IISE Transactions

Taylor & Francis
Taylor & Francis Group

IISE Transactions

ISSN: 2472-5854 (Print) 2472-5862 (Online) Journal homepage: https://www.tandfonline.com/loi/uiie21

Gaussian process based optimization algorithms with input uncertainty

Haowei Wang, Jun Yuan & Szu Hui Ng

To cite this article: Haowei Wang, Jun Yuan & Szu Hui Ng (2020) Gaussian process based optimization algorithms with input uncertainty, IISE Transactions, 52:4, 377-393, DOI: 10.1080/24725854.2019.1639859

To link to this article: https://doi.org/10.1080/24725854.2019.1639859

IISE TRANSACTIONS
2020, VOL. 52, NO. 4, 377–393
https://doi.org/10.1080/24725854.2019.1639859

Gaussian process based optimization algorithms with input uncertainty

Haowei Wang[a], Jun Yuan[b], and Szu Hui Ng[a]

[a]Department of Industrial Systems Engineering and Management, National University of Singapore, Singapore; [b]China Institute of FTZ Supply Chain, Shanghai Maritime University, Shanghai, P.R. China

ABSTRACT

Metamodels as cheap approximation models for expensive to evaluate functions have been commonly used in simulation optimization problems. Among various types of metamodels, the Gaussian Process (GP) model is popular for both deterministic and stochastic simulation optimization problems. However, input uncertainty is usually ignored in simulation optimization problems, and thus current GP-based optimization algorithms do not incorporate input uncertainty. This article aims to refine the current GP-based optimization algorithms to solve the stochastic simulation optimization problems when input uncertainty is considered. The comprehensive numerical results indicate that our refined algorithms with input uncertainty can find optimal designs more efficiently than the existing algorithms when input uncertainty is present.

ARTICLE HISTORY
Received 21 November 2018
Accepted 22 June 2019

KEYWORDS
Gaussian process; metamodel; stochastic simulation optimization; input uncertainty

1. Introduction

In practice, one common task is to optimize complex real systems (Shim *et al.*, 2002). However, most systems are often analytically intractable, due to their complexity, making optimizing such systems challenging. For example, the (*s*, *S*) inventory problem is mathematically intractable, due to random lead times (Jalali and van Nieuwenhuyse, 2015). To handle this problem, simulation models are widely used to imitate the complex real systems. Simulation optimization, or optimization via simulation, aims to find the optimal design that minimizes/maximizes some simulation output values. In this article, we only consider the minimization formulation. Generally, the deterministic simulation optimization problem is formulated as

$$\min_{\boldsymbol{x}\in\mathscr{X}} f(\boldsymbol{x}) \tag{1}$$

where $\boldsymbol{x} \in \mathbb{R}^d$ is a design parameter that belongs to a box-constraint design space $\mathscr{X}$. At any design parameter $\boldsymbol{x_0}$, $f(\boldsymbol{x_0})$ can be accurately observed with deterministic simulation output.

Unlike deterministic simulation whose outputs are deterministic, stochastic simulation is driven by some random input process, and hence, its outputs have variability. In this article, we assume a parametric model for the input process and term $\boldsymbol{\lambda} \in \mathbb{R}^l$ as the input parameter. For example, in an inventory simulation, the customers' random demand is the random input process. For stochastic simulation, we are interested in the expected simulation output $f(\boldsymbol{x}, \boldsymbol{\lambda})$ which is a function of both the design parameter $\boldsymbol{x}$ and the input parameter $\boldsymbol{\lambda}$. At any design point $\boldsymbol{x_0}$, given the input parameter $\boldsymbol{\lambda}$, $f(\boldsymbol{x_0}, \boldsymbol{\lambda})$ is not directly observable, and can only be estimated from samples of stochastic simulation output $y(\boldsymbol{x_0}, \boldsymbol{\lambda}) = f(\boldsymbol{x_0}, \boldsymbol{\lambda}) + \epsilon(\boldsymbol{x_0}, \boldsymbol{\lambda})$. $\epsilon(\boldsymbol{x_0}, \boldsymbol{\lambda})$ represents the output variability, whose distribution is assumed to be zero mean and finite variance. Given $\boldsymbol{\lambda}$, the objective of stochastic simulation optimization is to minimize the expected simulation output $\min_{\boldsymbol{x}\in\mathscr{X}} \mathrm{E}[y(\boldsymbol{x}, \boldsymbol{\lambda})]$. In most stochastic simulation optimization literature, the input parameter $\boldsymbol{\lambda}$ is assumed to be known, and thus, the dependence of simulation output $y(\boldsymbol{x}, \boldsymbol{\lambda})$ on $\boldsymbol{\lambda}$ is usually omitted. Hence, the most common form of the stochastic simulation optimization problem is

$$\min_{\boldsymbol{x}\in\mathscr{X}} \quad f(\boldsymbol{x}) = \mathrm{E}[y(\boldsymbol{x})]. \tag{2}$$

Currently, there are three main streams of methods to solve problem (1) or (2) with a continuous design space, including direct metaheuristics (Ólafsson, 2006), direct gradient-based methods (Kushner and Yin, 2003) and metamodel-based methods (Barton and Meckesheimer, 2006). Among them, metamodel-based methods require no strict assumption on objective functions, and are easier and cheaper to apply. Xu *et al.* (2015) gives a comprehensive review on simulation optimization methods.

For real applications, simulation runs can be very time-consuming. In this case, the Gaussian process (GP) based optimization algorithm, a type of metamodel-based method, is a popular choice which can efficiently utilize the limited simulation runs (Preuss *et al.*, 2012). The most popular GP-based optimization algorithm is the Efficient Global Optimization (EGO) algorithm (Jones *et al.*, 1998), in which the Expected Improvement (EI) infill criterion is used to find the next design point that has the largest potential

CONTACT Szu Hui Ng isensh@nus.edu.sg
Color versions of one or more of the figures in the article can be found online at www.tandfonline.com/uiie.
Supplemental data for this article is available online at the publishers website.

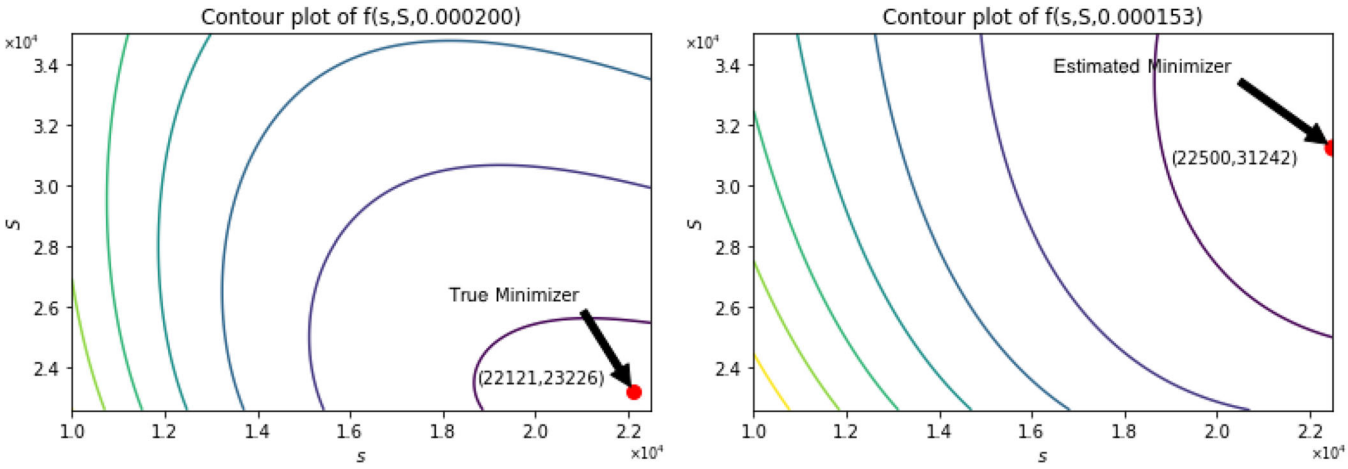

Figure 1. The contour plot for different plug-in values of input parameter for the (*s*, *S*) problem expected cost function. The circle point indicates the location of the minimizer.

improvement of the objective function value. EI and its variants have been widely applied to solve both deterministic (Jones *et al.*, 1998; Kleijnen, 2015) and stochastic (Huang *et al.*, 2006; Quan *et al.*, 2013) simulation optimization problems. In addition to these algorithms based on EI, there are other GP-based algorithms such as the Knowledge Gradient (KG) (Frazier *et al.*, 2009) which concentrates on the improvement of the conditional mean, and the Informational Approach for Global Optimization (IAGO) (Villemonteix *et al.*, 2009) and the Expected Excursion Volume (EEV) (Picheny, 2015) which both directly focus on the uncertainty in minimizers, but with different uncertainty measures. More details about these algorithms will be discussed in Section 3.2.

In reality, however, the true value of the input parameter $\boldsymbol{\lambda}$ is usually unknown, and instead estimated from real world data. Most current simulation optimization methods use the plug-in input parameter $\hat{\boldsymbol{\lambda}}$ estimated from the real world data as if it were the true input parameter, and simulation optimization under the estimated plug-in input parameter is carried out:

$$\min_{\boldsymbol{x}\in\mathscr{X}} \quad f(\boldsymbol{x},\hat{\boldsymbol{\lambda}}) = \mathrm{E}\left[y(\boldsymbol{x},\hat{\boldsymbol{\lambda}})\right] \tag{3}$$

As real world data collected is finite, this leads to uncertainty associated with the estimation (Lam, 2016). This type of uncertainty is formally termed as input uncertainty.

Example 1.1. Here, we take the popular (*s*, *S*) inventory problem (Bashyam and Fu, 1998) to illustrate the potential risk of ignoring input uncertainty in simulation optimization. The design parameter is $\boldsymbol{x} = [s, S]^T$ where s is the basic ordering level and S is the order-up-to level. Suppose the random demand ξ follows $\exp(\boldsymbol{\lambda_0})$ and $\boldsymbol{\lambda_0}$ is the true input parameter. The objective is to find the optimal design $\boldsymbol{x}^* = [s^*, S^*]^T$ that minimizes the long-run expected cost: $f(s, S, \boldsymbol{\lambda_0})$. This objective function is

$$f(s,S,\boldsymbol{\lambda}) = \frac{1}{\boldsymbol{\lambda}} + [100 + s - \frac{1}{\boldsymbol{\lambda}} + 0.5\lambda(S^2 - s^2) + \frac{101}{\boldsymbol{\lambda}} e^{-\lambda s}]/[1 + \boldsymbol{\lambda}(S-s)].$$

The true value $\boldsymbol{\lambda_0}$ is usually unknown in practice, and the only available information is a finite set of historical sales data $\boldsymbol{D_h} : \{\xi_1, \xi_2, \ldots, \xi_h\}$, observed from the true distribution $\exp(\boldsymbol{\lambda_0})$. The most common approach to solve this problem is to first obtain a maximum likelihood estimator for $\boldsymbol{\lambda_0}$, $\hat{\boldsymbol{\lambda}} = 1/(1/h\sum_{i=1}^{h}\xi_i)$, and then treat it as if it were the true parameter (ignoring the input uncertainty). GP-based optimization algorithms then focus on solving $\min_{s,S} f(s, S, \hat{\boldsymbol{\lambda}})$. However, different finite sets of data can lead to different estimated $\hat{\boldsymbol{\lambda}}$, and thus, resulting in different optimal design obtained from $\min_{s,S} f(s, S, \hat{\boldsymbol{\lambda}})$. For example, with 100 observations from $\exp(\boldsymbol{\lambda_0} = 0.0002)$, we can obtain an estimate $\hat{\boldsymbol{\lambda}} = 0.000\,153$. As seen from Figure 1, the minimizer of the expected cost function will change significantly from the true optimal (22 121, 23 226) to (22 500, 31 242) with this plug-in $\hat{\boldsymbol{\lambda}}$. In other words, the optimal design can be seriously misled when ignoring input uncertainty.

To avoid the potential risk incurred from a poorly estimated input parameter, Zhou and Xie (2015) proposed to incorporate input uncertainty into the simulation optimization problem, and formulated the optimization problem as follows:

$$\min_{\boldsymbol{x}\in\mathscr{X}} \quad \mathrm{E}_{\boldsymbol{\lambda}}\left[f(\boldsymbol{x},\boldsymbol{\lambda})\right] = \mathrm{E}_{\boldsymbol{\lambda}}\left[\mathrm{E}\left[y(\boldsymbol{x},\boldsymbol{\lambda})\right]\right], \tag{4}$$

where the original objective function is now averaged over all input parameters. They showed that this formulation of the objective is consistent with the problem when the true parameter is known, and illustrated the advantages of problem (4) over problem (3) with small real world data size. Wu *et al.* (2018) further showed the consistency of the optimal solution under some mild conditions. Recently, Pearce and Branke (2017) extended EGO and KG to incorporate input uncertainty to solve problem (4) and showed that their proposed methods outperformed the usual plug-in methods. Other than this work, most GP-based optimization algorithms, however, do not consider the input uncertainty, and cannot be directly applied to solve problem (4). In this article, we acknowledge the risk in ignoring input uncertainty (as highlighted in our motivating example), and refine four GP-based simulation optimization algorithms to solve problem (4). Although this work is closely related to Pearce and Branke (2017), it differs in the following contributions:

1. We use a two-step framework to refine EGO and KG, with the sequential design points selected using our modified infill criteria in the first step and the sequential input points selected using the Integrated Mean Squared Error criterion in the next step. We show that our two-step approach can match the approach of Pearce and Branke (2017) in performance, but require less evaluation time.
2. In addition to EGO and KG, we also refine IAGO and EEV using the two-step framework.
3. A comprehensive comparison among the proposed algorithms is conducted and suggestions for choice and application are provided.

The rest of this article is organized as follows. In Section 2, we will briefly review input uncertainty quantification in the simulation literature and the current works in stochastic optimization with input uncertainty. In Section 3, the basics of the GP model and GP-based optimization algorithms are reviewed. Then in Section 4, details of our refined GP-based algorithms with input uncertainty will be described. In Section 5, a comprehensive comparison of several GP-based optimization algorithms, with and without input uncertainty, is conducted. Section 6 concludes the article and highlights future research directions.

2. Literature review

The quantification of input uncertainty in stochastic simulation has been studied over the years. The research to characterize and quantify it has been developed along two lines. The first is from a frequentist approach, which includes the delta method (Cheng and Holland, 1997), two-point method (Cheng and Holland, 2004), bootstrap sampling (Barton and Schruben, 1993) and metamodel-assisted bootstrap (Barton *et al.*, 2013). And the other is from a Bayesian approach, where prior and posterior distributions are used to characterize the input uncertainty. Both direct sampling (Zouaoui and Wilson, 2003) and metamodel-assisted methods (Xie *et al.*, 2014) have been developed to propagate input uncertainty to quantify the simulation output variability. A comprehensive review of these approaches can be found in Lam (2016).

Traditional simulation optimization approaches (Xu *et al.*, 2015) focused on the development of methods under known input distributions. With the recognition of the impact of input uncertainty on system performance, attention has grown to incorporate input uncertainty in simulation to manage the risk. Early pioneering works considering input uncertainty focused on ranking and selection problems, which have a finite decision space (Corlu and Biller, 2013; Fan *et al.*, 2013; Zhang and Ding, 2016). More recently, researchers have started to look at continuous decision space simulation optimization under input uncertainty. To avoid overly conservative solutions from distributionally robust optimization, Zhou and Xie (2015) proposed an alternative Bayesian Risk Optimization (BRO) framework which quantifies input uncertainty from a Bayesian perspective and provided four risk formulations: mean, mean-variance, Value-at-Risk (VaR), and Conditional Value-at-Risk (CVaR). Zhou and Wu (2017) and Wu *et al.* (2018) further established the consistency of the BRO objective functions and optimal solutions. Zhu *et al.* (2016, 2018) extended the gradient-based adaptive stochastic search method to the optimization of VaR, CVaR formulation in BRO over a continuous decision space. Pearce and Branke (2017) solved the mean formulation by considering the input uncertainty in the infill criteria in EGO and KG. Lakshmanan and Venkateswaran (2017) combined stochastic kriging metamodel-assisted bootstrapping with a modified EGO algorithm to sequentially search for the optimum and estimate the worst case bound of the simulation optimization.

Similar to the recent works, in this article, we focus on continuous decision space optimization, and among the three main streams of simulation optimization approaches (direct metaheuristic, direct gradient-based and metamodel-based as outlined in Section 1), we will focus on the metamodel-based approach, and specifically the GP-based optimization approach.

