Noise and Hotel Revenue Management in Simulation-based Optimization

Manuel Dalcastagné

University of Trento - Italy

Department of Information Engineering and Computer Science

Advisor: Prof. Roberto Battiti

October 2021

Acknowledgements

I would like to thank my advisor, Prof. Roberto Battiti, for his help and advice during the PhD. His expertise helped me to sharpen my research methodology and skills, as well as my attitude towards science. I extend my sincere gratitude to Prof. Jürgen Branke for his valuable feedback on this thesis, and for the insightful conversations which helped me to move further the research that ultimately converged into this dissertation. Moreover, I would like to thank Prof. Amir Atiya for the time he spent to review this work and his useful feedback.
My thanks also go to past and present members of the LION lab, in particular to Prof. Mauro Brunato and Dr. Andrea Mariello, for the useful discussions during the PhD.
I am deeply grateful to my parents and grandmother, who gifted me with all the support, the love and the resources needed to let me find my own direction to explore that complex problem called life. Finally, I would like to express my gratitude to my friends and in particular to Michela, whose unwavering support helped and sustained me in one of the toughest periods of my existence.

Abstract

Several exact and approximate dynamic programming formulations have already been proposed to solve hotel revenue management (RM) problems. To obtain tractable solutions, these methods are often bound by simplifying assumptions which prevent their application on large and dynamic complex systems. This dissertation introduces HotelSimu, a flexible simulation-based optimization approach for hotel RM, and investigates possible approaches to increase the efficiency of black-box optimization methods in the presence of noise. In fact, HotelSimu employs black-box optimization and stochastic simulation to find the dynamic pricing policy which is expected to maximize the revenue of a given hotel in a certain period of time. However, the simulation output is noisy and different solutions should be compared in a statistically significant manner. Various black-box heuristics based on variations of random local search are investigated and integrated with statistical analysis techniques in order to manage efficiently the optimization budget.

Keywords

Simulation-based Optimization, Black-box Optimization, Noise, Hotel Revenue Management, Statistical Analysis, Optimal Computing Budget Allocation, Indifference Zone Selection

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Need for simulation-based optimization

Standard optimization methods require the mathematical formulation of a function to be optimized. In order to solve an optimization problem, techniques like linear or dynamic programming (DP) usually need a set of equations which models the behavior and the constraints of the problem.
In real-world scenarios, an explicit formulation is rarely available or ready-to-use [1]. Also, mathematically formulating an objective function can often be difficult, especially in the stochastic setting. Not only models can be linear or non-linear, and discrete or continuous, but also based on deterministic or probabilistic assumptions which might cause the objective function to contain several integrals [2, 3]. Once a formulation has been constructed, various optimization methods can be employed to find a solution which is exact or approximated, optimal or suboptimal, available in closed-form or iteratively found by improving an initial solution. The choice of the most suitable method depends on the characteristics of the problem at hand and on the needs of the

decision maker.
Although mathematical models are widely used in research, they are often tied to simplifying assumptions which might be necessary to obtain tractable models. Unfortunately, these models may result to be too simplistic for real-world applications [2]. A possible solution comes from the adoption of optimization methods which do not require a mathematical formulation in closed-form, but only numerical evaluations of the objective function at any given solution. These methods are called numerical, derivative-free, black-box or model-free, because they do not require any *a priori* knowledge of the structure of the objective function [1, 2]. Therefore, to solve complex problems where a closed-form solution is hard to construct, an alternative comes from the combination of simulation and black-box optimization. Simulation models mimic the behavior of dynamic systems, while black-box methods use the numerical output of simulators in order to evaluate possible solutions of the problem. The resulting framework is called simulation-based optimization, and it is particularly useful in those scenarios where a stochastic system can be easily simulated, whereas a closed-form mathematical model with a tractable solution is hard to find [2].

1.2 Complex dynamic systems

When modeling a dynamic system for optimization purposes, the formulation of the problem is the first step towards its solution. It is necessary to find a suitable model that comprehensively describes the state of the system, and to define how the system transitions from a state to another.
A system is usually modeled as a set of entities that interact with each

other as an environment. The state at time t is determined as a set of parameters which defines the characteristics of each entity, together with any available information that can be useful to take decisions. Transitions depend on the relationships between the entities of the system, and the frequency of the transitions is modeled by considering discrete or continuous time intervals. In the discrete case, the state changes at discrete points in time (e.g. a customer enters or leaves a queue at the postal office). In the continuous case, the state evolves continuosly over time (e.g. the flow of liquids in fluid dynamics). Systems which are completely discrete or continuous are not common, but in most problems one of the two time variations predominates and the whole problem is considered as discrete or continuous [4].
In complex dynamic systems based on discrete-time intervals and probabilistic assumptions, the size and complexity of the problem prevents a straightforward application of classical optimization methods. Defining explicitely a state which leads to tractable models may be difficult, because the state space corresponds to all possible combinations of the parameters' values which define the entities. So, the dimensionality of the problem explodes very quickly together with the number of parameters. This problem is called the *curse of dimensionality* [2, 3]. Furthermore, in stochastic systems, the entities of the environment are modeled as random variables which follow some probability distributions. Generically, the larger the number of random variables the more complicated is the analysis. This is especially true in the case of mathematical models, since the larger the number of random variables in a system the harder it is to derive closed-form expressions for the objective function [2].

1.3 Different types of optimization

In operations research, optimization problems of complex stochastic scenarios are usually formulated according to control optimization or parametric optimization paradigms [2]. In control optimization, the goal is to determine an optimal control in each state of a system in order to optimize some performance measure. Depending on the assumption of discrete or continuous time intervals, control optimization problems can be solved respectively by adopting DP or optimal learning formulations [3, 5]. In parametric optimization, the goal is to find the values of the decision variables which optimize an objective function, and these problems are traditionally solved using mathematical methods like linear or non-linear programming.
Prior to 1950, it had been established that many mathematical problems could be approximated numerically by means of a sampling experiment in a Monte Carlo fashion [6]. The process of neutron transport was the first application of simulations in the modern era [7, 8]. However, the terminology used for simulations can be misleading in this context. Most works in the literature refer to any kind of simulation as *Monte Carlo*, regardless of the simulation's purpose or complexity. Thus, to highlight effectively the different roles of simulation in control and parametric optimization, this dissertation differentiates between static and dynamic simulation models [4]. Static models, which correspond to the traditional Monte Carlo approach, estimate values of interest by probabilistically representing a system at a particular point in time through repeated random sampling. In contrast, dynamic models represent the evolution of a system over time and capture the internal logic of the process. An example is discrete-event simulation, where the state of the system changes at discrete time intervals accord-

ing to probability distributions which model the relationships between entities.
Simulation models may be used in combination with mathematical programming or black-box methods, in order to solve stochastic problems through optimization [2]. In the first scenario, random variables are used in the closed-form model of the problem, and static simulations estimate the expectations in closed-form solutions. In the second scenario, probability distributions are employed to generate random samples and mimic the behavior of the system. Therefore, dynamic simulations estimate the objective function at different values of the decision variables.

1.3.1 Control optimization

DP formulations are employed to model a discrete-time dynamic system which changes its state over a finite number of time intervals, also called stages. At each stage some form of control has to be applied to the system, in order to take a decision which leads to the optimal solution.
In control optimization problems, the optimal control at each stage is the best action (deterministic case) or the best policy (stochastic case) which can be selected from a set of feasible possibilities that are available at the current stage. Policies are functions which map all possible states at a stage into actions, and in the stochastic setting they are necessary because the formulation at each stage involves some form of randomness which might depend on the current state.
Each stage is associated to a cost function which is additive over time. For an initial given state, the total cost of a control sequence corresponds to the sum of the cost of all the states from the initial to the

final state. In the discrete-time setting, the objective of DP is to find the control sequence with minimum total cost. According to Bellman's principle of optimality [9], the optimal cost function can be constructed backwards in a sequential manner, by computing first the optimal cost function from the states at the last stage, then for the states at the previous stage and so on, until the initial stage is reached. Since the optimal cost function at each stage is based on the optimal costs of previous stages in a backwards manner, the cost function at the initial stage is guaranteed to be optimal. Unfortunately, notice that at a certain stage the computation of the cost function must be done for all its possible states, and in practice doing so is often quite expensive because the state space might be significantly large [3]. Also, the information used to take optimal decisions becomes progressively available as the system evolves. Thus, at each stage, not all information is known: only partial information about the solutions of already visited subproblems is at disposal, and information about yet-to-be-visited solutions is not available.

In the stochastic setting, the cost function is based on the expectations of all the random variables involved in the model [3, 5]. This formulation requires the explicit calculation of several expectations, which can be computed exactly or approximately. Furthermore, transitions between states become probabilistic. A significant part of the operations research literature defines methods for finding exact expressions of transition probabilities in complex dynamic systems for different applications. Unfortunately, these formulations are often complicated and contain multiple integrals or advanced algebra [2]. As the complexity of the problem grows, the more complicated is to find tractable solutions based on exact DP formulations. A possible alternative is to approximate the optimal cost function at each stage using a more

tractable function. According to this approximation, the control at each state is obtained by optimization of the cost over a limited horizon (number of stages), plus an approximation of the optimal future cost. The latter may be computed by a variety of methods, possibly involving simulation or some heuristic. Thus, the use of static simulation (model-free) often allows for implementations that do not require a mathematical model (model-based). For additional information about possible methods to provide DP approximations, please refer to [3].

1.3.2 Parametric optimization

When the closed form of the objective function is too hard to define, or the transition probabilities in the DP formulation lead to untractable models, using dynamic simulations to approximate the objective function is a valid alternative.
In discrete-event simulation the state of the system changes at discrete time intervals, so each time that an event is realized from the probability distribution used to model the respective entity. Dynamic simulations reduce complex models to a set of basic events and interactions, opening the possibility to encode a system through a set of rules which define the simulation logic. The level of representation detail depends on the requirements set by the decision maker, who might require the simulation to approximate the real system up to a certain accuracy level in order to properly justify meaningful decisions. However, the higher the level of simulation detail, the heavier the computational complexity. Also, the computational cost of simulations depends on the number of simulation replications used to take statistically significant decisions when comparing possible solutions. More formally, let F be a simulation that models a real-world problem, and its output to

depend on some decision variables x and on a random vector ξ which represents the random variables involved in the simulation model. The expectation of F is defined as

$$f(x) = \mathbb{E}[F(x, \xi)] \tag{1.1}$$

and it can be estimated by using a sample $\xi_1, ..., \xi_n$ of independent identically distributed (i.i.d.) realizations of the random vector ξ, in order to compute the sample mean as

$$\hat{f}_n(x) = \frac{1}{n} \sum_{i=1}^{n} F(x, \xi_i) \tag{1.2}$$

and the sample variance as

$$\hat{\sigma}_n^2(x) = \frac{1}{n-1} \sum_{i=1}^{n} (F(x, \xi_i) - \hat{f}_n(x))^2. \tag{1.3}$$

If the sample $\xi_1, ..., \xi_n$ is i.i.d., by the Law of Large Numbers, as n approaches infinity $\hat{f}_n(x)$ converges to $f(x)$ and so $\hat{f}_n(x)$ is an unbiased estimator of $f(x)$. Moreover, if the variance of F is finite, by the Central Limit Theorem $\hat{f}_n(x)$ asymptotically follows a normal distribution with mean $f(x)$ and variance σ^2/n, where σ^2 is the variance of F. As a consequence, the accuracy of the estimation increases with the sample size n and decreases, but this also increments the computational burden.

1.4 Noise management

In simulation-based optimization, the optimization of a dynamic simulation model is usually subject to a limited budget constraint. Because of the noisy and computationally expensive nature of simulations, this budget should be efficiently allocated in order to obtain good and reliable solutions as soon as possible. If the state space is discrete and sufficiently small as in the case of a few hundreds of possible alternatives, statistical analysis techniques can be employed in a brute force manner to select the best alternative [10, 11]. But, when the state space is continuous or significantly large, black-box methods might be a more efficient choice in order to find a globally or locally optimal solution. Simulation-based optimization allows the decision maker to systematically search a large decision space without being restricted to a few alternatives. This capability greatly broadens the scope of simulation as an analysis tool for complex system design [12].
The estimator defined in Equation (1.2) can be used by black-box techniques to compute an approximation of the objective function at different solutions, in order to optimize $f(x)$. Therefore, in a minimization problem, the goal is

$$\min_{x \in \Theta} f(x), \tag{1.4}$$

where Θ is the constraints-defined region in which x assumes values. The output of F follows some distribution which may (or may not) vary across Θ: in the first case the noise is said to be heteroscedastic, while it is homoscedastic otherwise. But, since the noiseless value of the objective function is distorted, making correct comparisons between candidate solutions is not straightforward. Given any configuration x, $f(x) - \hat{f}_n(x)$ defines an error $\epsilon_n(x)$ that goes to 0 only in the limit of n

going to infinity. Thus, when comparing two configurations x_1 and x_2, $f(x_1) = \hat{f}_{n_1}(x_1) + \epsilon_{n_1}(x_1)$, $f(x_2) = \hat{f}_{n_2}(x_2) + \epsilon_{n_2}(x_2)$ and the difference between the estimators can be written as

$$f(x_1) - f(x_2) = \hat{f}_{n_1}(x_1) + \epsilon_{n_1}(x_1) - \hat{f}_{n_2}(x_2) - \epsilon_{n_2}(x_2), \tag{1.5}$$

which means that n_1 and n_2 should be chosen properly in order to take a statistically significant decision. If the noise is too high with respect to the difference between the true values of two candidates (*signal*), and so the signal-to-noise ratio is too low, the outcome of a comparison might be erroneous.

As stated by [13, 14], an established practice in simulation-based optimization is to introduce a statistical analysis after the optimization by using ranking and selection (R&S) algorithms [15, 16, 14, 17, 18]. This analysis aims at selecting, in a statistically significant manner, the best solution x^* which performs better among the finite set of k possibilities found during the optimization. According to this strategy, during the search each solution is estimated by using a static number of replications which is determined manually *a priori*, before the optimization. Unfortunately, a posterior analysis introduces additional computational burden and possibly optimal solutions might not even be analyzed in the R&S phase, because they are not visited during the search. Heuristic algorithms provide no optimality guarantee and, due to the presence of noise, estimates may be inaccurate. As a consequence, improving solutions could be discarded and the search might never explore some portions of the search space which would further improve x^*.

Statistical analysis techniques can also be integrated into black-box optimization algorithms during the search [19, 20, 21, 22, 23]. Since the

output of simulators is stochastic, the quality of each solution visited throughout the optimization should be estimated. Therefore, to obtain effective simulation-based optimization strategies, finding a tradeoff between precision of estimation and CPU time is essential. In order to deal with the presence of noise, black-box optimization methods generically have two alternatives: to increase the strength of the signal, or to reduce the effect of noise. In the first case, the signal is improved by adapting the search region according to the signal-to-noise ratio. Multiple variants of this strategy have been studied in the field of evolutionary algorithms, and it has been shown that the adaptation of the search region during the optimization has a relevant impact [24, 25]. In the second case, the effect of noise can be reduced by combining multiple solutions located in a restricted area of the search space (*implicit averaging*) [26, 24, 27, 25], or by evaluating multiple times each solution (*explicit averaging*) [28, 29, 30, 31, 20].
An analysis of the role that various heuristic black-box optimization techniques have in simulation-based optimization is provided in [32, 33], while other surveys which cover several aspects of simulation-based optimization are [2, 14, 18, 34, 35]. However, as highlighted by [36], less attention has been given to the impact that the method used to estimate the objective function has on the performance of search algorithms employed in simulation-based optimization approaches. Probably due to the implicit averaging effect, population-based algorithms gained a lot of attention in the noisy optimization literature. But, to the best of the writer's knowledge, no theoretical guarantee justifies this choice over single-point randomized algorithms like simulated annealing or other local search variants.

1.4.1 Variance reduction

Variance reduction techniques like common random numbers (CRN), control variates or antithetic sampling help to reduce the computational burden required to compare solutions in noisy optimization problems [37, 38].
Due to the simplicity of application, CRN are widely adopted in simulations [17]. For each comparison, common seeds are used to initialize and synchronize the pseudorandom number streams of paired simulations. The objective is to induce positive correlation among the performance of the solutions to be compared, in order to reduce the variance of their difference [39]. In fact, it is well known that the variance of the difference of two independent variables X and Y corresponds to the difference of the respective variances. But, if the variables are dependent, then

$$Var[X - Y] = Var[X] + Var[Y] - 2\, Cov[X, Y]. \tag{1.6}$$

Therefore, $Var[\hat{f}_{n_1}(x_1) - \hat{f}_{n_2}(x_2)]$ changes according to the amount of correlation between $\hat{f}_{n_1}(x_1)$ and $\hat{f}_{n_2}(x_2)$. Without using CRN, the realizations used to compute $\hat{f}_{n_1}(x_1)$ and $\hat{f}_{n_2}(x_2)$ are i.i.d, and because of that $Cov[\hat{f}_{n_1}(x_1), \hat{f}_{n_2}(x_2)] = 0$. On the contrary, using CRN synchronizes the realizations and introduces a positive correlation between $\hat{f}_{n_1}(x_1)$ and $\hat{f}_{n_2}(x_2)$. As a consequence, $Cov[\hat{f}_{n_1}(x_1), \hat{f}_{n_2}(x_2)] > 0$ and the total variance is reduced.

1.4.2 CRN and seed management

CRN synchronize the pseudorandom streams of the simulations used to evaluate different solutions. The same seed is employed to initialize

the pseudorandom generator in order to obtain evaluations based on the same stochastic realizations. However, there are multiple ways to manage the seeds used throughout the optimization.

In the statistical techniques proposed in this dissertation, the performance of a solution x is initially estimated by using $\xi_s, \ldots, \xi_{s+n_0}$ i.i.d. realizations, where s is the starting seed of the optimization and n_0 is the initial sample size. Once x has been evaluated on a seed i, which defines the respective realization ξ_i, the evaluation $F(x, \xi_i)$ is memorized. If a comparison which involves x requires $j > n_0$ realizations in order to take a statistically significant decision, and the last in-memory evaluation used to estimate x is based on ξ_i with $n_0 < i < j$, the missing realizations are $\xi_{i+1}, \ldots, \xi_j$. Otherwise, if $i \geq j$, in-memory evaluations are reused.

By following this approach the reuse of samples is maximized, but the optimization depends on the sequence of realizations initialized by ξ_s and might overfit. Erroneous decisions might be taken if a particular sequence is not sufficiently representative of the simulated problem at hand, especially with small sample sizes. However, this effect is counterbalanced as the optimization moves towards local optima. Due to the qualitative similarity of solutions, larger sample sizes are required in order to take statistically significant decisions. Therefore, the sample becomes more representative and reduces the risk of overfitting.

To further improve the robustness of the procedure, one can discard completely or partially the samples saved in memory. For example, after each comparison the best solution could drop all the previous evaluations, and a new set of realizations based on different seeds could be sampled. Another less computationally expensive option could substitute one or more evaluations of the best solution, always after each comparison, with other evaluations based on new seeds.

1.4.3 R&S algorithms

R&S problems were first formulated during the 1950s for agricultural and clinical applications [40]. Back then, selecting the best alternative among a discrete set of solutions was a common problem. However, samples needed to be collected through physical experiments like clinical trials, which were time-consuming and possibly required a significant amount of resources to setup the experiments. Thus, empirical data was often gathered in batches and statistical analysis techniques meant to detect significant differences among multiple alternatives were designed in a stage-wise fashion. Starting from the 1990s, this paradigm began to change due to the increasing number of experiments which were run through simulations, which required little setup time.

As stated by [41], a stage occurs whenever the simulation of a configuration is started in order to evaluate it. The idea of two-stage procedures is to first gather an initial sample of the configurations in the set, and then to define the sample size required to guarantee that the best configuration is selected in the second stage. Single-stage procedures cannot guarantee to find x^* among a set of k alternatives when the variances of the configurations are unknown [42]. In contrast, two-stage procedures can provide that guarantee [43].

In the simulation scenario samples can be collected sequentially, leading to the development of fully-sequential R&S procedures. Differently from two-stage procedures, fully sequential methods sequentially compute single observations and eliminate statistically inferior configurations as soon as a statistically significant decision can be taken. But, since most fully-sequential procedures use the Bonferroni inequality to guarantee the PCS, they tend to be too conservative [44]. For more

information about the evolution or R&S, please refer to [45].

1.5 Simulation for revenue management

Many problems in science, engineering, and finance are nowadays solved through the adoption of static or dynamic simulations [4, 8]. Some relevant areas of application include manufacturing, construction engineering, logistics and distribution, traffic management, healthcare, materials engineering and risk analysis. Another field with a significant simulation-based stream in the literature is revenue management (RM), where exact or approximate dynamic programming formulations have been applied to RM problems, mostly in combination with static simulations. But, in order to obtain tractable models, many of these procedures are bound by simplifying assumptions which limit their application in complex and large systems.

Initially developed by airlines in the 1970s, RM has become a common business practice in many other industries. The purpose of RM is to match supply and demand in order to maximize the revenue of a particular firm [46]. In fact, RM has been defined as the application of information systems and pricing strategies to allocate the right capacity of supply to the right customer at the right price at the right time [47]. However, while general RM ideas easily apply to different fields, each industry has specific characteristics which change the application of RM techniques. For example, successful strategies for airlines are not automatically good solutions for hotels and the two problems need to be considered separately [46].

When allocating a specific resource according to demand from different customer classes, airlines and hotels consumers can take decisions from different sets of possible actions. In the case of airlines, resources

are single flight legs and demand classes are leg itineraries, where a flight leg is a direct flight from an airport A to an airport B with no stops in-between. In the case of hotels, resources are single nights and demand classes are combinations of consecutive nights. The difference lies in the fact that airline customers generally travel at most two or three legs, while in hotels customers might stay for a week or more [48]. Another example is given by the group size of bookings: in hotels the group size is the number of rooms reserved for a booking, while in airlines it is the number of seats reserved for a certain leg. But in hotels a room might contain more than a person and in airlines each seat holds a person only. So capacities of resources change differently. Also, in airlines itineraries might be composed by multiple legs offered by different companies, while this rarely is an option taken in consideration by customers of the hotel industry. This dissertation takes a focus on the application of dynamic simulations on hotel RM.

1.6 Structure of the thesis

The remainder of the thesis is as follows.
Chapter 2 presents a flexible simulation-based optimization approach based on dynamic pricing for hotel RM [49]. Demand is simulated using a novel set of parametric models based on the RIM quantifiers [50], whose parameters are daily statistics which can be estimated from data. These models allow to change the curves parametrically, redistributing demand along the booking horizon, without requiring any change of advance historical data. Bookings and cancellations associated to each day are distributed along the booking horizon with a non-homogeneous Poisson process, where demand expectations of each day are defined by the parametric models. The hotel manager can in-

ject new information in the system, adapting pricing policies to the mutated conditions of the market.
Chapter 3 compares various black-box methods based on variations of local search methods in the context of noisy optimization [51]. In particular, a widely used population-based method called Covariance Matrix Adaptation Evolutionary System (CMA-ES) [52, 53] is tested in various noise scenarios as well as other randomized algorithms which are extended with a statistical analysis technique based on the probabilities of making an error of type I and type II. As will be shown empirically, the effect and the limits of implicit averaging depend on the type and the amount of noise in the objective function.
Chapter 4 investigates possible integrations of black-box optimization with the optimal computing budget allocation (OCBA) [10] and the indifference-zone (IZ) [54] formulations, two popular R&S methods which have been widely explored in the traditional R&S scenario. Differently from previous approaches, which evaluate solutions only according to independent random realizations, positive correlation is exploited in order to take statistically significant decisions as soon as possible. Correlation is especially important in noisy scenarios like simulation-based optimization, where CRN are commonly used to reduce the variance of samples [55].
Chapter 5 summarizes results and provides possible directions for future work.

Chapter 2

Simulation for Hotel Revenue Management

2.1 Hotel revenue management

Information technology drastically changed how people plan travels and accomodations. In fact, tools such as online travel agencies (OTAs) or price comparison websites are now extensively used [56], and hotels are no longer forced to sell their rooms only through traditional intermediaries. Many hotel chains have already adopted RM techniques to manage their availability of rooms, in order to maximize their revenue. The situation is very different in the case of independent hotels, where the management of prices is often still manually tailored and RM is indirectly applied only on a subset of rooms through intermediaries like OTAs.

