Proceedings of the 2024 Winter Simulation Conference
H. Lam, E. Azar, D. Batur, S. Gao, W. Xie, S. R. Hunter, and M. D. Rossetti, eds.

RELIABLE ONLINE DECISION MAKING WITH COVARIATES

Heng Luo[1], Zhiyang Liang[1], and L. Jeff Hong[1,2]

[1]School of Data Science, Fudan University, Shanghai, CHINA
[2]School of Management, Fudan University, Shanghai, CHINA

ABSTRACT

In online decision makings, the challenge often lies in finding the optimal solution quickly based on the covariates observed in real time. In this paper we propose to use the framework of offline simulation and online application to develop algorithms that are capable of solving online simulation optimization problems. In the offline stage, the algorithms solve the optimization problems many times based on different values of the covariates and build predictive models of the optimal solution with respect to the covariates. In the online stage, once the covariates are observed, the optimal solution may be quickly determined by the predictive model. We focus on online strongly convex simulation optimization problems and propose to use different algorithms to construct the predictive models. We derive the rate of convergence of the optimality gaps of the proposed algorithms, and develop a finite-sample statistical measure of the optimality gap when these algorithms are used.

1 INTRODUCTION

Simulation optimization (SO) refers to a class of optimization problems (Fan et al. 2024), where the objective functions and constraints do not have closed analytical forms, and the objective function values can only be evaluated by running simulation experiments based on a simulation model. It is widely recognized that simulation experiments often require a significant amount of time. Therefore, simulation optimization is often applied to offline decision making problems where there is enough time to execute the optimization algorithms. In recent years, there is a growing need to use simulation models to support real-time decision makings (called online simulation optimization), where a part of the input parameters (called covariates) to the simulation model are unknown in advance and are only observed in real time. For instance, in the area of healthcare, medical treatments need to be adjusted in real-time based on patients' age, gender, baseline disease severity, and other information in hopes of achieving better personalized treatment outcome (Bertsimas et al. 2019). In urban transportation, the optimal driving decisions depend on various covariates, such as traffic flow, road conditions, and weather. Making these decisions in real-time can significantly improve efficiency, reduce congestion, and enhance the overall system performance (Katrakazas et al. 2015). In the online revenue management problem, consumers arrive sequentially and request a subset of goods, each offering a certain bid price for their demand, the seller must make an irrevocable decision in real-time to maximize revenue (Agrawal et al. 2014). All of the above examples can be considered as online simulation optimization problems. However, in practice, the available time is typically not enough to execute a simulation optimization algorithm once the values of the covariates are observed. This is the central challenge of online simulation optimization.

The use of slow simulation to support fast applications has been considered in the simulation literature. Hong and Jiang (2019) proposed an offline-simulation-online-application (OSOA) framework to address this issue and the key is to differentiate the offline stage and the online stage. In the offline stage, there is significant amount of time, but the exact values of the covariates are unknown. One can conduct simulation studies at a chosen set of points from the covariate space and build a predictive model using machine learning techniques. Then, in the online stage, once the values of the covariates are observed, one may use

the predictive model to guide decision makings without running additional simulation experiments. Based on the framework, Jiang et al. (2020) propose to build a logistic regression model using data generated from offline simulation experiments, and leverage this model to predict portfolio risk measures and classify risk levels at any given time and given market conditions. Shen et al. (2021) propose to learn the mean performance functions of all alternatives through offline simulation experiments and use these functions to compare alternatives and select the best after the values of covariates are observed, and they call the problem "ranking and selection with covariates" and apply their algorithms to personalized therapy selections. In this paper our goal is to leverage the OSOA framework and solve the online simulation optimization problems through offline simulation experiments and machine learning.

The use of offline computation to support online decision makings is not new to optimization. Recently, it has been explored by a sequence of papers in the optimization literature. There are in general two different types of approaches. The first approach learns the objective function with respect to both the covariates and the decision variables in the offline stage and solve the optimization problem with the learned objective function in the online stage after the covariates are observed. The second approach learns the optimal solution with respect to the covariates. Hannah et al. (2010), Hanasusanto and Kuhn (2013), Bertsimas and McCord (2019) and Gao et al. (2019) all consider the first approach, with the first three using local kernel methods to fit the objective function while the last using stochastic kriging. Similarly, Elmachtoub and Grigas (2022) employ machine learning models to predict certain parameters or components of the objective function based on observed covariates, the Smart Predict-then-Optimize (SPO) framework introduced in their work is a novel approach that integrates predictive modeling directly with the optimization process. Additionally, Kannan et al. (2022) discuss an approach that incorporates covariate information into the sample average approximation method for stochastic optimization problems, this method uses historical data, including covariates, to estimate the distribution of uncertain parameters and to learn a predictive model for the objective function. But Ban and Rudin (2019), Amos et al. (2023) and Keslin et al. (2022) adopt the second approach in their respective studies. Ban and Rudin (2019) focus on addressing the newsvendor problem, where optimal inventory decisions are made under uncertain future demand. They utilize machine learning techniques to predict the optimal ordering quantity directly from covariates. Their research demonstrates that their approach effectively reduces the optimality gap to zero, showcasing its efficacy in enhancing decision-making under uncertainty in inventory management. Amos et al. (2023) present a method for training neural networks or other machine learning models to quickly approximate solutions for a class of optimization problems, significantly reducing the computational time and resources required for each instance. However, the approach lacks rigorous theoretical guarantees in some cases, particularly when the model's prediction error is high or when the problem conditions vary significantly. Keslin et al. (2022) advocate for a classification-based approach in their work. This method involves solving ranking-and-selection (R&S) problems offline across various covariate values and then treating real-time decision-making as a classification task. By leveraging classification techniques, this approach provides a probabilistic guarantee on the selected system at each design point, independent of the ultimate choice of classification metamodel. Bertsimas and Kallus (2020) and Bertsimas and Koduri (2022) explore both approaches, they consider a more complex scenario where the distribution of the random parameters is conditional on the realization of the covariate. Nevertheless, the essence remains the prediction of the objective function and the optimal solution function.