3. Background and basics

3.1. Stochastic GP model

Let $y(\boldsymbol{x})$ be the stochastic simulation output at $\boldsymbol{x}$, and we assume that $y(\boldsymbol{x})$ is a realization of a random process that can be described by the model:

$$y(\boldsymbol{x}) = f(\boldsymbol{x}) + \epsilon(\boldsymbol{x}), \tag{5}$$

where $f(\boldsymbol{x}) = \mathrm{E}[y(\boldsymbol{x})]$ is the expected simulation output which describes the mean of the process, and $\epsilon(\boldsymbol{x})$ is the stochastic noise that is assumed to follow an independent normal distribution across $\boldsymbol{x}$ with mean 0 and finite constant variance σ_ϵ^2. We further assume that $f(\boldsymbol{x})$ can be modeled as a GP F with mean μ and stationary covariance function Σ_F, denoted as $F \sim \mathrm{GP}(\mu, \Sigma_F)$. This form of the stochastic GP model is also termed as the Nugget Effect Kriging Model (Cressie, 1992). In this article, we assume that $\mu = 0$, i.e., the GP prior mean is zero. This assumption is commonly used in many applications (Villemonteix *et al.*, 2009; Binois *et al.*, 2019). The covariance function measures the spatial relationship of F at any two points $\boldsymbol{x}$ and $\boldsymbol{x}'$: $\Sigma_F(\boldsymbol{x}, \boldsymbol{x}') = \mathrm{Cov}(F(\boldsymbol{x}), F(\boldsymbol{x}')) = \tau^2 R_F(\boldsymbol{x}, \boldsymbol{x}'; \boldsymbol{\theta})$, where R_F is the correlation function, and τ and $\boldsymbol{\theta}$ are the parameters. Many possible covariance functions can be chosen, such as the squared exponential and the Matern covariance functions.

Suppose that the initial experiment design points are $\{\boldsymbol{x_1}, \boldsymbol{x_2}, \ldots, \boldsymbol{x_n}\}$. For each design point $\boldsymbol{x_i}$, m replicates of simulation runs are conducted. Denote the stochastic simulation output sample mean vector at these points as $\bar{\mathcal{Y}}_{\boldsymbol{n}} = [\bar{y}(\boldsymbol{x_1}), \bar{y}(\boldsymbol{x_2}), \ldots, \bar{y}(\boldsymbol{x_n})]^T$. Conditional on $\bar{\mathcal{Y}}_{\boldsymbol{n}}$, F can be updated and the posterior distribution $F|\bar{\mathcal{Y}}_{\boldsymbol{n}} \sim \mathrm{GP}(m_n', k_n')$ can be derived:

$$m_n'(\boldsymbol{x}) = \mathrm{E}\left[F(\boldsymbol{x})|\bar{\mathcal{Y}}_{\boldsymbol{n}}\right] = \tau^2 R_F(\boldsymbol{x}, \cdot; \boldsymbol{\theta})^T \left[\tau^2 R_F(\boldsymbol{\theta}) + \boldsymbol{\Sigma}_{\boldsymbol{\epsilon}}\right]^{-1} \bar{\mathcal{Y}}_{\boldsymbol{n}}, \tag{6}$$

$$\begin{aligned} k_n'(\boldsymbol{x}, \boldsymbol{x}') = \mathrm{Cov}\left[F(\boldsymbol{x}), F(\boldsymbol{x}')|\bar{\mathcal{Y}}_{\boldsymbol{n}}\right] &= \tau^2 R_F(\boldsymbol{x}, \boldsymbol{x}'; \boldsymbol{\theta}) \\ &- \tau^4 R_F(\boldsymbol{x}, \cdot; \boldsymbol{\theta})^T \left[\tau^2 R_F(\boldsymbol{\theta}) + \boldsymbol{\Sigma}_{\boldsymbol{\epsilon}}\right]^{-1} R_F(\boldsymbol{x}', \cdot; \boldsymbol{\theta}), \end{aligned} \tag{7}$$

where $f(\boldsymbol{x})$ can be predicted by the conditional mean $m'_n(\boldsymbol{x})$. The conditional variance of F is $s'^2_n(\boldsymbol{x}) = k'_n(\boldsymbol{x},\boldsymbol{x})$. $\tau^2 R_F(\boldsymbol{x},\cdot;\boldsymbol{\theta})^T = [\mathrm{Cov}[F(\boldsymbol{x}),F(\boldsymbol{x_1})],...,\mathrm{Cov}[F(\boldsymbol{x}),F(\boldsymbol{x_n})]]^T$ is a $n\times 1$ vector. $\tau^2 R_F(\boldsymbol{\theta})$ is the $n\times n$ covariance matrix across all observed points. $\boldsymbol{\Sigma_\epsilon} = \mathrm{Diag}\{\frac{1}{m}\sigma^2_\epsilon \ \cdots, \frac{1}{m}\sigma^2_\epsilon\} = \frac{1}{m}\sigma^2_\epsilon \boldsymbol{I}$ is a $n\times n$ matrix. One common approach is to estimate τ and $\boldsymbol{\theta}$ with the maximum log-likelihood (Rasmussen and Williams, 2006) or maximum a posteriori (Ng and Yin, 2012), and we can use pooled variance of the simulation outputs to estimate σ^2_ϵ. As with all metamodels, before its application for prediction or optimization (as required in our optimization algorithms), validation of the GP model is first required. Here, we recommend the use of Leave-one-out cross-validation which has been shown to be almost unbiased (Efron, 1983).

For the deterministic GP, the simulation output is deterministic. Therefore, $\epsilon(\boldsymbol{x}), \sigma^2_\epsilon$, and $\boldsymbol{\Sigma_\epsilon}$ will vanish in all the above equations with the other components unchanged and replications are not needed ($m=1$). For the stochastic GP with heteroscedastic noise (Ankenman *et al.*, 2010; Yin *et al.*, 2011), the variance of the stochastic noise $\sigma^2_\epsilon(\boldsymbol{x})$ will depend on $\boldsymbol{x}$ (not a constant value).

3.2. Overview of GP-based optimization algorithms

For deterministic simulation optimization (1), the objective is to minimize the deterministic simulation output $f(\boldsymbol{x})$. Conditional on the initial deterministic simulation outputs $\boldsymbol{f_n} = [f(\boldsymbol{x_1}),...,f(\boldsymbol{x_n})]^T$, a deterministic GP model can be obtained $F_n \triangleq F|\boldsymbol{f_n} \sim \mathrm{GP}(m_n, k_n)$ as the metamodel for $f(\boldsymbol{x})$ with mean m_n and covariance k_n. The GP-based optimization problem becomes: $\min_{\boldsymbol{x}} m_n(\boldsymbol{x})$.

For stochastic simulation optimization (2), the objective is to minimize the expected stochastic simulation output. Conditional on the initial outputs mean $\bar{\boldsymbol{y}}_{\boldsymbol{n}} = [\bar{y}(\boldsymbol{x_1}),..., \bar{y}(\boldsymbol{x_n})]^T$, a stochastic GP metamodel is obtained $F'_n \triangleq F|\bar{\boldsymbol{y}}_{\boldsymbol{n}} \sim \mathrm{GP}(m'_n, k'_n)$ for $f(\boldsymbol{x})$. The GP-based optimization problem becomes $\min_{\boldsymbol{x}} m'_n(\boldsymbol{x})$.

Generally, a GP-based optimization algorithm uses the GP model to adaptively choose sequential points to evaluate in order to locate global optimizers with as few evaluations as possible. More specifically, a deterministic (or stochastic) GP model F_n (or F'_n) can be obtained based on initial deterministic (or stochastic) simulation outputs vector $\boldsymbol{f_n}$ (or $\bar{\boldsymbol{y}}_{\boldsymbol{n}}$). Then an infill criterion is calculated based on this obtained GP model to determine the next design point $\boldsymbol{x_{n+1}}$. After the simulation output at $\boldsymbol{x_{n+1}}$ is observed, the GP model can be updated to F_{n+1} (or F'_{n+1}) and a new design point $\boldsymbol{x_{n+2}}$ is selected again. The procedure is iterated until some stopping criteria are met or the simulation budget is exhausted. As the GP model F_n (or F'_n) updating procedure continues, its conditional mean m_n (or m'_n) can better approximate the objective function $f(\boldsymbol{x})$, and thus, the approximation accuracy of the global minimizers $\boldsymbol{x}^*$ is improved.

In the following, we will review four GP-based optimization algorithms, i.e., EGO, KG, IAGO and EEV. Furthermore, we extend the EEV algorithm for stochastic simulation optimization. These four algorithms differ primarily in the design of the infill criteria. In this article, we focus on systems with homoscedastic noise. Readers can refer to Jalali *et al.* (2017) for a review of algorithms for heteroscedastic noise.

3.2.1. EGO

EGO (Jones *et al.*, 1998) is first introduced to solve the deterministic simulation optimization problem (1). The infill criterion of EGO is called EI, Which is designed to focus on the improvement of the objective function value $\mathrm{EI}(\boldsymbol{x_{n+1}}) = \mathrm{E}_{F_n}[(f_n^{\min} - F_n(\boldsymbol{x_{n+1}}))^+ | \boldsymbol{f_n}]$, where $f_n^{\min} = \min\{f(\boldsymbol{x_1}),...,f(\boldsymbol{x_n})\}$ is the observed current best objective function value. At the candidate point $\boldsymbol{x_{n+1}}$, the objective function value $f(\boldsymbol{x_{n+1}})$ is unobserved, but we have $F_n(\boldsymbol{x_{n+1}})$ as the Gaussian variable to account for the uncertainty of $f(\boldsymbol{x_{n+1}})$. Hence, the improvement over the current best $f_n^{\min}$ given by $(f_n^{\min} - F_n(\boldsymbol{x_{n+1}}))^+$ is averaged over the uncertainty and the EI can be calculated analytically:

$$\begin{aligned}\mathrm{EI}(\boldsymbol{x_{n+1}}) &= \mathrm{E}_{F_n}\left[\left(f_n^{\min} - F_n(\boldsymbol{x_{n+1}})\right)^+ | \boldsymbol{f_n}\right] \\ &= \Delta\Phi\left(\frac{\Delta}{s_n(\boldsymbol{x_{n+1}})}\right) + s_n(\boldsymbol{x_{n+1}})\phi\left(\frac{\Delta}{s_n(\boldsymbol{x_{n+1}})}\right)\end{aligned} \tag{8}$$

where $\Delta = f_n^{\min} - m_n(\boldsymbol{x_{n+1}})$, Φ and ϕ are the cumulative distribution function and probability distribution function of the standard normal random variable. With Equation (8), the next design point is selected by maximizing the EI criterion with $\mathrm{argmax}_{\boldsymbol{x_{n+1}}} \mathrm{EI}(\boldsymbol{x_{n+1}})$.

This EI criterion not only aims to choose points that can reduce the function value most, but also considers the uncertainty at unobserved points, enabling it to balance the search within the local area of the current optimizer and at unexplored regions away from previously observed points.

The challenge to use EI to solve stochastic simulation optimization problems is to estimate the unknown current best objective function value $f_n^{\min}$. One direct approach is to use some plug-in value T to estimate the current best $f_n^{\min}$. The EI with this plug-in T can be formulated as

$$\begin{aligned}\mathrm{EI}_T(\boldsymbol{x_{n+1}}) &= \mathrm{E}_{F'_n}\left[\left(T - F'_n(\boldsymbol{x_{n+1}})\right)^+ | \bar{\boldsymbol{y}}_{\boldsymbol{n}}\right] \\ &= \Delta\Phi\left(\frac{\Delta}{s'_n(\boldsymbol{x_{n+1}})}\right) + s'_n(\boldsymbol{x_{n+1}})\phi\left(\frac{\Delta}{s'_n(\boldsymbol{x_{n+1}})}\right),\end{aligned} \tag{9}$$

where $\Delta = T - m'_n(\boldsymbol{x_{n+1}})$. Adopting this EI with the plug-in T idea, Huang *et al.* (2006) added an additional multiplicative factor to Equation (9) to account for the diminishing value of sampling at the same points, and proposed the Augmented Expected Improvement (AEI) criterion:

$$\mathrm{AEI}(\boldsymbol{x_{n+1}}) = \mathrm{EI}_T(\boldsymbol{x_{n+1}}) \times \left(1 - \frac{\sigma_\epsilon}{\sqrt{s'^2_n(\boldsymbol{x_{n+1}}) + \sigma^2_\epsilon}}\right).$$

This multiplicative factor penalizes additional replications at a particular location, and thus helps promote exploration. It is noted that the AEI requires information about σ_ϵ. When information about σ_ϵ is scarce, using $T = \min\{m'_n(\boldsymbol{x_1}),...,$

$m'_n(\boldsymbol{x_n})\}$ in Equation (9) can provide reasonable results (Picheny *et al.*, 2013).

3.2.2. KG

The KG approach (Frazier *et al.*, 2009) is proposed to solve the stochastic simulation optimization problem (2). The main motivation for KG lies in the fact that for the stochastic case, the conditional mean $m'_n(\boldsymbol{x})$ may be closer to the true value $f(\boldsymbol{x})$ than the observed sample mean $\bar{y}(\boldsymbol{x})$. Thus, KG focuses on the effect of the next evaluation point on the conditional mean $m'_n(\boldsymbol{x})$, and the infill criterion is designed as the improvement of the conditional mean:

$$\begin{aligned} \mathrm{I}'_n(\boldsymbol{x_{n+1}}) &= \min_{\boldsymbol{x}}\left(m'_n(\boldsymbol{x})\right) - \min_{\boldsymbol{x}}\left(m'_{n+1}(\boldsymbol{x})\right) \\ &= \min_{\boldsymbol{x}}\left(m'_n(\boldsymbol{x})\right) - \min_{\boldsymbol{x}}\left[m'_n(\boldsymbol{x}) + \frac{k'_n(\boldsymbol{x},\boldsymbol{x_{n+1}})}{\sqrt{s'^2_n(\boldsymbol{x_{n+1}}) + \sigma^2_\epsilon}} Z\right] \end{aligned} \tag{10}$$

where Z is the standard normal random variable. The updated conditional mean $m'_{n+1}(\boldsymbol{x})$ for any candidate point $\boldsymbol{x_{n+1}}$ is calculated using the GP updating formula. $m'_n(\boldsymbol{x}), s'^2_n(\boldsymbol{x_{n+1}})$, and $k'_n(\boldsymbol{x},\boldsymbol{x_{n+1}})$ are obtained from the stochastic GP F'_n. As Z is a standard normal random variable, the improvement function is random and the expectation is taken with respect to Z. KG aims to select the next design point that best improves the minimum of conditional mean and the next design point is selected with $\mathrm{argmax}_{\boldsymbol{x_{n+1}}}\mathrm{E}_Z[\mathrm{I}'_n(\boldsymbol{x_{n+1}})]$.