Optimization problems related to hotel RM are usually expressed following two approaches: capacity control [57, 58, 59, 60, 61, 62, 63, 48], where the decision variable is the amount of offered supply, and dynamic pricing [64, 65, 66, 67, 23], where the price represents the decision

variable. In both cases, several mathematical optimization methods have already been proposed to maximize revenue [46, 68, 69]. Many of these formulations assume that demand is independent from the chosen policy. More complex scenarios, where demand is influenced by other factors (e.g., price), are more difficult to handle and closed-form solutions are rarely available [70]. Demand is usually considered as a known deterministic function or as a stochastic function following a known distribution family with unknown parameters. Also, if stochastic cancellations are considered, the CPU time for solving the problem tends to grow exponentially and approaches like dynamic programming are effective only in specific cases [71]. A possible solution to mitigate the complexity of the model is approximate dynamic programming [60, 23, 48], where the problem is partitioned into simpler subproblems. Nonetheless, even approximate models cannot provide sufficiently tractable solutions for realistic scenarios because of the large number of possible states [72].

The maximization of revenue can be achieved by using dynamic simulations in combination with black-box optimization [73, 74]. The analytical model is substituted with a simulator of many inter-related processes like reservations, cancellations, no-shows, walk-ins, and black-box optimization is employed to find the policy which maximizes the revenue. An effective technique to maximize revenue and simulate different stochastic aspects of the hotel booking scenario is discrete-event simulation [75]. The generation of reservations and cancellations leads to a distribution of possible revenues, and the expected value of the distribution is considered as the variable to be maximized. For example, [66] employ a dynamic simulation approach to simulate demand as the result of many stochastic processes, and [67] also considers the effect of price on demand.

Existing simulation-based optimization approaches create empirical demand curves which cannot be easily modified if the current market situation deviates from the past [67, 65, 66]. This Chapter introduces HotelSimu, a simulation-based optimization approach for hotel RM with flexible parametric demand models which can be modified in order to adapt pricing policies to mutated market conditions. Furthermore, reservation requests and cancellations are not grouped into disjoint sets of events like in [66], but occur in an interleaved manner.
The structure of the Chapter is as follows. Section 2.2 provides a detailed analysis of relevant dynamic programming schemese for hotel RM. Section 2.3 describes HotelSimu, and defines more in detail its parametric models. Section 2.4 provides some details about the optimization procedure, and Section 2.5 shows the applicability of our models to a set of hotels in Trento, Italy. Results show that the use of HotelSimu leads to an average revenue increase similar to that of other dynamic pricing strategies, even though only aggregated data is used. Finally, Section 2.6 provides the main implications for the hotel manager.

2.2 Simulation and dynamic pricing

A possible dynamic pricing approach for hotel RM is proposed by [65], and it is based on the idea that hotels usually define a discrete set of price categories, where each category corresponds to a certain number of rooms. Low-priced categories are assigned to early reservations and, as they get booked, high-priced categories are used. As a consequence, assigning too many rooms to the lower-priced category will lead to more bookings, but at the expense of potentially losing higher revenue from higher-priced categories (lost opportunity). The opposite

problem would happen exchanging low-priced with high-priced rooms. The goal of the paper is to find the optimal price for each category, in each night, to maximize the total revenue. The objective function is defined as the sum over nights of the multiplication of the price of each night and the number of rooms reserved in that night. Moreover, the only restriction in the model is that the total number of reservations for each night does not exceed the associated capacity. The optimization problem, formulated as a non-linear programming problem, takes into account future bookings and their probabilities, considering the price elasticity of demand as well. In fact, price elasticity is considered separately from arrivals; a price drop results in an increase in demand, and vice versa. Furthermore, instead of using pre-defined probability distributions in order to approximate arrivals, historical data is used to estimate demand.

Another dynamic pricing approach is proposed by [67], and it employes a full simulation-based optimization approach in order to propose prices in a dynamic manner. The idea is based on the fact that usually hotels define a reference price for each seasonality that they consider (i.e. high season, middle season, low season). This reference price can be seen as the average price that hotel managers are willing to offer for their rooms in different periods of the year. As a consequence, when considering historical data, the implicit assumption is that the state of the hotel in different seasons also depends on the reference price that has been proposed. This means that the price elasticity of demand can be set by using the reference price as starting point.

The structure of the simulation-based optimization approach proposed by [67] involves three main entities: the simulation model that simulates the performance of different configurations of the pricing model, the optimization strategy that uses the results of the simulator to guide

the optimization process in order to maximize the objective function, and the dynamic pricing model to be optimized.
A dynamic simulator is employed to mimic the booking process of the hotel, in order to estimate the value of the objective function which can be obtained by proposing prices according to a specific configuration of the multipliers of the dynamic pricing model [66]. In fact, given a configuration of the parameters, the price of each reservation request in the simulation can be computed. The objective function to be maximized is the total revenue, defined by the sum of the revenue of all reservations in the simulation which are not canceled before the arrival of the customer. Once a configuration has been evaluated, it is compared with respect to the current best solution. Then the black-box optimization model proposes another configuration to be evaluated and the optimization loop continues as long as the budget, which is set before the whole optimization starts, is not over.
The dynamic pricing model is built using four linear multipliers which, for each reservation request, are used to propose a price obtained by multiplying the reference price. In fact, these multipliers take values in the interval $[0.60, 1.40]$, without modifying the reference price too much so that it would be too cheap or too expensive for the customers. Thus, each multiplier changes the reference price according to the value it assumes: a multiplier value lower than 1 corresponds to a discount and a value larger than 1 is a price increase. It is assumed that proposed prices influence demand, and this relation is expressed through a price elasticity function which considers the reference price as starting point: prices higher than the reference price reduce demand and viceversa. As a consequence, customers are going to accept or deny reservation offers with different probabilities. The four linear multipliers represent different features that characterize a reservation and

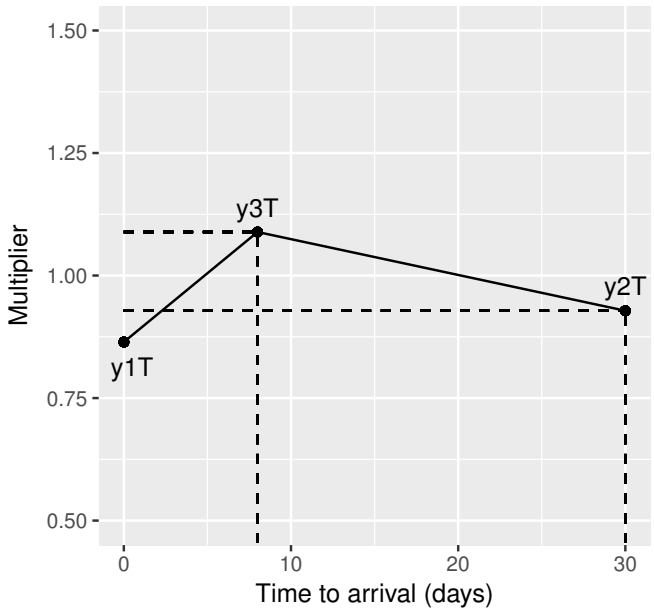


Figure 2.1: Time to arrival multiplier, defined between 0 and 30, the booking horizon. The x-axis defines the number of days between booking and arrival date.

influence its price:

- M_T, the number of days between booking and arrival date
- M_C, the capacity of the hotel at booking time
- M_L, the length of stay (LoS)
- M_S, the number of reserved rooms (group size)

The time-to-arrival multiplier M_T is the only multiplier defined as a piecewise linear function. Its shape, which is guaranteed by constraints defined on the parameters of the multiplier, can be seen in Figure 2.1. The x-axis defines the time between booking and arrival date, where time equal to 0 means that the booking date corresponds to the arrival date (walk-in customers). The number of days in advance in which

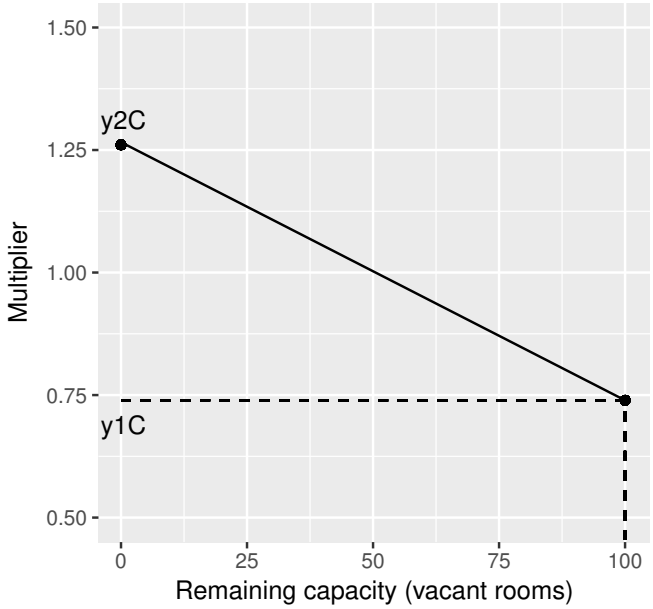


Figure 2.2: Capacity multiplier, defined between 0 and 100, the total capacity of the hotel. The x-axis defines the remaining capacity: prices are higher when remaining capacity is lower.

booking is possible is called booking horizon. The y-axis represents the value of the multiplier itself. The idea is that when the booking date is still far away from the arrival date, prices should be lower in order to encourage bookings. So, in the last day of the booking horizon, prices should be lower than the reference price and $y2T$ defines the multiplier value in that day. As the arrival date gets closer, prices are raised until a certain point $y3T$, which usually is defined a few days before the arrival date, when eventual vacancies need to be filled. Thus, bookings should be encouraged again by lowering the multiplier to $y1T$.

The capacity multiplier M_C is defined as a monotonically decrescent function. Its shape can be seen in Figure 2.2, where the x-axis defines the remaining capacity of the hotel. The idea is that when the

remaining capacity is low, the value of remaining rooms increases and so prices should increase as well. In contrast, when there are many vacant rooms, the value of rooms is lower and prices are decreased. As a consequence, when prices are higher, more expensive rooms are going to be booked by high-paying customers. $y1C$ and $y2C$ define the slope of the multiplier.
Similar arguments apply for the other multipliers. The LoS multiplier M_L and the group size multiplier M_S are also defined as monotonically decrescent functions. As the length of stay or the number of rooms of a reservation grows up to a limit set by the user, the relative price should be lowered. Slopes of the multipliers are defined by the relative parameters. As previously mentioned, the optimization strategy maximizes the objective function by changing the values of the parameters of the multipliers, and evaluating the performance of each configuration. The parameters of the multipliers are enumerated in table 2.1. Notice that, if such parameters are not optimized, then the offered price corresponds to the reference price and the multipliers have a horizontally flat shape. More formally, let us define a reservation R as a vector of the features that characterize it, so

$$R = \{R_T, R_C, R_L, R_S\} \tag{2.1}$$

where R_T is the time to arrival, R_C is the remaining capacity of the hotel when the reservation happens, R_L is the LoS and R_S is the group size. The final unit price P of a reservation R corresponds to the price for 1 room and 1 night used to compute the final price of reservation offers. P is given by the product of the reference price P_{ref} and the multipliers function M, limited around P_{ref} by Δ in the interval $[(1-\Delta)P_{ref}, (1+\Delta)P_{ref}]$, with a saturation speed proportional

Multiplier	Parameters
M_T	y1T, y2T, y3T
M_C	y2C
M_L	y2L
M_S	y2G

Table 2.1: HotelSimu's dynamic pricing model parameters

to η. Let us define it as

$$P(R) = P_{ref} \cdot F(M(R), \Delta, \eta) \tag{2.2}$$

where

$$M(R) = M_T(R_T) \cdot M_C(R_C) \cdot M_L(R_L) \cdot M_S(R_S) \tag{2.3}$$

and

$$F(M(R), \Delta, \eta) = (1 - \Delta) + 2\Delta \cdot \Phi(\eta \cdot (M(R) - 1)), \tag{2.4}$$

where $\Phi(\cdot)$ is the cumulative distribution function (c.d.f.) of the standard normal distribution. As previously mentioned, the final price obtained using the multipliers influences demand according to a price elasticity function. This function is defined as a probit function which, given a price P_{norm} normalized respect to the reference price, maps it to a certain demand index DI. A probit function Φ^{-1}, which is the inverse of the c.d.f. of the standard normal distribution, has been used in order to consider saturation effects for large price variations. Small changes of the reference price should influence the customers' decision less than large variations. DI defines the demand associated

to a certain price, and it is computed as

$$DI(P_{norm}) = \Phi^{-1}(\frac{P_{norm} - 1}{a}) + 0.5, \tag{2.5}$$

where a demand index equal to 1 is the mean demand obtained when using the reference price. The scaling parameter a defines the slope of the function, mapping prices to demand. According to this parameter, price changes have more or less influence on the decision of customers. So, given the normalized price of a reservation, the simulator checks the relative demand factor x and there are two possibilities. If x is lower than 1, then the reservation is considered as accepted with a probability equal to x; otherwise it is rejected. In contrast, if x is greater than 1, the reservation is accepted and a new reservation is generated with a probability equal to $x - 1$.
The simulator used to assess the performance of configurations models the booking scenario of a hotel by considering stochastic reservations, arrivals and cancellations [66]. It also models reservations with variable length of stay, no-shows and group reservations. More precisely, these processes are simulated forward in time after having estimated the parameters of distributions from historical data. And since all the involved processes are modeled as probabilistic distributions, the same simulation has to be repeated multiple times in a Monte Carlo fashion in order to obtain a sufficiently accurate estimation.

2.3 HotelSimu

The main components of HotelSimu are shown in Figure 2.3. An event generator simulates the reservation requests and the cancellations. A registry stores the information about the state of the hotel, in partic-

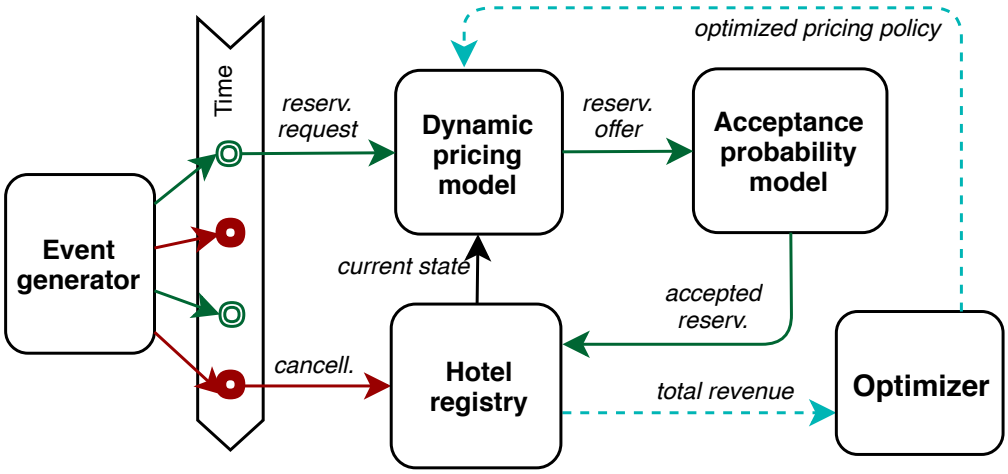


Figure 2.3: HotelSimu overview. Reservation requests and cancellations are interspersed. The state of the hotel after one complete simulation is used by the optimizer to compute the total revenue and adjust the pricing policy.

ular accepted reservations and room availability. A dynamic pricing model proposes an offer for each reservation request, and an acceptance probability model simulates the stochastic process by which customers accept or discard reservation offers. An optimizer searches for the optimal pricing policy to maximize revenue.

2.3.1 Definitions

Let us define the main concepts and the notation used throughout the rest of the Chapter.

Definition 1. A *reservation request* (RR) is an event characterized by the following features. The *reservation day* (RR_{res}), which is the day the request occurs. The *arrival day* (RR_{arr}), which is the day the customer arrives at the hotel. The *length of stay* (RR_{los}), which is the number of nights reserved. The *size* (RR_{size}), which is the number of

rooms reserved.

Definition 2. A *reservation offer* (RO) is an admissible reservation request (for which there is room availability) characterized by the *price* (RO_{price}) proposed by the hotel, which depends on the features of RR.

Definition 3. An *accepted reservation* or simply *reservation* (R) is a reservation offer accepted by the customer. It is registered on the hotel registry and it effectively changes room availability.

Definition 4. The *acceptance probability* of a reservation offer ($\text{Pr}_{\text{accept}}(RO)$) is the probability that a customer accepts RO and the proposed price, and therefore is equal to the probability that RO is registered on the book.

Definition 5. The *state of the hotel* $S(t)$ is defined as the state of the booking registry at time t, which corresponds to the historical records up to t as well as the set of reservations for future arrival days that are in the registry at time t.

Definition 6. Given two days identified by $i, j \in \{0, 1, 2, \dots\}$, the number of days between i and j, or their *distance*, is $d(i, j) = d(j, i) = |i - j| \geq 0$.

Definition 7. Given a reservation R, the *time-to-arrival* of R is $R_{\text{TTA}} = d(R_{\text{res}}, R_{\text{arr}})$. If $R_{\text{TTA}} = 0$, a customer makes a reservation on the arrival day or arrives at the hotel with no reservation and the customer is referred to as a *walk-in* user.

Definition 8. The booking time window or *booking horizon* (BH) is the maximum time-to-arrival allowed by the hotel.

Definition 9. A *cancellation* (C) is characterized by the *cancellation day* (C_{day}), which is the day the event occurs, and the *reservation* (C_{res}), which is the reservation on the book that is canceled by the customer. When a reservation is canceled, it is removed from the hotel registry and the associated rooms can be booked by other customers.

Definition 10. The *cancellation probability*, t days before arrival of a reservation R ($\text{Pr}_{\text{cancel}}(R, t)$), is the probability that the customer associated with R cancels it exactly t days before arrival, with $t \in [0, R_{\text{TTA}}]$. According to this definition, the probability that R is canceled within its lifetime is

$$\text{Pr}_{\text{cancel}}(R) = \sum_{t \in [0, R_{\text{TTA}}]} \text{Pr}_{\text{cancel}}(R, t). \tag{2.6}$$

Definition 11. The *reservation requests horizon* (RH) is the set of all the reservation days to be simulated. It corresponds to the values that each R_{res} can assume during the simulation.

Definition 12. The *arrivals horizon* (AH) is the set of all possible arrival days. It corresponds to the values that each R_{arr} can assume during the simulation.

Definition 13. The *optimization horizon* (OH) is the set of arrival days for which there is the need of an optimal dynamic pricing policy to maximize revenue.

For each simulated reservation day $r \in \text{RH}$, a random sequence of $\mathcal{C}_r$ cancellations and $\mathcal{R}_r$ reservation requests is generated. Each reservation request is associated with an arrival day $a \in \text{AH}$ following or coinciding to r ($a \succeq r$), and each cancellation is associated with a

registered reservation. The proposal of a price depends on a reservation request and on the state of the hotel at the moment the event occurs. Once a price has been proposed to the customer, a reservation is accepted according to the acceptance probability model. It is then registered into the hotel registry and, if a cancellation does not occur until the end of the simulation, it is considered in the evaluation of the total revenue to be passed to the optimizer. As concerns the optimization, one objective function evaluation corresponds to the average total revenue of several simulation runs, with respect to the reservations recorded in the registry within the OH.

2.3.2 Simulation of reservation requests

Let $\mathcal{R}_r^a$, $r \in \mathrm{RH}, a \in \mathrm{AH}$, be the number of reservation requests generated on day r that are associated with arrival day a. The total number of requests generated within RH and associated with one arrival day is therefore given by:

$$\mathcal{R}^a = \sum_{\substack{r \in \mathrm{RH} \\ r \preceq a}} \mathcal{R}_r^a, \tag{2.7}$$

where $\preceq$ describes the relation *precedes or coincides to*. The series of reservations generated within RH and associated with a defines its *reservation curve*. In fact, the expected total number of reservation requests associated with one arrival day can be seen as the result of several independent processes, which occur on each simulated day within the BH of an arrival day:

$$\mathbb{E}[\mathcal{R}^a] = \Lambda(a) = \sum_{i=0}^{\mathrm{BH}} \lambda(i, a), \tag{2.8}$$

where $\lambda(i, a)$ is the expected number of reservation requests occurring i days before the arrival day a. If historical data is available, one can estimate directly $\lambda(i, a)$ for each i and a. However, to avoid the computational load of a point-wise estimation and to facilitate what-if analyses, each $\lambda(i, a)$ is defined by the following model:

$$\begin{aligned} \lambda_\alpha(i, a) &= \Lambda(a) \times Q_\alpha(i, \mathrm{BH}) \\ &= \Lambda(a) \times \left(\left(\frac{\mathrm{BH} + 1 - i}{\mathrm{BH} + 1} \right)^\alpha - \left(\frac{\mathrm{BH} - i}{\mathrm{BH} + 1} \right)^\alpha \right), \end{aligned} \quad (2.9)$$

with $i = 0, 1, \ldots, \mathrm{BH}$, $a \in \mathrm{AH}$, and for any parameter $\alpha > 0$. Equation 2.9 is used to formulate a parametric reservation curve as a monotonically decreasing function, whose maximum value corresponds to the number of walk-in customers of a. The model is then parametrically set in order to obtain curves like the ones in [66], which are estimated from historical data. Similarly to the RIM quantifiers proposed in [50], $Q_\alpha(i, \mathrm{BH})$ defines a weight for each day i in order to distribute $\Lambda(a)$ along the BH, while also guaranteeing a few properties:

- it defines a function with discrete domain and continuous values;
- it sums up to 1:

  $$\sum_{i=0}^{BH} Q_\alpha(i, \mathrm{BH}) = 1,$$

  for any $\alpha > 0$ and therefore can represent a discrete probability distribution or a normalized curve;
- it can model different reservation scenarios through α, from a constant curve ($\alpha = 1$) to increasing and decreasing curves (see Figure 2.4);

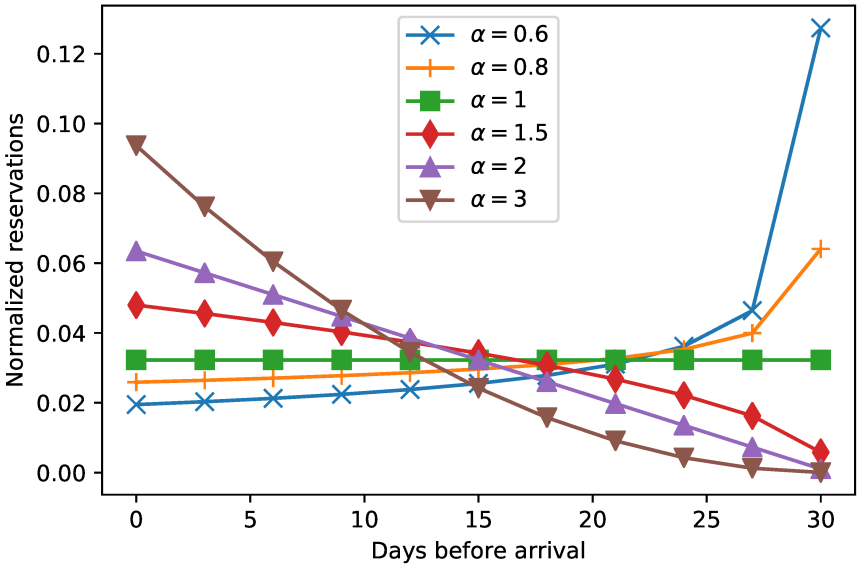


Figure 2.4: $Q_\alpha(i, \mathrm{BH})$ for $\mathrm{BH} = 30$ and for different values of α.

- it provides a simple way of finding α from the ratio of walk-in users with respect to the total number of reservations, that is, $Q_\alpha(0, \mathrm{BH})$.

In the current implementation, the reservation requests are assumed to follow a non-homogeneous Poisson process with an expected value given by the parametric model, so $\mathcal{R}^a \sim \mathrm{Poisson}(\Lambda(a))$. Therefore, reservation requests are generated for each simulated day according to the following model:

$$\begin{cases} \mathcal{R}_r^a \sim \mathrm{Poisson}(\lambda_\alpha(i, a)) & \text{if } i \leq \mathrm{BH}, \\ \mathcal{R}_r^a = 0 & \text{otherwise.} \end{cases} \tag{2.10}$$

Poisson processes are usually chosen to model arrival processes [76] and can represent the arrival of reservation requests with a minimum set of parameters. In [66] a binomial distribution is used, with additional constraints on the variance of samples in order to set the success

probability and the number of trials. However, a binomial distribution converges to a Poisson distribution when the number of trials (e.g., customers generating requests) grows. Removing the limit on the pool of customers that can generate new reservations makes the model more realistic, since the number of possible customers is usually unbounded and independent from the capacity of the hotel. The use of $\Lambda(a)$ is based on the assumption that it is possible to estimate the expected number of accepted reservation requests for a specific arrival day which are not canceled ($\mathcal{R}^a_{\text{accept}}$). Similarly, it is assumed that one has access to the expected number of reservation requests for a specific arrival day that are accepted by the customers and canceled ($\mathcal{R}^a_{\text{cancel}}$). $\mathcal{R}^a_{\text{accept}}$ can be approximated by the expected number of arrivals, while $\mathcal{R}^a_{\text{cancel}}$ can be seen as the expected number of cancellations.

