springer.

Ling, C. K., Low, K. H., and Jaillet, P. (2016). Gaussian process planning with lipschitz continuous reward functions: Towards unifying bayesian optimization, active learning, and beyond. In *AAAI*, pages 1860–1866.

Liu, H., Cai, J., and Ong, Y.-S. (2018). Remarks on multi-output gaussian process regression. *Knowledge-Based Systems*, 144:102–121.

Marengo, M. C. (2014). Urban simulation models: Contributions as analysis-methodology in a project of urban renewal. *Current Urban Studies*, 2(03):298.

Martine, G., Marshall, A., et al. (2007). State of world population 2007: unleashing the potential of urban growth. In *State of world population 2007: unleashing the potential of urban growth*. UNFPA.

Matheron, G. (1963). Principles of geostatistics. *Economic geology*, 58(8):1246–1266.

Maxwell, M. S., Restrepo, M., Henderson, S. G., and Topaloglu, H. (2010). Approximate dynamic programming for ambulance redeployment. *INFORMS Journal on Computing*, 22(2):266–281.

Mayorga, M. E., Bandara, D., and McLay, L. A. (2013). Districting and dispatching policies for emergency medical service systems to improve patient survival. *IIE Transactions on Healthcare Systems Engineering*, 3(1):39–56.

McClarren, R. G. (2018). Gaussian process emulators and surrogate models. In *Uncertainty Quantification and Predictive Computational Science*, pages 257–274. Springer.

McCormack, R. and Coates, G. (2015). A simulation model to enable the optimization of ambulance fleet allocation and base station location for increased patient survival. *European Journal of Operational Research*, 247(1):294–309.

McCulloch, W. S. and Pitts, W. (1943). A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133.

McElreath, R. (2020). *Statistical rethinking: A Bayesian course with examples in R and Stan*. CRC press.

McKay, M. D., Beckman, R. J., and Conover, W. J. (2000). A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics*, 42(1):55–61.

Mietzner, D. and Reger, G. (2004). Scenario approaches: history, differences, advantages and disadvantages. In *EU-US Seminar: New Technology Foresight, Forecasting and Assessment Methods, Seville, May*, pages 13–14.

Mitchell, T. M. et al. (1997). *Machine learning*. McGraw-hill New York.

Moore, T. and Pulidindi, J. (2013). Understanding urban transportation systems: An action guide for city leaders. *National League of Cities. Available online at: www.nlc.org. Accessed on 05/10/20*, 31.

Moreno-Muñoz, P., Artés-Rodríguez, A., and Álvarez, M. A. (2018). Heterogeneous multi-output gaussian process prediction. *arXiv preprint arXiv:1805.07633*.

Murphy, K. P. (2012). *Machine learning: a probabilistic perspective*. MIT press.

Pearl, J. (2009). *Causality*. Cambridge university press.

Pfingsten, T. (2006). Bayesian active learning for sensitivity analysis. In *European Conference on Machine Learning*, pages 353–364. Springer.

Pons, P. T. and Markovchick, V. J. (2002). Eight minutes or less: does the ambulance response time guideline impact trauma patient outcome? *The Journal of Emergency Medicine*, 23(1):43 – 48.

Poulton, M., Noulas, A., Weston, D., and Roussos, G. (2019). Modeling metropolitan-area ambulance mobility under blue light conditions. *IEEE Access*, 7:1390–1403.

Rasmussen, C. E. and Williams, C. (2006). *Gaussian processes for machine learning (Adaptive computation and machine learning)*. The MIT Press.

Repede, J. F. and Bernardo, J. J. (1994). Developing and validating a decision support system for locating emergency medical vehicles in louisville, kentucky. *European journal of operational research*, 75(3):567–581.

Restrepo, M., Henderson, S. G., and Topaloglu, H. (2009). Erlang loss models for the static deployment of ambulances. *Health care management science*, 12(1):67.

ReVelle, C. and Hogan, K. (1989). The maximum availability location problem. *Transportation science*, 23(3):192–200.

Robert, C. (2014). Machine learning, a probabilistic perspective.

Rodrigue, J.-P., Comtois, C., and Slack, B. (2016). *The geography of transport systems*. Routledge.

Roy, N. and McCallum, A. (2001). Toward optimal active learning through monte carlo estimation of error reduction. *ICML, Williamstown*, pages 441–448.

Samuel, A. L. (2000). Some studies in machine learning using the game of checkers. *IBM Journal of research and development*, 44(1.2):206–226.

Sanchez, P. J. (2007). Fundamentals of simulation modeling. In *Simulation Conference, 2007 Winter*, pages 54–62. IEEE.

Sanchez, S. M. (2005). Work smarter, not harder: guidelines for designing simulation experiments. In *Simulation Conference, 2005 Proceedings of the Winter*, pages 14–pp. IEEE.

Schmid, V. (2012). Solving the dynamic ambulance relocation and dispatching problem using approximate dynamic programming. *European journal of operational research*, 219(3):611–621.

Schmid, V. and Doerner, K. F. (2010). Ambulance location and relocation problems with time-dependent travel times. *European journal of operational research*, 207(3):1293–1303.

Schulz, E., Speekenbrink, M., and Krause, A. (2017). A tutorial on gaussian process regression with a focus on exploration-exploitation scenarios. *bioRxiv*, page 095190.

Settles, B. (2010). Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison.

Settles, B. and Craven, M. (2008). An analysis of active learning strategies for sequence labeling tasks. In *Proceedings of the conference on empirical methods in natural language processing*, pages 1070–1079. Association for Computational Linguistics.

Settles, B., Craven, M., and Ray, S. (2008). Multiple-instance active learning. In *Advances in neural information processing systems*, pages 1289–1296.