3.2.3. IAGO

IAGO is first designed to solve the deterministic simulation optimization problem (1). In contrast with EGO and KG that aim to reduce the function value, the IAGO approach focuses directly on the uncertainty (information) of the global minimizers $\boldsymbol{x}^*$. IAGO uses the conditional entropy of $\boldsymbol{x}^*$ as the measure to quantify this uncertainty. The infill criterion, see Equation (11), is defined as the expected conditional entropy that measures the uncertainty reduction in $\boldsymbol{x}^*$ expected from any candidate point $\boldsymbol{x_{n+1}}$. Villemonteix *et al.* (2009) approximated this expected conditional entropy with

$$\begin{aligned} H\left(\boldsymbol{x}^*|\boldsymbol{f_n}, F_n(\boldsymbol{x_{n+1}})\right) &= \sum_{i=1}^{M}\frac{1}{M}H\left(\boldsymbol{x}^*|\boldsymbol{f_n}, F_n(\boldsymbol{x_{n+1}}) = \gamma_i\right) \\ &= -\frac{1}{M}\sum_{i=1}^{M}\sum_{\boldsymbol{x}\in\mathbb{D}} P_{\boldsymbol{x}^*|\boldsymbol{f_n},\gamma_i}(\boldsymbol{x})\log_2 P_{\boldsymbol{x}^*|\boldsymbol{f_n},\gamma_i}(\boldsymbol{x}). \end{aligned} \tag{11}$$

To ease the computational burden, $F_n(\boldsymbol{x})$ is quantized as a discrete random variable $Q(F_n(\boldsymbol{x}))$ with Q as the quantization operator that takes values in $\{\gamma_1, \gamma_2, ..., \gamma_M\}$. $\{\gamma_1, \gamma_2, ..., \gamma_M\}$ are selected to satisfy for $\forall i, P(Q(F_n(\boldsymbol{x})) = \gamma_i) = (1/M)$ and $\forall u \in \mathbb{R}, Q(u) = \gamma_k$ with $k = \mathrm{argmin}_i|\gamma_k - u|$, based on the fact that $F_n(\boldsymbol{x})$ is Gaussian variable (Villemonteix *et al.*, 2009). In addition, when computing the conditional entropy $H(\boldsymbol{x}^*|\boldsymbol{f_n}, F_n(\boldsymbol{x_{n+1}}) = \gamma_i)$, a discrete version of the continuous candidate space, $\mathbb{D} \subseteq \mathscr{X}$ is used (Villemonteix *et al.*, 2009). Then the conditional distribution of $\boldsymbol{x}^* \in \mathbb{D}$ is discrete, and $P_{\boldsymbol{x}^*|\boldsymbol{f_n},\gamma_i}(\boldsymbol{x})$ is the conditional probability mass function of $\boldsymbol{x}^*$, which can be approximated as follows. First, N_{sp} random sample paths of $F(\boldsymbol{x})$ satisfying $F(\boldsymbol{x_1}) = f(\boldsymbol{x_1}), ..., F(\boldsymbol{x_n}) = f(\boldsymbol{x_n}), F(\boldsymbol{x_{n+1}}) = \gamma_i$ are generated. Then minimizers of these N_{sp} sample paths are obtained, and $P_{\boldsymbol{x}^*|\boldsymbol{f_n},\gamma_i}(\boldsymbol{x})$ is estimated using the obtained sample of minimizers:

$$\hat{P}_{\boldsymbol{x}^*|\boldsymbol{f_n},\gamma_i}(\boldsymbol{x}) = \frac{1}{N_{sp}}\sum_{i=1}^{N_{sp}}\delta_i(\boldsymbol{x}),$$

where $\delta_i(\boldsymbol{x}) = 1$ when $\boldsymbol{x}$ is the global minimizer of the ith sample path, for $i = 1, 2, ..., N_{sp}$. When $H(\boldsymbol{x}^*|\boldsymbol{f_n}, F_n(\boldsymbol{x_{n+1}}))$ decreases, it means that the distribution of $\boldsymbol{x}^*$ will get more peaked, and hence, the uncertainty remaining in $\boldsymbol{x}^*$ decreases. Therefore, the candidate point $\boldsymbol{x_{n+1}}$ that reduces the uncertainty in $\boldsymbol{x}^*$ most is selected with $\mathrm{argmin}_{\boldsymbol{x_{n+1}}}H(\boldsymbol{x}^*|\boldsymbol{f_n}, F_n(\boldsymbol{x_{n+1}}))$.

Yuan *et al.* extended the IAGO to the stochastic case with homoscedastic noise and the next design point is selected by $\mathrm{argmin}_{\boldsymbol{x_{n+1}}}H(\boldsymbol{x}^*|\bar{\boldsymbol{y}}_{\boldsymbol{n}}, F'_n(\boldsymbol{x_{n+1}}))$. The only difference is that stochastic simulation output $\bar{\boldsymbol{y}}_{\boldsymbol{n}}$ and stochastic GP F'_n are used instead in the infill criterion.

3.2.4. EEV

Similar to IAGO, the EEV approach proposed by Picheny (2015) also directly focuses on the uncertainty in $\boldsymbol{x}^*$, and is introduced to solve the deterministic simulation optimization problem (1). To cope with the high computational efforts required by IAGO (for expensive sample paths generation), the EEV approach defines a simpler to compute uncertainty measure: $ev_n = \int_{\mathscr{X}} P(F_n(\boldsymbol{x}) \le f_n^{\min})d\boldsymbol{x}$. ev_n represents the integrated probability of improvement over all possible candidate points in $\mathscr{X}$. A small value of ev_n indicates a low uncertainty in $\boldsymbol{x}^*$ as the potential improvement from any additional candidate point is limited. If the next design point is $\boldsymbol{x_{n+1}}$, the updated uncertainty is

$$ev_{n+1}(\boldsymbol{x_{n+1}}) = \int_{\mathscr{X}} P\left(F_{n+1}(\boldsymbol{x}) \le \min\left\{f_n^{\min}, F_n(\boldsymbol{x_{n+1}})\right\}\right)d\boldsymbol{x}, \tag{12}$$

where $F_{n+1}(\boldsymbol{x})$ is the updated GP. The benefit of evaluating at a candidate point $\boldsymbol{x_{n+1}}$ can be measured by the uncertainty reduction offered by that candidate point: $ev_n - ev_{n+1}(\boldsymbol{x_{n+1}})$; thus a small $ev_{n+1}(\boldsymbol{x_{n+1}})$ provides a large uncertainty reduction. $ev_{n+1}(\boldsymbol{x_{n+1}})$ is random due to the randomness of $F_n(\boldsymbol{x_{n+1}})$, thus we take expectation w.r.t. $F_n(\boldsymbol{x_{n+1}})$. The next design point that can best reduce the expected uncertainty can then be selected with $\mathrm{argmin}_{\boldsymbol{x_{n+1}}}\mathrm{E}_{F_n}[ev_{n+1}(\boldsymbol{x_{n+1}})]$.

To extend the EEV approach for stochastic simulation optimization, here we replace the current best deterministic objective value $f_n^{\min}$ with the current best estimated mean value T. We then formulate the infill criterion as $ev'_{n+1}(\boldsymbol{x_{n+1}}) = \int_{\mathscr{X}} P(F'_{n+1}(\boldsymbol{x}) \le \min\{T, F'_n(\boldsymbol{x_{n+1}})\})d\boldsymbol{x}$, where $F'_{n+1}(\boldsymbol{x})$ is the updated stochastic GP. Drawing parallels from the extension of EGO for stochastic simulation optimization in Section 3.2.1, we choose $T = \min\{m'_n(\boldsymbol{x_1}), ..., m'_n(\boldsymbol{x_n})\}$. The next design point is then selected with $\mathrm{argmin}_{\boldsymbol{x_{n+1}}}\mathrm{E}_{F'_n}[ev'_{n+1}(\boldsymbol{x_{n+1}})]$.

4. GP-based algorithms with input uncertainty

As illustrated by the motivating example in Example 1.1, the simulation optimization with estimated plug-in inputs can result in suboptimal designs. To address this problem, this article takes a Bayesian perspective and treats the input parameter $\boldsymbol{\lambda}$ as a random variable. Specifically, we apply a Bayesian approach, conditional on the real word data $\boldsymbol{D_h} = \{\xi_1, ..., \xi_h\}$, to compute the posterior distribution of $\boldsymbol{\lambda}$ to quantify input uncertainty: $p(\boldsymbol{\lambda}|\boldsymbol{D_h}) \propto p(\boldsymbol{\lambda}) \cdot p(\boldsymbol{D_h}|\boldsymbol{\lambda}) = p(\boldsymbol{\lambda}) \cdot \Pi_{i=1}^{h} \pi_0(\xi_i; \boldsymbol{\lambda})$ where $p(\boldsymbol{\lambda})$ represents the prior knowledge about $\boldsymbol{\lambda}$. We also assume that the random input process follows a known parametric distribution family $\pi_0(\cdot; \boldsymbol{\lambda})$, but the parameter $\boldsymbol{\lambda}$ is unknown. In other words, only input-parameter uncertainty is considered.

One advantage of using the Bayesian approach is that we can incorporate prior knowledge about the real input process and the input parameters in the prior distributions. This approach systematically enables the accounting and quantification of these uncertainties along with the data observations to update the knowledge on these parameters consistently (even for small sample sizes without the general need for asymptotics). This view is general, and if no prior knowledge is available or assumed, non-informative priors can be used. Selecting priors however can be an onerous task. Berger (1985) provided details for eliciting prior information for parameters and distributions from decision makers or observable quantities. For mathematical convenience and interpretability, conjugate priors are often used, where the prior information can be captured in its parameters. Non-informative hyperparameters can also be used in these conjugate priors. Furthermore, hierarchical priors can be used to reduce their impact on the posterior when the sample size is small (Gelman *et al.*, 2013). Non-informative or reference priors that are designed to minimally affect the posterior can also be applied when no prior information is available. Kass and Wasserman (1996) provided a detailed discussion on these priors. In our examples in Section 5, we use the weakly-informative conjugate priors as an illustration.

4.1. Simulation optimization with input uncertainty

In order to hedge the risk of poorly estimated input parameter leading to suboptimal designs, we focus on the stochastic simulation optimization problem (4), rewritten here as

$$\min_{\boldsymbol{x} \in \mathscr{X}} \quad g(\boldsymbol{x}) = \mathrm{E}_{\boldsymbol{\lambda}}\big[f(\boldsymbol{x}, \boldsymbol{\lambda})\big], \tag{13}$$

where $\boldsymbol{x} = [x_1, x_2, ..., x_d]^T \in \mathbb{R}^d$ is the design parameter, and $\boldsymbol{\lambda} = [\lambda_1, \lambda_2, ..., \lambda_l]^T \in \mathbb{R}^l$ is the input parameter. $y(\boldsymbol{x}, \boldsymbol{\lambda}) = f(\boldsymbol{x}, \boldsymbol{\lambda}) + \epsilon$ is the stochastic simulation output, and $f(\boldsymbol{x}, \boldsymbol{\lambda}) = \mathrm{E}[y(\boldsymbol{x}, \boldsymbol{\lambda})]$ is the expected simulation output. The stochastic noise is assumed to have a constant variance: $\epsilon \sim N(0, \sigma_\epsilon^2)$. This objective explicitly accounts for the input uncertainty in stochastic simulation optimization, and is less conservative than the robust optimization objective. In the following, we first formulate the GP model for the objective (13) before going into the details of the four refined GP-based optimization algorithms to solve obective (13).

4.2. GP model for g(x)

In order to refine the GP-based optimization algorithms to solve the expected objective (13), a GP model $G(\boldsymbol{x})$ for $g(\boldsymbol{x})$ is required. If the simulation budget is sufficiently large, then direct simulation methods (Chick, 2001; Zouaoui and Wilson, 2003) which require a large number of samples at each design point can be applied to estimate observations of $g(\boldsymbol{x})$. However, this method is not practical for time-consuming simulation models which are the focus of our work. Thus now the simulation outputs on $g(\boldsymbol{x})$ are not directly observable and the available information are the simulation outputs on $f(\boldsymbol{x}, \boldsymbol{\lambda})$. Considering now the bivariate inputs $\boldsymbol{x}$ and $\boldsymbol{\lambda}$ with similar spatial structures, a stochastic GP model $F_n(\boldsymbol{x}, \boldsymbol{\lambda})$ for $f(\boldsymbol{x}, \boldsymbol{\lambda})$ can be fit. In the following, we will first illustrate how to build the stochastic GP model for $f(\boldsymbol{x}, \boldsymbol{\lambda})$, and then propose to use the random process $G_n(\boldsymbol{x}) = \mathrm{E}_{\boldsymbol{\lambda}}[F_n(\boldsymbol{x}, \boldsymbol{\lambda})]$ to model $g(\boldsymbol{x}) = \mathrm{E}_{\boldsymbol{\lambda}}[f(\boldsymbol{x}, \boldsymbol{\lambda})]$.

4.2.1. Stochastic GP model of f(x, λ)

Suppose the initial sample points are $\{(\boldsymbol{x_1}, \boldsymbol{\lambda_1}), (\boldsymbol{x_2}, \boldsymbol{\lambda_2}), ..., (\boldsymbol{x_n}, \boldsymbol{\lambda_n})\}$. At each sample point $(\boldsymbol{x_i}, \boldsymbol{\lambda_i})$, m replicates of the simulation runs are carried out. The observed simulation output sample mean vector is $\bar{\boldsymbol{Y}}_{\boldsymbol{n}} = [\bar{y}(\boldsymbol{x_1}, \boldsymbol{\lambda_1}), ..., \bar{y}(\boldsymbol{x_i}, \boldsymbol{\lambda_i}), ..., \bar{y}(\boldsymbol{x_n}, \boldsymbol{\lambda_n})]^T$. Following the results in Equations (6) and (7), a stochastic GP model $F_n(\boldsymbol{x}, \boldsymbol{\lambda}) \triangleq F|\bar{\boldsymbol{Y}}_{\boldsymbol{n}} \sim \mathrm{GP}(m_n, k_n)$ for $f(\boldsymbol{x}, \boldsymbol{\lambda})$ can be obtained with

$$m_n(\boldsymbol{x}, \boldsymbol{\lambda}) = \tau^2 R_F((\boldsymbol{x}, \boldsymbol{\lambda}), \cdot; \boldsymbol{\theta})^T \big[\tau^2 R_F(\boldsymbol{\theta}) + \boldsymbol{\Sigma}_{\boldsymbol{\epsilon}}\big]^{-1} \bar{\boldsymbol{Y}}_{\boldsymbol{n}}, \tag{14}$$

$$\begin{aligned} k_n\big((\boldsymbol{x}, \boldsymbol{\lambda}), (\boldsymbol{x}', \boldsymbol{\lambda}')\big) &= \tau^2 R_F\big((\boldsymbol{x}, \boldsymbol{\lambda}), (\boldsymbol{x}', \boldsymbol{\lambda}'); \boldsymbol{\theta}\big) \\ &- \tau^4 R_F((\boldsymbol{x}, \boldsymbol{\lambda}), \cdot; \boldsymbol{\theta})^T \big[\tau^2 R_F(\boldsymbol{\theta}) + \boldsymbol{\Sigma}_{\boldsymbol{\epsilon}}\big]^{-1} R_F\big((\boldsymbol{x}', \boldsymbol{\lambda}'), \cdot; \boldsymbol{\theta}\big), \end{aligned} \tag{15}$$

Again, the conditional mean $m_n(\boldsymbol{x}, \boldsymbol{\lambda})$ is used to predict $f(\boldsymbol{x}, \boldsymbol{\lambda})$. $k_n((\boldsymbol{x}, \boldsymbol{\lambda}), (\boldsymbol{x}', \boldsymbol{\lambda}'))$ is the covariance between any two points. $\boldsymbol{\Sigma}_{\boldsymbol{\epsilon}} = \mathrm{Diag}\{\frac{1}{m}\sigma_\epsilon^2 \ \cdots, \frac{1}{m}\sigma_\epsilon^2\}$ is the $n \times n$ covariance matrix of ϵ. Compared with Equations (6) and (7), we now have the uncertain input parameter $\boldsymbol{\lambda}$ in Equations (14) and (15).

We explicitly model the input parameter in our model of the mean response in order to propagate the input uncertainty onto the output mean as seen in the objective of Equation (13). This model form that incorporates the uncertain input parameter is similar to the form in Xie *et al.* (2014), but as their focus is only on quantifying overall uncertainty and not in system optimization, they do not explicitly model the design parameter in their model.

4.2.2. Approximated GP model for g(x)

With the stochastic GP model $F_n(\boldsymbol{x}, \boldsymbol{\lambda})$ for $f(\boldsymbol{x}, \boldsymbol{\lambda})$ obtained, we consider the random process $G_n(\boldsymbol{x}) = \mathrm{E}_{\boldsymbol{\lambda}}[F_n(\boldsymbol{x}, \boldsymbol{\lambda})]$ as a model for $g(\boldsymbol{x}) = \mathrm{E}_{\boldsymbol{\lambda}}[f(\boldsymbol{x}, \boldsymbol{\lambda})]$:

$$G_n(\boldsymbol{x}) = \mathrm{E}_{\boldsymbol{\lambda}}[F_n(\boldsymbol{x}, \boldsymbol{\lambda})] = \int_{\boldsymbol{\lambda}} F_n(\boldsymbol{x}, \boldsymbol{\lambda}) \cdot p(\boldsymbol{\lambda}|\boldsymbol{D_h}) d\boldsymbol{\lambda}. \tag{16}$$

As discussed at the beginning of Section 4, we characterize the input uncertainty with the posterior $p(\boldsymbol{\lambda}|\boldsymbol{D_h})$. It can be

shown that $G_n(\boldsymbol{x})$ is a GP by rewriting the integral as the limit of Riemann sums (De Oliveira and Kone, 2015). The mean and covariance of $G_n(\boldsymbol{x})$ can be derived as follows:

$$\begin{aligned}\mathrm{E}[G_n(\boldsymbol{x})] &= \int_{\boldsymbol{\lambda}} \mathrm{E}[F_n(\boldsymbol{x},\boldsymbol{\lambda})]\cdot p(\boldsymbol{\lambda}|\boldsymbol{D_h})d\boldsymbol{\lambda} \\ &= \int_{\boldsymbol{\lambda}} m_n(\boldsymbol{x},\boldsymbol{\lambda})\cdot p(\boldsymbol{\lambda}|\boldsymbol{D_h})d\boldsymbol{\lambda}\end{aligned} \tag{17}$$

$$\begin{aligned}&\mathrm{Cov}\big[G_n(\boldsymbol{x}), G_n(\boldsymbol{x}')\big] \\ &= \int_{\boldsymbol{\lambda}}\int_{\boldsymbol{\lambda}'} p(\boldsymbol{\lambda}|\boldsymbol{D_h})p(\boldsymbol{\lambda}'|\boldsymbol{D_h})k_n\big((\boldsymbol{x},\boldsymbol{\lambda}),(\boldsymbol{x}',\boldsymbol{\lambda}')\big)d\boldsymbol{\lambda}'d\boldsymbol{\lambda}.\end{aligned} \tag{18}$$

In the following, we discuss a special case of exact evaluation of Equations (17), (18) and describe a Monte Carlo approach for the general case.