HotelSimu includes also a model of the acceptance probability $\text{Pr}_{\text{accept}}(RO)$. A model of probabilities (possibly one for each admissible input) can be estimated from data retrieved by an online booking platform, where one can keep track of users that search for a room and decide to finalize the reservation or leave the website. One can also estimate the expected acceptance probability $\mathbb{E}[\text{Pr}_{\text{accept}}(RO)]$ as the expected fraction of reservation requests that are finalized by the users after the search. Therefore, the expected total number of reservation requests (accepted or rejected) associated with one arrival day can be estimated as follows:

$$\mathbb{E}[\mathcal{R}^a] = \Lambda(a) \approx \frac{\mathcal{R}^a_{\text{accept}} + \mathcal{R}^a_{\text{cancel}}}{\mathbb{E}[\text{Pr}_{\text{accept}}(RO)]}. \tag{2.11}$$

2.3.3 Simulation of nights and rooms

Let $nights^a$ be the expected number of nights for a reservation associated with an arrival day a. Analogously, $rooms^a$ is the expected

number of rooms. *max-nights*^{a} and *max-rooms*^{a} represent the limits imposed by the hotel manager. Since each reservation request includes at least one night and one room, and the distributions of additional nights and rooms observed in [66] have an exponential-decay behavior which can be simulated by a Beta distribution, the discrete probability distribution of the number of additional nights/rooms is modeled as

$$\Pr(X-1=k) = \int_{\frac{k}{\max(X)}}^{\frac{k+1}{\max(X)}} \frac{(1-x)^{\frac{\max(X)}{\text{avg}(X)-0.5}-2}}{\text{B}(1, \frac{\max(X)}{\text{avg}(X)-0.5}-1)} dx, \tag{2.12}$$

where X is the number of nights/rooms, $X-1$ is the number of additional nights/rooms, $\max(X)$ is either *max-nights*^{a} or *max-rooms*^{a}, and $\text{avg}(X)$ is either *nights*^{a} or *rooms*^{a}. $k = 0, 1, \ldots, \max(X)-1$, and $\text{B}(\alpha, \beta)$ is the Beta function with parameters α and β.

The previously defined distribution is a discrete analogue of a (continuous) Beta distribution with $\alpha = 1$ and $\beta = \frac{\max(X)}{\text{avg}(X)-0.5} - 1$. The value of α is chosen so as to have a distribution with an exponential-decay profile. β is chosen so as to have an expected value approximately equal to $\text{avg}(X)-1$. This is achieved by imposing the equality of the expected value of the (continuous) Beta distribution, which is $\frac{\alpha}{\alpha+\beta}$, to the expected number of additional nights/rooms rescaled to $[0, 1]$, which is $\frac{\text{avg}(X)-0.5}{\max(X)}$. A correction of 0.5 is considered to account for the discretization error (due to the approximation of the continuous distribution using discrete intervals), in order to position rescaled expected values of the distribution in the middle of the discretization interval. Experiments show that the maximum error between the expected values and the empirical averages of the discrete analogue with $\max(X) = 5$ is at most 0.33, for expected values equal to $0, 0.1, 0.2, \ldots, \max(X)-1$.

Even though modeling the length of stay or the number of rooms as Bernoulli or Poisson processes provides a simple and exact way of imposing the expected value, it is not applicable to our context, which cannot be reduced to a coin toss or to an arrival process. In the literature, the Beta distribution is often used to model unknown probability distributions, with shapes that can be controlled by the parameters α and β. By building a discrete analogue of a Beta distribution, it is possible to exploit its macroscopic features and to obtain a realistic model of the variable of interest. A similar model can be defined also for *group* reservations, which usually follow a different distribution from that of the length of stay of *normal* reservations. This can be easily achieved by considering a different value for $\text{avg}(X)$. By following (2.12), an instance of the random variable X, which is either RR_{los} or RR_{size}, is generated as $X = 1 + \lfloor Y \times \max(X) \rfloor$, where $Y \sim \text{Beta}(1, \frac{\max(X)}{\text{avg}(X)-0.5} - 1)$.

2.3.4 Simulation of cancellations

Under the same assumptions of Section 2.3.2, and by analogy to (2.6), the probability that a reservation is canceled during its lifetime can be seen as the summation of the probabilities that a reservation is canceled exactly on a specific day within its lifetime:

$$\Pr_{\text{cancel}}(R) = \Omega(a) = \sum_{i=0}^{R_{\text{TTA}}} \omega(i, a), \tag{2.13}$$

where $\omega(i, a)$ is the probability that R is canceled exactly i days before the arrival day a, with i within its lifetime. Each $\omega(i, a)$ is defined by

the following parametric model:

$$\begin{aligned}\omega_\alpha(i,a) &= \Omega(a) \times Q_\alpha(i, R_{\text{TTA}}) \\ &= \Omega(a) \times \left(\left(\frac{R_{\text{TTA}} + 1 - i}{R_{\text{TTA}} + 1} \right)^\alpha - \left(\frac{R_{\text{TTA}} - i}{R_{\text{TTA}} + 1} \right)^\alpha \right) \end{aligned} \quad (2.14)$$

with $i = 0, 1, \ldots, R_{\text{TTA}}$, $a = R_{\text{arr}}$, and for any parameter $\alpha > 0$. In this context one can also find α from the fraction of cancellations that occur on the last day ($Q_\alpha(0, R_{\text{TTA}})$), which includes the so-called *no-shows*. $\Omega(a)$ can be estimated as follows:

$$\Omega(a) \approx \frac{\mathcal{R}^a_{\text{cancel}}}{\mathcal{R}^a_{\text{cancel}} + \mathcal{R}^a_{\text{accept}}}, \quad (2.15)$$

with an arrival day $a = R_{\text{arr}}$. In HotelSimu, different stochastic cancellation scenarios can be simulated by changing $\omega_\alpha(i, a)$ through $\Omega(a)$ and α.

2.4 Optimizing the noisy simulator

Since simulations employs stochastic processes, the performance of each solution corresponds to a distribution of results. The expected value of the distribution is used as an approximation of the objective function to be optimized, so the optimization operates in the presence of noise.

In the literature, multiple works tested diverse heuristic algorithms on noisy functions, and they have shown that population-based approaches like CMA-ES are a good choice to optimize noisy functions [77, 78, 79, 25]. In fact, instead of relying only on a single solution, at each iteration CMA-ES combines a subset of its candidate solu-

tions in order to direct the search in the most promising direction. By combining multiple solutions located in a restricted area of the search space, the impact of noise is decreased due to an *implicit averaging* effect [80, 25]. Moreover, to further reduce the effect of noise on the optimization, the performance of each solution can be computed as the mean of the outcome of multiple simulations. From probability theory, one knows that the effect of noise can be reduced by evaluating multiple times each solution [80].

CMA-ES is an evolutionary optimization algorithm in which a multivariate normal distribution $N(\mu_t, M_t)$ is used to sample solutions, where t defines the iteration of the algorithm. At each iteration, the mean μ_t defines the center of the distribution, while the covariance matrix M_t determines shape and orientation of the ellipsoid corresponding to $N(\mu_t, M_t)$. Also, a step size σ_t controls the spread of the distribution as a percentage of the search space. Iteratively, CMA-ES follows the following steps. First, a population of λ solutions is sampled from $N(\mu_t, M_t)$. Second, candidates are evaluated and ranked according to the respective evaluations. Third, the best $\lfloor \frac{\lambda}{2} \rfloor$ results are used to update μ_t and M_t, in order to move the search towards the most promising search direction. Fourth, σ_t is increased or decreased according to the length of the so-called evolution paths. Evolution paths are weighted vector sums of the last points visited by the algorithm. They provide information about the correlations among points, and they are used to find the direction recently followed by the optimization. If consecutive steps are going in the same direction, the same distance could be covered by longer steps and the current path is too long. If consecutive steps are not going in the same direction, single steps tend to cancel each other out and so the current path is too short.

CMA-ES is executed with $\lambda = 4 + \lfloor 3 \log d \rfloor$ and $\sigma = 0.5$, where d is

the dimensionality of the objective function and $\sigma \in (0, 20]$. The size of the population is the one suggested by the authors of [52], who have also tested that CMA-ES with this population size is a robust and fast local search method [81]. Also, all the standard stopping criterias of CMA-ES are active[82]. Each time a stopping criteria is triggered, the algorithm is restarted from another randomly generated point in the search space, with a new population of the same size.

2.5 Results

The following experiments show how HotelSimu can be used to search for the optimal pricing policies that maximize the total revenue of a set of hotels of different sizes. Tests assume the presence of only one category of rooms in the hotel, and that at least historical data about final demand is available. However, if advance historical data is also available and empirical demand curves can be estimated, the models can be calibrated using optimization algorithms [83].

2.5.1 Setup of the experiments

Reservation curves are considered as monotonically decreasing functions with 40% of the customers treated as walk-in users, and these models are calibrated according to the curves estimated from historical data in [66]. The goodness of this choice is also confirmed by data collected by the Italian Institute of Statistics (Istat) on the features of trips[1], which show that approximately 40% of the interviewed people travel without booking. As a consequence, it is reasonable to assume

[1] http://dati.istat.it/?lang=en; section: Communication,culture,trips/Trips/Trips and their characteristics.

that the remaining 60% of the reservations is monotonically distributed in the BH in a decreasing fashion as moving away from the walk-in day. Also, it is assumed that the maximum number of cancellations occurs on the last day, and this number is fixed to 40% of the total number of cancellations. BH is fixed to 180 days, the maximum number of nights for one reservation to 10, and the maximum number of rooms to 4. As concerns the pricing policy, the model proposed in [67] is used. The multipliers vary around 1, and each multiplier changes the reference price according to the value it assumes. The price RO_{price} proposed to a customer corresponds to the unit price for 1 room and 1 night, and it is computed as

$$RO_{\text{price}} = price^a \cdot \xi(RR_{\text{TTA}}, RR_{\text{los}}, RR_{\text{size}}, S, \Delta, \eta), \tag{2.16}$$

where $price^a$ is the expected unit price for customers arriving on day a, and $\xi(\cdot)$ is a function of the reservation request features and of the hotel registry, with average value equal to 1. This function smoothly adjusts the price within the interval $[(1-\Delta)price^a, (1+\Delta)price^a]$, with a slope proportional to η:

$$\begin{aligned} &\xi(RR_{\text{TTA}}, RR_{\text{los}}, RR_{\text{size}}, S, \Delta, \eta) = \xi(t, l, s, S, \Delta, \eta) = \\ &= (1 - \Delta) + 2\Delta \cdot \Phi(\eta \cdot (M_T(t)M_L(l)M_S(s)M_C(S) - 1)). \end{aligned} \tag{2.17}$$

$\Phi(\cdot)$ is the c.d.f. of the standard normal distribution, and $M_T(\cdot)$, $M_L(\cdot)$, $M_S(\cdot)$ and $M_C(\cdot)$ are functions (or *multipliers*) of the time-to-arrival, the length of stay, the number of rooms and the remaining hotel capacity at the moment the reservation request is generated, respectively. The parameters of the multipliers are set as $T_0 = 30$ and $C_0 = L_0 = G_0 = 1.6$. Also, $\eta = 3$ and $\Delta = 0.6$ in order to pro-

pose prices with a maximum increase/decrease of 60% with respect to $price^a$.

The effect on the room demand of changing the unit price is modeled by the acceptance probability, which is defined similarly to [66]. When the proposed price is equal to the average price of reservations with the same arrival day, the acceptance probability is set to 0.5, to model the absence of any preference about accepting or rejecting the reservation. With prices fixed to the average values, the expected number of accepted reservations is equal to half of the total number of reservation requests. The expected percentage of accepted reservations increases when the price decreases and decreases otherwise. This phenomenon, called *price elasticity*, is modeled by the following function:

$$\Pr_{\text{accept}}(RO) = 1 - \Phi(\rho \cdot (RO_{\text{price}} - price^a)), \tag{2.18}$$

where $\Phi(\cdot)$ is the c.d.f. of the standard normal distribution, and ρ is a parameter that controls the slope of the function and allows us to consider different price elasticity scenarios. In the experiments, ρ is chosen so that $\Pr_{\text{accept}}(RO) \approx 1$ when there is a discount of at least 50% and $\Pr_{\text{accept}}(RO) \approx 0$ when the price increases of at least 50%.

The applicability of HotelSimu is empirically shown on 10 hotels in Trento, Italy. Representative hotels have been selected from the official open data of the Province of Trento[2], as reported in Table 2.2. The information on the average arrivals and the average number of nights per reservation is taken from the Statistics Institute of the Province of Trento (Ispat)[3]. No information is available about the average number of rooms per reservation, which is assumed to be equal to 1. Data on

[2]http://dati.trentino.it/dataset/esercizi-alberghieri.

[3]http://www.statistica.provincia.tn.it, section “Annuari del Turismo".

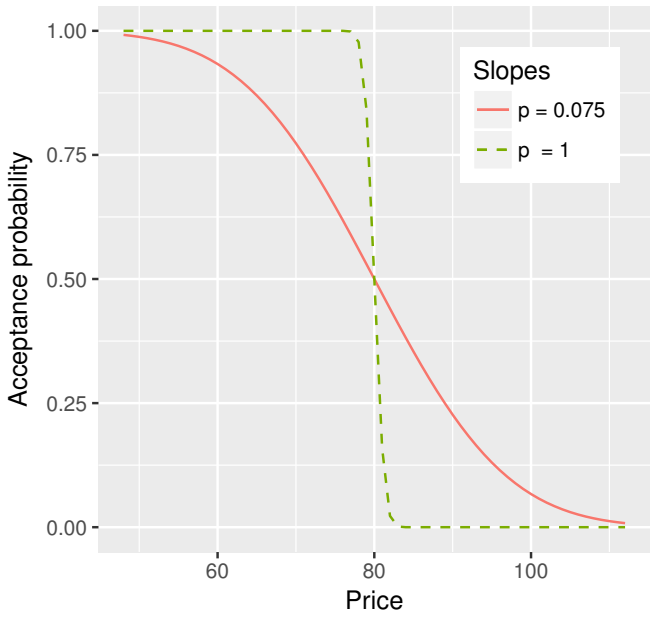


Figure 2.5: Price elasticity model of HotelSimu with different slope coefficients, defined around a reference price of 80

arrivals has been disaggregated and mapped onto each hotel according to the respective capacity, under the assumption that bigger hotels usually register more arrivals than smaller hotels. Real aggregated data on tourists and different hotels has been used to simulate time series of reservations and cancellations, and these time series are considered as a baseline to be compared to the outcome of the optimization.

In the experiments RH starts on July 1st, 2017, and ends on December 31st, 2018. AH starts on July 1st, 2017, and ends on January 31st, 2019. OH starts on January 1st, 2018, and ends on December 31st, 2018.

The optimization has a budget of 300 iterations (for a maximum running time of 5/6 hours), and each iteration estimates the total revenue as the average of 20 simulation runs, all with the same parameter configuration, for a total of 6000 simulations within one optimization run.

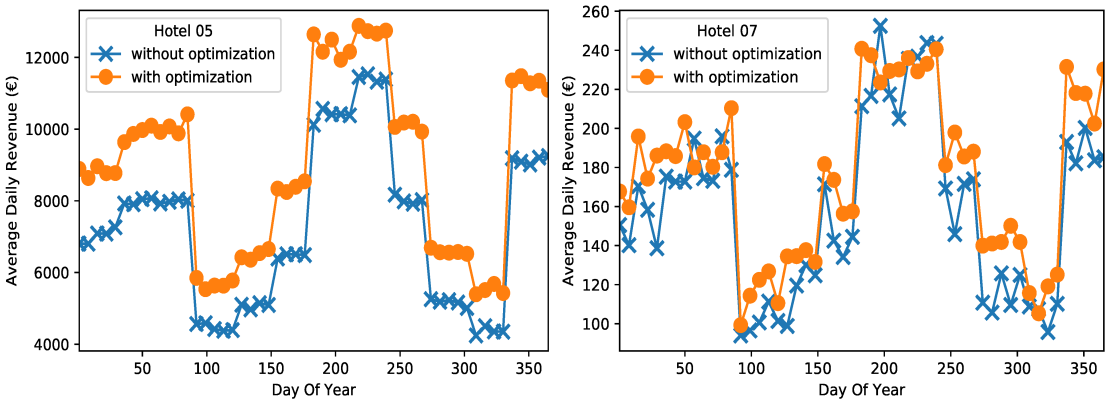
Hotel	Rooms	Price (€)	Arrivals	Occupancy	Revenue	Optimization	Simulation
01	52	120.00	48.2±0.5	47.6±0.6	18.4±0.6	11000±109.0	2.0 ±0.012
02	34	69.50	50.6±0.6	50.6±0.7	20.4±0.7	10800±82.33	1.91±0.006
03	136	290.00	51.6±0.3	52.6±0.3	21.6±0.3	18400±258.5	3.54±0.030
04	46	153.33	44.0±0.5	44.2±0.6	17.8±0.6	9910±97.61	1.78±0.008
05	113	136.675	**55.5±0.3**	**55.2±0.4**	**23.1±0.4**	16700±241.2	3.19±0.031
06	37	74.00	46.9±0.6	45.8±0.6	17.8±0.6	9570±87.87	1.69±0.012
07	9	39.00	**38.2±1.1**	**37.7±1.3**	**12.8±1.2**	7370±17.70	1.25±0.010
08	22	216.50	43.2±0.7	42.0±0.8	18.0±0.8	7870±35.67	1.35±0.004
09	14	66.50	42.0±0.9	41.8±1.0	17.7±1.0	7740±32.26	1.3 ±0.010
10	19	82.67	41.8±0.8	40.5±0.9	17.3±0.9	8650±46.21	1.49±0.008

Table 2.2: Characteristics of hotels used for the tests, and results. Arrivals, occupancy (as room-nights) and revenue after optimization are expressed as percentage increase, where maximum and minimum values are in bold. Optimization total CPU time and single-run simulation CPU time are defined in seconds.

The optimization is repeated 10 times. Each experiment is started from an initial solution which has been generated by a uniform distribution defined over the search space. Tests have been run on virtual machines using a KVM hypervisor (1 per hotel), each one with 512 MB of RAM and 1 CPU (1 core) at 2.1 GHz.

2.5.2 Results on arrivals, occupancy and revenue

Table 2.2 reports the results on customer arrivals, occupancy and total revenue as the percentage increase led by the optimized pricing model with respect to the configuration with the multipliers equal to 1. Results are expressed in terms of averages and standard errors, and they are statistically significant according to Welch's two-tailed unequal variances t-test [84], with a significance level $\alpha = 0.01$. A unit of occupancy corresponds to the so-called *room-night*, which is a room occupied for one night.

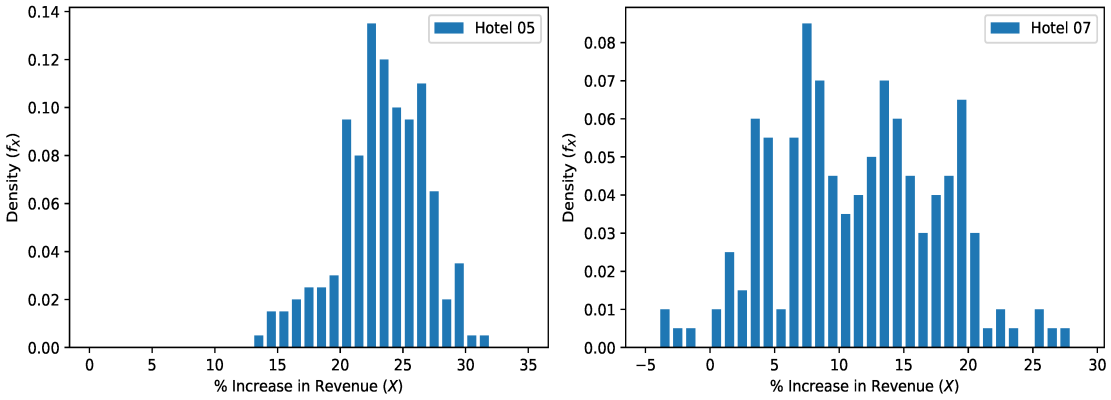


(a) Hotel 05 (b) Hotel 07

Figure 2.6: Average daily revenue over time (one value per week).

The outcome of experiments is promising for all the hotels, with a minimum of 12.8% increase in revenue, 37.7% in occupancy and 38.2% in arrivals. The maximum increase in revenue is reached for Hotel 05, with a value of 23.1%. The minimum values are reached for small hotels, where the limited number of rooms leads to fewer arrivals and then relatively low revenues. In this context, there is also more variability, since the hotel can become full with few reservations, thus leading to the rejection of more requests. Experiments suggest that higher revenues can be obtained for medium and big hotels, where the system exploits the capacity of the hotel to increase the number of arrivals. The time series of the average daily revenue during the year of interest for the best and worst scenarios are reported in Figure 2.6.

For Hotel 05, it is evident that the time series produced by the optimized model is significantly higher than that produced without optimization. In this case, there is less chance of having a loss in revenue because of an optimistic configuration found during the optimization process. For Hotel 07, the two time series are not significantly different because of the higher uncertainty caused by the small dimension



(a) Hotel 05 (b) Hotel 07

Figure 2.7: Average daily revenue distribution (one value per week).

of the hotel. This leads to higher risk and to the possibility of having a loss (with probability ≈ 0.03), as it is evident from the distribution of the increase in revenue in Figure 2.7. These results are in accordance with the expected behavior of non-homogeneous Poisson distributions, whose coefficient of variation decreases as the expected value increases. In the context of hotel demand, this property implies that for smaller hotels, which can accommodate a limited number of guests and therefore are characterized by less arrivals, the coefficient of variation is higher than that of large hotels. As a consequence, the increased variability for small hotels leads to higher risk of losses, as empirically shown by our results.

2.6 Conclusions

This Chapter introduced HotelSimu, a flexible simulation-based optimization approach which can be used for maximizing the revenue of hotels. Since the simulation output is noisy, a population-based algorithm that has already been studied in the literature has been employed

to optimize the simulator.
HotelSimu models stochastic arrivals and cancellations in an interleaved fashion, considering several characteristics of reservation requests in order to propose dynamic prices and it models the effect that price variations have on demand (price elasticity). Its parametric models, based on the RIM quantifiers, allow the hotel manager to adapt pricing policies to dynamic market conditions, and to analyze different booking scenarios by changing a compact set of meaningful parameters. Seasonal averages can be set even on a day-by-day basis, thus allowing the hotel manager to adapt the pricing policy to special events and to consider monthly as well as weekly seasonal effects.
The case study shows that the parametric models lead to results similar to other dynamic pricing models in the literature, while relying only on aggregated data. The average revenue increase is $\approx 19\%$ with respect to the original pricing policies, and the risk of losses is absent for medium-big hotels and limited for small hotels, with a maximum loss probability of ≈ 0.03. Moreover, experiments show that HotelSimu can simulate one year and a half in ≈ 2 seconds on average on a low-end machine. Also, a complete optimization can be run within one night.

Chapter 3

Heuristic Search Strategies for Noisy Optimization

3.1 Heuristics for noisy optimization

In many optimization problems, the evaluation of candidate solutions is affected by noise. Possible sources of noise include physical measurement limitations, or the stochastic component employed in simulations. Similarly, in machine learning, the diversity of data used to train and test models adds a layer of uncertainty to the problem. Different models are usually compared using cross-validation approaches, but comparisons are not guaranteed to be correct. The same happens in simulation-based optimization, where the output of a stochastic simulator is used in combination with black-box methods in order to optimize an objective function.

In order to deal with noise, various heuristics have been proposed and studied [85, 78, 24, 79, 25]. In particular, many studies employ variants of evolutionary algorithms, which adopt a set of candidate solutions (*population*) subject to local perturbative search and stronger

diversification means, often described with terms derived from genetics. Since these algorithms iteratively employ a population to explore the search space and propose new solutions, they are considered to be robust to the presence of noise [86, 25]. If a black-box method is population-based, the impact of noise can be decreased by incrementing the number of candidates (*size*) of the population [26, 24, 27]. This effect, called *implicit averaging*, has been studied using normally distributed noise with zero mean and different values of constant standard deviations [87, 85, 88]. However, as shown by [85], increasing indiscriminately the population size can be counterproductive. A similar approach is followed by shrinking-ball methods, where the value of the objective function is approximated by averaging single evaluations sampled within a hypersphere defined around the configuration to be evaluated. The neighborhood of a solution x is defined by a ball of a certain radius r, which can be reduced as the optimization progresses as in [89] or it can be kept constant as in [90].