In this paper we are interested in the second approach and focus our research on the properties of the optimality gap between the predicted and the true optimal solutions. In the literature, Bertsimas and Koduri (2022) have demonstrated that, for general convex optimization problems, the optimality gap converges to zero as the number of design points goes to infinity. However, the rate of such convergence has not been studied. We want to emphasize that the rate of convergence is important, because it creates a unified framework to compare different algorithms and it provides insights on how to select algorithms for different types of problems. Moreover, while both convergence and rate of convergence are important, they are

nonetheless asymptotic properties. When these algorithms are used in practice, the number of design points is fixed and there is a need to provide a measure of optimality gap in this context.

In this paper we first provide the detailed setup of online simulation optimization problems and the algorithms for building predictors of the optimal solutions. In particular, we propose to use nearest neighbor, k nearest neighbors and kernel ridge regression algorithms to fit the predictors. We then analyze the rate of convergence for these algorithms and we find that the algorithms are particularly efficient for online strongly convex simulation optimization problems where the dimension of the covariates is small while the dimension of the decision variables is large. Furthermore, motivated by the conformal prediction idea of Keslin et al. (2022), by taking advantage of the randomness in the covariates, we develop a finite-sample measure of the optimality gap for practical implementation of these algorithms based on the concept of prediction interval. The numerical results show that the algorithms and the finite-sample measure of the optimality gap work well.

The rest of the paper is organized as follows. The detailed problem settings and the algorithms are introduced in Section 2. The rate of convergence and the prediction interval of the optimality gaps of the proposed algorithms are studied in Section 3. The numerical experiments are reported in Section 4, followed by conclusions in Section 5.

2 PROBLEM SETTINGS AND ALGORITHMS

The ultimate goal is to address the challenge of solving stochastic optimization problems with covariates. The problem can be formulated as:

$$\theta^*(x) = \arg\min_{\theta\in\Theta} \mathbb{E}[f(\theta,x,\xi)],$$

where θ is the decision variable, x denotes the covariates, ξ is a random variable representing uncertainty, and $\mathbb{E}[f(\theta,x,\xi)]$ is the expected value of the objective function $f(\theta,x,\xi)$. Here, $\Theta \subset \mathbb{R}^{d_\theta}$ and $\mathscr{X} \subset \mathbb{R}^{d_x}$ denote the domains of θ and x, respectively. Solving this problem to find the exact optimal solution is challenging due to the inherent randomness and computational demands, we leave the problem for a future paper, and consider in this paper only the deterministic version of this problem, formulated as:

$$\theta^*(x) = \arg\min_{\theta\in\Theta} f(\theta,x), \tag{1}$$

where $f(\theta,x)$ denotes the objective function, which may be evaluated by running time-consuming simulation experiments when the decision variables θ and the covariates x are given.

We assume that the objective function $f(\theta,x)$ is strongly convex when the covariates x are given, so it may be solved exactly when there is sufficient time. We denote the unique optimal solution as $\theta^*(x)$ to emphasize its dependence on the covariates x. In the offline stage, we first use the space-gridding method to choose a set of design points in the covariates domain $\mathscr{X}$, and denote them by $\{x_1,x_2,\ldots,x_n\}$. Then, for every design point x_i, we solve the corresponding optimization problem to find the unique optimal solution $\theta^*(x_i)$ for all $i = 1,2,\ldots,n$. Given the data pairs $\{x_i,\theta^*(x_i)\}_{i=1}^n$, we propose to build a predictive model $\hat{\theta}^*(x)$, which is a function from $\mathbb{R}^{d_x}$ to $\mathbb{R}^{d_\theta}$, or more precisely, from $\mathscr{X}$ to Θ. Then, in the online stage, when the covariates are observed, say $x = x^{new}$, we can immediately predict the optimal solution $\hat{\theta}^*(x^{new})$ based on the predictive model and use it to support real-time decision makings.

Different from our proposed method, another approach to solve the online simulation optimization problem (1) is to build a predictive model of $f(\theta,x)$. This approach was used by Hannah et al. (2010) and Bertsimas and McCord (2019). In the offine stage, it uses a set of design points $\{(\theta_i,x_i),f(\theta_i,x_i)\}_{i=1}^n$ to build a predictive model of the objective function, denoted by $\tilde{f}(\theta,x)$, which is a function from $\mathbb{R}^{d_x+d_\theta}$ to $\mathbb{R}$, or more precisely, from $\mathscr{X}\times\Theta$ to $\mathbb{R}$. Then, in the online stage, when the covariates are observed, say $x = x^{new}$, it solves the optimization problem $\tilde{\theta}^*(x^{new}) = \arg\min_{\theta\in\Theta}\tilde{f}(\theta,x^{new})$. Notice that $\tilde{f}(\theta,x^{new})$ is a known function. Therefore, it can be solved quickly without running additional simulation experiments.

While this approach is intuitive, it differs from our proposed method in two ways. First, it fits a predictive model from $\mathbb{R}^{d_x+d_\theta}$ to $\mathbb{R}$ while ours fit a predictive model from $\mathbb{R}^{d_x}$ to $\mathbb{R}^{d_\theta}$. It is an important objective of this paper to understand the differences between these two predictive models. Second, when building the predictive model $\tilde{f}(\theta,x)$, it is often difficult to incorporate the structural property that $f(\theta,x)$ is strongly convex in θ when x is fixed. Therefore, it typically ignores such property, while ours explicitly take advantage of this property to solve the optimization problems in the offline stage.

