Interval uncertainty propagation by a parallel Bayesian global optimization method

Chao Dang[a], Pengfei Wei[b,*], Matthias G.R. Faes[c], Marcos A. Valdebenito[d], Michael Beer[a,e,f]

[a] *Institute for Risk and Reliability, Leibniz University Hannover, Callinstr. 34, Hannover 30167, Germany*
[b] *School of Mechanics, Civil Engineering and Architecture, Northwestern Polytechnical University, Xi'an 710072, PR China*
[c] *Chair for Reliability Engineering, TU Dortmund University, Leonhard-Euler-Str. 5, Dortmund 44227, Germany*
[d] *Faculty of Engineering and Sciences, Universidad Adolfo Ibáñez, Av. Padre Hurtado 750, 2562340 Viña del Mar, Chile*
[e] *Institute for Risk and Uncertainty, University of Liverpool, Liverpool L69 7ZF, United Kingdom*
[f] *International Joint Research Center for Resilient Infrastructure & International Joint Research Center for Engineering Reliability and Stochastic Mechanics, Tongji University, Shanghai 200092, PR China*

Abstract

This paper is concerned with approximating **the scalar response of a complex** computational model subjected to multiple **input** interval variables. **Such task is formulated as finding both the global minimum and maximum of a computationally expensive black-box function over a prescribed hyper-rectangle. On this basis, a novel non-intrusive method, called 'triple-engine parallel Bayesian global optimization', is proposed. The method begins by assuming a Gaussian process prior (which can also be interpreted as a surrogate model) over the response function. The main contribution lies in developing a novel infill sampling criterion, i.e., triple-engine pseudo expected improvement strategy, to identify multiple promising points for minimization and/or maximization based on the past observations at each iteration. By doing so, these identified points can be evaluated on the real response function in parallel. Besides, another potential benefit is that both the lower and upper bounds of the model response can be obtained with a single run of the developed method.** Four numerical examples with varying complexity are investigated to demonstrate the proposed method against some existing techniques, and results indicate that significant computational savings can be achieved by making full use of prior knowledge and parallel computing.

Keywords: Interval uncertainty propagation, Bayesian global optimization, Gaussian process, Infill sampling criterion, Parallel computing

Abbreviations

3-D	three-dimensional
BGO	Bayesian global optimization
EI	expected improvement
EI-MAX	expected improvement for maximum
EI-MIN	expected improvement for minimum
GP	Gaussian process
I-MLQMC	interval multilevel quasi-Monte Carlo
LHS	Latin hypercube sampling
N-PBGO	non-parallel Bayesian global optimization
NLML	negative log marginal likelihood
PBGO	parallel Bayesian global optimization
PEI	pseudo expected improvement
PEI-MAX	pseudo expected improvement for maximum
PEI-MIN	pseudo expected improvement for minimum
PEI-MIN-MAX	pseudo expected improvement for minimum and maximum
T-PBGO	triple-engine parallel Bayesian global optimization
T-PEI	triple-engine pseudo expected improvement
TLBO	teaching–learning-based optimization

1. Introduction

Along with the rapid development of computation techniques, deterministic numerical analysis has made great progress in various fields over the past several decades [1]. In this context, all parameters of a computational model designed to describe underlying structures or systems are typically treated as precise (crisp) numbers. This kind of numerical analysis, however, is essentially not suitable for situations where non-determinism has to be properly considered, which is the common case for a broad range of modern science and engineering disciplines. A typical example of such situations is the design and analysis of

*Corresponding author
Email addresses: chao.dang@irz.uni-hannover.de (Chao Dang), pengfeiwei@nwpu.edu.cn (Pengfei Wei), matthias.faes@tu-dortmund.de (Matthias G.R. Faes), marcos.valdebenito@uai.cl (Marcos A. Valdebenito), beer@irz.uni-hannover.de (Michael Beer)

engineering systems at an early stage where many aspects could be only imprecisely known. Alternatively, non-deterministic numerical analysis is emerging as an exciting research frontier with new opportunities and also challenges. **Such opportunities and challenges arise throughout the whole analysis, e.g., non-determinism characterization on the input side and response uncertainty quantification on the output side.**