Multiple works compared various heuristic algorithms with evolutionary strategies, and they have shown that population-based approaches are a good choice to optimize noisy functions [78, 79, 25]. However, these studies mostly compare evolutionary strategies only with algorithms which propose candidate solutions deterministically, without considering any extension based on statistical analysis techniques. In fact, the effect of noise can be reduced by evaluating multiple times each solution [28, 29, 30, 31, 20]. An example is given by simulated annealing [91], which has been extended by adapting the number of samples per solution based on some statistical analysis [19, 21]. However, studies which compare the performance of diverse randomized algorithms extended with statistical methods are not common in the literature. As will be shown empirically, even single-point randomized

methods extended with statistical techniques can perform better than regular population-based methods. The effects and the limits of implicit averaging depends on the type and the amount of noise in the objective function.
This Chapter compares the efficiency of different heuristic search strategies in the presence of noise, in order to investigate the effects that different components of these strategies have on the performance. Differently from previous studies, all the heuristic search strategies employed in this study are randomized, and algorithms not based on populations are extended using a reactive sample size scheme based on paired t-tests. The rest of the chapter is structured as follows. Section 3.2 gives an overview of the reactive sample size scheme. Section 3.3 outlines the heuristic search algorithms which have been used in the experiments, and comments their components. Section 3.4 defines the experiments and analyzes the results.

3.2 Reactive sample size

In this Chapter, in order to deal with the presence of noise, heuristic techniques which do not use a population are extended with a reactive sample size algorithm based on Student's paired t-test.
At each iteration of the optimization process, a new configuration x_{new} is proposed by a heuristic algorithm and must be compared against the current best configuration $x_{current}$. For each comparison, the algorithm reactively changes the sample size n used to evaluate $\hat{f}_n(x_{new})$ and $\hat{f}_n(x_{current})$, according to observed statistical evidence. Significant differences are detected by considering the relationship between probabilities α and β of making an error of type I and type II. To remind the reader, given a null hypothesis H_0 and an alternative hypothesis

H_1, α is the probability to reject H_0 when H_0 is true and β is the probability to fail to reject H_0 when H_0 is false.
In noisy optimization, it is not necessary to detect a difference between the estimators, which would require a two-sided test. In contrast, using a one-sided test to assess if $\hat{f}_n(x_{new}) > \hat{f}_n(x_{current})$ or $\hat{f}_n(x_{new}) < \hat{f}_n(x_{current})$ is sufficient, depending on whether one is maximizing or minimizing the objective function. Upper-tailed tests are used in the case of maximization and lower-tailed tests for minimization. The following definitions consider a maximization problem, where null and alternative hypothesis of the upper-tailed test on the difference in means are defined as

$$\begin{aligned} H_0 &: \hat{f}_n(x_{new}) - \hat{f}_n(x_{current}) \leq \delta_{H_0} \\ H_1 &: \hat{f}_n(x_{new}) - \hat{f}_n(x_{current}) > \delta_{H_0}, \end{aligned} \tag{3.1}$$

where $\delta_{H_0} = f(x_{new}) - f(x_{current}) = 0$ defines the value for which the null hypothesis is assumed to be true. To statistically determine if H_1 is true, paired or unpaired evaluations can be used to compute a statistic. The reactive algorithm uses paired evaluations to compute Student's paired t-test statistic by evaluating x_{new} and $x_{current}$ on the same realizations. And since the mean of paired differences corresponds to the difference of means, the statistic to test is

$$T_{n-1} = \frac{\delta_{obs} - \delta_{H_0}}{\hat{\sigma}_n / \sqrt{n}}, \tag{3.2}$$

where the observed difference $\delta_{obs} = \hat{f}_n(x_{new}) - \hat{f}_n(x_{current})$ and s is the sample standard deviation of paired evaluations. Furthermore, since σ of F is approximated with $\hat{\sigma}_n$, T_{n-1} follows a t-distribution with $n-1$ degrees of freedom (d.o.f.) and the algorithm requires $n \geq 2$.

The statistic in equation (3.2) can be used to compute the p-value, which defines how unlikely it is to observe a statistic such as T_{n-1} if H_0 is true. If this is too unlikely (threshold defined by the user-defined probability α of making an error of type I), then H_0 should be rejected. However, it is also important to consider the probability β to make an error of type II, so to fail to reject H_0 when H_0 is false. To compute β, a specific value for which H_1 is assumed to be true has to be defined, because β depends on the difference δ_{obs} for which H_0 is assumed to be false and H_1 to be true. In fact, assuming that H_1 is true, if T_{n-1} falls in the acceptance region of H_0 then it is unlikely that H_0 is false. In the reactive algorithm, the value for which H_1 is assumed to be true is δ_{obs}. When H_1 is true, T_{n-1} follows a noncentral t-distribution with $n-1$ d.o.f. Unfortunately the noncentral t-distribution has a complex density function, but [92] propose a method to approximate it using the t-distribution. Thus, in the one-tailed case, β can be approximated as

$$\beta = 1 - T_{n-1}(\delta_{norm}\sqrt{n} - t_{n-1,\alpha}) + T_{n-1}(-\delta_{norm}\sqrt{n} - t_{n-1,\alpha}) \quad (3.3)$$

where T_{n-1} is the c.d.f. of the t-distribution with $n-1$ d.o.f., $\delta_{norm} = \delta_{obs}/\hat{\sigma}_n$ and $t_{n-1,\alpha}$ is the quantile x of the t-distribution with $n-1$ d.o.f. such that $T_{n-1}(x) = \alpha$. As a consequence, given a paired sample, equation (3.3) can be used to iteratively find the minimum sample size n that should be used to test a one-tailed hypothesis with error probabilities α and β as

$$n = \frac{(t_{n-1,\alpha} + t_{n-1,\beta})^2}{\delta_{norm}^2}. \quad (3.4)$$

Algorithm 1 Statistical guard

1: **procedure** STATISTICALGUARD($x_{new}, x_{current}$)
2: Compute $\delta_{observed}, \hat{\sigma}_n, \delta_{norm}$
3: p-value $\leftarrow 1 - T_{n-1}(\frac{\delta_{observed}}{\hat{\sigma}_n/\sqrt{n}})$
4: $\beta \leftarrow 1 - T_{n-1}(\delta_{norm}\sqrt{n} - t_{n-1,\alpha}) + T_{n-1}(-\delta_{norm}\sqrt{n} - t_{n-1,\alpha})$
5: **if** $\beta \leq \beta_{req}$ **then**
6: **if** p-value $\leq \alpha_{req}$ **then**
7: $x_{current} \leftarrow x_{new}$
8: $n_{current} \leftarrow max(n, n_{current})$
9: return

However, in real world problems, one might not be interested to correctly detect very small differences between means. It might be too computationally expensive to statistically differentiate between two means that are very similar. Precisely, if x_{new} and $x_{current}$ have a very similar performance and δ_{obs} is smaller than a certain user-defined δ_{heu}, the comparison can be performed heuristically according to the values of $\hat{f}_n(x_{new})$ and $\hat{f}_n(x_{current})$. The value of δ_{heu} is expressed as a percentage of $\hat{f}_n(x_{current})$, because in many cases the user does not know *a priori* the performance of the best possible solution which can be found during the optimization. Furthermore, because of the stochasticity of the objective function, the algorithm also considers the impact of the sample standard deviation $\hat{\sigma}_n$. Before checking if a comparison should be done heuristically, δ_{heu} is normalized by $\hat{\sigma}_n$ and, to apply the heuristic solution, the algorithm checks if

$$\delta_{norm} < \frac{\delta_{heu}}{\hat{\sigma}_n}. \tag{3.5}$$

In the heuristic solution, $\hat{f}_n(x_{new})$ is evaluated at least as many times and on the same set of seeds as $\hat{f}_n(x_{current})$. In fact, the reactive scheme

keeps track of the sample size $n_{current}$ used to evaluate $\hat{f}_n(x_{new})$ when Equation 3.5 is verified, and any sample size which lowers the current value is not accepted. As a consequence, the sample size used in the heuristic case can only increase as the optimization advances. As suggested in [14], the sample size should increase as the optimization proceeeds towards a local minimum, because it is more difficult to detect the difference between solutions which tend to have similar values. When the current evaluation is far from any minimum present in Θ, a small sample size should be sufficient to distinguish the performance of different configurations. As the process goes towards locally optimal points, comparing diverse solutions becomes harder and so additional evaluations are necessary.
When increasing n using (3.4), the update is not done in one-shot but iteratively, updating the pair $\hat{f}_n(x_{new})$ and $\hat{f}_n(x_{current})$ with one evaluation at the time. This is done because (3.4) defines an estimation of the minimum sample size required to test the hypothesis, which is based on the observed sample. But the observed sample is not always representative of the whole population, and so estimations might not be correct. As n increases, the sample is going to be more and more representative; but by checking iteratively p-value and β a lot of unnecessary evaluations are saved. In fact, at each update, p-value and β are compared with α_{req} and β_{req}; if such requirements are satisfied, then a statistically significant decision can be taken. The same is done when Equation 3.5 is verified and the heuristic solution is applied. In fact, by computing iteratively p-value and β even in the heuristic solution, a statistical guard checks if sufficient statistical evidence has been observed in order to take a statistically significant decision (Algorithm 1).

3.2.1 Parameters of the algorithm

The main parameters of the algorithm are the required probability α_{req} of making an error of type I, the required probability β_{req} of making an error of type II and the minimum required difference δ_{heu} between averages to apply a statistical test. To provide additional flexibility to the algorithm, a minimum number n_{min} and a maximum number n_{max} of function evaluations can be set for each configuration in comparisons. If during a comparison n_{max} is reached, but α_{req} and β_{req} have not yet been satisfied, a decision is taken by considering only the values of $\hat{f}_n(x_{new})$ and $\hat{f}_n(x_{current})$. In the experiments $n_{min} = 2$ and $n_{max} = \infty$, so the algorithm autonomously decides when to stop the sampling process.

3.2.2 Outline of the algorithm

The reactive sample size algorithm is outlined in Algorithm 2. At each step of the optimization process, given two configurations x_{new} and $x_{current}$, the algorithm first evaluates both configurations on n_{min} times in order to obtain $\hat{f}_n(x_{new})$ and $\hat{f}_n(x_{current})$, where $n = n_{min}$. Then, p-value and β are computed using (3.2) and (3.3), respectively. If these values satisfy the probabilities of making an error of type I and II required by the user, then a decision which is considered to be statistically significant can be taken; otherwise, the sample size of both estimators is increased with one additional evaluation based on the same ξ_i. In both cases, if Equation 3.5 is verified, the decision is taken heuristically by considering only the values of $\hat{f}_n(x_{new})$ and $\hat{f}_n(x_{current})$. The optimization continues as long as there is a sufficient amount of budget left, where the budget is defined as number of ob-

Algorithm 2 Reactive sample size algorithm

1: **procedure** REACTIVECOMPARISON($x_{new}, x_{current}$)
2: Compute $\hat{f}_{nmin}(x_{new})$
3: statisticalGuard($x_{new}, x_{current}$)
4: $n \leftarrow (t_{n-1,\alpha} + t_{n-1,\beta})^2/\delta^2_{norm}$.
5: $i \leftarrow n_{min}$
6: **while** $i \leq n$ **do**
7: Update $\hat{f}_i(x_{new})$ and $\hat{f}_i(x_{current})$ using ξ_{i+1}
8: statisticalGuard($x_{new}, x_{current}$)
9: **if** $\delta_{norm} \leq \delta_{heu}/\hat{\sigma}_n$ **then** ▷ Heuristic decision
10: $k \leftarrow i$
11: **while** $k \leq n_{current}$ **do**
12: Update $\hat{f}_k(x_{new})$ using $F(x_{new}, \xi_{k+1})$
13: statisticalGuard($x_{new}, x_{current}$)
14: $k \leftarrow k + 1$
15: heuristicDecision($x_{new}, x_{current}, n_{current}$)
16: $i \leftarrow i + 1$
17: **if** $i = n_{max}$ **then**
18: heuristicDecision($x_{new}, x_{current}, n_{max}$)
19: **if** $i = n + 1$ **then**
20: $n \leftarrow (t_{n-1,\alpha} + t_{n-1,\beta})^2/\delta^2_{norm}$.

jective function evaluations.

Algorithm 2 is called by a heuristic optimization process when some x_{new} has to be compared against $x_{current}$, with $x_{new} \neq x_{current}$. In Algorithm 2, on line 7 a new evaluation $F(x_{current}, \xi_i)$ is computed only if $\hat{f}_i(x_{current})$ has not been evaluated on ξ_i yet. On line 20, n is computed again in the case that the initial sample produces an underestimate of the minimum sample size required to statistically differentiate between the performance of x_{new} and $x_{current}$.

3.3 Optimization algorithms

The optimization algorithms employed in the experiments are Random Search (RS), the Reactive Affine Shaker (RAS) [93] and the Covariance Matrix Adaptation Evolutionary Strategy (CMA-ES) [52, 82]. RS is a simple stochastic local search algorithm which is often used as baseline for comparisons, while RAS and CMA-ES are more advanced stochastic schemes which adapt step size and direction of the search during the optimization.

In RS, a new candidate solution x_{new} is sampled from an interval defined in a neighborhood of the current best solution $x_{current}$, according to a uniform distribution. A step size σ is used to define, as a percentage of the intervals which define Θ along each dimension, the boundaries of the local search region located around $x_{current}$. Consequently, diverse step sizes correspond to search policies with different levels of locality. A step size of 1 would make the search global, and the optimization would correspond to pure random search.

In RAS, a local search region is adapted by an affine transformation. The aim is to scout for local minima in the attraction basin where the initial point falls. The step size σ and the direction of the search region are adapted in order to maintain heuristically the largest possible movement per function evaluation. The search occurs by generating points in a stochastic manner, with a uniform probability in the search region, following a *double shot* strategy. A single displacement Δ is generated, and two specular points $x_{current} + \Delta$ and $x_{current} - \Delta$ are considered for evaluation. An evaluation is successful if the objective function value in at least one of the two candidates is better than $\hat{f}(x_{current})$. The search region is modified according to the outcome of the comparisons. It is compressed if both comparisons are unsuccess-

ful, and it is expanded otherwise. In both cases, the search region is modified according to an expansion factor $\rho > 1$.

CMA-ES [52, 82] is an evolutionary optimization paradigm in which configurations are sampled from a multivariate normal distribution $N(\mu_t, M_t)$, where t defines the iteration of the algorithm. At each iteration, the mean μ_t defines the center of the distribution, the covariance matrix M_t determines shape and orientation of the ellipsoid corresponding to $N(\mu_t, M_t)$, and a step size σ_t controls the spread of the distribution as a percentage of the intervals which define each dimension of Θ. The ellipsoid is the local search region used by the algorithm to explore the search space and propose candidate solutions. Iteratively, CMA-ES follows four steps. First, it samples a fixed number λ of new configurations from $N(\mu_t, M_t)$, creating a population. Second, candidates are evaluated and ranked according to the quality of the evaluations. Third, the best $\lfloor \frac{\lambda}{2} \rfloor$ results are used to update $N(\mu_t, M_t)$, in order to move the search towards the most promising search direction. Fourth, σ_t is increased or decreased according to the length of the so-called evolution paths, in order to maximize the expected improvement of the optimization. This last step is explained more in detail in the following subsection. Also, CMA-ES has been extended with an uncertainty handling (UH) method, to deal with possible noise in the objective function [53]. In this version of CMA-ES, referred as UH-CMA-ES, the uncertainty is measured by rank changes among members of the population. Once each solution of the population has been evaluated and ranked, a few additional evaluations are taken and the population is ranked again. By doing so the algorithm tries to estimate the amount of noise in the evaluations, in order to increase σ_t and prevent the signal-to-noise ratio from becoming too low.

3.3.1 On CMA-ES step-size adaptation

The adaptation of σ_t, also called cumulative step-size adaptation, is based on the evolution paths mentioned in Section 3.3. An evolution path is a weighted vector sum of the last points successively visited by the algorithm. It provides information about the correlations between points, and it can be used to detect the direction of consecutive steps taken by the optimization. If consecutive steps are going in the same direction (scalar product greater than zero), the same distance could be covered by longer steps. If consecutive steps are not going in the same direction (scalar product lower than zero), single steps tend to cancel each other out and the current path is too short. To make successive steps more efficient, σ_t is changed accordingly. The step size determines the signal strength used by CMA-ES to estimate the direction of the gradient. If the steps of the algorithm are very small, the signal is also likely low and therefore the signal-to-noise ratio becomes small as well. Also, it has been shown that in noisy optimization the cumulative step-size adaptation may result in premature convergence [94].

3.4 Experiments

Before showing the results about the performance of different randomized algorithms in various noisy scenarios and higher dimensions, a preliminary study which investigates the effect of implicit averaging on CMA-ES is presented.

3.4.1 Setup

Each experiment is based on 100 macroreplications, where each optimization process has a budget (number of function evaluations) of 5000. Initial solutions are generated according to a uniform distribution defined on the interval of each dimension of Θ. In each experiment, algorithms start the optimization from a different randomly generated solution x_0 and consider the same local search region around it. Then, the local search region is iteratively modified according to the algorithm. Apart from CMA-ES and UH-CMA-ES, the algorithms are employed in two versions: with a naive scheme which uses 1 evaluation for each solution, and the reactive sample size scheme. The acronym of each algorithm is preceded with N in the former case and with R in the latter. The parameters of the reactive method are set as $\alpha_{req} = 0.1$, $\beta_{req} = 0.4$ and $\delta_{heu} = 0.01$. At the end of the optimization, the solution x^* with the best measured performance is returned.
In the first set of experiments on CMA-ES, restarts are not considered. The standard implementation of CMA-ES includes various stopping criterias and restart policies [82], but they have been deactivated in order to simplify the analysis of the different components of the algorithm. In the second set of experiments, global restarts are activated and based on a single stopping criterion: if the current best solution does not improve by at least 10% in $k = 500$ function evaluations, a restart is done. An exception is given by RAS, which possibly needs to be restarted because of its double-shot strategy. In fact, if x_0 is generated nearby the boundaries of the search space, the double shot strategy might be unable to generate a valid configuration.
RS uses $\sigma \in \{0.1, 0.2\}$, while RAS employs $\sigma \in \{0.1, 0.2\}$ and $\rho = 2$. CMA-ES and UH-CMA-ES adopt $\sigma \in \{1.0, 2.0\}$, because the library

used to implement the algorithm defines $\sigma \in (0, 10]$[1] and not in $(0, 1]$. Also, $\lambda = 4 + \lfloor 3 \log d \rfloor$, where d is the dimensionality of the objective function. The values of ρ and λ are the ones suggested respectively by the authors of [93, 82].

3.4.2 Benchmarking functions and noise models

In order to test diverse heuristic strategies in the presence of noise, Sphere and Rastrigin functions have been extended with multiple types and levels of noise. As in other works in the literature, both functions are optimized in $[-5.12, 5.12]^d$, where d is the number of dimensions. To evaluate the impact of noise on the optimization, a standard practice in the literature is to extend deterministic functions by introducing multiplicative or additive noise. In the case of multiplicative noise, a percentage ϵ of $f(x)$ is added to $f(x)$ according to a displacement generated using a standard normal distribution:

$$f(x, \epsilon) = f(x) + f(x) \cdot \epsilon \cdot N(0, 1). \tag{3.6}$$

This kind of noise is typical of devices which take physical measurements, like speed cameras, where values are guaranteed to be accurate up to a certain percentage of the measured quantity. However, as the optimization proceeds towards lower values, the noise decreases. This means that as the optimization approaches the global optimum, it is easier to move into the right direction and, if the function value at the global optimum is equal to zero, there is almost no noise in its proximity.

Although such a situation might be true in many real-world scenarios,

[1] https://github.com/beniz/libcmaes

there exist other problems where the noise does not always go to zero as the global optimum (if any) is approached. As examples, consider the optimization of simulation models, or the tuning of the hyperparameters of machine learning algorithms. In this case, the noise can be simulated by adopting additive noise. However, determining the amount of noise to add is up to the practitioner. In fact, additive noise is usually normally distributed with zero mean and constant standard deviation σ_ϵ. Since this kind of perturbations does not depend on the signal, the signal-to-noise ratio might cause problems only when approaching the minimum and its effects are going to be very different from function to function.
To avoid these drawbacks, a possibility is to define additive noise as normally distributed with zero mean and dynamic standard deviation. Since the step size used by randomized algorithms impacts the strength of the signal, in order to test harder noise scenarios it makes sense to set the noise level according to the step size. So, given a point x in Θ and a percentage ϵ, the dynamic standard deviation σ_i along each dimension i is computed as follows. Compute lower bound $l_i = x_i - \epsilon \cdot \Theta_i$ and upper bound $u_i = x_i + \epsilon \cdot \Theta_i$, where Θ_i is the interval in which the function is defined along dimension i. Then, find the minimum m and the maximum M among $\{f(l_i), f(x_i), f(u_i)\}$. Finally, $\sigma_i = |m - M|$ and the additive noise model is defined as

$$f(x, \epsilon) = f(x) + \sum_{i=1}^{N} N(0, \frac{\sigma_i}{k}), \tag{3.7}$$

where N is the dimensionality of the function and k is a constant used to control the amount of noise. In the experiments, $\epsilon = 0.1$ and $k \in \{1, 2, 3, 6\}$. Therefore, while using this model of noise, the

distortion of the signal is set according to the maximum signal which can be detected by an algorithm which adopts a fixed step size (like RS). With $k = 6$, 99.7% of the noise is generated within the intervals which define the local search region. With $k = 3, k = 2, k = 1$ the same is true respectively for $81.86\%, 68, 27\%, 34.14\%$ of the noise.

3.4.3 Average loss signal per iteration

In a noisy scenario, in order to understand how the algorithm behaves with populations of different sizes, a possible way to proceed is to measure the magnitude of the error made when estimating the gradient. To do so, at each iteration, the population of candidate solutions $p = \{x_1, ..., x_m\}$ is ranked in two ways. Firstly, according to the noisy ranking $\hat{r}$ based on $\hat{f}_1(x_1), ..., \hat{f}_1(x_m)$. Secondly, following the noiseless ranking r defined by $f(x_1), ..., f(x_m)$. Then, the signal loss L is defined as the difference between the signal of these two rankings, where the signal of a ranking is the sum of the absolute differences among the ordered values used for ranking. More formally, in the case of $\hat{r}$ and r,

$$\hat{s}(\hat{r}, p) = \sum_{i=1}^{m-1} |\hat{f}_1(x_i) - \hat{f}_1(x_{i+1})| \tag{3.8}$$

and

$$s(r, p) = \sum_{i=1}^{m-1} |f(x_i) - f(x_{i+1})|, \tag{3.9}$$

where the set $\{1, ..., m-1\}$ is ordered respectively according to $\hat{r}$ and r. Therefore,

$$L(\hat{r}, r, p) = \hat{s}(\hat{r}, p) - s(r, p) \tag{3.10}$$

and the average signal loss per iteration is defined as

$$\mathbb{E}(L) = \frac{1}{N} \sum_{i=1}^{N} L(\hat{r}_i, r_i, p_i), \tag{3.11}$$

where N is the number of iterations of the algorithm. By following this procedure, the optimization estimates the gradient according to $\hat{r}$ and it is possible to measure by how much the optimization goes in the wrong direction. Also, by comparing $\hat{r}$ and r, the number of misrankings among each population's candidates can be computed, as well as the average misrankings' percentage $\mathbb{E}(M)$ of the whole optimization.

3.4.4 Results with larger populations

This set of experiments is based on the bidimensional Sphere function. Tables 3.1 - 3.3 contain the results of the experiments that investigate the effect of implicit averaging on CMA-ES, where the best performances are highlighted in bold. Tests adopt $\sigma = 1.0$ and $\lambda \in \{6, 36, 60\}$.

As it is possible to see in the tables, in all cases $\mathbb{E}(L)$ keeps increasing as the population grows. Larger populations can potentially detect more signal, but are unable to do so. In fact, as Figures 3.1a - 3.1c show, larger populations contribute to mantain a larger step size and so to make wider steps. However, as the amount of noise increases, misrankings are going to happen first among similarly ranked candidates and then also between solutions ranked farther away from each other. Also, the magnitude of the errors increases with the noise. Consequently, larger populations tend to lose more signal and to guide the optimization farther away from the true direction of the gradient. The

Table 3.1: Mean and standard error representing the performance of CMA-ES with $\sigma = 1.0$ on the Sphere function with $d = 2$ and multiple levels of constant additive noise.