In the rest of this section, we will present a variety of methods to fit the predictive model $\hat{\theta}^*(x)$ and propose the overall algorithm to solve the online simulation optimization problem (1). We will analyze the algorithm's asymptotic behaviors in Section 3.

2.1 K Nearest Neighbors and Nearest Neighbor

First, we propose a method to build the predictive model $\hat{\theta}^*(x)$ using the k nearest neighbors (KNN) algorithm (Devroye 1978), which is a famous yet very simple nonparametric surface fitting algorithm. The predict model is as follows:

$$\hat{\theta}^*_{knn}(x) = \frac{1}{k}\sum_{l=1}^{k} \theta^*\left(x_l^{nei}\right),$$

where $\{x_1^{nei}, x_2^{nei}, \ldots, x_k^{nei}\}$ denote the k nearest neighbors of the covariates $x \in \mathscr{X}$ from the design points $\{x_1, x_2, \ldots, x_n\}$. Notice that this predictive model leverages the similarities among the design points and provides a quick and often reasonably accurate approximation of the optimal solution. It is particularly useful for online simulation optimization problems, where directly solving the simulation optimization problem in the online stage may be computationally prohibitive.

The nearest neighbor (NN) predictor is a special case of the KNN predictor with $k = 1$, i.e.,

$$\hat{\theta}^*_{nn}(x) = \theta^*\left(x^{nei}\right).$$

The NN predictor is particularly interesting, because there is no randomness in the response $\theta^*(x)$. Therefore, it is not necessary to balance the trade-off between bias and variance that is typical for KNN algorithms when selecting k, and it may be beneficial to set $k = 1$.

2.2 Kernel Ridge Regression

Kernel ridge regression (KRR) is a machine learning algorithm that combines ridge regression with the kernel trick. It is particularly useful to handle non-linear relationships between the independent variables and the dependent variable. Now we have data $\{x_i, \theta^*(x_i)\}_{i=1}^n$ sampled by a space-gridding method in offline stage. For a new covariate x, the estimate of $\theta^*(x)$ produced by KRR is

$$\hat{\theta}^*_{krr}(x) = \mathbf{k}_\Phi^{\mathrm{T}}(x)(K_\Phi + \lambda I)^{-1}\theta(\mathbf{x}),$$

where $\mathbf{k}_\Phi(x) = (\Phi(x-x_1), \Phi(x-x_2), \ldots, \Phi(x-x_n))^{\mathrm{T}}$, $K_\Phi = (\Phi(x_j - x_k))_{jk}$, $\theta(\mathbf{x}) = (\theta^*(x_1), \theta^*(x_2), \ldots, \theta^*(x_n))^{\mathrm{T}}$ and $\lambda \geq 0$ is a hyperparameter used to improve stability.

In this work, Matérn-type kernels are used. They have the form

$$\Phi(l) = \frac{1}{\Gamma(\nu)2^{\nu-1}}(2\sqrt{\nu}\rho\|l\|)^\nu K_\nu(2\sqrt{\nu}\rho\|l\|).$$

From Tuo and Wu (2016), Matérn-type kernels have a spectral density

$$f_\Phi(\omega) = \pi^{-d/2}\frac{\Gamma(\nu + d/2)}{\Gamma(\nu)}(4\nu\rho^2)^\nu(4\nu\rho^2 + \|\omega\|^2)^{-(\nu+d/2)},$$

where $\rho, \nu > 0$, K_ν is the modified Bessel function of the second kind, l is the input, d is the dimension of space, and $\|\cdot\|$ denotes the Euclidean distance.

2.3 Algorithm Procedure

After introducing methods for predicting optimal solution, we now introduce our algorithm for online simulation optimization. The algorithm is designed to undergo a two-stage process of training and prediction, where the first stage is offline (denoted as Algorithm 1) and the second stage is online (denoted as Algorithm 2). In the algorithm, we will use the concept of the optimality gap, which is defined as

$$\Delta(x) = f(\hat{\theta}^*(x), x) - f(\theta^*(x), x),$$

where $\theta^*(x)$ is the true optimal solution that is unknown to us.

Algorithm 1 Offline Simulation (Build a predictor by using three algorithms).

Input: Number of offline covariates points n and $m \geqslant 100$, hyperparameters k (KNN) or λ(KRR).
Specify the n covariates points: Use a space-gridding method to find n covariates points $\{x_1, x_2, \ldots, x_n\}$ in $\mathscr{X}$.
Specify the m covariates points: Generate m covariates points $\{x'_1, x'_2, \ldots, x'_m\}$ from its distribution (or from the historical data).
First stage:
for $i = 1 : n$ **do**
Call an algorithm to solve $\theta^*(x_i) = \arg\min_{\theta \in \Theta} f(\theta, x_i)$, get the optimal solution $\theta^*(x_i)$.
end for
Second stage:
for $j = 1 : m$ **do**
Call an algorithm to solve $\theta^*(x'_j) = \arg\min_{\theta \in \Theta} f(\theta, x'_j)$.
Call one of the following algorithms to produce an estimate $\hat{\theta}^*(x'_j)$.
NN model: Find x'_j's one nearest neighbor x^{nei} in $\{x_1, x_2, \ldots, x_n\}$. Let $\hat{\theta}^*(x'_j) = \theta^*\left(x^{nei}\right)$.
Or KNN model: Find x'_j's k nearest neighbors $\{x_1^{nei}, x_2^{nei}, \ldots, x_k^{nei}\}$ in $\{x_1, x_2, \ldots, x_n\}$. Let $\hat{\theta}^*(x'_j) = \frac{1}{k}\sum_{l=1}^{k} \theta^*\left(x_l^{nei}\right)$.
Or KRR model: Let $\hat{\theta}^*(x'_j) = \mathbf{k}_{\Phi}^{\mathrm{T}}(x'_j)(\mathbf{K} + \lambda \mathbf{I})^{-1}\theta(\mathbf{x})$.
Get the estimated gap $\Delta(x'_j) = f(\hat{\theta}^*(x'_j), x'_j) - f(\theta^*(x'_j), x'_j)$.
end for
Output: $\Delta(x'_j),\ j = 1, 2, \ldots, m$.