In general, three types of approaches are available for modelling non-determinism: probabilistic approach, imprecise probabilistic approach and non-probabilistic approach [2]. On the basis of classical probability theory and statistical techniques, the probabilistic approach is most widely used. Herein, an uncertain parameter is modelled as a random variable with a precisely known probability distribution. Thus, it is often challenging to apply the probabilistic approach in reality since a large amount of high-quality data is required to infer an accurate probability distribution. Against this background, by generalizing traditional probability and statistics concepts, the imprecise probabilistic approach has evolved as a powerful and elegant framework for quantifying uncertainty from incomplete information [3, 4]. Within this approach, one needs to assign a pair of lower and upper probabilities to an event, rather than a single probability. On the other hand, the non-probabilistic approach, such as interval models and fuzzy sets [5], is also gaining increasing interest for non-determinism modelling, **especially when the available information is limited**. With the interval concept, a non-deterministic parameter is treated as an interval variable specified by a pair of numbers, i.e., the lower and upper bounds, and potentially a function modelling the auto-dependencies among multiple interval parameters [6]. Thus, instead of a full probability distribution the analyst only needs to determine the bounds and auto-dependency functions, which can be easily and objectively acquired from a small number of samples. The present study limits its scope to interval uncertainty.

There have been plenty of methods developed to propagate interval uncertainty via a computational model, which can be roughly classified into four kinds. The first kind of methods is based on using the interval arithmetic of Moore, e.g., refer to [7]. Despite its efficiency, the interval calculus cannot trace parameter dependency by definition (the so-called dependency problem), which therefore can lead to a severe overestimation of the size of a response interval. **Recent developments are focused on limiting**

the overestimation by, e.g., accounting for dependency among interval variables [8–12] or using interval fields [13, 14], parameterizing intervals via trigonometric functions [15, 16] and representing intervals by affine arithmetic [17, 18], etc. Although these methods are able to provide sharp bounds within reasonable computational cost, their applicability is still limited due to the intrusive nature of interval arithmetic. The approximate analytical methods that rely on constructing a simplified approximation of true response function falls in the second group. Typical examples of such methods include, Taylor series expansion methods [19–23] and Chebyshev series expansion methods [24, 25], which are intrusive and non-intrusive, respectively. However, these Taylor methods tend to lose accuracy when the considered problem involves large uncertainty (i.e.,the widths of interval variables being large) and/or highly nonlinear behavior. For these Chebyshev methods, the required number of response function evaluations grows exponentially with the number of dimensions. As for the third type, the vertex method [26, 27] and interval multilevel quasi-Monte Carlo (I-MLQMC)) [28, 29] are non-intrusive and can produce accurate response bounds under certain conditions. The classical vertex method is exact on the premise that the response function is monotonic with respect to d interval parameters, while at the cost of 2^d model evaluations. More strictly, the I-MLQMC method requires a linearity assumption on the response function. As such, these two methods suffer from non-linearity and/or dimensionality.

In the last group, global optimization methods are naturally applicable to the topic of interval numerical analysis. In this context, several studies have been conducted by directly using, e.g., genetic algorithm [30, 31]. Generally, global optimization algorithms require a large number of model evaluations to find the minimum/maximum, and hence can be computationally demanding especially when each such evaluation is expensive. To alleviate the computation burden, a cheap-to-evaluate surrogate model can be adopted to substitute the original computational model based on some observations. Along this line, Kriging-assisted global optimization (formally called Bayesian global optimization (BGO)) algorithms are attracting increasing attention due to their high efficiency for optimizing expensive black-box functions. A typical BGO method starts by building an initial Kriging model for the objective function based on a small number of observations, and then refines the initial model by sequentially selecting more updating points according to a

infill sampling criterion [32]. Existing studies then focus more on developing efficient infill sampling criteria so as to reduce the total number of function evaluations. On this aspect, representative works in the context of interval uncertainty propagation include the maximum improvement criterion [33], expected improvement criterion [34, 35] and a comparison study of several criteria [36]. **It is shown that these methods exhibit encouraging features regarding the computational efficiency and accuracy for computationally expensive black-box problems over other existing methods. Despite these advantages, one of the major limitations of the existing BGO methods is that they are sequential in nature and hence unsuitable for parallelization, or at least high-level parallelization, hindering the potential benefits from parallel distributed processing.**

In this paper, a parallel Bayesian global optimization (PBGO) method is proposed for estimating the response bounds of a computational model in the presence of interval variables. Our main objective is to further reduce the computational time of existing BGO methods by making use of parallelism. For this purpose, a novel infill sampling criterion is developed to select multiple points at each iteration, and hence corresponding model evaluations can be distributed on multiple processing cores simultaneously. **Such parallelisation is relevant when the model at hand is computationally intensive and parallel computing facilities are available. Besides, in contrast to the traditional way of searching the lower and upper bounds of a scalar response quantity via two separate optimization problems, we consider it only as one problem.** Following the developed scheme, the lower and upper bounds can be obtained simultaneously with a single run. Last but not least, a Matlab implementation of the developed algorithm is also readily available to the public [1].