		Constant additive			
λ	N	σ_ϵ	$\mathbb{E}(M)$	$\mathbb{E}(L)$	$f(x^*)$
6	833	0.1	81.92 ± 0.07	0.01 ± 0.00	0.02 ± 0.00
6	833	1.0	82.55 ± 0.06	0.05 ± 0.00	0.22 ± 0.00
6	833	2.0	82.74 ± 0.06	0.10 ± 0.00	0.42 ± 0.01
6	833	3.0	82.85 ± 0.07	0.14 ± 0.01	0.75 ± 0.01
36	138	0.1	93.26 ± 0.31	0.01 ± 0.00	0.01 ± 0.00
36	138	1.0	95.72 ± 0.08	0.14 ± 0.01	0.05 ± 0.00
36	138	2.0	95.98 ± 0.08	0.30 ± 0.03	0.10 ± 0.00
36	138	3.0	96.20 ± 0.05	0.40 ± 0.03	0.14 ± 0.00
60	83	0.1	93.75 ± 0.45	0.02 ± 0.00	**0.00±0.00**
60	83	1.0	97.05 ± 0.12	0.19 ± 0.01	**0.04±0.00**
60	83	2.0	97.50 ± 0.04	0.35 ± 0.03	**0.08±0.00**
60	83	3.0	97.59 ± 0.05	0.52 ± 0.04	**0.12±0.00**

Table 3.2: Mean and standard error representing the performance of CMA-ES with $\sigma = 1.0$ on the Sphere function with $d = 2$ and multiple levels of multiplicative noise.

		Multiplicative			
λ	N	σ_ϵ	$\mathbb{E}(M)$	$\mathbb{E}(L)$	$f(x^*)$
6	833	0.1	13.37±0.34	0.00±0.00	**0.00±0.00**
6	833	0.2	24.34±0.17	0.01±0.00	**0.00±0.00**
6	833	0.3	33.82±0.61	0.03±0.01	**0.28±0.02**
6	833	0.4	43.78±0.76	0.06±0.02	**2.28±0.05**
36	138	0.1	53.48±0.19	0.04±0.01	**0.00±0.00**
36	138	0.2	70.86±0.13	0.07±0.01	**0.00±0.00**
36	138	0.3	79.03±0.11	0.14±0.02	0.38±0.02
36	138	0.4	84.30±0.08	0.28±0.04	4.63±0.06
60	83	0.1	66.15±0.11	0.05±0.01	**0.00±0.00**
60	83	0.2	80.16±0.07	0.13±0.02	**0.00±0.00**
60	83	0.3	86.12±0.06	0.22±0.03	0.58±0.02
60	83	0.4	90.09±0.04	0.48±0.05	6.56±0.06

Table 3.3: Mean and standard error representing the performance of CMA-ES with $\sigma = 1.0$ on the Sphere function with $d = 2$ and multiple levels of dynamic additive noise.

		Dynamic additive			
λ	N	σ_ϵ	$\mathbb{E}(M)$	$\mathbb{E}(L)$	$f(x^*)$
6	833	6	82.28±0.07	0.04±0.00	**0.15±0.00**
6	833	3	82.39±0.08	0.12±0.01	**1.45±0.02**
6	833	2	82.61±0.08	0.23±0.01	**3.26±0.04**
6	833	1	82.50±0.12	0.97±0.13	9.09±0.06
36	138	6	95.72±0.06	0.16±0.01	0.35±0.01
36	138	3	95.85±0.04	0.53±0.03	2.26±0.02
36	138	2	95.87±0.05	1.17±0.06	4.22±0.03
36	138	1	96.22±0.04	3.32±0.25	**9.07±0.06**
60	83	6	97.10±0.10	0.23±0.02	0.40±0.01
60	83	3	97.25±0.04	0.78±0.05	2.48±0.02
60	83	2	97.33±0.03	1.73±0.09	5.24±0.04
60	83	1	97.64±0.03	4.47±0.22	10.04±0.05

effect of the misrankings depends on the amount of noise. Even if $\mathbb{E}(M)$ increases with the population size, it is not implied that the performance of the optimization deteriorates. For example, Table 3.1 shows that in the case of constant additive noise, larger populations obtain better results. This happens because the signal-to-noise ratio is sufficiently high to avoid misrankings which would guide the optimization towards a significantly wrong direction. The amount of noise starts creating problems only when approaching the minimum, as shown in Figure 3.2. With this amount of noise, even N-RS is able to perform comparably well. In contrast, in the case of dynamic additive noise, the signal-to-noise ratio is approximately the same throughout the search space and results are expected to deteriorate much more, as confirmed by the results in Table 3.3. Even in the case of multiplicative noise, larger populations possibly worsen the performance of the optimiza-

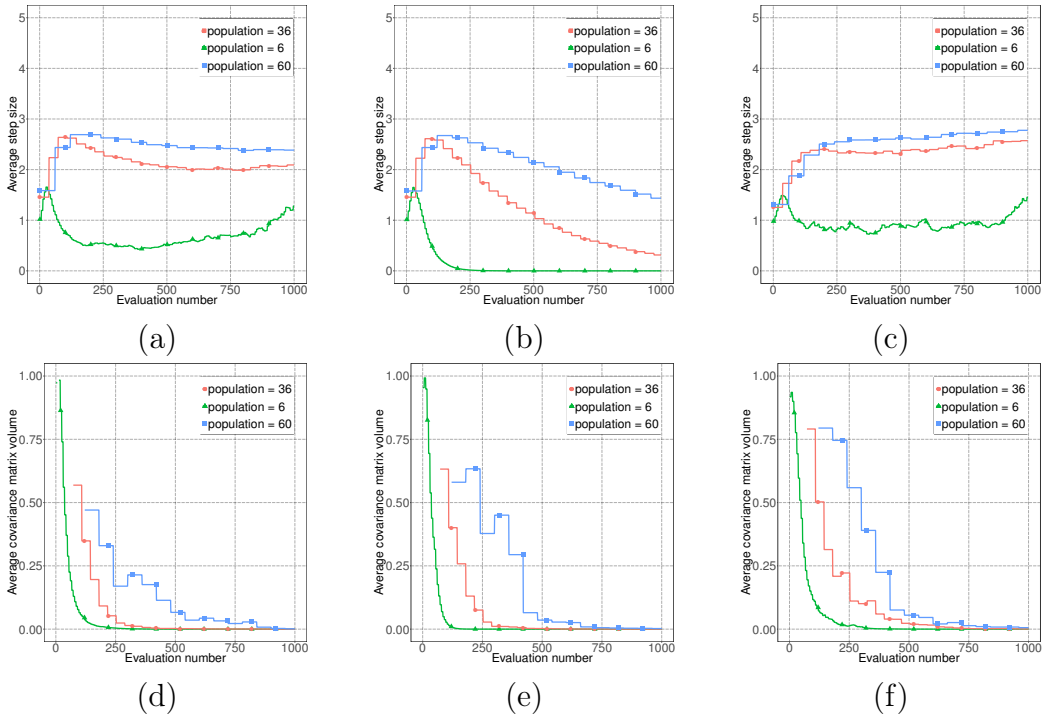
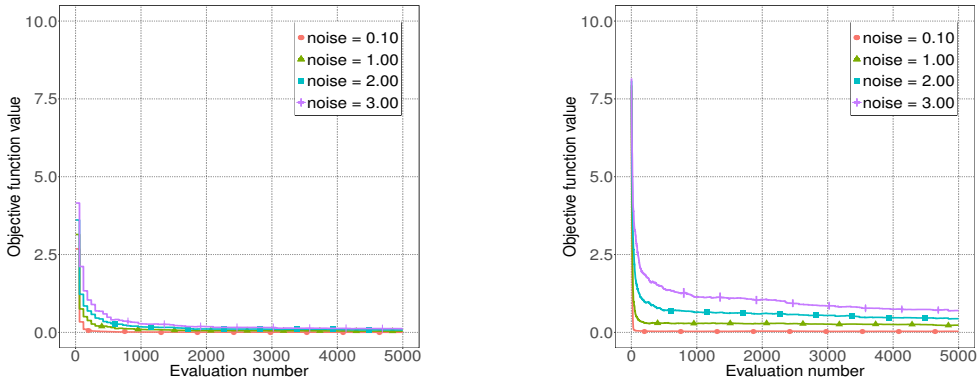


Figure 3.1: Each row shows respectively average variation of CMA-ES step size (3.1a) and CMA-ES covariance matrix volume (3.1d), with $\sigma = 1.0$ and $\lambda \in \{6, 36, 60\}$. Each column refers respectively to a particular case of the diverse types of noise used in Table 3.1. More precisely, they show the cases with constant additive noise with $\sigma = 1.0$ (first column), multiplicative noise with $\epsilon = 0.2$ (second column) and dynamic additive noise with $k = 3$ (third column).

tion. Therefore, when the signal-to-noise ratio is low and misrankings happen among further positions, increasing the population size or the number of parents is not going to significantly improve the robustness of the optimization. In this case, it is preferable to increase the sample size used to estimate each candidate solution.

It is also worth noticing the premature convergence of the covariance matrix. Figures 3.1d - 3.1f show that the volume of the covariance matrix goes to zero in the first part of the optimization. After that,



(a) CMA-ES (b) N-RS

Figure 3.2: Noiseless mean and standard error of the solutions with best measured performance found during the optimization, using CMA-ES ($\sigma = 1.0$, $\lambda = 60$) and N-RS ($\sigma = 0.1$) on the Sphere function with $d = 2$ and multiple levels of constant additive noise.

CMA-ES is no longer able to propose significantly different solutions.

3.4.5 Results with larger sample size

These experiments are based on both Sphere and Rastrigin functions, with $d \in \{2, 10\}$. Results in Tables 3.4 - 3.6 show that, with high levels of noise, a simple optimization algorithm such as R-RS performs better than more complex algorithms like RAS, CMA-ES or UH-CMA-ES. Without increasing the sample size of estimators, using a population of solutions is not able to compete with single-point algorithms which adapt the sample size of estimators according to empirical evidence. Furthermore, in this context, UH-CMA-ES might even worsen the performance. As shown in Figure 3.3d, increasing the step size according to observed misrankings provides better results when the initial step size is very low with respect to the distortion caused by the noise. Figure 3.3a shows that a larger step size can improve the efficiency of

Table 3.4: Noiseless mean and standard error of the solutions with best measured performance found during the optimization by different optimizers, with multiple levels of dynamic additive noise applied to the Sphere function ($d = 2$).

Optimizer	σ	$k = 1$	$k = 2$	$k = 3$	$k = 6$
R-RS	0.1	**1.14±0.01**	**0.29±0.00**	**0.15±0.00**	**0.05±0.00**
R-RS	0.2	1.25±0.02	0.45±0.01	0.26±0.00	**0.05±0.00**
R-RAS	0.1	4.12±0.06	1.37±0.03	0.78±0.03	0.12±0.01
R-RAS	0.2	3.41±0.05	1.35±0.04	0.55±0.03	**0.05±0.00**
CMA-ES	1.0	13.95±0.05	5.72±0.04	2.45±0.03	0.44±0.01
CMA-ES	2.0	14.76±0.06	5.00±0.04	2.14±0.02	0.41±0.00
UH-CMA-ES	1.0	14.71±0.05	5.09±0.03	2.15±0.01	0.45±0.01
UH-CMA-ES	2.0	15.46±0.05	5.66±0.04	2.03±0.02	0.38±0.00

Table 3.5: Noiseless mean and standard error of the solutions with best measured performance found during the optimization by different optimizers, with multiple levels of dynamic additive noise applied to the Sphere function ($d = 10$).

Optimizer	σ	$k = 1$	$k = 2$	$k = 3$	$k = 6$
R-RS	0.1	4.29±0.02	1.75±0.01	**0.81±0.00**	**0.29±0.00**
R-RS	0.2	**3.77±0.02**	**1.74±0.01**	1.16±0.01	0.51±0.00
R-RAS	0.1	6.85±0.03	4.27±0.02	2.59±0.02	0.83±0.01
R-RAS	0.2	7.63±0.03	4.63±0.03	3.34±0.03	0.70±0.01
CMA-ES	1.0	11.98±0.03	7.22±0.03	3.13±0.02	0.69±0.01
CMA-ES	2.0	12.17±0.02	7.47±0.03	3.17±0.02	0.71±0.00
UH-CMA-ES	1.0	12.96±0.02	9.86±0.03	5.26±0.03	1.02±0.01
UH-CMA-ES	2.0	12.79±0.02	10.33±0.02	6.09±0.03	0.98±0.01

Table 3.6: Noiseless mean and standard error of the solutions with best measured performance found during the optimization by different optimizers, with multiple levels of dynamic additive noise applied to the Rastrigin function ($d = 2$).

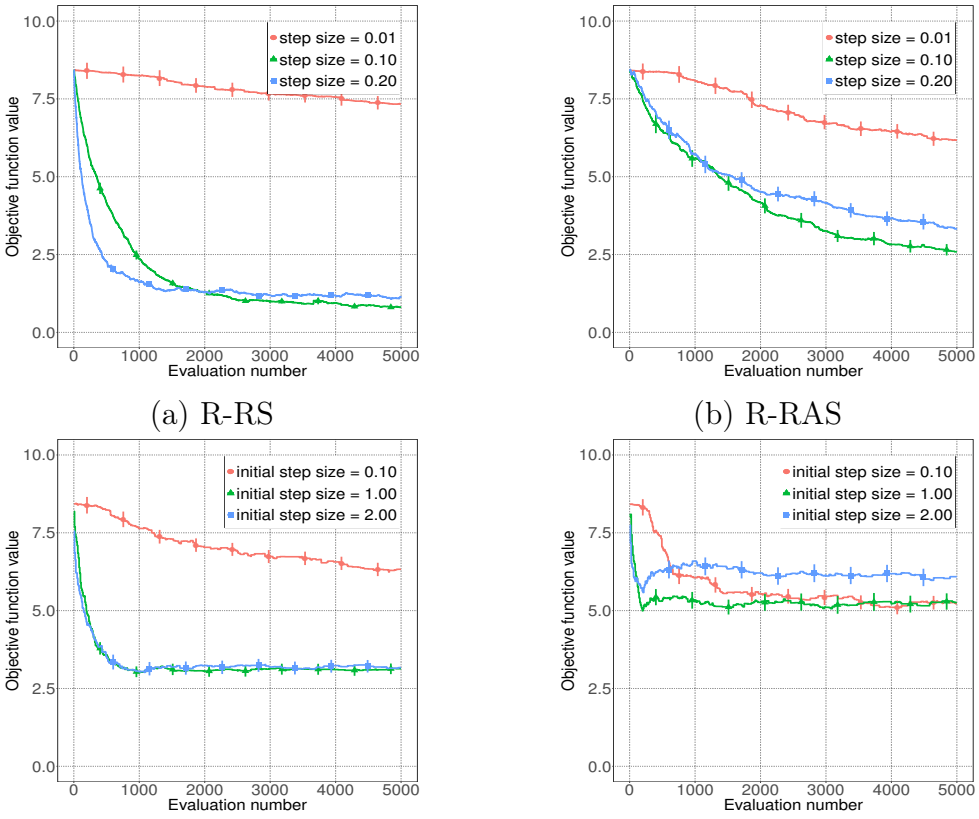
Optimizer	σ	$k = 1$	$k = 2$	$k = 3$	$k = 6$
R-RS	0.1	**1.87±0.02**	**0.58±0.01**	**0.30±0.00**	**0.13±0.00**
R-RS	0.2	2.19±0.02	0.77±0.01	0.48±0.01	0.25±0.00
R-RAS	0.1	7.68±0.07	5.65±0.05	5.10±0.05	3.68±0.03
R-RAS	0.2	6.61±0.06	4.77±0.04	4.61±0.04	2.86±0.03
CMA-ES	1.0	16.85±0.07	6.62±0.06	3.58±0.03	1.03±0.01
CMA-ES	2.0	17.48±0.07	6.16±0.05	2.85±0.02	0.98±0.00
UH-CMA-ES	1.0	16.79±0.08	4.73±0.03	2.98±0.02	1.11±0.00
UH-CMA-ES	2.0	17.67±0.07	5.21±0.05	2.11±0.01	0.95±0.01

Table 3.7: Noiseless mean and standard error of the solutions with best measured performance found during the optimization by different optimizers, with multiple levels of dynamic additive noise applied to the Rastrigin function ($d = 10$).

Optimizer	σ	$k = 1$	$k = 2$	$k = 3$	$k = 6$
R-RS	0.1	10.94±0.03	8.07±0.02	7.29±0.01	6.60±0.01
R-RS	0.2	**10.30±0.03**	**7.24±0.02**	6.40±0.01	5.93±0.01
R-RAS	0.1	13.20±0.03	9.55±0.03	8.47±0.02	6.97±0.02
R-RAS	0.2	13.98±0.04	10.16±0.03	8.35±0.03	6.88±0.02
CMA-ES	1.0	18.83±0.03	12.37±0.05	**6.21±0.04**	2.24±0.01
CMA-ES	2.0	19.59±0.03	13.31±0.04	6.46±0.04	**2.13±0.01**
UH-CMA-ES	1.0	19.89±0.03	15.94±0.03	11.02±0.03	3.95±0.02
UH-CMA-ES	2.0	19.71±0.03	16.57±0.04	12.64±0.03	3.91±0.02

the optimization. On average, compared solutions correspond to more different estimators and a lower sample size is required to make statistically significant comparisons. However, since a larger step size also implies reducing the sampling granularity, the optimization enhances the global search phase and the convergence speed tends to decrease. For the same reason, as shown in Figure 3.3b, the effectiveness of the double-shot strategy in a noisy environment is questionable. Although such an approach can be a good strategy for deterministic optimization, on each iteration very similar configurations are compared, the signal-to-noise ratio tends to be low and the sample size required to make statistically significant comparisons increases. Furthermore, as the search region is compressed, this effect is further enhanced.

In noisy scenarios, step-size adaptation mechanisms and adaptations of the search space are potentially counterproductive. In deterministic functions, compressions of the search region usually lead to a better exploitation of the local structure of the objective function. However, because of the presence of noise, decisions to compress the search region might be wrong and therefore the optimization might prematurely



(a) R-RS (b) R-RAS

(c) CMA-ES (d) UH-CMA-ES

Figure 3.3: Noiseless mean and standard error of the solutions with best measured performance found during the optimization, using different step sizes and dynamic additive noise (Sphere function values, with $d = 10$ and $k = 3$).

converge to false local optima.

In larger dimensions, the situation changes in the case of Rastrigin function with lower levels of noise. However, it is expected that a population-based algorithm performs better than single-point algorithms in the case of a multimodal function like Rastrigin. Combining a set of candidate solutions at each iteration gives the algorithm the ability to adapt to the local topology of the objective function, reducing the risk to get stuck in local optima.

3.5 Conclusions

This Chapter investigated different components of diverse heuristic strategies in the context of noisy optimization. A preliminary study on the bidimensional Sphere function showed how the *implicit averaging* effect of population-based algorithms does not always improve the optimization as the size of the population is increased, and analyzed how different amounts of noise change the impact of this effect on the optimization. Randomized algorithms not based on populations have been extended using a statistical analysis technique to deal with the presence of noise, and they have been compared with CMA-ES and UH-CMA-ES.

Results in Section 3.4.4 confirm the findings of [85]. The analysis provides an explanation about the reason for which larger populations do not always improve the optimization, and a higher sample size should be preferred when the signal-to-noise ratio is too low. Furthermore, these results also agree with [25]: in the presence of noise, step length control mechanisms are crucial to the performance of the optimization. If optimization methods are extended with statistical analysis techniques, the resolution at which the search space is explored matters significantly. With lower step sizes, solutions correspond to more similar function evaluations, and the sample size required to statistically determine a difference increases.

Chapter 4

Comparisons in Simulation-based Optimization

4.1 Diverse comparisons during the search

In many optimization problems, it might not be possible to find the optimal solution in acceptable (polynomial) CPU times [95]. Furthermore, according to the application of interest, determining a sufficiently good solution might be more relevant than finding a solution which is guaranteed to be optimal. Heuristic search algorithms provide several methods for this purpose [96, 1]. For example, local search strategies can find good solutions trough the iterative improvement of an initial solution, by gathering information on the objective function from within the neighborhood of the current solution. The target is not to generate a sequence of solutions that provably converges to the global optimum, but to improve the quality of visited solutions as quickly as possible [25].

Optimization might be further complicated by the presence of noise, due to several reasons: measurement errors, probabilistic modeling,

and others. This is also the case in simulation-based optimization, where the output of a stochastic simulator is used to estimate the objective function. Therefore, to obtain effective simulation-based optimization strategies, finding a tradeoff between precision of estimation and CPU time is essential.

Statistical analysis techniques can be integrated into heuristic search algorithms [19, 20, 21, 22, 23]. This Chapter investigates possible heuristic search integrations of the optimal computing budget allocation (OCBA) and the indifference-zone (IZ) formulations, two popular R&S methods which have been widely explored in the traditional R&S scenario. Differently from previous approaches [20, 22, 23], which evaluate solutions only by assuming independent random realizations, positive correlation is exploited in order to take statistically significant decisions as soon as possible. Correlation is especially important in noisy scenarios like simulation-based optimization, because it reduces the computational burden required to compare estimators [55].

In this Chapter, building on the approximation of [97], OCBA is extended with a stopping rule which considers positive correlation among the samples. Results show that, by evaluating simultaneously multiple stopping criteria based on different assumptions about the samples, the efficiency of the optimization significantly improves. Also, to further enhance the exploration-vs-exploitation tradeoff of the search, the IZ formulation is considered. Tests are run for the particular case that $k = 2$, as it happens in local search methods like simulated annealing or variants of random local search, but results are likely to be extended to the general case where $k > 2$. Experiments consider the OCBA approximation by [97] and a more conservative hypothesis testing procedure based on the probabilities of making errors of type I and type II. Results show that diverse assumptions on the samples lead to different

probability guarantees, with a significant impact on the efficiency of the optimization.
The rest of the Chapter is structured as follows. Section 4.2 describes related work about OCBA, IZ selection and different assumptions of statistical testing. Section 4.3 defines possible ways to improve the efficiency of heuristic search in simulation-based optimization. Section 4.4 outlines the design of the experiments and analyzes the results.

4.2 Statistical analysis methods

Given a finite set of k solutions, whose performance can be evaluated only through sampling, the purpose of R&S methods is to identify the best performing solution. This objective is usually bound by two possible constraints: guaranteeing a given fixed precision of correct selection (PCS), or achieving the best possible PCS within a given fixed sampling budget [54]. Since the primary goal of fixed-precision procedures is to guarantee a specified PCS, they tend to be more conservative with respect to fixed-budget procedures.

4.2.1 Optimal computing budget allocation

In the heuristic search setting the objective is to improve the quality of visited solutions as soon as possible. OCBA is among the best performing fixed-budget procedures [16], so it is a good choice for integrating a statistical selection procedure within heuristic search techniques.
Many methods in the R&S literature are statistically conservative, and essentially allocate simulation replications to competing configurations proportionally to the estimated variance without considering the value of estimated means. OCBA is a less conservative sequential R&S pro-

cedure based on Bayesian statistics, and it considers both estimated variances and means. Given an estimate of the mean and the variance of each competing configuration, OCBA defines the ratios between the sample sizes of the k configurations which lead to the maximization of an approximation of the PCS. At each step of the allocation procedure, a part Δ of the remaining budget is assigned to the configurations according to these ratios. In order to avoid a poor allocation of the budget at each iteration, especially in the first stages of the allocation, Δ should be as small as possible and can be set as 1 [10]. The allocation continues sequentially until a stopping criteria which provides an approximated probability of correct selection (APCS) is satisfied. However, the original allocation rule and stopping criteria of OCBA do not consider correlation among the samples [10].
Some authors derived optimal allocation rules to be applied in the presence of positive correlation between paired evaluations [98, 99]. As the correlation vanishes, the optimal allocation policy corresponds to the original OCBA rules based on independent samples [98]. In particular, if $k = 2$ and variances are the same, equal allocation is optimal for any positively correlated samples. Otherwise, the greater the difference in the variances, the higher level of positive correlation is required for the equal allocation to be optimal. Also, if the variance difference is sufficiently large, equal allocation can never be optimal. However, these works have some limits. Firstly, exact solutions are available only if $k = 2$. For the general case, only approximated asymptotic allocation rules are derived and there is no guarantee of optimality in a fully-sequential setting. Secondly, the PCS of the best found solution cannot be computed directly and can only be estimated in a Monte Carlo fashion. Thirdly, a significantly large initial sample size (10 or 20) is required in order to compute accurately the optimal allocation.