Proposition 4.1. *With a prior* $\boldsymbol{\lambda}\sim\mathrm{Dirichlet}(\boldsymbol{\alpha}=\mathbf{1})$ $(\boldsymbol{\alpha}=[\alpha_1,\ldots,\alpha_K]^T, \boldsymbol{\lambda}=[\lambda_1,\ldots,\lambda_K]^T$ *with* $\boldsymbol{\lambda}\in[0,1]^K, \sum_{k=1}^K\lambda_k=1)$, *and a categorical distribution likelihood with parameter* $\boldsymbol{\lambda}$ *for the real world data in* $\boldsymbol{D_h}=\{\boldsymbol{\xi_1},\ldots,\boldsymbol{\xi_h}\}$ *(use dummy coding to represent* $\boldsymbol{\xi_i}$ *as a K-dimensional vector), then the posterior of* $\boldsymbol{\lambda}$ *is* $\mathrm{Dirichlet}(\boldsymbol{\alpha}')$ *with* $\boldsymbol{\alpha}'=\boldsymbol{\alpha}+\sum_{i=1}^h\boldsymbol{\xi_i}$, *i.e.*

$$p(\boldsymbol{\lambda}|\boldsymbol{D_h})=\frac{\prod_{k=1}^K\lambda_k^{\alpha'_k-1}}{\mathrm{B}(\boldsymbol{\alpha}')},$$

and the prior covariance function for the GP model is assumed to be the exponential covariance function:

$$\begin{aligned}&\tau^2R_F((\boldsymbol{x},\boldsymbol{\lambda}),(\boldsymbol{x}',\boldsymbol{\lambda}');\boldsymbol{\theta}=[\boldsymbol{\theta_1}^T,\boldsymbol{\theta_2}^T]^T) \\ &\qquad=\tau^2\exp\left(-\frac{|\boldsymbol{x}-\boldsymbol{x}'|}{2\boldsymbol{\theta}_1^2}-\frac{|\boldsymbol{\lambda}-\boldsymbol{\lambda}'|}{2\boldsymbol{\theta}_2^2}\right).\end{aligned}$$

Then model (16) is a GP with

$$\begin{aligned}\mathrm{E}[G_n(\boldsymbol{x})] &= \frac{1}{B(\boldsymbol{\alpha}')}\sum_{i=1}^n C_i(\boldsymbol{x})\prod_{k=1}^K\int_{\lambda_k}\exp\left(\frac{-|\lambda_k-\lambda_{i,k}|}{2\theta_{2,k}^2}\right)\lambda_k^{\alpha'_k-1}d\lambda_k \\ \mathrm{Cov}\big[G_n(\boldsymbol{x}),G_n(\boldsymbol{x}')\big] &= \frac{\tau^2}{B(\boldsymbol{\alpha}')^2}\exp\left(\frac{-|\boldsymbol{x}-\boldsymbol{x}'|}{2\boldsymbol{\theta}_1{}^2}\right)\prod_{k=1}^K\int_{\lambda_k}\int_{\lambda'_k}\lambda_k^{\alpha'_k-1}\lambda_k'^{\alpha'_k-1}\exp\left(\frac{-|\lambda_k-\lambda'_k|}{2\theta_{2,k}^2}\right)d\lambda_k d\lambda'_k \\ &\quad-\frac{1}{B(\boldsymbol{\alpha}')^2}\sum_{j=1}^n\sum_{i=1}^n c_{ij}\exp\left(\frac{-|\boldsymbol{x}-\boldsymbol{x_i}|-|\boldsymbol{x}'-\boldsymbol{x_j}|}{2\boldsymbol{\theta}_1{}^2}\right) \\ &\quad\prod_{k=1}^K\left\{\int_{\lambda_k}\exp\left(-\frac{|\lambda_k-\lambda_{i,k}|}{2\theta_{2,k}^2}\right)\lambda_k^{\alpha'_k-1}d\lambda_k\int_{\lambda'_k}\exp\left(-\frac{|\lambda'_k-\lambda_{j,k}|}{2\theta_{2,k}^2}\right)\lambda_k'^{\alpha'_k-1}d\lambda'_k\right\}\end{aligned}$$

where λ_k *is the kth dimension of the input parameter* $\boldsymbol{\lambda}$, $\lambda_{i,k}$ *is the kth dimension of the sample* $\boldsymbol{\lambda_i}$.

$$C_i(x)=\exp\left(-\frac{|\boldsymbol{x}-\boldsymbol{x_i}|}{2\boldsymbol{\theta}_1{}^2}\right)\left(\tau^2[\tau^2R_F(\boldsymbol{\theta})+\boldsymbol{\Sigma}_\epsilon]^{-1}\bar{\boldsymbol{Y}}_{\boldsymbol{n}}\right)_i$$

and $c_{ij}=(\tau^4[\tau^2R_F(\boldsymbol{\theta})+\boldsymbol{\Sigma}_\epsilon]^{-1})_{ij}$.

The analytical form of the mean and covariance function in Proposition 4.1 can be derived using the integral

$$\int_0^\lambda\exp(c\lambda)\lambda^n d\lambda=e^{c\lambda}\sum_{p=0}^n(-1)^{n-p}\frac{n!}{p!c^{n-p+1}}\lambda^p,$$

for $c\neq0$ and $n\geq0$ several times and the detailed proof is provided in Online Supplement Appendix A. The special case in Proposition 4.1 is applicable for simulation of systems with categorical real world data such as random sampling outcomes (accept/reject) or (Type I/Type II/Type III).

For general covariance functions and distributions on $\boldsymbol{\lambda}$, the integration in Equations (17) and (18) can be done numerically. Specifically, samples of $\{\boldsymbol{\lambda_1},\ldots,\boldsymbol{\lambda_{N_{MC}}}\}$ can be generated from $p(\boldsymbol{\lambda}|\boldsymbol{D_h})$ to compute

$$\mathrm{E}[G_n(\boldsymbol{x})]\approx\mu_n(\boldsymbol{x})=\frac{1}{N_{MC}}\sum_{i=1}^{N_{MC}}m_n(\boldsymbol{x},\boldsymbol{\lambda_i}), \tag{19}$$

$$\begin{aligned}&\mathrm{Cov}\big[G_n(\boldsymbol{x}),G_n(\boldsymbol{x}')\big]\approx c_n(\boldsymbol{x},\boldsymbol{x}') \\ &\qquad=\frac{1}{N_{MC}^2}\sum_{i=1}^{N_{MC}}\sum_{j=1}^{N_{MC}}k_n\big((\boldsymbol{x},\boldsymbol{\lambda_i}),(\boldsymbol{x}',\boldsymbol{\lambda_j})\big),\end{aligned} \tag{20}$$

$\mathrm{Var}[G_n(\boldsymbol{x})]\approx\sigma_n^2(\boldsymbol{x})=c_n(\boldsymbol{x},\boldsymbol{x})$. In fact, we have

$$\mu_n(\boldsymbol{x})=\mathrm{E}\left[\frac{1}{N_{MC}}\sum_{i=1}^{N_{MC}}F_n(\boldsymbol{x},\boldsymbol{\lambda_i})\right]$$

and

$$c_n(\boldsymbol{x},\boldsymbol{x}')=\mathrm{Cov}\left(\frac{1}{N_{MC}}\sum_{i=1}^{N_{MC}}F_n(\boldsymbol{x},\boldsymbol{\lambda_i}),\frac{1}{N_{MC}}\sum_{j=1}^{N_{MC}}F_n(\boldsymbol{x}',\boldsymbol{\lambda_j})\right).$$

Furthermore, as

$$\hat{G}_n(\boldsymbol{x})=\frac{1}{N_{MC}}\sum_{i=1}^{N_{MC}}F_n(\boldsymbol{x},\boldsymbol{\lambda_i}) \tag{21}$$

is a finite sum of Gaussian random variable, $\hat{G}_n(\boldsymbol{x})$ is also a GP, with mean $\mu_n(\boldsymbol{x})$ and covariance $c_n(\boldsymbol{x},\boldsymbol{x}')$. The above

Monte Carlo approximation $\hat{G}_n(\boldsymbol{x})$ can then be used to approximate $G_n(\boldsymbol{x})$. When non-conjugate priors are applied resulting in complicated posteriors, Markov Chain Monte Carlo (MCMC) techniques can be applied to sample $\boldsymbol{\lambda}$. Brooks *et al.* (2011) and Hoffman and Gelman (2014) have provided various fast MCMC techniques to sample from the posterior. As with all Monte Carlo approximations, the sample size N_{MC} influences the approximation error and generally, the approximation error will converge to zero at the rate of $O(\frac{1}{\sqrt{N_{MC}}})$ (Neal, 1993). Convergence diagnostics should be applied to ensure convergence of the estimate (Robert and Casella, 2004). Several variance reduction techniques can also be used to improve the accuracy (Robert and Casella, 2004). In our examples in Section 5, we use $N_{MC} = 100$ as recommended in Neal (1993) to balance the accuracy and the computation efficiency.

4.3. GP-based algorithms with input uncertainty to select x_{n+1}

In Section 3.2, the details of EGO, KG, IAGO and EEV are reviewed for both deterministic and stochastic simulation optimization problems where the input parameter is assumed to be known. In the following, we will refine these four GP-based algorithms to consider input uncertainty for stochastic simulation problems to solve Equation (13) with the approximated GP model $\hat{G}_n(\boldsymbol{x})$. For the special case (Proposition 4.1), the exact analytical form of $G(\boldsymbol{x})$ can be applied.

4.3.1. EGO with input uncertainty

The EGO algorithm with EI focuses on the improvement of the objective function value. In order to refine EI for the objective function $g(\boldsymbol{x})$, we need to specify the current best value of $g(\boldsymbol{x})$ and the predictive distribution of $g(\boldsymbol{x}_{n+1})$ at any candidate point $\boldsymbol{x}_{n+1}$. Here, we estimate the predictive distribution at $\boldsymbol{x}_{n+1}$ with the normal random variable $\hat{G}_n(\boldsymbol{x}_{n+1})$ obtained from Equation (21). Hence, we can define the EI-based infill criterion with input uncertainty as

$$\mathrm{EI}_T(\boldsymbol{x}_{n+1}) = \mathrm{E}_{\hat{G}_n}\left[\left(T - \hat{G}_n(\boldsymbol{x}_{n+1})\right)^+ | \bar{\boldsymbol{Y}}_{\boldsymbol{n}}\right],$$

where T is the current best value for $g(\boldsymbol{x})$. To determine T, we draw parallels from the extension of EGO to the stochastic case in Section 3.2.1, and choose $T = \min\{\mu_n(\boldsymbol{x}_1), \ldots, \mu_n(\boldsymbol{x}_n)\}$. The expectation is taken with respect to $\hat{G}_n(\boldsymbol{x}_{n+1})$, and thus the next design point is selected with $\mathrm{argmax}_{\boldsymbol{x}_{n+1}} \mathrm{EI}_T(\boldsymbol{x}_{n+1})$.

4.3.2. KG with input uncertainty

Adapting KG to solve Equation (13), the updated and the current conditional mean for $g(\boldsymbol{x})$ need to be specified. Here, we use the current and updated conditional mean in the improvement function with that from $\hat{G}_n(\boldsymbol{x})$. Hence, the refined KG infill criterion for any candidate point $\boldsymbol{x}_{n+1}$ is

$$\begin{aligned}\mathrm{E}_Z\left[\mathrm{I}_n(\boldsymbol{x}_{n+1})\right] &= \mathrm{E}_Z\left[\min_{\boldsymbol{x}} \mu_n(\boldsymbol{x}) - \min_{\boldsymbol{x}} \mu_{n+1}(\boldsymbol{x})\right] \\ &= \min_{\boldsymbol{x}} \mu_n(\boldsymbol{x}) - \mathrm{E}_Z\left[\min_{\boldsymbol{x}}\left(\mu_n(\boldsymbol{x}) + \frac{c_n(\boldsymbol{x}, \boldsymbol{x}_{n+1})}{\sigma_n(\boldsymbol{x})} Z\right)\right],\end{aligned}$$

where $\mu_{n+1}(\boldsymbol{x})$ can be calculated by the GP updating formula, and the next design point is selected as the point that can best improve the minimum of conditional mean with $\mathrm{argmax}_{\boldsymbol{x}_{n+1}} \mathrm{E}_Z[\mathrm{I}_n(\boldsymbol{x}_{n+1})]$.

4.3.3. IAGO with input uncertainty

For the IAGO approach, we now aim to quantify the uncertainty in $\boldsymbol{x}^* = \mathrm{argmin}_{\boldsymbol{x}} g(\boldsymbol{x})$ using the entropy measure. Here, the expected conditional entropy of $\boldsymbol{x}^*$ is $H(\boldsymbol{x}^* | \bar{\boldsymbol{Y}}_{\boldsymbol{n}}, \hat{G}_n(\boldsymbol{x}_{n+1}))$, where the conditions are $\bar{\boldsymbol{Y}}_{\boldsymbol{n}}$ (the observed stochastic simulation output mean on $f(\boldsymbol{x}, \boldsymbol{\lambda})$), and $\hat{G}_n(\boldsymbol{x}_{n+1})$ (the predictive distribution of $g(\boldsymbol{x})$ at $\boldsymbol{x}_{n+1}$). With the quantization technique described in Section 3.2.3, we can approximate it with

$$\begin{aligned}H\left(\boldsymbol{x}^* | \bar{\boldsymbol{Y}}_{\boldsymbol{n}}, \hat{G}_n(\boldsymbol{x}_{n+1})\right) &= \sum_{i=1}^{M} \frac{1}{M} H\left(\boldsymbol{x}^* | \bar{\boldsymbol{Y}}_{\boldsymbol{n}}, \hat{G}_n(\boldsymbol{x}_{n+1})\right) = \gamma_i) \\ &= -\frac{1}{M} \sum_{i=1}^{M} \sum_{\boldsymbol{x} \in \mathbb{D}} P_{\boldsymbol{x}^* | \bar{\boldsymbol{Y}}_{\boldsymbol{n}}, \gamma_i}(\boldsymbol{x}) \log_2 P_{\boldsymbol{x}^* | \bar{\boldsymbol{Y}}_{\boldsymbol{n}}, \gamma_i}(\boldsymbol{x}).\end{aligned}$$

To evaluate the conditional distribution $P_{\boldsymbol{x}^* | \bar{\boldsymbol{Y}}_{\boldsymbol{n}}, \gamma_i}(\boldsymbol{x})$, we propose to approximate it through the distribution of the minimizers of the conditional sample paths from $\hat{G}|_{\bar{\boldsymbol{Y}}_{\boldsymbol{n}}, \gamma_i} \triangleq \hat{G}_n|_{\hat{G}_n(\boldsymbol{x}_{n+1}) = \gamma_i}$. Generating conditional sample paths from $\hat{G}|_{\bar{\boldsymbol{Y}}_{\boldsymbol{n}}, \gamma_i}$ is equivalent to constructing conditional sample paths from $\hat{G}_n$, satisfying the condition $\hat{G}_n(\boldsymbol{x}_{n+1}) = \gamma_i$. We use the "Conditioning by Kriging" approach (Villemonteix *et al.*, 2009) to transform the sample paths from $\hat{G}_n$ to sample paths of $\hat{G}|_{\bar{\boldsymbol{Y}}_{\boldsymbol{n}}, \gamma_i}$. For computational savings, we can generate conditional sample paths through Eigen-decomposition of the covariance matrix (Gentle, 2009), which requires the covariance matrix of $\hat{G}_n$ to be positive semi-definite. In fact, the covariance matrix of $\hat{G}_n$ is positive semi-definite, as the covariance function of F is positive semi-definite. With the expected conditional entropy approximated, the next design point that best reduces the uncertainty in $\boldsymbol{x}^*$ is selected with $\mathrm{argmin}_{\boldsymbol{x}_{n+1}} H(\boldsymbol{x}^* | \bar{\boldsymbol{Y}}_{\boldsymbol{n}}, \hat{G}_n(\boldsymbol{x}_{n+1}))$.