Algorithm 2 Online Application (Predict the optimal solution).

Input: Significance level α, $\Delta(x'_j),\ j = 1, 2, \ldots, m$, from Algorithm 1. The covariates point x we want to optimize.
Call the predictor to predict $\hat{\theta}^*(x)$. Use $\hat{\theta}^*(x)$ to estimate $\theta^*(x)$, which can guarantee

$$\Pr\{f(\hat{\theta}^*(x), x) - f(\theta^*(x), x) \leqslant \Delta_{1-\alpha}\} \geqslant 1 - \alpha,$$

where the assessment $\Delta_{1-\alpha} = \Delta_{(j^*)}$, the j^* order statistic of $\{\Delta(x'_1), \Delta(x'_2), \ldots, \Delta(x'_m)\}$, where is chosen as prescribed by Section 3.2.
Output: $\hat{\theta}^*(x)$, $\Delta_{1-\alpha}$.

3 RATE OF CONVERGENCE AND PREDICTION INTERVAL OF OPTIMALITY GAP

To evaluate the performance of online simulation optimization algorithms, a natural performance measure is the optimality gap $\Delta(x) = f(\hat{\theta}^*(x), x) - f(\theta^*(x), x)$, which measures the difference between the predicted

optimal solution and the true optimal solution in terms of the objective values. In Section 3.1 we analyze the rate of the optimality gap converging to zero, and in Section 3.2 we show how to build a prediction interval of the optimality gap that quantifies how good the predicted solution may be.

3.1 Rate of Convergence

In the context of predicting optimal solution, Bertsimas and Koduri (2022) prove that the optimality gap converges to zero, but they do not establish the rate of convergence. However, its rate of convergence is important, as it provides insights into the accuracy and the applicability of the algorithm.

To establish the rate of convergence of the optimality gap, we need the following conditions.

Condition 1 The covariates domain $\mathscr{X} \subset \mathbb{R}^{d_x}$ is a compact and convex set.

Condition 2 For any $x \in \mathscr{X}$, the objective function $f(\theta,x)$ is strongly convex in θ, which means for some $\mu > 0$, the function $f(\theta,x) - \frac{\mu}{2}\|\theta\|^2$ is convex in θ. Furthermore, assume f is q times continuous differentiable in θ for some $q \geqslant 2$.

Condition 3 The optimal solution $\theta^*(x)$ is obtained in the interior of Θ with $\frac{\partial}{\partial\theta} f(\theta,x) \mid_{\theta=\theta^*(x)} = 0$.

Based on the conditions, we have the following lemma that characterizes the smoothness of the optimal solution with respect to the covariates.

Lemma 1. *If conditions 1-3 hold, $\theta^*(x)$ is a $q-1$ times continuous differentiable function of x.*

Proof. Firstly, $\theta^*(x)$ is unique for any fixed x by condition 1, which implies $\theta^*(x)$ is a function of x. Secondly, by condition 2, $\frac{\partial}{\partial\theta} f(\theta,x)$ is $q-1$ times continuous differentiable. Also, with condition 1 $\frac{\partial^2}{\partial\theta^2} f(\theta,x) \mid_{\theta=\theta^*(x)}$ is invertible. Then by the implicit function theorem (Krantz and Parks 2002), $\theta^*(x)$ that satisfies condition 3 is a $q-1$ times continuous differentiable function in x. □

To establish the rate of convergence of the optimality gap, we also need to characterize how the design points $\{x_1, x_2, \ldots, x_n\}$ cover the design space $\mathscr{X}$. To that end, we introduce the concept of fill distance, which is defined as follows.

Definition 1 The fill distance of design points $X_n = \{x_1, x_2, \ldots, x_n\} \subseteq \mathscr{X}$ is

$$h_{X_n,\mathscr{X}} = \sup_{x\in\mathscr{X}} \inf_{x_j\in X_n} \left\|x - x_j\right\|_2.$$

The optimality gaps of our algorithms are highly related to the fill distance of the design. In this work, we use the uniform grid design which, according to (Wendland 2004), has a fill distance $h_{X_n,\mathscr{X}} \asymp n^{-\frac{1}{d_x}}$. Therefore, there exists some constant $c_0 > 0$ such that $h_{X_n,\mathscr{X}} \leqslant c_0 n^{-\frac{1}{d_x}}$.

Now we are ready to state and prove the following theorems on the rate of convergence of the optimality gaps with different surface fitting algorithms, including the NN, KNN and KRR algorithms.

Theorem 1 Under Conditions 1-3, for any fixed $x \in \mathscr{X}$, the optimality gap produced by the NN algorithm satisfies $\Delta(x) = \mathrm{O}(d_\theta^2 n^{-\frac{2}{d_x}})$.