Variance known

The original OCBA formulation assumes known means and variances in order to derive PCS guarantees based on normal distributions. From a Bayesian perspective, under a non-informative prior, it is possible to make the assumption that $f(x)$ follows a normal prior $N(\mu_x, \sigma_x^2)$ with unknown μ_x and known σ_x^2 [10]. After simulation $F(x, \xi_i)$ is run, the posterior distribution $p_i(f(x)|F(x, \xi_i))$ can be built by updating the prior with the output of $F(x, \xi_i)$. Since σ_x^2 is known, p_i approximately follows a normal distribution with mean $\hat{f}_n(x)$ and variance σ_x^2/n [100]. According to [10], in practice σ_x^2 can be approximated using $\hat{\sigma}_n^2(x)$.
At each step of the optimization, a configuration x_1 must be compared against some other configuration x_2. Since

$$\hat{f}_{n_1}(x_1) - \hat{f}_{n_2}(x_2) \approx N\left(\hat{f}_{n_1}(x_1) - \hat{f}_{n_2}(x_2), \frac{\hat{\sigma}_{n_1}^2(x_1)}{n_1} + \frac{\hat{\sigma}_{n_2}^2(x_2)}{n_2}\right), \tag{4.1}$$

$$P\{\hat{f}_{n_1}(x_1) - \hat{f}_{n_2}(x_2) > 0\} \approx \Phi\left(\frac{\hat{f}_{n_1}(x_1) - \hat{f}_{n_2}(x_2)}{\sqrt{\frac{\hat{\sigma}_{n_1}^2(x_1)}{n_1} + \frac{\hat{\sigma}_{n_2}^2(x_2)}{n_2}}}\right) \tag{4.2}$$

where Φ is the c.d.f. of the standard normal distribution. So, given k configurations which are approximately normally distributed with means $\hat{f}_{n_1}(x_1), \cdots, \hat{f}_{n_k}(x_k)$ and variances $\hat{\sigma}_{n_1}^2(x_1), \cdots, \hat{\sigma}_{n_k}^2(x_k)$, estimated with sample sizes $n_1, \cdots, n_k$, the Approximate Probability of Correct Selection (APCS) is computed as

$$1 - \prod_{i=1, i\neq b}^{k} \Phi\left(\frac{\hat{f}_{n_i}(x_i) - \hat{f}_{n_b}(x_b)}{\sqrt{\frac{\hat{\sigma}_{n_i}^2(x_i)}{n_i} + \frac{\hat{\sigma}_{n_b}^2(x_b)}{n_b}}}\right), \tag{4.3}$$

where b is the index of the solution with the best sample mean.

Variance unknown

Since in practice means and variances are approximated using estimators, t distributions can be used in order to be consistent with the probabilistic assumptions linked to the unknown variance [97, 16]. Therefore, from a Bayesian perspective, under a non-informative prior, it is possible to make the assumption that $f(x)$ follows a Normal-gamma prior $N(\mu_x, \sigma_x)$ with unknown μ_x and unknown σ_x^2 [10]. After simulation $F(x, \xi_i)$ is run, the posterior distribution $p_i(f(x)|F(x, \xi_i))$ can be built by updating the prior with the output of $F(x, \xi_i)$. Since σ_x^2 is unknown, p_i follows a non-standardized t distribution T with sample mean $\hat{f}_n(x)$, precision $n/\hat{\sigma}_n^2(x)$ and $n-1$ d.o.f. [101].
At each step of the optimization, a configuration x_1 is proposed by a heuristic algorithm and must be compared against another configuration x_2. As [97, 20] state, since

$$\hat{f}_{n_1}(x_1) - \hat{f}_{n_2}(x_2) \approx St\left(\mu, r, \nu\right), \tag{4.4}$$

with location $\mu = \hat{f}_{n_1}(x_1) - \hat{f}_{n_2}(x_2)$, precision $r = (\sqrt{\hat{\sigma}_{n_1}^2(x_1)/n_1 + \hat{\sigma}_{n_2}^2(x_2)/n_2})^{-1}$ and ν is Welch's approximation [102] for the d.o.f., then

$$P\{\hat{f}_{n_1}(x_1) - \hat{f}_{n_2}(x_2) > 0\} \approx \Phi_\nu \left(\frac{\hat{f}_{n_1}(x_1) - \hat{f}_{n_2}(x_2)}{\sqrt{\frac{\hat{\sigma}_{n_1}^2(x_1)}{n_1} + \frac{\hat{\sigma}_{n_2}^2(x_2)}{n_2}}} \right) \tag{4.5}$$

where Φ_ν is the c.d.f. of the standardized t distribution and ν corresponds to the d.o.f. defined by Welch's approximation. So, given k configurations which approximately follow a non-standardized t distribu-

tion with locations $\hat{f}_{n_1}(x_1), \cdots, \hat{f}_{n_k}(x_k)$ and variances $\hat{\sigma}^2_{n_1}(x_1), \cdots, \hat{\sigma}^2_{n_k}(x_k)$, estimated with sample sizes $n_1, \cdots, n_k$, Slepian's inequality can be used to compute the APCS as

$$1 - \prod_{i=1, i \neq b}^{k} \Phi_\nu \left(\frac{\hat{f}_{n_i}(x_i) - \hat{f}_{n_b}(x_b)}{\sqrt{\frac{\hat{\sigma}^2_{n_i}(x_i)}{n_i} + \frac{\hat{\sigma}^2_{n_b}(x_b)}{n_b}}} \right). \tag{4.6}$$

4.2.2 Indifference-zone methods

In real world problems, one might not be interested to correctly detect very small differences between solutions. The difference between the means of solutions may be smaller than a certain threshold δ_{iz}, called the IZ parameter, which defines the minimum difference considered to be worth correctly detecting. Thus, if considering solutions which have means that fall into the indifference zone, it does not matter which one of these good alternatives is selected as the best. Therefore, the target is to provide a guarantee of probability of good selection (PGS) rather than the original PCS.

The IZ formulation was first introduced by [40]. Subsequent works include [103, 43], who proposed solutions based on a two-stage approach, and [104, 41, 105, 106, 55], who adopted a fully sequential approach. The goal of IZ procedures is to discard configurations as soon as possible while guaranteeing the PCS, to reduce the computational burden required to find x^*. But, as highlighted by [44], since most fully sequential procedures use the Bonferroni inequality to guarantee the PCS, they are too conservative. Examples are [41, 105, 106, 55]. Unfortunately, all these methods assume that no solution has been sampled before the beginning of the R&S procedure. Also, they do not exploit

samples obtained previously, and already visited solutions are always evaluated anew. In contrast, during the heuristic search, using information about already visited solutions is desirable. But extending IZ techniques in order to exploit previous samples (and proving their validity), is not straightforward [107].
Differently from the previously mentioned R&S approaches, [105, 55] propose methods to be applied during the heuristic search. Both works are based on the results of [108, 109], who improved the PCS bounds of Paulson's procedure [104]. These approaches use a fixed triangular region, also called continuation region, to decide the sample size which should be used for each comparison in order to guarantee the PCS of the procedure. [105] introduce a fully-sequential scheme called Sequential Selection with Memory (SSM), which eliminates solutions as soon as they are considered as statistically inferior with respect to the best. The algorithm memorizes samples found during the search, in order to reuse them during the optimization. Then, at each iteration of SSM, an additional observation is taken for each surviving solution; solutions might be eliminated if the cumulative sum between the evaluations of the solution and the best falls out from the continuation region. [55] propose fully-sequential R&S methods for problems in which solutions are generated sequentially. They propose single-elimination approaches, where each solution is considered only once, and stop-and-go schemes where each solution is considered throughout the whole optimization. However, in order to fulfill the PCS requirement using the triangular continuation region, a large number of evaluations is necessary. Both procedures aim at guaranteeing a fixed-precision and are based on the Bonferroni inequality. Thus, they are more conservative with respect to OCBA and therefore constitute less effective choices for integration with heuristic search algorithms.

4.2.3 Paired and unpaired comparisons

In hypothesis testing, the difference in means can be tested according to diverse assumptions about the samples used to compute the estimators. In fact, samples can be considered as paired or unpaired. In simulation-based optimization approaches which adopt CRN, the assumption which leads to the minimum sample size required to take a statistically significant decision depends on the amount of positive correlation introduced by CRN.
At each step of a heuristic search algorithm, in the case that $k = 2$, a new configuration x_{new} is proposed and must be compared against the current best configuration $x_{current}$. The assumption of paired or unpaired samples changes the statistic used to compute the p-value, and also the d.o.f. of the distribution followed by the difference in means. Therefore, diverse assumptions lead to different probability guarantees, with a significant impact on the efficiency of the optimization.

Student's and Welch's t-tests

The following definitions extend the formulation provided in Section 3.2, by considering the possibility that the sample sizes might differ. In this case, null and alternative hypothesis of the upper-tailed test on the difference in means are defined as

$$\begin{aligned} H_0 &: \hat{f}_{n_1}(x_{new}) - \hat{f}_{n_2}(x_{current}) \leq \delta_{H_0} \\ H_1 &: \hat{f}_{n_1}(x_{new}) - \hat{f}_{n_2}(x_{current}) > \delta_{H_0}, \end{aligned} \tag{4.7}$$

where $\delta_{H_0} = f(x_{new}) - f(x_{current}) = 0$ defines the value for which the null hypothesis is assumed to be true. To take a statistically significant decision, samples can be assumed to be paired or unpaired and, in the

latter case, assuming equal or diverse variances.
In the first case, using Student's paired t-test, the mean of paired differences corresponds to the difference of means and the statistic to test is

$$T_{n-1} = \frac{\delta_{obs} - \delta_{H_0}}{\hat{\sigma}_n/\sqrt{n}}, \tag{4.8}$$

where $n = n_1 = n_2$, observed difference $\delta_{obs} = \hat{f}_n(x_{new}) - \hat{f}_n(x_{current})$ and $\hat{\sigma}_n$ is the sample standard deviation of paired evaluations. Also, since σ of F is approximated by using the sample standard deviation of the differences, T_{n-1} follows a standardized t-distribution with $n-1$ d.o.f. In the second case, assuming equal variances, Student's unpaired t-test defines the statistic to test as

$$T_{n_1+n_2-2} = \frac{\delta_{obs} - \delta_{H_0}}{\hat{\sigma}_p\sqrt{\frac{1}{n_1} + \frac{1}{n_2}}}, \tag{4.9}$$

where $\hat{\sigma}_p$ is the pooled estimator of the variance of δ_{obs} and $T_{n_1+n_2-2}$ follows a standardized t-distribution with $n_1 + n_2 - 2$ d.o.f. Otherwise, assuming unequal variances, Welch's approximation determines the statistic to test as

$$T_\nu = \frac{\delta_{obs} - \delta_{H_0}}{\sqrt{\frac{\hat{\sigma}^2_{n_1}}{n_1} + \frac{\hat{\sigma}^2_{n_2}}{n_2}}}, \tag{4.10}$$

where ν corresponds to the d.o.f defined by Welch's approximation. The aforementioned statistics are used to compute the p-value, which defines how unlikely it is to observe the statistic if H_0 is true. If this is too unlikely , then H_0 should be rejected. However, it is also important to consider the probability β to make an error of type II, so to fail to reject H_0 when H_0 is false. To compute β, a specific value for which

H_1 is assumed to be true has to be defined, because β depends on the difference δ_{obs} for which H_0 is assumed to be false and H_1 to be true. When H_1 is true, the statistic follows a noncentral t-distribution. According to [92], in the case of homoscedastic noise, β can be computed according to a generic formula. Assuming known variances and means, let the probability of making an error of type II to be computed as

$$\beta = 1-\Phi(\delta_{norm}\sqrt{n}-(\sigma_{new}/\sigma_{current})z_{\alpha/2})-\Phi(-\delta_{norm}\sqrt{n}-\sigma_{new}/\sigma_{current})z_{\alpha/2}) \tag{4.11}$$

where $\sigma_{new} = \sigma(x_{new})$, $\sigma_{current} = \sigma(x_{current})$, $\delta_{norm} = f(x_{current}) - f(x_{new})/\sigma_{current}$ and $z_{\alpha/2}$ is the quantile x of the standard normal distribution such that $\Phi(x) = \alpha/2$. In the case of paired comparisons $n = n_1 = n_2$, otherwise n corresponds to the sample size of the smallest sample. Power size calculations for one-sided tests are obtained by doubling the value of α. However, if $\sigma_{new} \neq \sigma_{current}$, (4.11) does not yield a closed solution and a result can be obtained through iterative methods. Thus, assuming that variances and means are estimated using sample statistics, in the one-tailed case with paired samples

$$\beta \approx 1 - \Phi_{n-1}(\delta_{norm}\sqrt{n} - t_{n-1,\alpha}) - \Phi_{n-1}(-\delta_{norm}\sqrt{n} - t_{n-1,\alpha}) \tag{4.12}$$

where Φ_{n-1} is c.d.f. of the standardized t-distribution with $n-1$ d.o.f., $\delta_{norm} = (\delta_{obs} - \delta_{H_0})/\hat{\sigma}_n$ and $t_{n-1,\alpha}$ is the quantile x of the standardized t-distribution with $n-1$ d.o.f. such that $T_{n-1}(x) = \alpha$. In the one-tailed case with unpaired samples

$$\beta \approx 1-\Phi_{n(k+1)-2}(\delta_{norm}\sqrt{n}-t_{n(k+1)-2,\alpha})-\Phi_{n(k+1)-2}(-\delta_{norm}\sqrt{n}-t_{n(k+1)-2,\alpha}) \tag{4.13}$$

where k is the ratio between the two samples used in the comparison and $\delta_{norm} = (\delta_{obs} - \delta_{H_0})/\hat{\sigma}_p$. Unfortunately, in the case of Welch's approximation, the computation of β is more complicated. [110] propose an iterative method, which can be adapted to the one-tailed case. This approach is based on the numerical evaluation of the c.d.f. of a noncentral t-distribution that is necessary to compute Welch's approximation, and its integration with respect to a Beta distribution. The computation of the c.d.f. of the noncentral t-distribution is based on [111].

On the assumptions of parametric tests

Parametric tests are usually more powerful than their non-parametric alternatives, but they make some assumptions. Student's t-test assumes that F is normally distributed and the variances of samples are at least approximately similar. When the sample sizes of the groups are equal, Student's t-test is robust to unequal variances as long as the estimators do not deviate significantly from the true values. But, in the case that both variances and sample sizes are significantly different across samples, the test can be biased [112, 113]. Also, Student's t-test robustness to unequal variances relies on the assumption that distributions are at least approximately normally distributed. This it typically the case in simulation, because the output of a simulator is estimated using an average metric that tends to satisfy the Central Limit Theorem [10]. Although the normality assumption is not always valid, it is often possible to batch a number of evaluations so that normality is approximately satisfied [20, 16]. Therefore, if the sample sizes are equal, Student's t-test can be a valid statistic to adopt even in the case that variances are different. Also, notice that as the quality

of solutions becomes more and more similar throughout the search, comparisons are going to require a larger sample size in order to take a decision. As a consequence, even comparisons based on paired samples iteratively become more robust. However, if sample sizes are significantly different across samples, Welch's t-test is more robust [114]. Whenever a positive correlation exists within paired samples, the denominator for the paired t-test is smaller than its counterpart in the unpaired t-test. Even if a paired test has an advantage in the case of positive correlation, at the same time it has a loss of $n - 1$ d.o.f with respect to an unpaired test. Increasing the d.o.f. of a test changes the shape of the distribution of the statistic, improving its power. But it is worth noticing that as long as the d.o.f. are large enough (usually at least 30), the loss of d.o.f. does not significantly change the shape of the distribution. Therefore, the tradeoff lies between the reduced variance caused by the positive correlation among dependent samples and the increase of power caused by the additional d.o.f. of the statistic.

4.3 Integrations with R&S techniques

In the classic R&S setting, no solution is sampled before the beginning of the procedure, and already visited solutions are always evaluated anew. In contrast, during the heuristic search phase, using information about already visited solutions is desirable. Similarly to [105], in the proposed integrations all evaluations of solutions visited during the search are kept in memory, in order to avoid the waste of computational budget in the case of repeated comparisons.

4.3.1 Hypothesis testing procedure

By taking a statistical hypothesis testing (HT) perspective, significant differences between means can be detected by considering the probabilities α and β of making an error of type I and type II. A statistically significant decision can be taken according to the value of α when the power of the test, defined as $1 - \beta$, is sufficiently high. The relationship between α and β is based on the importance given to correctly detecting an effect on the population [115]. In most of the scientific literature, concluding that there *is* an effect when there is *no* effect in the population is usually considered four times as serious as concluding there is *no* effect when there *is* an effect in the population [116]. Therefore, if $\alpha = 0.05$, a generally accepted minimum level of power is 0.80.

Given a null hypothesis H_0 and an alternative hypothesis H_1, α defines the probability to reject H_0 (assuming H_0 is true and H_1 is false) and β corresponds to the probability to fail to reject H_0 (assuming H_1 is true and H_0 is false). The p-value can be computed by using Student's or Welch's t-tests for the difference in means. If the observed value of the chosen statistic is too unlikely, then H_0 should be rejected. But, in order to take a statistically significant decision, enough statistical evidence to provide a guarantee for β should also be gathered. To do so, a significant difference in means to be detected, for which H_0 is assumed to be false and H_1 to be true is necessary. This difference defines the minimum *effect* in the population which is important to detect, and it corresponds to δ_{norm} in the power formulas of Section 4.2.3.

According to this framework, given $k = 2$ solutions to be compared, a statistically significant decision can be taken as follows. Firstly, gather

an initial first-stage sample of $n_0 = 2$ evaluations for each solution. Secondly, in a fully-sequential manner, increment the sample of each solution with 1 additional evaluation based on a new realization ξ_i (unless the evaluation for that realization has been saved in memory). Because of the CRN assumption, the realization is the same for both solutions. Evaluations are sequentially requested until a statistically significant decision can be taken according to one of the stopping rules based on the probability guarantees defined in Section 4.2.3.
Differently from the procedure in Section 3.2, which uses the statistical guard mechanism to take a decision whenever the observed difference in means is lower than a certain threshold, this procedure is able to take a statistically significant decision within a margin of error defined by δ_{H_0}.

4.3.2 OCBA using paired and unpaired samples

As mentioned in Section 4.2.3, the difference in means can be tested according to diverse assumptions. By assuming paired or unpaired samples, different approximations of the PCS can be used to take statistically significant decisions.
In simulation-based optimization, the amount of positive correlation induced by CRN depends on how the internal models of the simulator react to different values of the decision variables. Thus, it is not possible to decide *a priori* which assumption leads to the most efficient stopping rule. The correlation among pairs of solutions might vary across Θ, and depend on the similarity of the solutions. Therefore, it is reasonable to expect that a single stopping rule based on paired or unpaired samples is not going to always be the most efficient choice. According to the comparison, assuming paired samples might be less

or more efficient than assuming unpaired samples. One can consider both stopping criteria simultaneously, taking a statistically significant decision as soon as one of the two APCS is satisfied. In the case of paired samples, according to Equation (4.8)

$$P\{\hat{f}_n(x_1) - \hat{f}_n(x_2) > \delta_{H_0}\} \approx T_{n-1} = \frac{\hat{f}_n(x_1) - \hat{f}_n(x_2) - \delta_{H_0}}{\hat{\sigma}_n/\sqrt{n}} \quad (4.14)$$

with $n = n_1 = n_2$. So, given k configurations whose paired differences follow a standardized t distribution with $n-1$ d.o.f, the APCS based on paired samples is computed as

$$1 - \prod_{i=1, i\neq b}^{k} \Phi_{n-1}\left(\frac{\hat{f}_n(x_i) - \hat{f}_n(x_b) - \delta_{H_{0ib}}}{\hat{\sigma}_n(x_i, x_b)/\sqrt{n}}\right), \quad (4.15)$$

where $\hat{\sigma}_n(x_i, x_b)$ is the sample standard deviation of paired evaluations of x_i and x_b. In the case of unpaired samples, the APCS is computed according to Welch's approximation as defined in Equation (4.6).

4.3.3 IZ selection

In the traditional R&S scenario, δ_{iz} is statically defined by the decision maker. The objective of the statistical analysis is to find the best solution among a static set of k alternatives by at least δ_{iz} (PCS case) or a solution which is at most δ_{iz} away from the best (PGS case). In this setting, regardless of the budget, the priority is to provide a fixed-precision guarantee. In contrast, if a R&S method is integrated within a search algorithm which is also bound by the same total budget, the precision of estimation impacts also the search policy and the target is to achieve the best possible PCS (or PGS) within a fixed budget. Thus,

the higher the precision, the less the search space is explored. A set of k solutions is compared at each iteration of the search, but the quality of the final result depends also on the exploration-vs-exploitation tradeoff of the search.
The magnitude of the objective function might change throughout the search space. A static δ_{iz} does not consider this variation, because it is formulated according to the best possible outcome that the decision maker expects from the optimization. However, this is just an approximation which must be manually defined and it is not easy to be correctly determined *a priori*. Therefore, one could define δ_{iz} as a percentage of the sample mean of the current best estimator, in order to consider an IZ parameter which changes according to the variations of magnitude during the search as

$$\delta_{iz} = p \cdot \hat{f}(x_{current}),$$

with $0 < p \leq 1$. So, statistically significant differences worth detecting become proportional to the magnitude of the objective function.

4.4 Experiments

Experiments are run on four benchmark functions extended with artificial noise and HotelSimu. The benchmarks are Sphere, Rastrigin, Griewank and Ackley functions, defined in two dimensions. All functions have a global optimum in $(0, 0)$ and, apart from the first one, are multimodal [117, 118].

4.4.1 Noise in benchmark functions

To evaluate the impact of noise on the optimization, a standard practice in the literature is to extend deterministic functions by introducing additive noise. In this Chapter, benchmarks are perturbed by introducing different levels of additive noise as

$$f(x) = f(x) + N(0, d \cdot \sigma_\epsilon) \tag{4.16}$$

where $\sigma_\epsilon \in \{1, 2, 3\}$ and d defines to the dimensions of the function. In the experiments, the additive noise is generated in order to simulate different levels of correlation induced by the adoption (or not) of CRN. The amount of correlation is controlled by sampling the noise of paired samples from a Bivariate Normal Distribution with correlation coefficient ρ. Multiple correlation levels are used to provide empirical evidence about how paired comparisons take advantage of the positive correlation between samples with respect to the help obtained by the additional d.o.f. employed by the statistics based on unpaired samples. Two random variables X_1, X_2 follow a Bivariate Normal Distribution with correlation coefficient ρ when

$$\begin{pmatrix} X_1 \\ X_2 \end{pmatrix} \sim N\left[\begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix}, \begin{pmatrix} \sigma_1^2 & \rho\sigma_1\sigma_2 \\ \rho\sigma_1\sigma_2 & \sigma_2^2 \end{pmatrix}\right] \tag{4.17}$$

Let Z_1, Z_2 be two independent random variables which follow a standard normal distribution. To sample from a Bivariate Normal Distribution, samples from Z_1, Z_2 are transcformed according to

$$\begin{aligned} X_1 &= \sigma_1 Z_1 + \mu_1 \\ X_2 &= \sigma_2(\rho Z_1 + \sqrt{1-\rho^2} Z_2) + \mu_2 \end{aligned} \tag{4.18}$$

where $\mu_1 = \mu_2 = 0$ and $\sigma_1 = \sigma_2 = d \cdot \sigma_\epsilon$. In the experiments, $\rho \in \{0.00, 0.25, 0.50, 0.75, 1.00\}$. In fact, if CRN are used, it is reasonable to assume that the correlation between simulations based on the same realizations is going to be non negative. Also, the correlation between nearby solutions in the search space is expected to be higher with respect to solutions which are far away.

4.4.2 Setup

Each experiment is based on 100 macroreplications initialized with a different ξ_s, and each macroreplication adopts a budget (number of function evaluations) of $500 \cdot d$, where d is the dimension of the problem. The initial solution of each macroreplication is generated according to a uniform distribution defined on the interval of each dimension of Θ. All values of benchmark functions are normalized by the dimensionality. Furthermore, in the experiments based on Hotelsimu, the ground truth is not available and the quality of each best found solution is approximated by building an estimator based on a fixed set of $n = 100$ seeds (which are disjoint from the ones used in the macroreplications). At the end of the optimization, the solution x^* with the best measured performance is returned.