4.3.4. EEV with input uncertainty

In order to adopt the EEV approach to solve Equation (13), we need to determine the current best of the objective $g(\boldsymbol{x})$ so that the integrated probability of improvement can be computed. With the extension of EEV to stochastic simulation in Section 3.2.4, here we can formulate the criterion to account for input uncertainty as

$$ev_{n+1}(\boldsymbol{x}_{n+1}) = \int_{\mathbb{X}} P\left(\hat{G}_{n+1}(\boldsymbol{x}) \le \min\left\{T, \hat{G}_n(\boldsymbol{x}_{n+1})\right\}\right) d\boldsymbol{x},$$

where $T = \min\{\mu_n(\boldsymbol{x}_1), \ldots, \mu_n(\boldsymbol{x}_n)\}$ is used to estimate the current best of $g(\boldsymbol{x})$. $\hat{G}_{n+1}(\boldsymbol{x})$ is the updated approximated GP at any candidate point $\boldsymbol{x}_{n+1}$. Similarly, the next design point that can reduce the expected uncertainty measure most is selected with $\mathrm{argmin}_{\boldsymbol{x}_{n+1}} \mathrm{E}_{\hat{G}_n}[ev_{n+1}(\boldsymbol{x}_{n+1})]$.

4.4. Selection of next input point λ_{n+1}

In the above, we formulated the infill criteria to select the next design point $\boldsymbol{x_{n+1}}$ for the simulation optimization problems that consider input uncertainty. However, the selection of next input point $\boldsymbol{\lambda_{n+1}}$ is also required in order to execute the simulation at $(\boldsymbol{x_{n+1}}, \boldsymbol{\lambda_{n+1}})$ and update the GP model to $F_{n+1}(\boldsymbol{x}, \boldsymbol{\lambda})$. It should be noted that Pearce and Branke (2017) also used a joint bivariate $(\boldsymbol{x}, \boldsymbol{\lambda})$ stochastic GP model for $f(\boldsymbol{x}, \boldsymbol{\lambda})$, but formulated their infill criteria to jointly select $\boldsymbol{x_{n+1}}$ and $\boldsymbol{\lambda_{n+1}}$. To account for the uncertainty of $\boldsymbol{\lambda}$, the posterior predictive distribution for $g(\boldsymbol{x})$ with respect to $\boldsymbol{\lambda}$ is obtained through Monte Carlo sampling from the distribution of $\boldsymbol{\lambda}$ within the infill criteria.

Here, we refined the GP algorithms with a two-step framework instead, by first choosing next design point $\boldsymbol{x_{n+1}}$ through our formulated infill criteria in Section 4.3, and then separately choosing the next input point $\boldsymbol{\lambda_{n+1}}$. We take this two-step approach as we see the purpose of sampling $\boldsymbol{\lambda_{n+1}}$ is to improve the model approximation accuracy for $g(\boldsymbol{x})$. The improvement of the model approximation accuracy will, in turn, benefit the performance of metamodel-based algorithms (Quan *et al.*, 2013). From Equation (21), we see that the approximation surface of $g(\boldsymbol{x})$ is the Monte Carlo (MC) results of the approximation surface of $f(\boldsymbol{x}, \boldsymbol{\lambda})$. Therefore, a good approximation surface for $f(\boldsymbol{x}, \boldsymbol{\lambda})$ can provide a good approximation surface for $g(\boldsymbol{x})$. As $\boldsymbol{\lambda_{n+1}}$ affects the updated approximation surface of $f(\boldsymbol{x}, \boldsymbol{\lambda})$ as well as $g(\boldsymbol{x})$, we treat it separately to address its impact on the approximation result of $g(\boldsymbol{x})$. Moreover, from a computational perspective, our two-step framework will reduce the dimension of the candidate search space to the separate $\boldsymbol{x}$ and $\boldsymbol{\lambda}$ spaces over the joint $\boldsymbol{x}$, $\boldsymbol{\lambda}$ space, and hence, reduce the number of Monte Carlo approximations required.

In the following, we will propose three possible ways of choosing $\boldsymbol{\lambda_{n+1}}$.

4.4.1. Random sampling to select λ_{n+1}

In the Random (RA) approach, $\boldsymbol{\lambda_{n+1}}$ is sampled directly from the posterior distribution $P(\boldsymbol{\lambda}|\boldsymbol{D_h})$. The idea here is to select $\boldsymbol{\lambda_{n+1}}$ with the purpose of improving the approximation surface of $f(\boldsymbol{x}, \boldsymbol{\lambda})$ at places where the MC samples of $\boldsymbol{\lambda}$ are more likely to occur in Equation (21) to approximate $g(\boldsymbol{x})$. If the posterior distribution is complicated (potentially arising from non-conjugate priors), Rejection Sampling, Sampling Importance Resampling, or MCMC algorithms (Brooks *et al.*, 2011) may be required.

4.4.2. Integrated mean squared error approach to selecting λ_{n+1}

A popular criterion for improving the global fitting accuracy of GP models is the Integrated Mean Squared Error (IMSE). Here, the IMSE can be applied for the selection of $\boldsymbol{\lambda_{n+1}}$ to best improve the global approximation surface of $f(\boldsymbol{x}, \boldsymbol{\lambda})$. Although the IMSE for the GP models does not have a closed-form, numerical methods such as quasi-Newton optimizer are commonly used (Sacks *et al.*, 1989). To ease computation, we apply a closed-form approximation method proposed by Binois *et al.* (2019). Recall the GP model $F_n(\boldsymbol{x}, \boldsymbol{\lambda})$ for $f(\boldsymbol{x}, \boldsymbol{\lambda})$ obtained from Section 4.2.1, the IMSE is then defined as

$$\text{IMSE}_n = \int_{\boldsymbol{\lambda}}\int_{\boldsymbol{x}} s_n^2(\boldsymbol{x}, \boldsymbol{\lambda}) d\boldsymbol{x} d\boldsymbol{\lambda}, \tag{22}$$

where $s_n^2(\boldsymbol{x}, \boldsymbol{\lambda})$ is the conditional variance of $F_n(\boldsymbol{x}, \boldsymbol{\lambda})$ at point $(\boldsymbol{x}, \boldsymbol{\lambda})$. If covariance function parameters $\boldsymbol{\theta}$, τ, and variance of stochastic noise σ_ϵ^2 are known, IMSE_n can be written as:

$$\text{IMSE}_n = E - tr\left[(\tau^2 R_F(\boldsymbol{\theta}) + \frac{1}{m}\sigma_\epsilon^2 \boldsymbol{I})^{-1} \boldsymbol{W_n}\right],$$

where $E = \int_{\boldsymbol{x}}\int_{\boldsymbol{\lambda}} k_n(\boldsymbol{x}, \boldsymbol{\lambda}) d\boldsymbol{x} d\boldsymbol{\lambda}$ and $\boldsymbol{W_n}$ is the $n \times n$ matrix with entries being the integral of kernel products:

$$w((\boldsymbol{x_i}, \boldsymbol{\lambda_i}), (\boldsymbol{x_j}, \boldsymbol{\lambda_j})) = \int_{\boldsymbol{x}}\int_{\boldsymbol{\lambda}} k_n((\boldsymbol{x_i}, \boldsymbol{\lambda_i}), (\boldsymbol{x}, \boldsymbol{\lambda})) \cdot k_n((\boldsymbol{x_j}, \boldsymbol{\lambda_j}), (\boldsymbol{x}, \boldsymbol{\lambda})) d\boldsymbol{x} d\boldsymbol{\lambda}.$$

Then the IMSE after evaluating at any candidate point of $\boldsymbol{\lambda}$ is

$$\begin{aligned}\text{IMSE}_{n+1}(\boldsymbol{\lambda}) = \text{IMSE}_n - &\left[s_n^2(\boldsymbol{x_{n+1}}, \boldsymbol{\lambda}) h(\boldsymbol{x_{n+1}}, \boldsymbol{\lambda})^T \boldsymbol{W_n} h(\boldsymbol{x_{n+1}}, \boldsymbol{\lambda})\right] \\ &+ 2w(\boldsymbol{x_{n+1}}, \boldsymbol{\lambda})^T h(\boldsymbol{x_{n+1}}, \boldsymbol{\lambda}) \\ &+ s_n^2(\boldsymbol{x_{n+1}}, \boldsymbol{\lambda})^{-1} w((\boldsymbol{x_{n+1}}, \boldsymbol{\lambda}), (\boldsymbol{x_{n+1}}, \boldsymbol{\lambda}))),\end{aligned}$$

where

$$h(\boldsymbol{x}, \boldsymbol{\lambda}) = -s_n^2(\boldsymbol{x}, \boldsymbol{\lambda})^{-1} (\tau^2 R_F(\boldsymbol{\theta}) + \frac{1}{m}\sigma_\epsilon^2 I)^{-1} k_n(\boldsymbol{x}, \boldsymbol{\lambda}).$$

Hence, the next input point that can improve the integrated mean square error of the GP model best is selected as $\boldsymbol{\lambda_{n+1}} = \text{argmin}_{\boldsymbol{\lambda}} \text{IMSE}_{n+1}(\boldsymbol{\lambda})$.

4.4.3. Distribution IMSE to select λ_{n+1}

The aforementioned IMSE criterion for $\boldsymbol{\lambda_{n+1}}$ in Section 4.4.2 assumes that all the candidate input points are uniformly distributed. In other words, we treat the model approximation accuracy of $f(\boldsymbol{x}, \boldsymbol{\lambda})$ throughout the input space $\boldsymbol{\lambda}$ with equal importance. An alternative approach is to incorporate the posterior distribution of $\boldsymbol{\lambda}$ into the criterion. This intuitively will give more importance to the MSE in areas where the density of $\boldsymbol{\lambda}$ is high, and these are precisely the areas that will appear more often in the MC samples used for approximating $g(\boldsymbol{x})$. In light of this, we propose a distribution IMSE (DI), aiming at improving approximation surface of $f(\boldsymbol{x}, \boldsymbol{\lambda})$ in areas where the density of $\boldsymbol{\lambda}$ is higher. We define this distribution IMSE as

$$DI_n = \int_{\boldsymbol{\lambda}}\int_{\boldsymbol{x}} s_n^2(\boldsymbol{x}, \boldsymbol{\lambda}) \cdot p(\boldsymbol{\lambda}|\boldsymbol{D_h}) d\boldsymbol{x} d\boldsymbol{\lambda}, \tag{23}$$

Then $\boldsymbol{\lambda_{n+1}}$ that minimizes the distribution IMSE is selected as $\boldsymbol{\lambda_{n+1}} = \text{argmin}_{\boldsymbol{\lambda}} \mathbf{DI_{n+1}}(\boldsymbol{\lambda})$.

Algorithm 1 below outlines the general steps for the GP-based algorithms with input uncertainty.

Algorithm 1. GP-Based Optimization Algorithms with Input Uncertainty

1. Given i.i.d. real world data $\boldsymbol{D_h} = \{\xi_1, \xi_2, ..., \xi_h\}$ of size h, and the prior $p(\boldsymbol{\lambda})$, derive posterior $p(\boldsymbol{\lambda}|\boldsymbol{D_h}) \propto p(\boldsymbol{\lambda}) \cdot p(\boldsymbol{D_h}|\boldsymbol{\lambda}) = p(\boldsymbol{\lambda}) \cdot \Pi_{i=1}^{h} \pi_0(\xi_i; \boldsymbol{\lambda})$;
2. **Initialization**: Generate $\{\boldsymbol{x_1}, \boldsymbol{x_2}, ..., \boldsymbol{x_n}\}$ using Latin Hypercube Sampling from Uniform distribution with support $\mathscr{X}$ and generate $\{\boldsymbol{\lambda_1}, \boldsymbol{\lambda_2}, ..., \boldsymbol{\lambda_n}\}$ using Latin Hypercube Sampling from $p(\boldsymbol{\lambda}|\boldsymbol{D_h})$ to obtain an initial set of sample points $\{(\boldsymbol{x_1}, \boldsymbol{\lambda_1}), (\boldsymbol{x_2}, \boldsymbol{\lambda_2}), ..., (\boldsymbol{x_n}, \boldsymbol{\lambda_n})\}$;
3. Run the simulation experiment at the initial points with m replications at each point and obtain the observed output sample mean vector $\bar{\boldsymbol{Y}}_{\boldsymbol{n}} = [\bar{y}(\boldsymbol{x_1}, \boldsymbol{\lambda_1}), ..., \bar{y}(\boldsymbol{x_i}, \boldsymbol{\lambda_i}), ..., \bar{y}(\boldsymbol{x_n}, \boldsymbol{\lambda_n})]^T$;
4. **Validation**: Based on $\bar{\boldsymbol{Y}}_{\boldsymbol{n}}$, construct a stochastic GP model $F_n(\boldsymbol{x}, \boldsymbol{\lambda})$; Perform cross-validation (e.g. Leave-one-out cross validation) to ensure that the metamodel is valid;
5. **while** $r \leq N-n$ **do** ▷ $N * m$ is the total budget
6. Generate $\boldsymbol{\lambda_1}, ..., \boldsymbol{\lambda}_{N_{MC}}$ from $p(\boldsymbol{\lambda}|\boldsymbol{D_h})$; Derive the predictive distribution $\hat{G}_n(\boldsymbol{x})$ for $g(\boldsymbol{x})$ using (19),(20),(21). When the special case applies, use analytical distribution from Proposition 4.1.
7. **Select design point** $\boldsymbol{x}_{n+1}$: use the refined EGO/KG/IAGO/EEV algorithms;
8. **Select input point** $\boldsymbol{\lambda}_{n+1}$: use RA/IMSE/DI criterion;
9. Run simulation experiments at $(\boldsymbol{x}_{n+1}, \boldsymbol{\lambda}_{n+1})$ with m replications and obtain the observed output mean $\bar{y}(\boldsymbol{x_1}, \boldsymbol{\lambda_1})$, set $\bar{\boldsymbol{Y}}_{n+1} = [\bar{\boldsymbol{Y}}_n, \bar{y}(\boldsymbol{x_1}, \boldsymbol{\lambda_1})]^T$;
10. **Update**: update the stochastic GP model $F_n(\boldsymbol{x}, \boldsymbol{\lambda})$ based on $\bar{\boldsymbol{Y}}_{n+1}$;
11. Set $r = r + 1$
12. **return** $\hat{\boldsymbol{x}}^* = \text{argmin}_{\boldsymbol{x}}\, \mu_N(\boldsymbol{x})$

5. Comprehensive comparison of different algorithms

In this section, we aim to comprehensively compare the various GP-based optimization algorithms with and without input uncertainty. Specifically, we are interested in the following research questions:

1. For our proposed algorithms with input uncertainty, algorithms proposed by Pearce and Branke (2017), and algorithms with plug-in input parameter, which one works better and when?
2. For proposed algorithms with input uncertainty, which method to choose the next input point $\boldsymbol{\lambda}_{n+1}$ (RA, IMSE, and DI) works better and when?
3. For the four refined algorithms with input uncertainty, which one works better and when?

5.1. Algorithms considered

We focus on the four types of GP-based optimization algorithms, EGO, KG, IAGO, and EEV. For each type, we test the algorithms with Plug-in Input (PI), and our proposed algorithms with input uncertainty. The PI algorithms are currently the most commonly used approaches in practice and the maximum likelihood estimator is used as the PI $\hat{\boldsymbol{\lambda}}$ for the following experiments. For the algorithms with input uncertainty, we test three different methods to select the next input point $\boldsymbol{\lambda}_{n+1}$ (RA, IMSE, and DI). The three methods only differ in the choice of next input point, and are compared to learn how the selection of the sequential input points can influence the performances. We also include the refined EGO and KG proposed by Pearce and Branke (2017) in the comparison. Overall, $4 \times 4 + 2 = 18$ algorithms are compared.