The remainder of the paper is organized as follows. Section 2 describes the interval analysis problem to be solved in this study. The proposed PBGO method is introduced in Section 3, with its relationship to other PBGO methods also being discussed. Four numerical examples are studied in Section 4 to demonstrate the performance of the developed method. In Section 5, some concluding remarks and perspectives are given to end the paper

[1] to be released upon acceptance of the paper

2. Problem formulation

Let us consider a computational model represented by a deterministic, continuous, and real-valued function $y = g(\boldsymbol{x}) : \mathbb{R}^d \to \mathbb{R}$. Here the model response y is a scalar quantity of interest, the g-function is assumed to be an expensive-to-evaluate black box, and the model input vector $\boldsymbol{x}$ consists of d variables, i.e., $\boldsymbol{x} = [x_1, x_2, \cdots, x_d]$.

Under the assumption that available information on the model inputs is poor or incomplete, we proceed to treat them with interval models. **For identifying intervals from real observations, one can refer to, e.g., [37, 38].** An interval vector $\boldsymbol{x}^I = [x_1^I, x_2^I, \cdots, x_d^I] \in \mathbb{IR}^d$ can be defined as:

$$\boldsymbol{x}^I = [\underline{\boldsymbol{x}}, \overline{\boldsymbol{x}}] = \left\{ \boldsymbol{x} \in \mathbb{R}^d | \underline{\boldsymbol{x}} \leq \boldsymbol{x} \leq \overline{\boldsymbol{x}} \right\}, \tag{1}$$

and its component x_i^I satisfies

$$x_i^I = [\underline{x}_i, \overline{x}_i] = \{ x \in \mathbb{R} | \underline{x}_i \leq x \leq \overline{x}_i \}, i = 1, 2, \cdots, d,$$

where $\underline{\boldsymbol{x}} = [\underline{x}_1, \underline{x}_2, \cdots, \underline{x}_d]$ and $\overline{\boldsymbol{x}} = [\overline{x}_1, \overline{x}_2, \cdots, \overline{x}_d]$ represent the lower and upper bounds of $\boldsymbol{x}^I$, respectively. Further, the midpoint $\boldsymbol{x}^C$ and radius $\boldsymbol{x}^R$ of $\boldsymbol{x}^I$ can be defined as:

$$\boldsymbol{x}^C = \frac{\underline{\boldsymbol{x}} + \overline{\boldsymbol{x}}}{2},$$

$$\boldsymbol{x}^R = \frac{\overline{\boldsymbol{x}} - \underline{\boldsymbol{x}}}{2}.$$

It follows that the interval vector defined in Eq. (1) can also be rewritten in terms of $\boldsymbol{x}^C$ and $\boldsymbol{x}^R$ as:

$$\boldsymbol{x}^I = \boldsymbol{x}^C + \delta\boldsymbol{x},$$

where $\delta\boldsymbol{x} \in [-1, 1]\boldsymbol{x}^R$. **For convenience, the interval variables are assumed to be independent. In fact, for dependent interval variables one can transform them into independent ones by applying a suitable transformation, e.g., [39].**

With the interval vector $\boldsymbol{x}^I$ as input, the g-function will also give rise to a interval output y^I in our context, i.e., $y^I = \left\{ y \in \mathbb{R} | y = g(\boldsymbol{x}), \boldsymbol{x} \in \boldsymbol{x}^I \right\}$. The resulting interval can be fully characterized by its lower and upper bounds, which correspond to the worst or best case of y^I that we might be interested in. Therefore,

the main objective is to determine the lower and upper bounds of y^I, which are naturally defined as the solutions of the following two optimization problems:

$$\underline{y} = \min_{\boldsymbol{x} \in \boldsymbol{x}^I} \{y | y = g(\boldsymbol{x})\}, \tag{2}$$

$$\overline{y} = \max_{\boldsymbol{x} \in \boldsymbol{x}^I} \{y | y = g(\boldsymbol{x})\}, \tag{3}$$

where $\underline{y}$ and $\overline{y}$ can be interpreted as the global minimum and maximum of $y = g(\boldsymbol{x})$ subject to $\boldsymbol{x} \in \boldsymbol{x}^I$, respectively.

Although their definitions are rather simple, the analytical solutions to $\underline{y}$ and $\overline{y}$ are unavailable for a general black-box problem. Thus, numerical approximation techniques are necessary and useful tools for practical applications. Existing numerical methods, however, still suffer from their respective limitations as discussed in the introduction section. This motivates us to develop a PBGO method for propagating interval uncertainty in the following section.