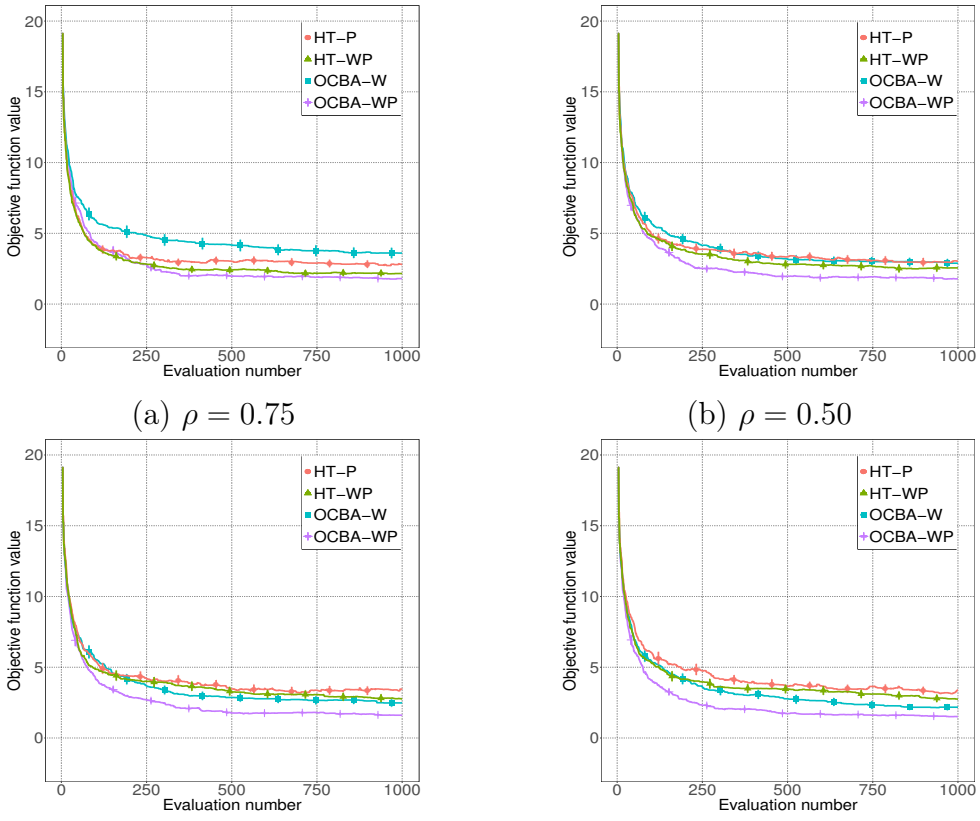
4.4.3 Local search algorithm

In each experiment, the algorithm starts the optimization from a randomly generated solution x_0 (according to a uniform distribution defined on Θ) and considers a local search region around x_0. The local search region is iteratively adapted according to a variant of random local search (RLS). In RLS, a new candidate solution x_{new} is sam-

pled from an interval defined around the current best solution $x_{current}$, according to some distribution. In the RLS variant adopted for the experiments, called dynamic RLS (D-RLS), a uniform distribution is used to sample new solutions. A step size p defines, as a percentage of the interval in which the function is defined, the boundaries of the interval around the best current solution in which new candidate solutions are sampled. Consequently, diverse values of p correspond to search policies with different levels of locality. A step size of 1 makes the search global, and the optimization corresponds to pure random search. Initially, and after each restart, $p = 0.5$. The neighborhood of $x_{current}$ is defined as a dynamic box which is enlarged each time x_{new} is better than $x_{current}$ and shrinked otherwise. In both cases, similarly to [93], the expansion factor has a ratio of 1/10. The search is restarted in two cases: whenever the search box becomes too small ($p < 0.01$), or when a stagnation is detected (the quality of the current best solution does not improve by at least 1% after 100 function evaluations). Restarts are of two different types: from randomly chosen points in Θ, or as the average of the best solutions found during the optimization before a restart. In order to balance exploration and exploitation of previous results, restarts are alternated.

4.4.4 Results

D-RLS is integrated with the statistical techniques enumerated in Section 4.3. Two sets of tests have been designed to consider integrations which are based on PCS or PGS guarantees, so in order to include or not the IZ formulation. In the first set, the HT and OCBA procedures are combined with D-RLS in order to provide PCS guarantees. In the second set, the HT and OCBA methods are extended according to the



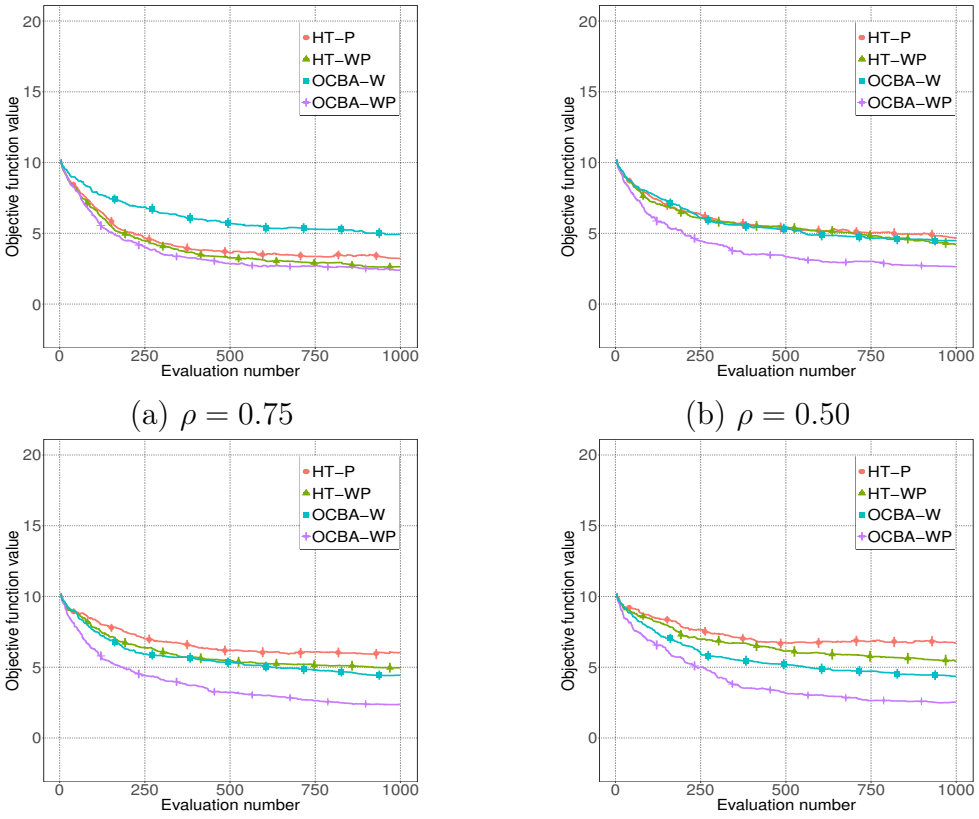
(a) $\rho = 0.75$ (b) $\rho = 0.50$

(c) $\rho = 0.25$ (d) $\rho = 0.00$

Figure 4.1: Noiseless mean and standard error of the solutions with best measured performance found during the optimization, at different levels of correlation (Rastrigin function values, with $d = 2$ and $\sigma = 3$).

IZ formulation in order to provide PGS guarantees.

Both procedures are tested with different probability guarantees which stop the sequential allocation of samples (stopping criteria) according to the statistics defined in Section 4.2.3. In the following analysis, stopping criteria are based on Student's t paired statistic (P), Welch's t unpaired statistic (W), or simultaneously on both statistics (PW). OCBA based on Welch's stopping criterion (OCBA-W) corresponds to the standard OCBA formulation based on the t-distribution approximation proposed by [97].



(a) $\rho = 0.75$ (b) $\rho = 0.50$

(c) $\rho = 0.25$ (d) $\rho = 0.00$

Figure 4.2: Noiseless mean and standard error of the solutions with best measured performance found during the optimization, at different levels of correlation (Ackley function values, with $d = 2$ and $\sigma = 3$).

First set of experiments

In the vast majority of cases, OCBA-WP obtains the best performance both in terms of efficiency and quality of the solution found at the end of the optimization. Figure 4.1 and Figure 4.2 show that on average OCBA-WP converges to the best found solution before the other comparison schemes at different levels of correlation. This achievement is reached in almost all experiments, with the exception of a few cases where HT-P or HT-WP outperform OCBA-WP when the correlation

Table 4.1: Noiseless mean and standard error of the solutions with best measured performance found during the optimization (Sphere function values, $d = 2$).

	$\rho = 0.25$			$\rho = 0.50$		
Optimizer	$\sigma = 1$	$\sigma = 2$	$\sigma = 3$	$\sigma = 1$	$\sigma = 2$	$\sigma = 3$
R-P	0.22 ± 0.04	0.51 ± 0.08	0.60 ± 0.10	0.20 ± 0.04	0.36 ± 0.05	$\mathbf{0.42 \pm 0.06}$
HT-W	0.34 ± 0.05	0.49 ± 0.09	0.71 ± 0.08	0.37 ± 0.05	0.59 ± 0.11	0.77 ± 0.12
HT-WP	0.21 ± 0.04	0.42 ± 0.05	0.60 ± 0.10	0.18 ± 0.03	$\mathbf{0.34 \pm 0.04}$	0.59 ± 0.06
OCBA-W	0.20 ± 0.02	0.45 ± 0.04	0.68 ± 0.06	0.21 ± 0.03	0.48 ± 0.06	0.68 ± 0.08
OCBA-WP	$\mathbf{0.19 \pm 0.02}$	$\mathbf{0.41 \pm 0.04}$	$\mathbf{0.47 \pm 0.05}$	$\mathbf{0.16 \pm 0.02}$	0.41 ± 0.05	0.47 ± 0.05

Table 4.2: Noiseless mean and standard error of the solutions with best measured performance found during the optimization (Rastrigin function values, $d = 2$).

	$\rho = 0.25$			$\rho = 0.50$		
Optimizer	$\sigma = 1$	$\sigma = 2$	$\sigma = 3$	$\sigma = 1$	$\sigma = 2$	$\sigma = 3$
HT-P	2.06 ± 0.27	2.74 ± 0.29	3.49 ± 0.34	2.17 ± 0.27	2.73 ± 0.29	3.03 ± 0.31
HT-W	2.28 ± 0.27	3.51 ± 0.39	3.67 ± 0.36	2.83 ± 0.35	3.67 ± 0.38	4.03 ± 0.38
HT-WP	1.84 ± 0.26	2.54 ± 0.26	2.76 ± 0.27	1.59 ± 0.23	2.29 ± 0.27	2.56 ± 0.29
OCBA-W	1.34 ± 0.24	2.04 ± 0.27	2.48 ± 0.26	1.72 ± 0.26	2.20 ± 0.26	2.89 ± 0.30
OCBA-WP	$\mathbf{0.70 \pm 0.05}$	$\mathbf{1.16 \pm 0.07}$	$\mathbf{1.60 \pm 0.09}$	$\mathbf{0.81 \pm 0.04}$	$\mathbf{1.09 \pm 0.08}$	$\mathbf{1.78 \pm 0.13}$

Table 4.3: Noiseless mean and standard error of the solutions with best measured performance found during the optimization (Griewank function values, $d = 2$).

	$\rho = 0.25$			$\rho = 0.50$		
Optimizer	$\sigma = 1$	$\sigma = 2$	$\sigma = 3$	$\sigma = 1$	$\sigma = 2$	$\sigma = 3$
HT-P	0.78 ± 0.06	0.93 ± 0.06	1.32 ± 0.14	0.67 ± 0.02	0.97 ± 0.10	0.96 ± 0.09
HT-W	0.99 ± 0.12	1.05 ± 0.09	1.47 ± 0.16	0.97 ± 0.08	1.06 ± 0.08	1.37 ± 0.15
HT-WP	0.83 ± 0.07	0.97 ± 0.05	1.18 ± 0.13	0.72 ± 0.04	0.95 ± 0.10	1.01 ± 0.05
OCBA-W	0.82 ± 0.61	1.04 ± 0.49	1.17 ± 0.69	0.82 ± 0.95	1.14 ± 1.18	1.36 ± 0.98
OCBA-WP	$\mathbf{0.71 \pm 0.18}$	$\mathbf{0.88 \pm 0.29}$	$\mathbf{1.01 \pm 0.48}$	$\mathbf{0.67 \pm 0.17}$	$\mathbf{0.82 \pm 0.33}$	$\mathbf{0.95 \pm 0.44}$

Table 4.4: Noiseless mean and standard error of the solutions with best measured performance found during the optimization (Ackley function values, $d = 2$).

	$\rho = 0.25$			$\rho = 0.50$		
Optimizer	$\sigma = 1$	$\sigma = 2$	$\sigma = 3$	$\sigma = 1$	$\sigma = 2$	$\sigma = 3$
HT-P	2.18 ± 0.23	4.02 ± 0.33	6.04 ± 0.35	1.92 ± 0.21	3.65 ± 0.34	4.64 ± 0.34
HT-W	3.06 ± 0.31	5.12 ± 0.35	6.54 ± 0.37	3.77 ± 0.36	5.25 ± 0.38	6.84 ± 0.34
HT-WP	1.98 ± 0.22	3.30 ± 0.30	4.96 ± 0.34	1.90 ± 0.24	2.80 ± 0.30	4.21 ± 0.33
OCBA-W	2.05 ± 0.25	2.95 ± 0.30	4.43 ± 0.33	1.69 ± 0.18	3.51 ± 0.32	4.49 ± 0.34
OCBA-WP	$\mathbf{0.70 \pm 0.05}$	$\mathbf{1.66 \pm 0.09}$	$\mathbf{2.38 \pm 0.13}$	$\mathbf{0.79 \pm 0.06}$	$\mathbf{1.78 \pm 0.11}$	$\mathbf{2.64 \pm 0.17}$

is high ($\rho = 0.75$ or $\rho = 1.00$). Tables 4.4 - 4.1 provide the noiseless value of the average best found solution at the end of the optimization with $\rho = 0.5$ and $\rho = 0.25$, at different levels of additive noise. Not all results are reported in the tables, but the outcomes are coherent with the observations above. The same outcome is observed in Figure 4.3, where the optimization is applied on a particular booking scenario in HotelSimu (the same use case scenario of Hotel 07 in Section 2.5.2). HT-WP performs similarly to OCBA-W under mild amounts of correlation ($\rho = 0.25$ or $\rho = 0.50$). As the correlation decreases, OCBA-W performs slightly better and viceversa. Although, with high levels of correlation ($\rho = 0.75$), even HT-P performs better with respect to OCBA-W. By considering simultaneously both paired and Welch stopping criteria, on average both HT and OCBA methods significantly improve their performance with respect to their counterparts which only consider stopping criteria singularly.

Second set of experiments

The IZ formulation improves the efficiency of both HT and OCBA procedures. The performance gap between HT and OCBA is drastically reduced, but OCBA-WP performs better with respect the other approaches in most case. Also, differently from the previous set of experiments, HT-WP achieves better results with respect to OCBA-W, the classical OCBA formulation. By requesting the null hypothesis value to be different from 0, an indifference zone is introduced in the difference in means in order to reduce the signal which is supposed to be correctly detected. As a consequence, comparisons become less conservative and more budget is available for the optimization. By considering the PGS, the difference in terms of performance between

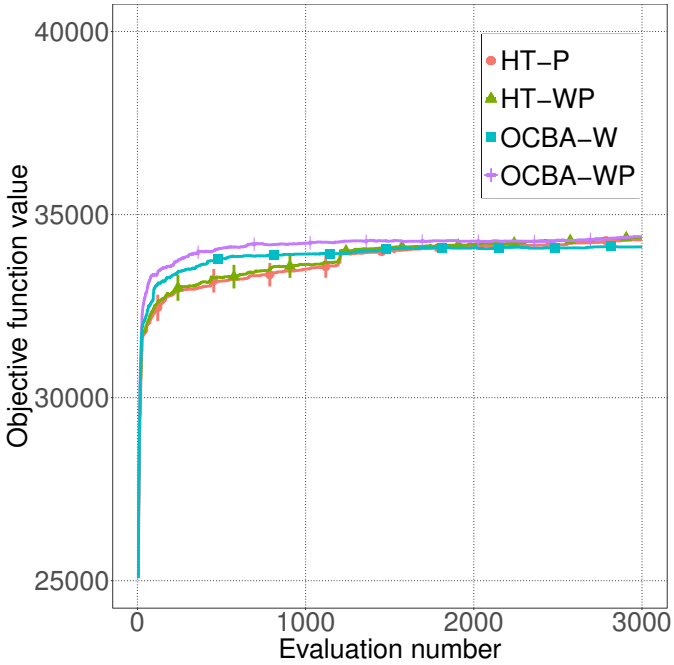


Figure 4.3: Estimated mean and standard error of the best solutions found during the optimization (HotelSimu revenue from hotel 07).

HT and OCBA is reduced. Tables 4.5 - 4.8 show a subset of the results on benchmark functions for the case where $\rho = 0.50$ and $\sigma = 3$. The adoption of an absolute δ_{abs} implicitly requires a manual adaptation which depends on the magnitude of the function, since there is no direct relationship between δ_{abs} and the function value. Defining a relative δ_{rel} removes this limit, and the performance is similar. However, even in this case, some manual tuning is required. A possible direction for future research involves the dynamic adaptation of the IZ parameter according to the step size of the algorithm.

Table 4.5: Noiseless mean and standard error of the solutions with best measured performance found during the optimization (Sphere function values, using IZ with $\rho = 0.50$ and $\sigma = 3$).

	absolute			relative		
Optimizer	$\delta_{abs} = 0.10$	$\delta_{abs} = 0.50$	$\delta_{abs} = 1.00$	$\delta_{rel} = 0.01$	$\delta_{rel} = 0.05$	$\delta_{rel} = 0.10$
HT-P	0.35 ± 0.04	$\mathbf{0.27 \pm 0.04}$	$\mathbf{0.27 \pm 0.03}$	0.39 ± 0.05	0.34 ± 0.04	$\mathbf{0.29 \pm 0.03}$
HT-W	0.62 ± 0.08	0.38 ± 0.04	0.43 ± 0.06	0.66 ± 0.08	0.61 ± 0.07	0.54 ± 0.05
HT-WP	0.57 ± 0.07	0.40 ± 0.03	0.37 ± 0.04	0.51 ± 0.05	0.51 ± 0.05	0.39 ± 0.04
OCBA-W	0.73 ± 0.08	0.53 ± 0.05	0.42 ± 0.04	0.62 ± 0.07	0.69 ± 0.07	0.48 ± 0.06
OCBA-WP	$\mathbf{0.31 \pm 0.04}$	0.31 ± 0.04	0.31 ± 0.04	$\mathbf{0.30 \pm 0.04}$	$\mathbf{0.33 \pm 0.04}$	0.31 ± 0.04

Table 4.6: Noiseless mean and standard error of the solutions with best measured performance found during the optimization (Rastrigin function values, using IZ with $\rho = 0.50$ and $\sigma = 3$).

	absolute			relative		
Optimizer	$\delta_{abs} = 0.10$	$\delta_{abs} = 0.50$	$\delta_{abs} = 1.00$	$\delta_{rel} = 0.01$	$\delta_{rel} = 0.05$	$\delta_{rel} = 0.10$
R-P	2.76 ± 0.30	2.58 ± 0.28	1.70 ± 0.18	2.60 ± 0.22	1.80 ± 0.13	1.68 ± 0.16
R-W	3.82 ± 0.36	2.46 ± 0.19	1.64 ± 0.11	3.20 ± 0.24	2.02 ± 0.13	1.92 ± 0.15
R-WP	2.31 ± 0.26	1.41 ± 0.10	$\mathbf{1.12 \pm 0.08}$	2.20 ± 0.21	1.34 ± 0.09	$\mathbf{1.23 \pm 0.08}$
OCBA-W	2.59 ± 0.27	1.83 ± 0.13	1.67 ± 0.10	2.17 ± 0.18	1.80 ± 0.12	1.81 ± 0.13
OCBA-WP	$\mathbf{1.33 \pm 0.08}$	$\mathbf{1.35 \pm 0.08}$	1.32 ± 0.08	$\mathbf{1.27 \pm 0.08}$	$\mathbf{1.33 \pm 0.08}$	1.32 ± 0.08

Table 4.7: Noiseless mean and standard error of the solutions with best measured performance found during the optimization (Griewank function values, using IZ with $\rho = 0.50$ and $\sigma = 3$).

	absolute			relative		
Optimizer	$\delta_{abs} = 0.10$	$\delta_{abs} = 0.50$	$\delta_{abs} = 1.00$	$\delta_{rel} = 0.01$	$\delta_{rel} = 0.05$	$\delta_{rel} = 0.10$
R-P	0.90 ± 0.04	0.82 ± 0.04	0.88 ± 0.11	0.88 ± 0.04	0.87 ± 0.04	0.83 ± 0.07
R-W	1.22 ± 0.12	1.08 ± 0.08	0.95 ± 0.05	1.41 ± 0.15	1.08 ± 0.06	1.00 ± 0.05
R-WP	0.92 ± 0.05	0.85 ± 0.04	0.87 ± 0.04	0.91 ± 0.04	0.93 ± 0.04	0.89 ± 0.04
OCBA-W	1.20 ± 0.08	1.14 ± 0.06	1.00 ± 0.04	1.28 ± 0.09	1.12 ± 0.07	0.83 ± 0.07
OCBA-WP	$\mathbf{0.79 \pm 0.03}$	$\mathbf{0.79 \pm 0.03}$	$\mathbf{0.79 \pm 0.03}$	$\mathbf{0.83 \pm 0.03}$	$\mathbf{0.79 \pm 0.03}$	$\mathbf{0.80 \pm 0.03}$

Table 4.8: Noiseless mean and standard error of the solutions with best measured performance found during the optimization (Ackley function values, using IZ with $\rho = 0.50$ and $\sigma = 3$).

	absolute			relative		
Optimizer	$\delta_{abs} = 0.10$	$\delta_{abs} = 0.50$	$\delta_{abs} = 1.00$	$\delta_{rel} = 0.01$	$\delta_{rel} = 0.05$	$\delta_{rel} = 0.10$
R-P	3.81 ± 0.30	2.06 ± 0.18	1.75 ± 0.18	3.84 ± 0.28	2.33 ± 0.20	1.76 ± 0.15
R-W	6.42 ± 0.38	4.23 ± 0.34	2.10 ± 0.16	6.33 ± 0.37	2.75 ± 0.17	1.99 ± 0.10
R-WP	3.48 ± 0.29	1.81 ± 0.11	$\mathbf{1.51 \pm 0.10}$	3.22 ± 0.28	1.96 ± 0.12	$\mathbf{1.57 \pm 0.07}$
OCBA-W	4.15 ± 0.32	2.51 ± 0.18	2.43 ± 0.11	4.00 ± 0.28	2.35 ± 0.10	1.76 ± 0.15
OCBA-WP	$\mathbf{1.67 \pm 0.08}$	$\mathbf{1.67 \pm 0.08}$	1.72 ± 0.08	$\mathbf{1.75 \pm 0.09}$	$\mathbf{1.68 \pm 0.08}$	1.65 ± 0.08

4.5 Conclusions

This Chapter integrated different statistical analysis techniques with a variant of random local search and investigated the respective effects on the heuristic search in simulation-based optimization. Since CRN are commonly used to reduce the impact of noise on the computational load of simulation-based optimization, tests on benchmark functions considered various levels of additive noise and positive correlation.
Results show that the adoption of multiple stopping criteria based on Student's paired t-test and Welch's unpaired t-test significantly improves the efficiency of the optimization. In particular, OCBA-WP improves its performance with respect to its classic formulation as the amount of correlation among samples increases. Also, even in the case that the samples are uncorrelated, OCBA-WP still performs better or approximately the same as OCBA-W. However, in the presence of negative correlation (as in the case that variance reduction techniques like antithetic sampling are employed), paired comparisons would be penalized and not particularly effective.
The introduction of the IZ parameter in the formulation improves significantly the efficiency of the optimization in all cases, because comparisons are not required to detect exact differences. Thus, statistically significant decisions can be taken using smaller sample sizes and more budget is available to explore the search space. Although it is possible that the best visited configuration is not selected in favour of a good alternative, considering the PGS to provide a probability guarantee leads to a faster convergence with respect to PCS. In the heuristic search setting, the impact of a wrong decision is not as serious as in the traditional R&S scenario. In the former case, even if a solution is discarded, a similar solution might be sampled again during

the optimization. In the latter case, once a solution is discarded, it is removed completely and not considered further. Also, as the quality of solutions becomes more and more similar throughout the search, comparisons are going to require a larger sample size in order to take a decision and comparisons iteratively become more robust.

Chapter 5

Conclusions

This dissertation introduced HotelSimu, a novel simulation-based optimization approach for hotel RM, and investigated various statistical analysis approaches which can be employed to improve the efficiency of black-box methods in noisy scenarios.
In order to maximize the revenue of an hotel, HotelSimu avoids the formulation of a mathematical model and employs a discrete-event simulator which mimics the processes that define the hotel booking scenario. The generation of multiple realizations of the stochastic process used to model the booking logic is employed to estimate the total revenue obtained by several dynamic pricing policies. The expectation of the revenue distribution is used by black-box optimization methods to find the dynamic pricing policy which is expected to maximize the revenue. Furthermore, HotelSimu's parametric demand models can be used to inject new information into the simulator, in order to adapt pricing policies to mutated market conditions and run *what-if* analyses. Possible directions of future work could involve the extension of the dynamic pricing model by using non-linear models like neural networks or Gaussian processes. Other extensions could include multiple

categories of customers and rooms in the simulation logic, leading to the creation of multiple price elasticity models and to the necessity of optimally allocating the availability of rooms while dynamically changing the price.