5.2. Test functions and factors considered

We test the above 18 algorithms with the Branin function, the Eggholder function, and the Hartmann-6 function. These three test functions are selected to cover a variety of problem characteristics and dimensions. As seen in Figure 2, the Branin function is two dimensional and its surface is relatively flat. The two-dimensional Eggholder has a steeper slope surface than the Branin, and is difficult to optimize due to a large number of local minima. The Hartmann-6 function is a higher-dimensional problem (six dimensional). For the Branin and Eggholder functions, we choose one of the two variables as the design parameter $\boldsymbol{x}$ and another one as the input parameter $\boldsymbol{\lambda}$. For the Hartmann-6, we choose four-out-of-the-six variables as the design parameters and another two as the two-dimensional input parameter $\boldsymbol{\lambda}$. The specific settings of the test functions are listed in Table 1. An (s, S) inventory problem is elaborated in the Online Supplement Appendix B, and a three-dimension correlated input problem is studied in Online Supplement Appendix C.

Several papers in the literature have studied the factors that can influence the performance of the GP-based optimization algorithms. The level of stochastic noise can have a large influence (Picheny *et al.*, 2013; Jalali *et al.*, 2017). When input uncertainty is present and considered, the input uncertainty level can be another important factor. Denote the output range of each test function as R, we consider two stochastic noise levels: $\sigma_\epsilon^2 = R \times 0.5\%$ to represent light stochastic noise, and $\sigma_\epsilon^2 = R \times 4\%$ for heavy stochastic noise (Huang *et al.*, 2006). The input uncertainty level is measured by the real world data size h. The larger the h value, the smaller the posterior distribution variance of $\boldsymbol{\lambda}$, and the less input uncertainty associated with the problem. In this

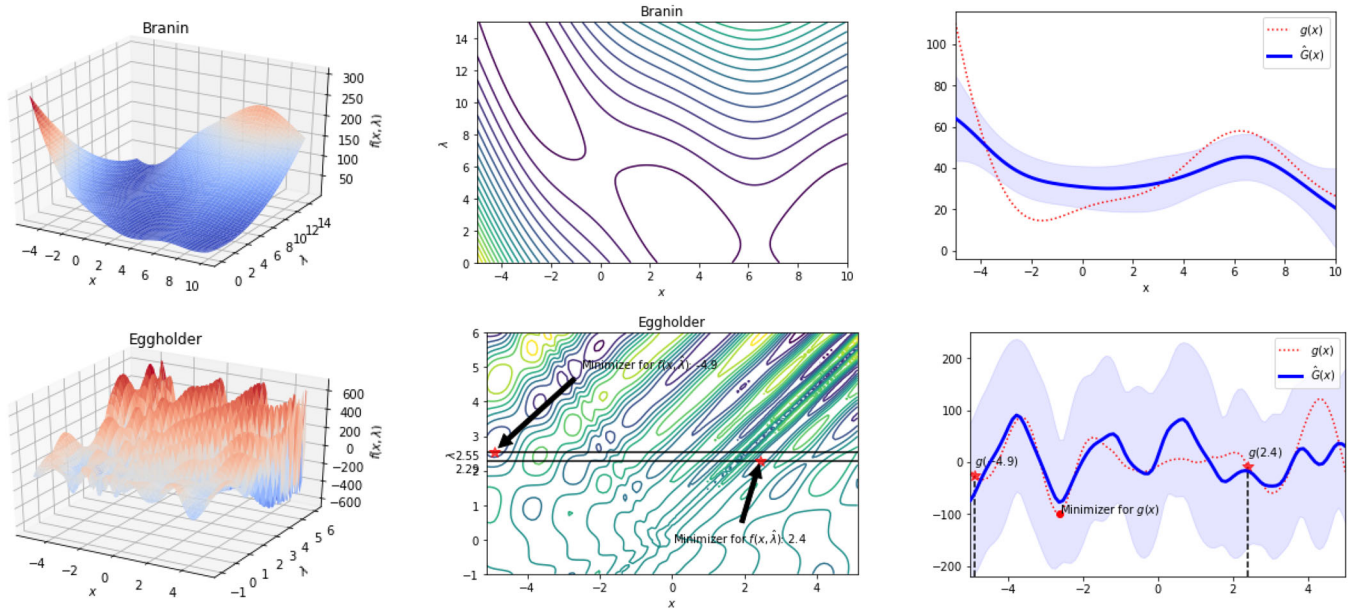

Figure 2. The first column graphs are the surface plots for Branin and Eggholder and the middle column graphs are the respective contour plots. The third column graphs are the initial approximation GP models $\hat{G}_n(\boldsymbol{x})$ for $g(\boldsymbol{x})$ of Branin and Eggholder for one macro-replication in the case of high input uncertainty and heavy stochastic noise.

Table 1. Analytical test functions.

Name	*Function details*
Branin	$f(\boldsymbol{x},\boldsymbol{\lambda}) = (\lambda - \frac{5.1x^2}{4\pi^2} + \frac{5x}{\pi} - 6)^2 + (10 - \frac{10}{8\pi})\cos(\boldsymbol{x}) + 10, \boldsymbol{x} \in [-5, 10]$
Eggholder	$f(\boldsymbol{x},\boldsymbol{\lambda}) = (-\tilde{\boldsymbol{x}} + 47) \cdot \sin(\sqrt{\lvert\tilde{\boldsymbol{x}} + \tilde{\boldsymbol{\lambda}}/2 + 47\rvert}) - \tilde{\boldsymbol{\lambda}} \cdot \sin(\sqrt{\lvert\tilde{\boldsymbol{\lambda}} - (\tilde{\boldsymbol{x}} + 47)\rvert})$ $\tilde{\boldsymbol{x}} = 100\boldsymbol{x}, \tilde{\boldsymbol{\lambda}} = 100\boldsymbol{\lambda}, \boldsymbol{x} \in [-5, 5]$
Hartmann-6	$f(\boldsymbol{x},\boldsymbol{\lambda}) = -\frac{1}{1.94}[2.58 + \sum_{i=1}^{4} \alpha_i \exp(-\sum_{j=1}^{4} A_{ij}(x_j - P_{ij})^2 - \sum_{j=5}^{6} A_{ij}(\lambda_j - P_{ij})^2)]$ α_i, A_{ij}, P_{ij} are fixed values, $x_j \in [0, 1]$, for $j = 1, \ldots, 4$

section, we use $h = 10$ for the high input uncertainty level and $h = 100$ for low input uncertainty.

5.3. Implementation details

In the experiment, we first consider a normal distribution for the real world data $\xi_1, \ldots, \xi_h \overset{i.i.d.}{\sim} N(\boldsymbol{\lambda_0}, 3)$, where $\boldsymbol{\lambda_0}$ is the unknown true input parameter. We use $N(0, 10)$ as the prior distribution for $\boldsymbol{\lambda}$, then the posterior is

$$N\left(\frac{1}{\frac{1}{100} + \frac{h}{9}} \frac{\sum_{i=1}^{h} \xi_i}{9}, \sqrt{\left(\frac{1}{100} + \frac{h}{9}\right)^{-1}}\right).$$

The true values of input parameter for the Branin, Eggholder, and Hartmann-6 are set to be $\boldsymbol{\lambda_0}$ =8, 2.5, 0.5, respectively.

Then we consider the case of $\xi_1, \ldots, \xi_h \overset{i.i.d.}{\sim} \exp(\boldsymbol{\lambda_0})$ (which is not symmetric compared with the normal distribution). In this case, we use the non-informative Jeffreys prior, $p(\boldsymbol{\lambda}) \propto 1/\boldsymbol{\lambda}$, for $\boldsymbol{\lambda}$, and the posterior distribution becomes Gamma$(h, \sum_{i=1}^{h} \xi_i)$. Here, we want to investigate whether the skewness of the real world data distribution will influence the performances of the algorithms.

The initial sample points are selected using Latin Hypercube Sampling with a Maximin criterion, and we apply the commonly used criterion 10-d (10 × dimension of function) to set the initial sample size. For the sequential stage, one single point is selected per iteration. The total simulation budget is set to be 40 for Branin and Eggholder functions, and 100 for the Hartmann-6 function.

In the following numerical examples, we use the stationary square exponential covariance function for the GP model, i.e.

$$\tau^2 R_F((\boldsymbol{x},\boldsymbol{\lambda}),(\boldsymbol{x}',\boldsymbol{\lambda}');\boldsymbol{\theta}) = \tau^2 \exp\left(-\sum_{i=1}^{d} \frac{(x_i - x_i')^2}{2\theta_{1,i}^2} - \sum_{j=1}^{l} \frac{(\lambda_j - \lambda_j')^2}{2\theta_{2,j}^2}\right),$$

where $\boldsymbol{\theta} = [\theta_{1,1}, \ldots, \theta_{1,d}, \theta_{2,1}, \ldots, \theta_{2,l}]^T$. Other stationary and non-stationary classes of covariance functions (Rasmussen and Williams, 2006) can also be applied to capture more complicated dependence of the design and/or input parameters.

5.4. Performance measure

We carry out 100 independent macro-replications to compare the different algorithms. For each macro-replication i, each algorithm starts with the same initial simulation observations. After consuming the total budget N, the true optimal design $\boldsymbol{x}^* = \text{argmin}_{\boldsymbol{x}} g(\boldsymbol{x})$ is approximated by

$$\hat{\boldsymbol{x}}_i^* = \arg\min_{\boldsymbol{x}} \hat{g}(\boldsymbol{x}) \tag{24}$$

where $\hat{g}(\boldsymbol{x}) = \mu_N(\boldsymbol{x})$ is the predictive mean for $g(\boldsymbol{x})$ obtained from the mean of $\hat{G}_N(\boldsymbol{x})$. Define GAP $= |g(\boldsymbol{x}^*) - g(\hat{\boldsymbol{x}}_i^*)|$ as the performance measure that directly measures the gap of the loss obtained from an approximated optimal design $\hat{\boldsymbol{x}}_i^*$. We visualize the GAP for different algorithms using boxplots, and focus on the median of GAP which represents the "typical" performance of each algorithm. We also report and compare the evaluation time for different algorithms.

Table 2. Average evaluation time of different algorithms per iteration in seconds.

Evaluation time(s)	*Branin*	*Eggholder*	*Hartmann-6*	*(s,S) Problem*
EGO_PI	0.76	0.94	2.72	1.06
KG_PI	0.90	1.25	20.48	7.96
EEV_PI	3.74	3.01	55.64	23.08
IAGO_PI	37.76	38.52	139.10	59.55
EGO_RA	2.21	2.58	7.80	4.78
KG_RA	8.12	8.76	102.80	12.48
EEV_RA	10.57	11.89	285.21	43.56
IAGO_RA	40.79	42.52	394.85	98.78
EGO_IMSE	3.45	3.42	10.12	6.04
KG_IMSE	8.80	9.29	110.12	16.72
EEV_IMSE	11.06	12.29	293.55	46.02
IAGO_IMSE	42.56	44.68	400.20	102.53
EGO_DI	4.98	5.26	12.05	7.97
KG_DI	10.24	11.02	111.84	18.69
EEV_DI	12.66	14.07	295.29	47.89
IAGO_DI	44.34	46.71	401.49	104.26
EGO_P&B	12.82	13.23	57.86	25.93
KG_P&B	123.51	125.58	885.72	248.65

5.5. Results and discussion

As the experiment results for the case where the real world data follow an exponential distribution are quite similar to the case of a normal distribution, we here only present the results for the normal case. All the algorithms are run on a server with 8 vCPUs and 52 GB memory. The average evaluation time of different algorithms are recorded in Table 2. The boxplots of GAP for the Eggholder, Branin, and Hartmann-6 are displayed in Figures 3, 4, 5 respectively. We use the median of GAP and the evaluation time to address the research questions above. We use Mood's median test (Mood, 1954) with a significant level of 0.05 to test whether there exist significant differences between the GAP medians of the different algorithms. From here on, "similar performance" implies no significant difference, and "better performance" implies significantly better.

1. For algorithms that consider input uncertainty and their counterparts with PIs, generally, algorithms with input uncertainty usually perform better when input uncertainty level is high. We also find that both algorithm types tend to have similar performances when input uncertainty is low. This can be expected because as input uncertainty gets smaller (real world data size gets larger), the MLE will get closer to the true input parameter, and posterior distribution of the input parameter

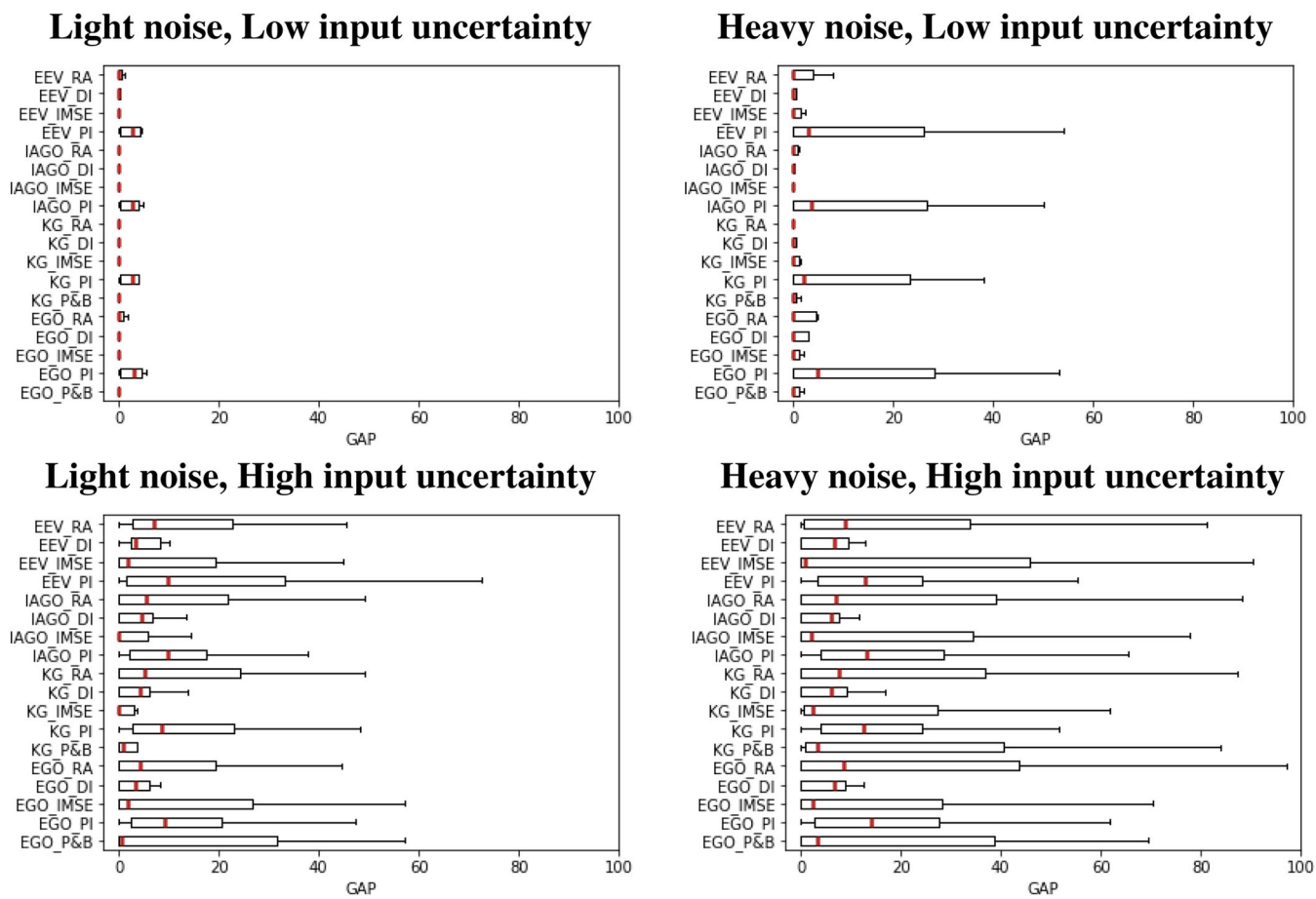

Figure 3. GAP for the Eggholder function for different algorithms. The bold lines inside the boxes are the median values; the edges of the boxes are the 25th and 75th percentiles.