Proof. Let $x^{nei} \in \{x_1, x_2, \ldots, x_n\}$ be the nearest neighbor of the fixed covariate x. By the nearest neighbor algorithm, the predicted optimal solution $\hat{\theta}^*(x)$ is $\theta^*(x^{nei})$. Furthermore, we know that $\|x - x^{nei}\|_2 = \min_{x^{nei}\in X_n} \|x - x^{nei}\|_2 \leqslant h_{X_n,\mathscr{X}} \leqslant c_0 n^{-\frac{1}{d_x}}$. Then, by Taylor's expansion when x is fixed,

$$\begin{aligned}
\Delta(x) &= f\left(\theta^*(x^{nei}),x\right) - f\left(\theta^*(x),x\right) \\
&= \frac{\partial}{\partial\theta} f(\theta^*(x),x)^\top (\theta^*(x) - \theta^*(x^{nei})) + \frac{1}{2}\left(\theta^*(x) - \theta^*(x^{nei})\right)^\top \frac{\partial^2}{\partial\theta^2} f\left(\theta^*(x),x\right)\left(\theta^*(x) - \theta^*(x^{nei})\right) \\
&\quad + o\left(\left\|\theta^*(x) - \hat{\theta}^*(x^{nei})\right\|_2^2\right),
\end{aligned}$$

from Conditions 2 and 3, $\frac{\partial}{\partial\theta}f(\theta^*(x),x)=0$, and $\frac{\partial^2}{\partial\theta^2}f(\theta^*(x),x)$ is non-zero and finite, so there exists some constant $c_1>0$, such that

$$\Delta(x)\leqslant c_1\|\theta^*(x)-\theta^*(x^{nei})\|_2^2. \tag{2}$$

We consider each dimension of $\theta^*(x)$. Let $\theta^*(x)=(\theta_1^*(x),\theta_2^*(x),\ldots,\theta_{d_\theta}^*(x))^{\mathrm{T}}$, thus

$$\left\|\theta^*(x)-\theta^*(x^{nei})\right\|_2\leqslant\sum_{i=1}^{d_\theta}\left\|\theta_i^*(x)-\theta_i^*(x^{nei})\right\|_2.$$

From lemma 1 we can know $\theta_i^*(x)$ is at least 1st-order continuous differentiable in a compact set $\mathscr{X}$, which implies $\theta_i^*(x)$ is Lipschitz, there exists some $c_2>0$,

$$\left\|\theta^*(x)-\theta^*(x^{nei})\right\|_2\leqslant c_2d_\theta\|x-x^{nei}\|_2\leqslant c_2c_0d_\theta n^{-\frac{1}{d_x}}. \tag{3}$$

Combined with inequality (2), the optimality gap satisfies

$$\Delta(x)\leqslant c_1c_2^2c_0^2d_\theta^2n^{-\frac{2}{d_x}},$$

which completes the proof. □

Theorem 2 Under Conditions 1-3, the optimality gap produced by the KNN algorithm satisfies $\Delta(x)=\mathrm{O}(k^2d_\theta^2n^{-\frac{2}{d_x}})$.

Proof. Find x's k nearest neighbors $X^{nei}=\{x_1^{nei},x_2^{nei},\ldots,x_k^{nei}\}\subseteq\{x_1,x_2,\ldots,x_n\}$. By the KNN algorithm, the predicted optimal solution $\hat{\theta}^*(x)$ is $\hat{\theta}_{knn}^*(x)=\frac{1}{k}\sum_{l=1}^k\theta^*\left(x_l^{nei}\right)$.

Let $x_w=\arg\max_{x^{nei}\in X^{nei}}\|\theta^*(x)-\theta^*(x^{nei})\|_2$, by the definition of fill distance,

$$\|x-x_w\|_2\leqslant\max_{x^{nei}\in X^{nei}}\|x-x^{nei}\|_2\leqslant kh_{X_n,\mathscr{X}}\leqslant c_0kn^{-\frac{1}{d_x}}.$$

Thus, by Equation (3),

$$\begin{aligned}\left\|\theta^*(x)-\hat{\theta}_{knn}^*(x)\right\|_2&=\left\|\theta^*(x)-\frac{1}{k}\sum_{l=1}^k\theta^*\left(x_l^{nei}\right)\right\|_2\\&\leqslant\frac{1}{k}\sum_{l=1}^k\left\|\theta^*(x)-\theta^*(x_l^{nei})\right\|_2\\&\leqslant\|\theta^*(x)-\theta^*(x_w)\|_2\leqslant c_2c_0kd_\theta n^{-\frac{1}{d_x}}.\end{aligned}$$

In a similar way to (2), we have

$$\Delta(x)\leqslant c_1c_2^2c_0^2k^2d_\theta^2n^{-\frac{2}{d_x}}.$$

This shows that $\Delta(x)=\mathrm{O}(k^2d_\theta^2n^{-\frac{2}{d_x}})$. □

To properly illustrate the rate of convergence of the optimality gap of the KRR algorithm, one additional condition is needed.

Condition 4 The smoothness of the kernel satisfies $0 < \nu \leqslant q-1$. And there exist constants $c_4 \geqslant c_3 > 0$ such that, for all $\omega \in \mathbb{R}^d$,

$$c_3(1+\|\omega\|^2)^{-(\nu+d/2)} \leqslant f_\Phi(\omega) \leqslant c_4(1+\|\omega\|^2)^{-(\nu+d/2)}.$$

For ease of analysis, we assume the smoothness of the Matérn-type kernel is smaller than the interpolated function. In practical scenarios, the value of q is typically quite large, making this assumption reasonably robust.