Black-box algorithms can deal with the presence of noise by combining multiple solutions located in a restricted area of the search space, or by evaluating multiple times each solution. The implicit averaging effect of population-based schemes is effective under mild noise conditions, but in the presence of larger amounts of noise statistical analysis techniques are necessary in order to effectively deal with the presence of noise. The increase of the populations' size does not always lead to an improvement of the optimization performance, and additional evaluations of the samples should be preferred whenever the signal-to-noise ratio is low. Future work should extend the current results in order to consider additional black-box strategies and more benchmarks. Also, it would be worth investigating the performance of more global search policies, which iteratively compare configurations located farther away in Θ and search more locally in different parts of Θ only if sufficient empirical evidence to do so is observed. Approaches like CoRSO [119] or Bayesian Optimization [120] could be good choices.

Statistical analysis techniques can be integrated within heuristic search methods, in order to efficiently manage the optimization budget and manage the number of replications of each solution in comparisons. The adoption of CRN in simulations induces a variable amount ot positive correlation between the samples of different solutions to be compared. Therefore, it is reasonable to exploit this correlation in order to take a statistically significant decision as soon as possible. Considering simultaneously multiple stopping criteria based on Student's t paired test and Welch's t unpaired test significantly improves the efficiency of

the optimization. In particular, as the amount of the correlation among samples increases, OCBA significantly improves its performance with respect to its classic formulation. Moreover, the introduction of the IZ parameter in the hypothesis formulation leads to stopping criteria based on PGS guarantees, which require lower samples with respect to their PCS counterpart. The efficiency of the optimization significantly improves since more budget is available to explore the search space, and finding smarter ways to set dinamically the IZ parameter (possibly with respect to the locality of the search or the quality of the current best solution) might lead to further improvements.

Bibliography

[1] Roberto Battiti and Mauro Brunato. *The LION way. Machine Learning* plus *Intelligent Optimization*. LIONlab, University of Trento, Italy, 2017.

[2] Abhijit Gosavi. *Simulation-Based Optimization: Parametric Optimization Techniques and Reinforcement Learning*. Springer Publishing Company, Incorporated, 2nd edition, 2014.

[3] Dimitri P Bertsekas. *Reinforcement learning and optimal control*. Athena Scientific Belmont, 2019.

[4] Jerry Banks, John S. Carson II, Barry L. Nelson, and David M. Nicol. *Discrete-Event System Simulation, 5th New Internatinal Edition*. Pearson Education, 2010.

[5] Dimitri P. Bertsekas. *Dynamic programming and optimal control: Vol. 1*. Athena scientific Belmont, 2000.

[6] John Harling. Simulation techniques in operations research—a review. *Operations Research*, 6(3):307–319, 1958.

[7] Nicholas Metropolis, Arianna W Rosenbluth, Marshall N Rosenbluth, Augusta H Teller, and Edward Teller. Equation of state

calculations by fast computing machines. *The journal of chemical physics*, 21(6):1087–1092, 1953.

[8] Dirk P Kroese, Thomas Taimre, and Zdravko I Botev. *Handbook of monte carlo methods*, volume 706. John Wiley & Sons, 2013.

[9] Richard Bellman. Dynamic programming. *Science*, 153(3731):34–37, 1966.

[10] Chun-Hung Chen and Loo Hay Lee. *Stochastic simulation optimization: an optimal computing budget allocation*, volume 1. World scientific, 2011.

[11] Jie Xu, Edward Huang, Chun-Hung Chen, and Loo Hay Lee. Simulation optimization: A review and exploration in the new era of cloud computing and big data. *Asia-Pacific Journal of Operational Research*, 32(03):1550019, 2015.

[12] Jie Xu, Edward Huang, Liam Hsieh, Loo Hay Lee, Qing-Shan Jia, and Chun-Hung Chen. Simulation optimization in the era of industrial 4.0 and the industrial internet. *Journal of Simulation*, 10(4):310–320, 2016.

[13] Justin Boesel, Barry L. Nelson, and Seong-Hee Kim. Using ranking and selection to "clean up" after simulation optimization. *Operations Research*, 51:814–825, 2000.

[14] L. J. Hong and B. L. Nelson. A brief introduction to optimization via simulation. In *Proceedings of the Winter Simulation Conference, 2009.*, pages 75–85, Dec 2009.

[15] Seong-Hee Kim and Barry L. Nelson. Chapter 17 selecting the best system. In Shane G. Henderson and Barry L. Nelson, edi-

tors, *Simulation*, volume 13 of *Handbooks in Operations Research and Management Science*, pages 501 – 534. Elsevier, 2006.

[16] Jürgen Branke, Stephen E. Chick, and Christian Schmidt. Selecting a selection procedure. *Management Science*, 53(12):1916–1932, 2007.

[17] R. Pasupathy and S. Ghosh. Simulation optimization: A concise overview and implementation guide. In H. Topaloglu, editor, *TutORials in Operations Research*, chapter 7, pages 122–150. INFORMS, 2013.

[18] L. Jeff Hong, Barry L. Nelson, and Jie Xu. *Discrete Optimization via Simulation*, pages 9–44. Springer New York, 2015.

[19] Mohamed A Ahmed and Talal M Alkhamis. Simulation-based optimization using simulated annealing with ranking and selection. *Computers & Operations Research*, 29(4):387–402, 2002.

[20] Christian Schmidt, Jürgen Branke, and Stephen E Chick. Integrating techniques from statistical ranking into evolutionary algorithms. In *Workshops on Applications of Evolutionary Computation*, pages 752–763. Springer, 2006.

[21] Jürgen Branke, Stephan Meisel, and Christian Schmidt. Simulated annealing in the presence of noise. *Journal of Heuristics*, 14(6):627–654, 2008.

[22] Hui Xiao and Loo Hay Lee. Simulation optimization using genetic algorithms with optimal computing budget allocation. *SIMULATION*, 90(10):1146–1157, 2014.

[23] Si Zhang, Jie Xu, Loo Hay Lee, Ek Peng Chew, Wai Peng Wong, and Chun-Hung Chen. Optimal computing budget allocation for particle swarm optimization in stochastic optimization. *IEEE Transactions on Evolutionary Computation*, 21(2):206–219, 2017.

[24] Hans-Georg Beyer. Evolutionary algorithms in noisy environments: Theoretical issues and guidelines for practice. *Computer methods in applied mechanics and engineering*, 186(2-4):239–267, 2000.

[25] Dirk V Arnold. *Noisy optimization with evolution strategies*, volume 8. Springer Science & Business Media, 2012.

[26] J Michael Fitzpatrick and John J Grefenstette. Genetic algorithms in noisy environments. *Machine learning*, 3(2-3):101–120, 1988.

[27] Dirk V Arnold and H-G Beyer. Investigation of the (/spl mu/,/spl lambda/)-es in the presence of noise. In *Proceedings of the 2001 Congress on Evolutionary Computation (IEEE Cat. No. 01TH8546)*, volume 1, pages 332–339. IEEE, 2001.

[28] Jürgen Branke, Christian Schmidt, and Hartmut Schmeck. Efficient fitness estimation in noisy environments. In *Proceedings of the 3rd Annual Conference on Genetic and Evolutionary Computation*, pages 243–250. Morgan Kaufmann Publishers Inc., 2001.

[29] Yasuhito Sano and Hajime Kita. Optimization of noisy fitness functions by means of genetic algorithms using history of search with test of estimation. In *Proceedings of the 2002 Congress*

on Evolutionary Computation. CEC'02 (Cat. No. 02TH8600), volume 1, pages 360–365. IEEE, 2002.

[30] Jürgen Branke and Christian Schmidt. Selection in the presence of noise. In *Genetic and Evolutionary Computation Conference*, pages 766–777. Springer, 2003.

[31] Erick Cantú-Paz. Adaptive sampling for noisy problems. In *Genetic and Evolutionary Computation Conference*, pages 947–958. Springer, 2004.

[32] Sigrún Andradóttir. Chapter 20 an overview of simulation optimization via random search. In Shane G. Henderson and Barry L. Nelson, editors, *Simulation*, volume 13 of *Handbooks in Operations Research and Management Science*, pages 617 – 631. Elsevier, 2006.

[33] Sigurdur Olafsson. Chapter 21 metaheuristics. In Shane G. Henderson and Barry L. Nelson, editors, *Simulation*, volume 13 of *Handbooks in Operations Research and Management Science*, pages 633 – 654. Elsevier, 2006.

[34] M. C. Fu, F. W. Glover, and J. April. Simulation optimization: a review, new developments, and applications. In *Proceedings of the Winter Simulation Conference, 2005.*, pages 13 pp.–, Dec 2005.

[35] Nanjing Jian and Shane Henderson. An introduction to simulation optimization. *Proceedings of the 2015 Winter Simulation Conference L. Yilmaz, W. K. V. Chan, I. Moon, T. M. K. Roeder, C. Macal, and M. D. Rossetti, eds.*, pages 1780–1794, 2015.

[36] D. D. Linz, Z. B. Zabinsky, S. Kiatsupaibul, and R. L. Smith. A computational comparison of simulation optimization methods using single observations within a shrinking ball on noisy black-box functions with mixed integer and continuous domains. In *2017 Winter Simulation Conference (WSC)*, pages 2045–2056, Dec 2017.

[37] Zdravko Botev and Ad Ridder. Variance reduction. *Wiley StatsRef: Statistics Reference Online*, pages 1–6, 2014.

[38] Hongyu Ren, Shengjia Zhao, and Stefano Ermon. Adaptive antithetic sampling for variance reduction. In *International Conference on Machine Learning*, pages 5420–5428. PMLR, 2019.

[39] GM Clark. Use of common random numbers in comparing alternatives. In *1990 Winter Simulation Conference Proceedings*, pages 367–368. IEEE Computer Society, 1990.

[40] Robert E. Bechhofer. A single-sample multiple decision procedure for ranking means of normal populations with known variances. *Ann. Math. Statist.*, 25(1):16–39, 03 1954.

[41] Seong-Hee Kim and Barry L. Nelson. A fully sequential procedure for indifference-zone selection in simulation. *ACM Trans. Model. Comput. Simul.*, 11(3):251–273, 2001.

[42] Edward J. Dudewicz and Siddharta R. Dalal. Allocation of observations in ranking and selection with unequal variances. *Sankhya*, B:37:28–78, 01 1975.

[43] Barry L. Nelson, Julie Swann, David Goldsman, and Wheyming Song. Simple procedures for selecting the best simulated system

when the number of alternatives is large. *Operations Research*, 49(6):950–963, 2001.

[44] Peter I. Frazier. A fully sequential elimination procedure for indifference-zone ranking and selection with tight bounds on probability of correct selection. *Operations Research*, 62(4):926–942, 2014.

[45] Michael C Fu and Shane G Henderson. History of seeking better solutions, aka simulation optimization. In *2017 Winter Simulation Conference (WSC)*, pages 131–157. IEEE, 2017.

[46] Stanislav Ivanov. *Hotel revenue management: From theory to practice*. Zangador, 2014.

[47] Sheryl E Kimes. Yield management: a tool for capacity-considered service firms. *Journal of operations management*, 8(4):348–363, 1989.

[48] N. Aydin and S.I. Birbil. Decomposition methods for dynamic room allocation in hotel revenue management. *European Journal of Operational Research*, 271(1):179–192, 2018.

[49] Andrea Mariello, Manuel Dalcastagné, and Mauro Brunato. Hotelsimu: Simulation-based optimization for hotel dynamic pricing. In *International Conference on Learning and Intelligent Optimization*, pages 341–355. Springer, 2020.

[50] Ronald R Yager. Quantifier guided aggregation using owa operators. *International Journal of Intelligent Systems*, 11(1):49–73, 1996.

[51] Manuel Dalcastagné. Heuristic search strategies for noisy optimization. In *International Conference on Learning and Intelligent Optimization*, pages 356–370. Springer, 2020.

[52] Nikolaus Hansen. The cma evolution strategy: a comparing review. In *Towards a new evolutionary computation*, pages 75–102. Springer, 2006.

[53] N. Hansen, A. S. P. Niederberger, L. Guzzella, and P. Koumoutsakos. A method for handling uncertainty in evolutionary optimization with an application to feedback control of combustion. *IEEE Transactions on Evolutionary Computation*, 13(1):180–197, 2009.

[54] L. Jeff Hong, Weiwei Fan, and Jun Luo. Review on ranking and selection: A new perspective. *Frontiers of Engineering Management*, 2021.

[55] L. Jeff Hong and Barry L. Nelson. Selecting the best system when systems are revealed sequentially. *IIE Transactions*, 39(7):723–734, 2007.

[56] Zheng Xiang, Vincent P Magnini, and Daniel R Fesenmaier. Information technology and consumer behavior in travel and tourism: Insights from travel planning using the internet. *Journal of Retailing and Consumer Services*, 22:244–249, 2015.

[57] Gabriel R Bitran and Susana V Mondschein. An application of yield management to the hotel industry considering multiple day stays. *Operations research*, 43(3):427–443, 1995.

[58] Timothy Kevin Baker and David A Collier. A comparative revenue analysis of hotel yield management heuristics. *Decision Sciences*, 30(1):239–263, 1999.

[59] Tat Y Choi and Vincent Cho. Towards a knowledge discovery framework for yield management in the Hong Kong hotel industry. *International Journal of Hospitality Management*, 19(1):17–31, 2000.

[60] Dimitris Bertsimas and Ioana Popescu. Revenue management in a dynamic network environment. *Transportation science*, 37(3):257–277, 2003.

[61] Kin-Keung Lai and Wan-Lung Ng. A stochastic approach to hotel revenue optimization. *Computers & Operations Research*, 32(5):1059–1072, 2005.

[62] S Liu, Kin Keung Lai, and Shouyang Wang. Booking models for hotel revenue management considering multiple-day stays. *International Journal of Revenue Management*, 2:78–91, 02 2008.

[63] José Guadix, Pablo Cortés, Luis Onieva, and Jesús Muñuzuri. Technology revenue management system for customer groups in hotels. *Journal of Business Research*, 63(5):519–527, 2010.

[64] Anton J Kleywegt. An optimal control problem of dynamic pricing. *School of Industrial and Systems Engineering, Georgia Institute of Technology (2001)*, 2001.

[65] Heba Abdel Aziz, Mohamed Saleh, Mohamed H. Rasmy, and Hisham Elshishiny. Dynamic room pricing model for hotel

revenue management systems. *Egyptian Informatics Journal*, 12(3):177–183, 2011.

[66] Athanasius Zakhary, Amir F Atiya, Hisham El-Shishiny, and Neamat E Gayar. Forecasting hotel arrivals and occupancy using monte carlo simulation. *Journal of Revenue and Pricing Management*, 10(4):344–366, 2011.

[67] Abd El-Moniem Bayoumi, Mohamed Saleh, Amir F Atiya, and Heba Abdel Aziz. Dynamic pricing for hotel revenue management using price multipliers. *Journal of Revenue and Pricing Management*, 12(3):271–285, 2013.

[68] Basak Denizci Guillet and Ibrahim Mohammed. Revenue management research in hospitality and tourism: A critical review of current literature and suggestions for future research. *International Journal of Contemporary Hospitality Management*, 27(4):526–560, 2015.

[69] Aldric Vives, Marta Jacob, and Margarita Payeras. Revenue management and price optimization techniques in the hotel sector: A critical literature review. *Tourism Economics*, 24:720–752, 2018.

[70] Gabriel Bitran and René Caldentey. An overview of pricing models for revenue management. *Manufacturing & Service Operations Management*, 5(3):203–229, 2003.

[71] Jeffrey I McGill and Garrett J Van Ryzin. Revenue management: Research overview and prospects. *Transportation science*, 33(2):233–256, 1999.

[72] Kemal Subulan, Adil Baykasoğlu, Derya Eren Akyol, and Gokalp Yildiz. Metaheuristic-based simulation optimization approach to network revenue management with an improved self-adjusting bid-price function. *The Engineering Economist*, 62(1):3–32, 2017.

[73] Dimitris Bertsimas and Sanne De Boer. Simulation-based booking limits for airline revenue management. *Operations Research*, 53(1):90–106, 2005.

[74] Gonçalo Figueira and Bernardo Almada-Lobo. Hybrid simulation–optimization methods: A taxonomy and discussion. *Simulation Modelling Practice and Theory*, 46:118–134, 2014.

[75] Dusan Stefanovic, Nenad Stefanovic, and B Radenkovic. Supply network modelling and simulation methodology. *Simulation Modelling Practice and Theory*, 17(4):743–766, 2009.

[76] Pablo Grube, Felipe Núñez, and Aldo Cipriano. An event-driven simulator for multi-line metro systems and its application to santiago de chile metropolitan rail network. *Simulation Modelling Practice and Theory*, 19(1):393–405, 2011.

[77] Soraya Rana, L. Darrell Whitley, and Ronald Cogswell. Searching in the presence of noise. In *Parallel Problem Solving from Nature — PPSN IV*, pages 198–207, Berlin, Heidelberg, 1996. Springer Berlin Heidelberg.

[78] Volker Nissen and Jörn Propach. On the robustness of population-based versus point-based optimization in the presence of noise. *IEEE Transactions on Evolutionary Computation*, 2(3):107–119, 1998.

[79] Dirk V Arnold and Hans-Georg Beyer. A comparison of evolution strategies with other direct search methods in the presence of noise. *Computational Optimization and Applications*, 24(1):135–159, 2003.

[80] Yaochu Jin and J. Branke. Evolutionary optimization in uncertain environments-a survey. *IEEE Transactions on Evolutionary Computation*, 9(3):303–317, 2005.

[81] Nikolaus Hansen and Andreas Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary computation*, 9(2):159–195, 2001.

[82] Nikolaus Hansen. Benchmarking a bi-population cma-es on the bbob-2009 function testbed. In *Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference: Late Breaking Papers*, pages 2389–2396. ACM, 2009.

[83] S. Vock, S. Enz, and C. Cleophas. Genetic algorithms for calibrating airline revenue management simulations. In *Proceedings of the Winter Simulation Conference 2014*, pages 264–275, Dec 2014.

[84] Bernard L Welch. The generalization ofstudent's' problem when several different population variances are involved. *Biometrika*, 34(1/2):28–35, 1947.

[85] Ulrich Hammel and Thomas Bäck. Evolution strategies on noisy functions how to improve convergence properties. In *International Conference on Parallel Problem Solving from Nature*, pages 159–168. Springer, 1994.

[86] Nikolaus Hansen, André SP Niederberger, Lino Guzzella, and Petros Koumoutsakos. A method for handling uncertainty in evolutionary optimization with an application to feedback control of combustion. *IEEE Transactions on Evolutionary Computation*, 13(1):180–197, 2008.

[87] David E Goldberg, Kalyanmoy Deb, James H Clark, et al. Genetic algorithms, noise, and the sizing of populations. *COMPLEX SYSTEMS-CHAMPAIGN-*, 6:333–333, 1992.

[88] George Harik, Erick Cantú-Paz, David E Goldberg, and Brad L Miller. The gambler's ruin problem, genetic algorithms, and the sizing of populations. *Evolutionary Computation*, 7(3):231–253, 1999.

[89] Sigrún Andradóttir and Andrei A. Prudius. Adaptive random search for continuous simulation optimization. *Naval Research Logistics (NRL)*, 57(6):583–604, 2010.

[90] S. Kiatsupaibul, R. L. Smith, and Z. B. Zabinsky. Improving hit-and-run with single observations for continuous simulation optimization. In *2015 Winter Simulation Conference (WSC)*, pages 3569–3576, Dec 2015.

[91] Peter JM Van Laarhoven and Emile HL Aarts. Simulated annealing. In *Simulated annealing: Theory and applications*, pages 7–15. Springer, 1987.

[92] William D. Dupont and Walton D. Plummer. Power and sample size calculations: A review and computer program. *Controlled Clinical Trials*, 11(2):116 – 128, 1990.

[93] Mauro Brunato and Roberto Battiti. *RASH: A Self-adaptive Random Search Method*, pages 95–117. Springer Berlin Heidelberg, Berlin, Heidelberg, 2008.

[94] Hans-Georg Beyer and Dirk V Arnold. Qualms regarding the optimality of cumulative path length control in csa/cma-evolution strategies. *Evolutionary Computation*, 11(1):19–28, 2003.

[95] Giorgio Ausiello, Pierluigi Crescenzi, Giorgio Gambosi, Viggo Kann, Alberto Marchetti-Spaccamela, and Marco Protasi. *Complexity and approximation: Combinatorial optimization problems and their approximability properties*. Springer Science & Business Media, 2012.

[96] Satyajith Amaran, Nikolaos V. Sahinidis, Bikram Sharda, and Scott J. Bury. Simulation optimization: a review of algorithms and applications. *Annals of Operations Research*, 240(1):351–380, May 2016.

[97] J. Branke, S. E. Chick, and C. Schmidt. New developments in ranking and selection: an empirical comparison of the three main approaches. In *Proceedings of the Winter Simulation Conference, 2005.*, pages 10 pp.–, 2005.

[98] Michael C Fu, Jian-Qiang Hu, Chun-Hung Chen, and Xiaoping Xiong. Simulation allocation for determining the best design in the presence of correlated sampling. *INFORMS Journal on Computing*, 19(1):101–111, 2007.

[99] Yijie Peng, Chun-Hung Chen, Michael C Fu, and Jian-Qiang Hu. Efficient simulation resource sharing and allocation for selecting

the best. *IEEE Transactions on Automatic Control*, 58(4):1017–1023, 2012.

[100] Morris H DeGroot. *Optimal statistical decisions*, volume 82. John Wiley & Sons, 2005.

[101] Koichiro Inoue and Stephen E Chick. Comparison of bayesian and frequentist assessments of uncertainty for selecting the best system. In *1998 Winter Simulation Conference. Proceedings (Cat. No. 98CH36274)*, volume 1, pages 727–734. IEEE, 1998.

[102] B. L. Welch. The significance of the difference between two means when the population variances are unequal. *Biometrika*, 29(3/4):350–362, 1938.

[103] Yosef Rinott. On two-stage selection procedures and related probability-inequalities. *Communications in Statistics - Theory and Methods*, 7(8):799–811, 1978.

[104] Edward Paulson. A sequential procedure for selecting the population with the largest mean from k normal populations. *Ann. Math. Statist.*, 35(1):174–180, 03 1964.

[105] Juta Pichitlamken, Barry Nelson, and L. Hong. A sequential procedure for neighborhood selection-of-the-best in optimization via simulation. *European Journal of Operational Research*, 173:283–298, 01 2006.

[106] L. Hong. Fully sequential indifference-zone selection procedures with variance-dependent sampling. *Naval Research Logistics (NRL)*, 53:464–476, 08 2006.

[107] Seong-Hee Kim and B. L. Nelson. Recent advances in ranking and selection. In *2007 Winter Simulation Conference*, pages 162–172, Dec 2007.

[108] Vaclav Fabian. Note on anderson's sequential procedures with triangular boundary. *The Annals of Statistics*, 2(1):170–176, 1974.

[109] Mark Hartmann. An improvement on paulson's procedure for selecting the poprlation with the largest mean from k normal populations with a common unknown variance. *Sequential Analysis*, 10(1-2):1–16, 1991.

[110] Show-Li Jan and Gwowen Shieh. Optimal sample sizes for welch's test under various allocation and cost considerations. *Behavior research methods*, 43(4):1014–1022, 2011.

[111] Russell V Lenth. Algorithm as 243: cumulative distribution function of the non-central t distribution. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 38(1):185–189, 1989.

[112] David M Erceg-Hurn and Vikki M Mirosevich. Modern robust statistical methods: an easy way to maximize the accuracy and power of your research. *American Psychologist*, 63(7):591, 2008.

[113] Kim F Nimon. Statistical assumptions of substantive analyses across the general linear model: a mini-review. *Frontiers in psychology*, 3:322, 2012.

[114] Marie Delacre, Daniel Lakens, and Christophe Leys. Why psychologists should by default use welch's t-test instead of student's t-test. *International Review of Social Psychology*, 30(1), 2017.

[115] Daniël Lakens. Calculating and reporting effect sizes to facilitate cumulative science: a practical primer for t-tests and anovas. *Frontiers in psychology*, 4:863, 2013.

[116] Jacob Cohen. *Statistical power analysis for the behavioral sciences*. Academic press, 2013.

[117] Marcin Molga and Czesław Smutnicki. Test functions for optimization needs. *Test functions for optimization needs*, 101:48, 2005.

[118] Momin Jamil and Xin-She Yang. A literature survey of benchmark functions for global optimisation problems. *International Journal of Mathematical Modelling and Numerical Optimisation*, 4(2):150–194, 2013.

[119] Mauro Brunato and Roberto Battiti. Corso (collaborative reactive search optimization): blending combinatorial and continuous local search. *Informatica*, 27(2):299–322, 2016.

[120] Bobak Shahriari, Kevin Swersky, Ziyu Wang, Ryan P Adams, and Nando De Freitas. Taking the human out of the loop: A review of bayesian optimization. *Proceedings of the IEEE*, 104(1):148–175, 2015.