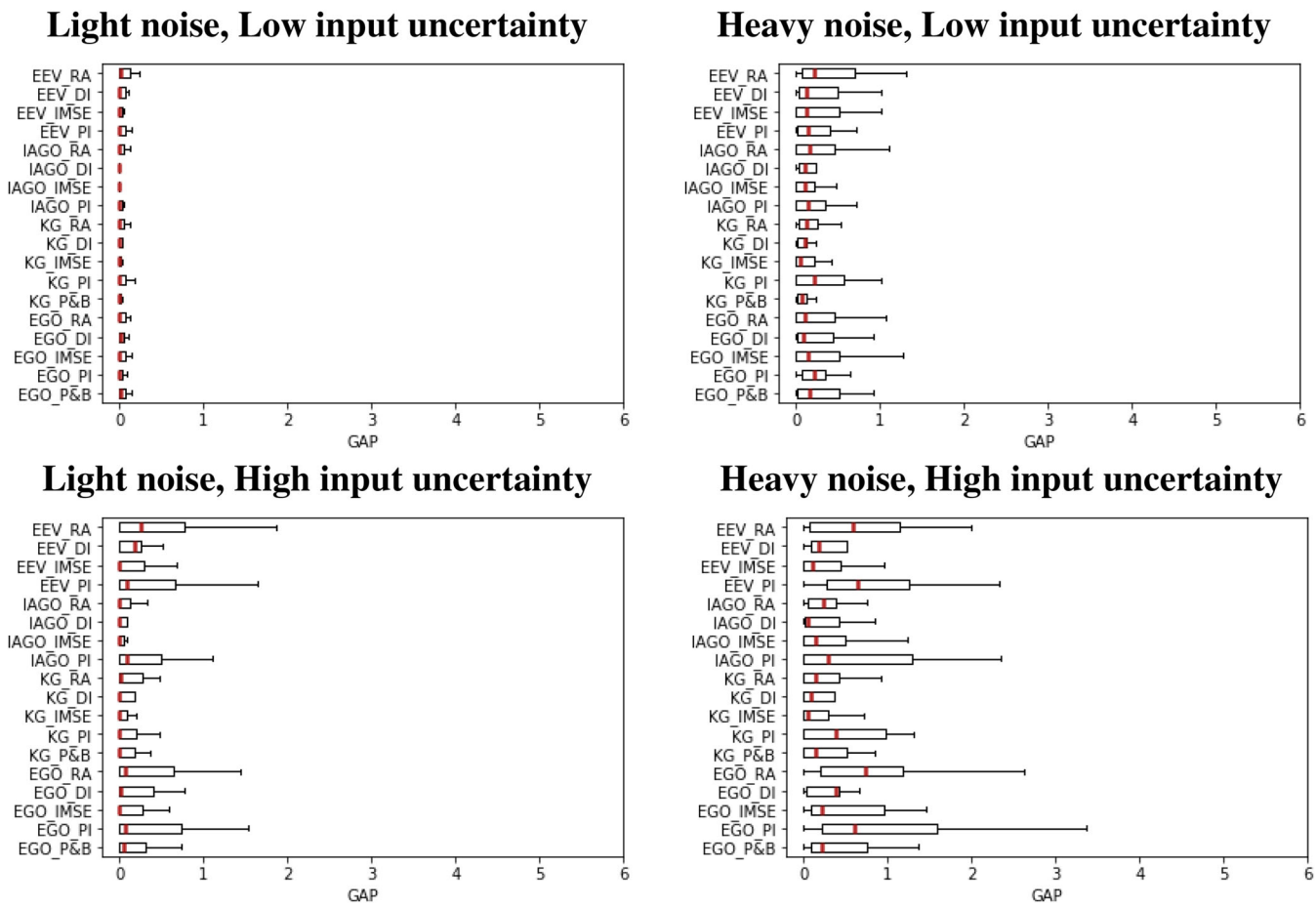

Figure 4. GAP for the Branin function for different algorithms. The bold lines inside the boxes are the median values; The edges of the boxes are the 25th and 75th percentiles.

will get more concentrated around the true input parameter. In the extreme case when input uncertainty vanishes (real world data is infinite), both MLE and posterior distribution degenerates to the true input parameter, and both algorithms are solving the same simulation optimization problem without input uncertainty. Still, it is worth noticing that for the Eggholder function, algorithms with input uncertainty can perform better even when the input uncertainty level is low. Looking more closely at the Eggholder function surface in Figure 2, we see that its surface is quite sensitive to $\boldsymbol{\lambda}$. Hence, a small variation of the plug-in value $\hat{\boldsymbol{\lambda}}$ can produce a significant change in optimal design $\hat{\boldsymbol{x}}^*$ obtained from $f(\boldsymbol{x}, \hat{\boldsymbol{\lambda}})$. Moreover, as the surface $g(\boldsymbol{x})$ is sensitive to $\boldsymbol{x}$, the obtained design $\hat{\boldsymbol{x}}^*$ will result in $g(\hat{\boldsymbol{x}}^*)$ deviating a lot from $g(\boldsymbol{x}^*)$. Comparing our refined EGO and KG with their counterparts proposed by Pearce and Branke (2017) (indicated as EGO_P&B and KG_P&B in the boxplots), we find the performances of our proposed EGO_IMSE, and KG_IMSE can always match the performances of EGO_P&B and KG_P&B. As for evaluation time, our proposed EGO_IMSE and KG_IMSE require less evaluation time than EGO_P&B and KG_P&B respectively. For each sequential step, for any candidate $(\boldsymbol{x_{n+1}}, \boldsymbol{\lambda_{n+1}})$, both EGO_P&B and KG_P&B require a MC method to compute the posterior distribution of $\hat{G}_{n+1}(\boldsymbol{x})$, whereas our proposed EGO_IMSE and KG_IMSE only require the MC method to compute the posterior distribution of $\hat{G}_{n+1}(\boldsymbol{x})$ at any design point $\boldsymbol{x_{n+1}}$. By reducing the candidate space from the joint $(\boldsymbol{x}, \boldsymbol{\lambda})$ space to the $\boldsymbol{x}$ space through our two-step framework, our proposed EGO_IMSE and KG_IMSE reduce the evaluation time as shown in Table 2, and this is especially significant when the problem dimension increases:

2. For the three methods for choosing $\boldsymbol{\lambda_{n+1}}$: RA, IMSE and DI, we investigate their median GAP performances. When input uncertainty is low, the performances of these three methods are quite similar. This is because with low input uncertainty, the posterior distribution of $\boldsymbol{\lambda}$ tends to be more concentrated around the true input parameter with narrower range, and thus there is no obvious difference in the $\boldsymbol{\lambda_{n+1}}$ selected from these three methods. When input uncertainty is high, the performances of the IMSE algorithms are usually better than DI and RA, with the RA performing consistently worst. To account for the good performance of IMSE, recall that getting an accurate approximation surface for $g(\boldsymbol{x})$ requires accurate MC samples of $f(\boldsymbol{x}, \boldsymbol{\lambda})$ throughout the domain of $\boldsymbol{\lambda}$ (see Equation (21)). The $\boldsymbol{\lambda_{n+1}}$ chosen by IMSE are more evenly spread over the domain, ensuring a more accurate surface of $f(\boldsymbol{x}, \boldsymbol{\lambda})$ as well as MC samples of it. The $\boldsymbol{\lambda_{n+1}}$ chosen by DI tend to be more concentrated at high density of $\boldsymbol{\lambda}$ areas. This will lead to

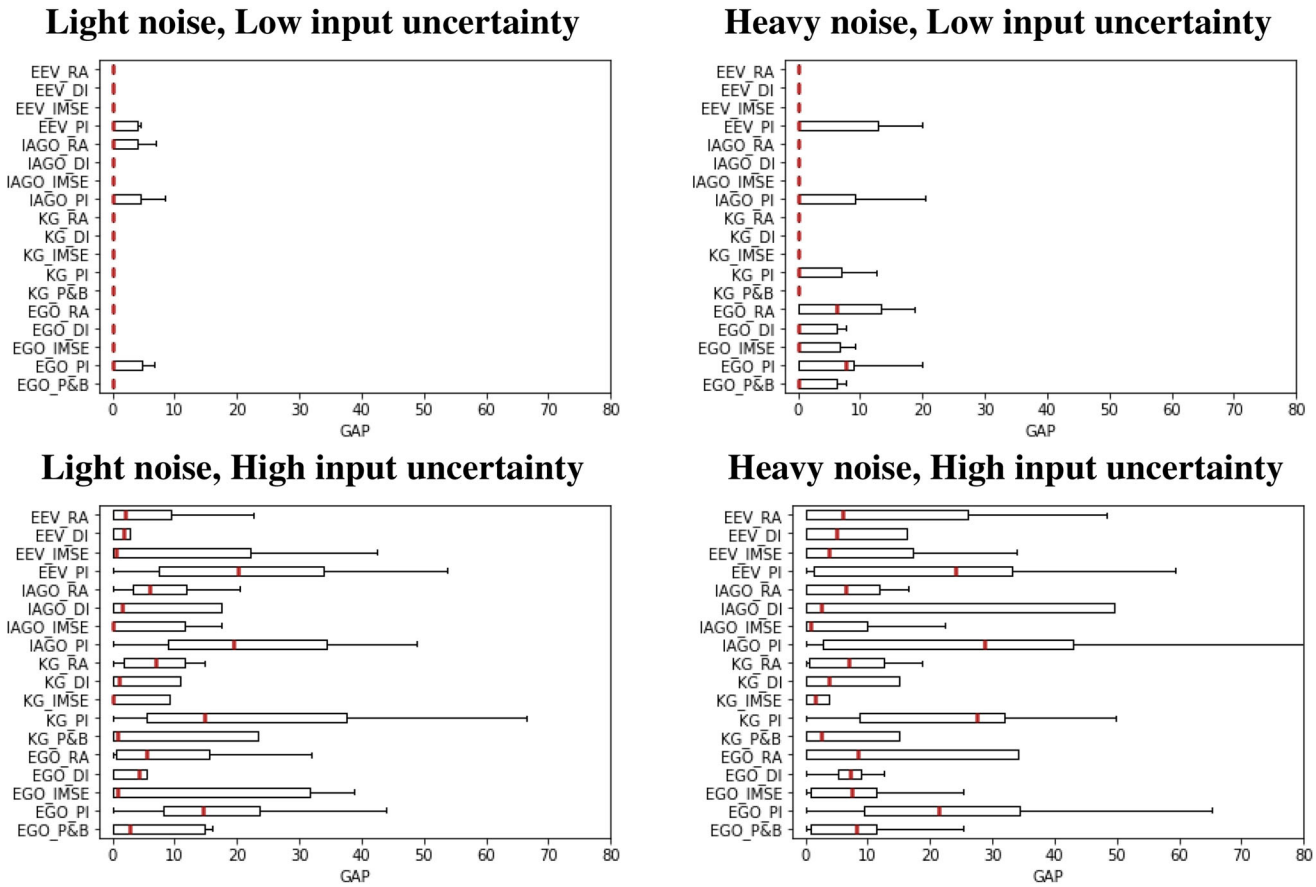

Figure 5. GAP for Hartmann-6 function for different algorithms. The bold lines inside the boxes are the median values; the edges of the boxes are the 25th and 75th percentiles.

more accurate approximations of $f(\boldsymbol{x}, \boldsymbol{\lambda})$ at high density areas, and less accurate ones at low density areas. The approximation of $g(\boldsymbol{x})$ will then not be as accurate, due to the less accurate approximation of $f(\boldsymbol{x}, \boldsymbol{\lambda})$ at these low density areas. Hence, focusing on improving the approximation of $f(\boldsymbol{x}, \boldsymbol{\lambda})$ in the high-density region of $\boldsymbol{\lambda}$ appears insufficient to improve the overall approximation of $g(\boldsymbol{x})$. Therefore, DI which favors $\boldsymbol{\lambda}_{n+1}$ at high-density areas has worse performance than IMSE. This problem is more severe for RA which have even fewer $\boldsymbol{\lambda}_{n+1}$ in low-density regions than DI. In all cases, RA performs worse than IMSE, as it is not able to cover a wide domain of $\boldsymbol{\lambda}$ to improve the global fitting for $f(\boldsymbol{x}, \boldsymbol{\lambda})$ with the given budget, to obtain accurate approximations for $g(\boldsymbol{x})$. However, when the simulation budget is increased from 40 to 80 (double the current simulation budget), the experiment results show that the performances of the RA method will be similar to IMSE, as the larger budget enables RA to spread its samples more widely across the domain of $\boldsymbol{\lambda}$.

The good performance of IMSE can therefore be attributed to its ability to provide a better spread of points in the input domain. This further motivates us to explore other fast-to-compute space-filling methods (which can also provide a good spread). Here, we adopt the sequential Latin Hypercube Sampling (LHS) with maximin criterion (Johnson *et al.*, 1990) to sequentially choose $\boldsymbol{\lambda}_{n+1}$ that can maximize the minimum distance between any existing point in the joint $\boldsymbol{x}$ and $\boldsymbol{\lambda}$ space. In all cases, the experiment results show that the LHS method does not perform better than the IMSE. As we have elaborated, the IMSE criterion has a focus on decreasing the global prediction error of $f(\boldsymbol{x}, \boldsymbol{\lambda})$ as we sequentially update the GP model $F_n(\boldsymbol{x}, \boldsymbol{\lambda})$ and this reduction of the global prediction error can help increase the approximation accuracy of $g(\boldsymbol{x})$. Although the LHS method can also provide a good spread of points in the domain, and in a sense reduce the overall prediction error of $f(\boldsymbol{x}, \boldsymbol{\lambda})$, it does not utilize the updated information of the GP model and thus is less efficient than IMSE in reducing the prediction error of $f(\boldsymbol{x}, \boldsymbol{\lambda})$ and $g(\boldsymbol{x})$.

In summary, the DI does not perform better than IMSE and requires longer evaluation times, so it is not recommended based on our experiment results. The RA method is only preferred with low input uncertainty since its performance can match the IMSE method, and it requires less evaluation time. Performances of the IMSE method tend to be more robust with respect to input uncertainty, which makes the IMSE method preferred when input uncertainty level is high.