Theorem 3 Under Conditions 1-4, the optimality gap produced by the KRR algorithm satisfies $\Delta(x) = \mathrm{O}(d_\theta^2 n^{-\frac{2\nu}{d_x}})$ for $\lambda \leqslant h_{X_n,\mathscr{X}}^{2\nu}$.

Proof. By the KRR algorithm, the predicted optimal solution $\hat{\theta}^*(x)$ is

$$\hat{\theta}^*_{krr}(x) = \mathbf{k}_\Phi^{\mathrm{T}}(x)(K_\Phi + \lambda I)^{-1}\theta(\mathbf{x}),$$

where $\mathbf{k}_\Phi(x) = (\Phi(x-x_1), \Phi(x-x_2), \ldots, \Phi(x-x_n))^{\mathrm{T}}$, $K_\Phi = (\Phi(x_j - x_k))_{jk}$, $\theta(\mathbf{x}) = (\theta^*(x_1), \theta^*(x_2), \ldots, \theta^*(x_n))^{\mathrm{T}}$ and λ is a tuning parameter. Kernel Φ satisfies condition 4.

Also consider each dimension of $\hat{\theta}^*_{krr}(x)$. Let $\hat{\theta}^*_{krr}(x) = (\hat{\theta}^*_1(x), \hat{\theta}^*_2(x), \ldots, \hat{\theta}^*_{d_\theta}(x))^{\mathrm{T}}$. A simplified result comes from Wendland and Rieger (2005) shows that for a finite constant $c_5 > 0$

$$\left\|\theta^*(x) - \hat{\theta}^*_{krr}(x)\right\|_2 \leqslant \sum_{i=1}^{d_\theta} \left\|\theta^*_i(x) - \hat{\theta}^*_i(x)\right\|_2 \leqslant d_\theta c_5 h^{\nu}_{X_n,\mathscr{X}} \leqslant c_5 c_0 d_\theta n^{-\frac{\nu}{d_x}},$$

if $\lambda \leqslant h^{2\nu}_{X_n,\mathscr{X}}$. In the same way to (2), the optimality gap satisfies

$$\Delta(x) \leqslant c_1 c_5^2 c_0^2 d_\theta^2 n^{-\frac{2\nu}{d_x}},$$

which leads to the conclusion $\Delta(x) = \mathrm{O}(d_\theta^2 n^{-\frac{2\nu}{d_x}})$. □

Remark 1 In Condition 4 the smoothness of the kernel ν is assumed to be equal or less than $q-1$. When the smoothness of the optimal solution $\theta^*(x)$ is not exactly known, one general choice for ν is $\nu = 1$, which makes the rate of convergence at least the same as the NN algorithm. If more information about q is provided, ν may be set a larger value to achieve a faster rate of convergence.

The rate of convergence results indicate that the optimality gap decreases exponentially with respect to the number of offline design points n. It is worth noting that compared to d_θ, d_x has a greater impact on the rate of convergence. This implies that the proposed algorithms are particularly efficient for online simulation optimization problems where the dimension of the covariates d_x is small while the dimension of the decision variables d_θ is large.

3.2 Prediction Interval

While the rate of convergence clearly characterizes the asymptotic behaviors of the optimality gaps under different algorithms as the number of design points goes to infinity, it does not provide a measure of the optimality gap when the algorithms are used in practice with an inevitable finite number of design points. To address this issue, a straight-forward approach is to provide a $1-\alpha$ prediction interval of the optimality gap in the form of $[0, \Delta_{1-\alpha}]$ such that

$$\Pr\{\Delta(x) \leqslant \Delta_{1-\alpha}\} \geqslant 1-\alpha. \tag{4}$$

Notice that Equation (4) is difficult to achieve without further knowledge of the objective function, if we want it to hold for any fixed $x \in \mathscr{X}$. To solve the problem, motivated by the conformal prediction idea of Keslin

et al. (2022), we take advantage of the randomness in the covariate x and treat it as a random vector. Suppose that we have an independent and identically distributed (i.i.d.) sample $\{x'_1, x'_2, \ldots, x'_m\}$ from the covariate distribution. Notice that the sample may be drawn from a simulation model or may be from the historical observations. Then, in the offline stage, we can use predictor $\hat{\theta}^*(x)$ developed by the algorithms proposed in Section 2 to predict their corresponding optimal solutions $\{\hat{\theta}^*(x'_1), \hat{\theta}^*(x'_2), \ldots, \hat{\theta}^*(x'_m)\}$. But at the same time, we can also solve the optimization problems to find the true optimal solutions $\{\theta^*(x'_1), \theta^*(x'_2), \ldots, \theta^*(x'_m)\}$. Then, we obtain a sample of the optimality gap $\{\Delta(x'_1), \Delta(x'_2), \ldots, \Delta(x'_m)\}$. Notice that this may be done in the offline stage, as there is typically enough time to run these experiments.

It is interesting to note that $\{\Delta(x'_1), \Delta(x'_2), \ldots, \Delta(x'_m)\}$ is also an i.i.d. sample, because the approach that we use to construct the predictor $\hat{\theta}^*(x)$ is completely deterministic and $\{x'_1, x'_2, \ldots, x'_m\}$ is an i.i.d. sample from the covariate distribution. Then, we may use the sample to construct an empirical distribution of the optimality gap $\Delta(x)$, where the randomness is in x. With the empirical distribution we may produce all kinds of statistics of the optimality gap, e.g., its mean, variance or quantiles. The prediction interval that we want in Equation (4) essentially requires the $1-\alpha$ quantile of the optimality gap, which may be estimated by j^*th order statistic of $\{\Delta(x'_1), \Delta(x'_2), \ldots, \Delta(x'_m)\}$, where $j^* = \lceil m(1-\alpha) \rceil$.