Seung, H. S., Opper, M., and Sompolinsky, H. (1992). Query by committee. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 287–294. ACM.

Snyder, L. V. and Daskin, M. S. (2005). Reliability models for facility location: the expected failure cost case. *Transportation Science*, 39(3):400–416.

Song, W., Han, K., Wang, Y., Friesz, T., and del Castillo, E. (2017). Statistical metamodeling of dynamic network loading. *Transportation Research Procedia*, 23:263–282.

Su, S. and Shih, C.-L. (2003). Modeling an emergency medical services system using computer simulation. *International journal of medical informatics*, 72(1-3):57–72.

Telford, J. K. (2007). A brief introduction to design of experiments. *Johns Hopkins APL Technical Digest*, 27(3):224–232.

United Nations (2018). *The World's Cities in 2018, Data Booklet, ST/ESA/SER.A/417*. Department of Economic and Social Affairs, Population Division.

United Nations (2020). *World Cities Report 2020: The Value of Sustainable Urbanization*. United Nations Human Settlements Programme (UN-Habitat).

Titsias, M. K. and Lázaro-Gredilla, M. (2011). Variational heteroscedastic gaussian process regression. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 841–848.

Toregas, C., Swain, R., ReVelle, C., and Bergman, L. (1971). The location of emergency service facilities. *Operations research*, 19(6):1363–1373.

Tufuor, E. O. A., Rilett, L. R., Nam, Y., and Beltran, A. H. (2018). Land suitability analysis for emergency medical services posts along state highways: A case study of california. *Transportation Research Record*, 2672(32):94–106.

Turing, A. M. (1950). Computing machinery and intelligence. *Mind*, 59(236):433–460.

Ünlüyurt, T. and Tunçer, Y. (2016). Estimating the performance of emergency medical service location models via discrete event simulation. *Computers & Industrial Engineering*, 102:467–475.

Vallverdú, J. (2015). *Bayesians Versus Frequentists: A Philosophical Debate on Statistical Reasoning*. Springer.

Van Beers, W. C. and Kleijnen, J. P. (2004). Kriging interpolation in simulation: a survey. In *Simulation Conference, 2004. Proceedings of the 2004 Winter*, volume 1. IEEE.

Van Beers, W. C. and Kleijnen, J. P. C. (2003). Kriging for interpolation in random simulation. *Journal of the Operational Research Society*, 54(3):255–262.

Van Essen, J. T., Hurink, J. L., Nickel, S., and Reuter, M. (2013). Models for ambulance planning on the strategic and the tactical level. *TU Eindhoven, Research School for Operations Management and Logistics, Netherlands (BETA)*.

Vapnik, V. (1963). Pattern recognition using generalized portrait method. *Automation and remote control*, 24:774–780.

Viana, F. A. (2016). A tutorial on latin hypercube design of experiments. *Quality and Reliability Engineering International*, 32(5):1975–1985.

Vlachos, A. (2008). A stopping criterion for active learning. *Computer Speech & Language*, 22(3):295–312.

Vlahov, D. and Galea, S. (2002). Urbanization, urbanicity, and health. *Journal of Urban Health*, 79(1):S1–S12.

Wang, C. and Neal, R. M. (2012). Gaussian process regression with heteroscedastic or non-gaussian residuals. *arXiv preprint arXiv:1212.6246*.

Wang, G. G. and Shan, S. (2006). Review of metamodeling techniques in support of engineering design optimization. In *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, volume 4255, pages 415–426.

Wang, L., Beeson, D., Akkaram, S., and Wiggs, G. (2005). Gaussian process metamodels for efficient probabilistic design in complex engineering design spaces. *ASME Paper No. DETC2005-85406*.

Wang, W., Cai, W., and Zhang, Y. (2014). Stability-based stopping criterion for active learning. In *Data Mining (ICDM), 2014 IEEE International Conference on*, pages 1019–1024. IEEE.

Wang, X. and Zhai, J. (2016). *Learning With Uncertainty*. CRC Press.

Weber, C. and Puissant, A. (2003). Urbanization pressure and modeling of urban growth: example of the tunis metropolitan area. *Remote Sensing of Environment*, 86(3):341 – 352. Urban Remote Sensing.

Xu, Z., Akella, R., and Zhang, Y. (2007). Incorporating diversity and density in active learning for relevance feedback. In *European Conference on Information Retrieval*, pages 246–257. Springer.

Yang, S., Hamedi, M., and Haghani, A. (2005). Online dispatching and routing model for emergency vehicles with area coverage constraints. *Transportation Research Record*, 1923(1):1–8.

Yue, Y., Marla, L., and Krishnan, R. (2012). An efficient simulation-based approach to ambulance fleet allocation and dynamic redeployment. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 26.

Zantalis, F., Koulouras, G., Karabetsos, S., and Kandris, D. (2019). A review of machine learning and iot in smart transportation. *Future Internet*, 11(4):94.

Zhang, L., He, X., Xiong, C., Zhu, Z., et al. (2014). Bayesian stochastic kriging metamodel for active traffic management of corridors. In *IIE Annual Conference. Proceedings*, page 1790. Institute of Industrial and Systems Engineers (IISE).

Zhu, J., Wang, H., Hovy, E., and Ma, M. (2010). Confidence-based stopping criteria for active learning for data annotation. *ACM Transactions on Speech and Language Processing (TSLP)*, 6(3):3.

Zhu, X., Zhang, P., Lin, X., and Shi, Y. (2007). Active learning from data streams. In *Seventh IEEE International Conference on Data Mining (ICDM 2007)*, pages 757–762. IEEE.