3. For the four GP-based algorithms with IMSE, they have similar satisfactory median GAP performance for all scenarios. It should be noted that for Hartmann-6, IAGO and KG have better performance compared with EGO and EEV when both input uncertainty and stochastic noise level are high. This is because EGO and EEV are

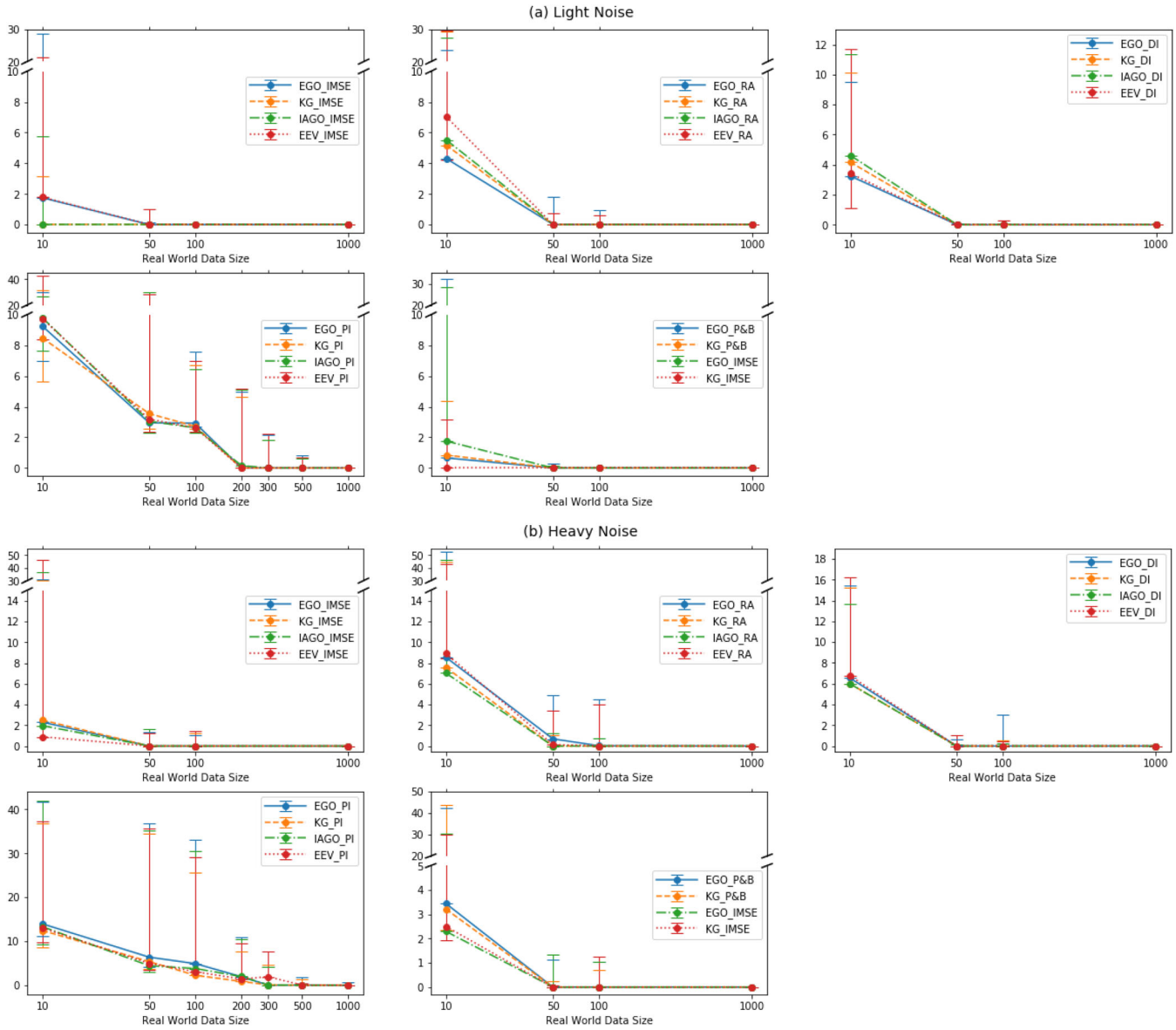

Figure 6. Convergence of GAP for the Eggholder function with light and heavy noise. The y-axis represents GAP. The solid circles represent the median values of the GAP, and the ends of the two lines from the circles represent the 3rd and 1st quartile values of the GAP.

initially proposed for deterministic simulation optimization. When extending these two algorithms for stochastic simulation optimization with input uncertainty, the estimation of T (the current best) is required, and this estimate can be highly noisy as the problem dimension increases. This can then mislead the selection of the sequential points. We see the median performance for EGO and EEV matches IAGO and KG when either stochastic noise level or input uncertainty level is low, as T is better estimated in these cases. Furthermore, the evaluation time for IAGO with IMSE, EEV with IMSE, and KG with IMSE are considerably long, and this can impede their application in practice. Although the performance of EGO with IMSE can be affected by the estimation of the current best, it is still competitive as it requires much less evaluation time.

To further analyze the empirical convergence characteristics of the algorithms, we conducted a GAP analysis for the Eggholder function. As seen from Figure 6, when the real world data size increases (input uncertainty decreases), the performances of all algorithms improve. The variances also noticeably shrink when the data size increases. As expected, the convergence is slower when the simulation noise is larger, with also larger variances in the GAP. However, with the exception of the plug-in algorithms, all other algorithms converge rather quickly (GAP goes to zero) at about $h = 50$. Then plug-in algorithms took four times and six times as many real world data (observations) as the IMSE algorithms to converge for the small noise and large noise cases respectively.

In summary, the following conclusions can be drawn. Algorithms with PI have good approximation accuracy only when input uncertainty level is low and when the surface of $f(\boldsymbol{x}, \boldsymbol{\lambda})$ is relatively insensitive to the input parameter $\boldsymbol{\lambda}$. Although it is not easy to define low (or high) input uncertainty level, a rough gauge can be the size of the real world input data available. However, it is important to highlight again that underestimating the input uncertainty level and applying

algorithms with PI can result in suboptimal designs. Overall, the algorithms with IMSE appear to provide good approximation results regardless of the input uncertainty level. Furthermore, among the four algorithms with IMSE, if the user's interest is on the approximation accuracy and have sufficient evaluation time budget, the IAGO and KG should be adopted, as they are robust and can consistently yield good approximation results for all scenarios; if the evaluation time budget is limited, EGO can be a favorable choice.

6. Conclusions

In this article, we first refine EGO, KG, IAGO, and EEV with a two-step framework to solve stochastic simulation optimization problems with input uncertainty. Specifically, we modify the infill criteria of these algorithms to solve Equation (13). We further conduct several numerical tests to compare the various GP-based optimization algorithms with and without input uncertainty. From the results, we find that:

1. GP-based optimization algorithms with input uncertainty can more accurately approximate the global minimizers than the commonly applied algorithms with estimated PI parameter. This highlights the need to incorporate input uncertainty in GP-based optimization algorithms, especially with high input uncertainty. Furthermore, our proposed EGO with IMSE and KG with IMSE have similar performance as their counterparts in Pearce and Branke (2017), but require much less evaluation time.
2. Using the IMSE criterion to select the next input point can improve the approximation accuracy for $g(\boldsymbol{x})$ most, and this becomes more significant as the input uncertainty level increases.
3. Although all four refined algorithms with IMSE can accurately approximate the global minimizers, the evaluation time for EGO with IMSE is much shorter, especially for higher-dimension problems. This can make it preferred in practice.

There are several future directions to consider. First, this article assumes the distribution family of the input process is known. This may not be realistic in practice, and there are usually more than one distribution family that can fit well the finite real world data. Thus, in addition to the uncertainty associated with input parameter (input-parameter uncertainty), the uncertainty in the distribution family (input-model uncertainty) can be incorporated into stochastic simulation optimization. Second, our proposed algorithms are currently designed for single-objective optimization problems. In practice, for example, the inventory problem, in addition to the expected total cost, the customers service rate is another important objective when designing the system. Both the expected total cost and the customers service rate depend on the input parameter, and thus a misspecified input parameter can mislead the derived Pareto set. Algorithms for the multi-objective optimization problem with input-parameter uncertainty have not yet been studied.

Notes on contributors

Haowei Wang is currently a Ph.D. candidate in the Department of Industrial Systems engineering and Management at the National University of Singapore. He received his B.Eng. degree in industrial engineering from Nanjing University in 2016. His research interests include uncertainty quantification, simulation metamodeling and optimization.

Jun Yuan is currently an associate professor with China Institute of FTZ Supply chain, Shanghai Maritime University. He received his B.E. degree in industrial engineering and management from Shanghai Jiao Tong University, Shanghai, P.R. China, in 2008, and a Ph.D. degree in industrial and systems engineering National University of Singapore, Singapore, in 2013. From 2014-2017, he worked as a research fellow at the National University of Singapore. His research interests include energy systems modeling, shipping energy systems, computer simulation, and machine learning.

Szu Hui Ng is an associate professor in the Department of Industrial Systems Engineering and Management at the National University of Singapore. She holds B.S., M.S., and Ph.D. degrees in industrial and operations engineering from the University of Michigan. Her research interests include computer simulation analysis and optimization, applications of simulation to maritime transportation and quality engineering. She is a member of IEEE and INFORMS, and a senior member of IISE.

References

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

Barton, R.R. and Meckesheimer, M. (2006) Metamodel-based simulation optimization. *Handbooks in Operations Research and Management Science*, **13**, 535–574.

Barton, R.R., Nelson, B.L. and Xie, W. (2013) Quantifying input uncertainty via simulation confidence intervals. *INFORMS Journal on Computing*, **26**(1), 74–87.

Barton, R.R. and Schruben, L.W. (1993) Uniform and bootstrap resampling of empirical distributions, in *Proceedings of the 25th Conference on Winter Simulation*, IEEE, Los Angeles, CA, pp. 503–508.

Bashyam, S. and Fu, M.C. (1998) Optimization of (s,S) inventory systems with random lead times and a service level constraint. *Management Science*, **44**(12-part-2), S243–S256.

Berger, J.O. (1985) *Statistical Decision Theory and Bayesian Analysis*.

Binois, M., Huang, J., Gramacy, R.B. and Ludkovski, M. (2019) Replication or exploration? Sequential design for stochastic simulation experiments. *Technometrics*, **61**(1), 7–23.

Brooks, S., Gelman, A., Jones, G. and Meng, X.-L. (2011) *Handbook of Markov Chain Monte Carlo*, CRC Press, Boca Raton, FL.

Cheng, R.C. and Holland, W. (1997) Sensitivity of computer simulation experiments to errors in input data. *Journal of Statistical Computation and Simulation*, **57**(1-4), 219–241.

Cheng, R.C. and Holland, W. (2004) Calculation of confidence intervals for simulation output. *ACM Transactions on Modeling and Computer Simulation (TOMACS)*, **14**(4), 344–362.

Chick, S.E. (2001) Input distribution selection for simulation experiments: Accounting for input uncertainty. *Operations Research*, **49**(5), 744–758.

Corlu, C.G. and Biller, B. (2013) A subset selection procedure under input parameter uncertainty, in *Proceedings of the 2013 Winter Simulation Conference*, IEEE Press, Piscataway, NJ, pp. 463–473.

Cressie, N. (1992) Statistics for spatial data. *Terra Nova*, **4**(5), 613–617.

De Oliveira, V. and Kone, B. (2015) Prediction intervals for integrals of Gaussian random fields. *Computational Statistics and Data Analysis*, **83**, 37–51.

Efron, B. (1983) Estimating the error rate of a prediction rule: improvement on cross-validation. *Journal of the American Statistical Association*, **78**(382), 316–331.

Fan, W., Hong, L.J. and Zhang, X. (2013) Robust selection of the best, in *Proceedings of the 2013 Winter Simulation Conference*, IEEE Press, Piscataway, NJ, pp. 868–876.

Frazier, P., Powell, W. and Dayanik, S. (2009) The knowledge-gradient policy for correlated normal beliefs. *INFORMS Journal on Computing*, **21**(4), 599–613.

Gelman, A., Stern, H.S., Carlin, J.B., Dunson, D.B., Vehtari, A. and Rubin, D.B. (2013) *Bayesian Data Analysis*, Chapman and Hall/CRC, Boca Raton, FL.

Gentle, J.E. (2009) *Computational Statistics*, Springer Science & Business Media. Springer, New York, NY.

Hoffman, M.D. and Gelman, A. (2014) The no-u-turn sampler: Adaptively setting path lengths in Hamiltonian Monte Carlo. *Journal of Machine Learning Research*, **15**(1), 1593–1623.

Huang, D., Allen, T.T., Notz, W.I. and Zeng, N. (2006) Global optimization of stochastic black-box systems via sequential kriging meta-models. *Journal of Global Optimization*, **34**(3), 441–466.

Jalali, H. and van Nieuwenhuyse, I. (2015) Simulation optimization in inventory replenishment: A classification. *IIE Transactions,* **47**(11), 1217–1235.

Jalali, H., van Nieuwenhuyse, I. and Picheny, V. (2017) Comparison of kriging-based algorithms for simulation optimization with heterogeneous noise. *European Journal of Operational Research*, **261**(1), 279–301.

Johnson, M.E., Moore, L.M. and Ylvisaker, D. (1990) Minimax and maximin distance designs. *Journal of Statistical Planning and Inference*, **26**(2), 131–148.

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

Kass, R.E. and Wasserman, L. (1996) The selection of prior distributions by formal rules. *Journal of the American Statistical Association*, **91**(435), 1343–1370.

Kleijnen, J.P. (2015) *Design and Analysis of Simulation Experiments*, Springer. Cham.

Kushner, H. and Yin, G. (2003) *Stochastic Approximation and Recursive Algorithms and Applications*, Springer-Verlag, New York, NY.

Lakshmanan, S. and Venkateswaran, J. (2017) Robust simulation based optimization with input uncertainty, in *Proceedings of the 2017 Winter Simulation Conference*, IEEE Press, Piscataway, NJ, pp. 2257–2267.

Lam, H. (2016) Advanced tutorial: Input uncertainty and robust analysis in stochastic simulation, in *Proceedings of the 2016 Winter Simulation Conference*, IEEE Press, pp. 178–192.

Mood, A.M. (1954) On the asymptotic efficiency of certain nonparametric two-sample tests. *The Annals of Mathematical Statistics*, **25**(3), 514–522.

Neal, D. (1993) Determining sample sizes for Monte Carlo integration. *The College Mathematics Journal*, **24**(3), 254–259.

Ng, S.H. and Yin, J. (2012) Bayesian kriging analysis and design for stochastic simulations. *ACM Transactions on Modeling and Computer Simulation,* **22**(3), pp. 17:1–17:26.

Ólafsson, S. (2006) Metaheuristics. *Handbooks in Operations Research and Management Science*, **13**, 633–654.

Pearce, M. and Branke, J. (2017) Bayesian simulation optimization with input uncertainty. in *Proceedings of the 2017 Winter Simulation Conference*, IEEE Press, Piscataway, NJ, pp. 2268–2278.

Picheny, V. (2015) Multiobjective optimization using Gaussian process emulators via stepwise uncertainty reduction. *Statistics and Computing*, **25**(6), 1265–1280.

Picheny, V., Wagner, T. and Ginsbourger, D. (2013) A benchmark of kriging-based infill criteria for noisy optimization. *Structural and Multidisciplinary Optimization*, **48**(3), 607–626.

Preuss, M., Wagner, T. and Ginsbourger, D. (2012) High-dimensional model-based optimization based on noisy evaluations of computer games, *in Learning and Intelligent Optimization*, Springer, Berlin Heidelberg, pp. 145–159.

Quan, N., Yin, J., Ng, S.H. and Lee, L.H. (2013) Simulation optimization via kriging: A sequential search using expected improvement with computing budget constraints *IIE Transactions*, **45**(7), 763–780.

Rasmussen, C.E. and Williams, C.K.I. (2006) *Gaussian Processes for Machine Learning*, MIT Press.

Robert, C.P. and Casella, G. (2004) *Monte Carlo Statistical Methods (Springer Texts in Statistics)*, Springer-Verlag, Berlin, Germany.

Sacks, J., Welch, W.J., Mitchell, T.J. and Wynn, H.P. (1989) Design and analysis of computer experiments. *Statistical Science*, **4**(4), 409–423.

Shim, J.P., Warkentin, M., Courtney, J.F., Power, D.J., Sharda, R. and Carlsson, C. (2002) Past, present, and future of decision support technology. *Decision Support Systems*, **33**(2), 111–126.

Villemonteix, J., Vazquez, E. and Walter, E. (2009) An informational approach to the global optimization of expensive-to-evaluate functions. *Journal of Global Optimization*, **44**(4), pp. 509–534.

Wu, D., Zhu, H. and Zhou, E. (2018) A Bayesian risk approach to data-driven stochastic optimization: Formulations and asymptotics. *SIAM Journal on Optimization*, **28**(2), 1588–1612.

Xie, W., Nelson, B.L. and Barton, R.R. (2014) A Bayesian framework for quantifying uncertainty in stochastic simulation. *Operations Research*, **62**(6), 1439–1452.

Xu, J., Huang, E., Chen, C.-H. and Lee, L.H. (2015) Simulation optimization: A review and exploration in the new era of cloud computing and big data. *Asia-Pacific Journal of Operational Research*, **32**(03), 1–34.

Yin, J., Ng, S.H. and Ng, K.M. (2011) Kriging metamodel with modified nugget-effect: The heteroscedastic variance case. *Computers & Industrial Engineering,* **61**(3), 760–777.

Zhang, X. and Ding, L. (2016) Sequential sampling for Bayesian robust ranking and selection, in *Proceedings of the 2016 Winter Simulation Conference*, IEEE Press, Piscataway, NJ, pp. 758–769.

Zhou, E. and Wu, D. (2017) Simulation optimization under input model uncertainty. *Advances in Modeling and Simulation*, Springer, Cham, pp. 219–247.

Zhou, E. and Xie, W. (2015) Simulation optimization when facing input uncertainty, in *Proceedings of the 2015 Winter Simulation Conference*, in IEEE Press, Piscataway, NJ, pp. 3714–3724.

Zhu, H., Hale, J. and Zhou, E. (2016) Optimizing conditional value-at-risk via gradient-based adaptive stochastic search, in *Proceedings of the 2016 Winter Simulation Conference*, IEEE Press, Piscataway, NJ, pp. 726–737.

Zhu, H., Hale, J. and Zhou, E. (2018) Simulation optimization of risk measures with adaptive risk levels. *Journal of Global Optimization*, **70**(4), 783–809.

Zouaoui, F. and Wilson, J.R. (2003) Accounting for parameter uncertainty in simulation input modeling. *IIE Transactions*, **35**(9), 781–792.