Notice that the prediction interval not only provides a statistical measure of the optimality gap in the online stage, it also creates a measure of the goodness of the predictor $\hat{\theta}^*(x)$ in the offline stage. If the estimated $\Delta_{1-\alpha}$ is large, it often means that the predictor is not accurate enough and a larger sample size n is needed to fit the predictor. This is an interesting direction to further improve the proposed algorithm, and we leave it for future research.

4 NUMERICAL EXPERIMENTS

In this section we conduct some preliminary numerical experiments to test the performances of the proposed algorithms. We use the following two commonly used test functions.

1. De Jong's function:

$$f(\theta, x) = \sum_{l=1}^{d} (\theta_l - x_l)^2.$$

   The global minimum $\theta^*(x)$ of this function is obtained at $\theta_l = x_l$, $l = 1, 2, ..., d$.
2. Axis parallel hyper-ellipsoid function:

$$f(\theta, x) = \sum_{l=1}^{d} l \cdot (\theta_l - \sqrt{x_l})^2.$$

   The global minimum $\theta^*(x)$ of this function is obtained at $\theta_l = \sqrt{x_l}$, $l = 1, 2, ..., d$.

Notice that $d_\theta = d_x = d$ for both test functions and we consider $d = 4, 6$ and 8 to understand the impact of dimensionality on the performances of the algorithms. We let $\Theta = \mathscr{X} = [0,4]^d$ as the domains of the decisions variables and the covariates. To test the performances of the prediction intervals, we set $\alpha = 0.05$. In the KNN algorithm, we set $k = 10$; and in the KRR algorithm, we set $\lambda = 10^{-4}$. Furthermore, we use $m = 200$ observations from the covariate distribution to build the empirical distribution of the optimality gap and to estimate the 95% quantile $\Delta_{95\%}$.

We first study the convergence behaviors of the optimality gaps for different algorithms for different dimensions. To this end, we use the estimated $\Delta_{95\%}$ values. Notice that $\Delta_{95\%}$ converges to zero as the number of design points n goes to infinity. We report the results of the numerical experiments in Figures 1 and 2. It can be observed from the figures that both the NN and KNN algorithms perform better than the KRR algorithm when the number of design points is small. However, as n increases, the advantage of the KRR algorithm becomes obvious. This is mainly due to the faster rate of convergence that the KRR algorithm has for smooth objective functions.

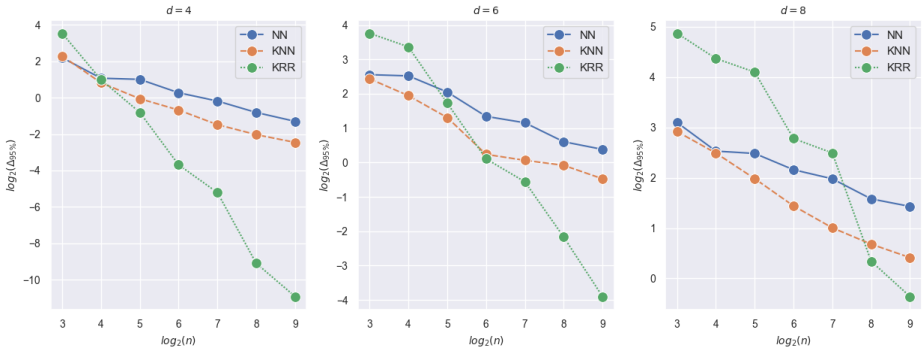

Figure 1: De Jong's function, the relation between n and $\Delta_{95\%}$.

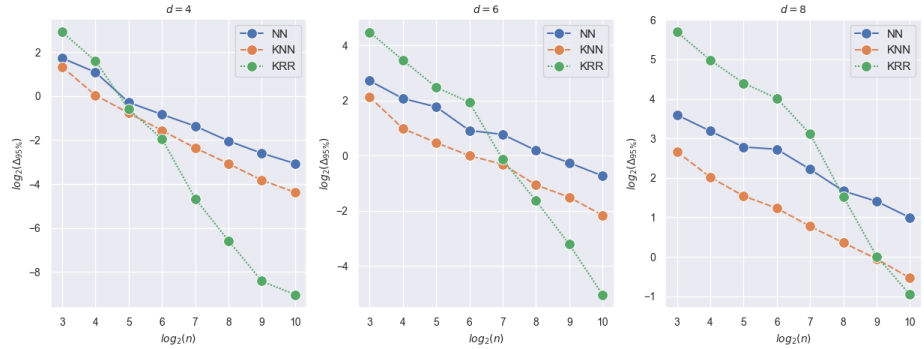

Figure 2: Axis parallel hyper-ellipsoid function, the relation between n and $\Delta_{95\%}$.

We then study the performances of the prediction intervals. To this end, we estimate $\Delta_{95\%}$ for all scenarios in the offline stage. Then, in the online stage, for each scenario, we randomly generate 200 test cases, evaluate the optimality gap for each case, and compare it with the corresponding $\Delta_{95\%}$. In Table 1, we report the observed probabilities that the test-case optimality gap is less than or equal to $\Delta_{95\%}$ for each scenario. From the table 1, it is clear that the observed probabilities are close to 0.95, demonstrating that the prediction interval works well in providing a statistical measure of the true optimality gap.

Table 1: Probabilities of $\{\Delta(x) \leqslant \Delta_{95\%}\}$ in test.

Test function	d = 4	d = 6	d = 8
De Jong's function	92.5%	94.5%	97.0%
Axis parallel hyper-ellipsoid function	96.0%	95.5%	94.0%

5 CONCLUSIONS

This paper presents algorithms that are designed based on the framework of offline simulation and online application for solving online simulation optimization problems. The offline stage involves finding the optimal solutions for a number of design points in the covariate space and learning a predictive model of the optimal solution for any values of the covariates, using the NN, KNN or KRR algorithms. Then, in the online stage, the optimal solution may be predicted instantly using the predictive model once the values of the covariates are observed. We derive the rate of convergence of the optimality gap of these algorithms and develop finite-sample statistical measure of the optimality gap when the algorithms are

used in practice. Our results reveal that the algorithms are particularly efficient for strongly convex online simulation optimization problems where the dimension of the covariate is small but the dimension of the decision variables is large.

REFERENCES

Agrawal, S., Z. Wang, and Y. Ye. 2014. "A dynamic near-optimal algorithm for online linear programming". *Operations Research* 62(4):876–890.

Amos, B. *et al*. 2023. "Tutorial on amortized optimization". *Foundations and Trends® in Machine Learning* 16(5):592–732.

Ban, G.-Y. and C. Rudin. 2019. "The big data newsvendor: Practical insights from machine learning". *Operations Research* 67(1):90–108.

Bertsimas, D. and N. Kallus. 2020. "From predictive to prescriptive analytics". *Management Science* 66(3):1025–1044.

Bertsimas, D. and N. Koduri. 2022. "Data-driven optimization: A reproducing kernel hilbert space approach". *Operations Research* 70(1):454–471.

Bertsimas, D., N. Korolko, and A. M. Weinstein. 2019. "Covariate-adaptive optimization in online clinical trials". *Operations Research* 67(4):1150–1161.

Bertsimas, D. and C. McCord. 2019. "From predictions to prescriptions in multistage optimization problems". *arXiv preprint arXiv:1904.11637*.

Devroye, L. 1978. "The uniform convergence of nearest neighbor regression function estimators and their application in optimization". *IEEE Transactions on Information Theory* 24(2):142–151.

Elmachtoub, A. N. and P. Grigas. 2022. "Smart "predict, then optimize"". *Management Science* 68(1):9–26.

Fan, W., L. J. Hong, G. Jiang, and J. Luo. 2024. "Review of Large-Scale Simulation Optimization". *arXiv preprint arXiv:2403.15669*.

Gao, S., C. Li, and J. Du. 2019. "Rate Analysis For Offline Simulation Online Application". In *2019 Winter Simulation Conference (WSC)*, 3468–3479 https://doi.org/10.1109/WSC40007.2019.9004834.

Hanasusanto, G. A. and D. Kuhn. 2013. "Robust data-driven dynamic programming". *Advances in Neural Information Processing Systems* 26.

Hannah, L., W. Powell, and D. Blei. 2010. "Nonparametric density estimation for stochastic optimization with an observable state variable". *Advances in Neural Information Processing Systems* 23.

Hong, L. J. and G. Jiang. 2019. "Offline simulation online application: A new framework of simulation-based decision making". *Asia-Pacific Journal of Operational Research* 36(06):1940015.

Jiang, G., L. J. Hong, and B. L. Nelson. 2020. "Online risk monitoring using offline simulation". *INFORMS Journal on Computing* 32(2):356–375.

Kannan, R., G. Bayraksan, and J. R. Luedtke. 2022. "Data-driven sample average approximation with covariate information". *arXiv preprint arXiv:2207.13554*.

Katrakazas, C., M. Quddus, W.-H. Chen, and L. Deka. 2015. "Real-time motion planning methods for autonomous on-road driving: State-of-the-art and future research directions". *Transportation Research Part C: Emerging Technologies* 60:416–442.

Keslin, G., B. L. Nelson, M. Plumlee, B. K. Pagnoncelli and H. Rahimian. 2022. "A classification method for ranking and selection with covariates". In *2022 Winter Simulation Conference (WSC)*, 1–12. IEEE.

Krantz, S. G. and H. R. Parks. 2002. *The implicit function theorem: history, theory, and applications*. Springer Science & Business Media.

Shen, H., L. J. Hong, and X. Zhang. 2021. "Ranking and selection with covariates for personalized decision making". *INFORMS Journal on Computing* 33(4):1500–1519.

Tuo, R. and J. C. Wu. 2016. "A theoretical framework for calibration in computer models: Parametrization, estimation and convergence properties". *SIAM/ASA Journal on Uncertainty Quantification* 4(1):767–795.

Wendland, H. 2004. *Scattered data approximation*, Volume 17. Cambridge university press.

Wendland, H. and C. Rieger. 2005. "Approximate interpolation with applications to selecting smoothing parameters". *Numerische Mathematik* 101:729–748.

AUTHOR BIOGRAPHIES

HENG LUO is a Ph.D. student in the School of Data Science at Fudan University in Shanghai, China. He received his bachelor's and master's degree from Hangzhou Dianzi University and Fudan University. His research interests include simulation optimization and statistical learning. His email address is 22110980024@m.fudan.edu.cn.

ZHIYANG LIANG is a Ph.D. student in the School of Data Science at Fudan University, Shanghai, China. He received his bachelor's degree from Fudan University. His email address is liangzy23@m.fudan.edu.cn.

L. JEFF HONG is the Fudan Distinguished Professor and Hongyi Chair Professor with joint appointment at School of Management and School of Data Science at Fudan University in Shanghai, China. His research interests include stochastic simulation, stochastic optimization, risk management and supply chain management. He is currently an associate editor of *Management Science* and *ACM Transactions on Modeling and Computer Simulation*, and an associate editor-in-chief of *Journal of Operations Research Society of China*. His email address is hong_liu@fudan.edu.cn.