3. Triple-engine parallel Bayesian global optimization

In this section, the propagation of interval uncertainty via an expensive black-box computational model is treated by a kind of Bayesian numerical method, i.e., the so-called Bayesian global optimization (BGO) [32]. Specifically, an efficient method, termed "Triple-engine parallel Bayesian global optimization" (T-PBGO), is proposed to approximate the lower and upper bounds of the model output y^I (defined in Eqs. (2) and (3)) when the model input is characterized by a interval vector $\boldsymbol{x}^I$ (defined in Eq. (1)). The proposed method makes use of the Gaussian process model and a newly developed infill sampling criterion, as will be introduced in what follows. For notational simplicity, the superscripts of $\boldsymbol{x}^I$ and y^I are omitted when there is no confusion.

3.1. Gaussian process model

Under the black-box assumption, no additional knowledge on the inner structure of the g-function is available and the only possibility for us is to evaluate it at some points. That is, we know nothing about the behavior of the g-function (e.g., concavity and linearity) before seeing any observations, let along its

minimum and maximum. The lack of knowledge on $g(\cdot)$ is referred to as a kind of epistemic uncertainty simply because it is numerically unknown until we actually evaluate it, and hence reduceable. Following a Bayesian approach, our prior beliefs on the g-function can be modeled by assigning a Bayesian prior distribution. In this study, we adopt a Gaussian process (GP) prior over g. In the following, we only give a brief introduction to the GP model, and for further details the reader can refer to [40]. The GP prior assumes that the g-function is a realization of a GP indexed by $\boldsymbol{x}$. To formalize this, we write the GP prior as:

$$\hat{g}_0(\boldsymbol{x}) \sim \mathcal{GP}(m_0(\boldsymbol{x}), k_0(\boldsymbol{x}, \boldsymbol{x}')) = m_0(\boldsymbol{x}) + Z(\boldsymbol{x}),$$

where $\hat{g}_0$ denotes the prior distribution of g; $m_0(\boldsymbol{x})$ is the mean function of the GP prior; $Z(\boldsymbol{x})$ is a stationary GP with zero-mean and covariance function $k_0(\boldsymbol{x}, \boldsymbol{x}')$. The GP prior is completely characterized by its prior mean function $m_0(\boldsymbol{x})$ and covariance function $k_0(\boldsymbol{x}, \boldsymbol{x}')$. The prior mean function reflects the general trend of the GP model, while the prior covariance function encodes the key features of the g-function, e.g., stationarity, isotropy, smoothness and periodicity. **There are many kinds of specific functional forms available in literature for the prior mean and covariance functions [40].** In this paper, without loss of generality, the prior mean function is assumed to be a constant (i.e., $m_0(\boldsymbol{x}) = \beta$) and the prior covariance function is of squared exponential form expressed as:

$$k_0(\boldsymbol{x}, \boldsymbol{x}') = \sigma_g^2 \exp\left[-\frac{1}{2}\left(\boldsymbol{x} - \boldsymbol{x}'\right)\boldsymbol{\Sigma}^{-1}\left(\boldsymbol{x} - \boldsymbol{x}'\right)^{\mathrm{T}}\right],$$

where σ_g^2 is the overall variance with $\sigma_g > 0$; $\boldsymbol{\Sigma} = \mathrm{diag}\left(l_1^2, l_2^2, \cdots, l_d^2\right)$ with $l_i > 0$ being the characteristic length-scale in i-th dimension; and $\mathrm{diag}(\cdot)$ denotes a diagonal matrix whose entries are equal to the argument values. The $d + 2$ free parameters β, σ_g^2 and $\{l_i\}_{i=1}^d$ are referred to hyper-parameters whose values need to be determined, denoted by $\boldsymbol{\theta} = \left\{\beta, \sigma_g^2, l_1, l_2, \cdots, l_d\right\}$.

Now assume that we have evaluated the g-function at several (e.g., $n \in \mathbb{Z}^+$) points. We aggregate the sampled points in a $n \times d$ matrix $\boldsymbol{X}$ with its j-th row being the j-th point $\boldsymbol{x}^{(j)}$, and the corresponding g-function values in a $n \times 1$ vector $\boldsymbol{y}$ with its j-th element being $y^{(j)}$, where $y^{(j)} = g(\boldsymbol{x}^{(j)})$. The set of

hyper-parameters can then be estimated by minimizing the negative log marginal likelihood (NLML) [40]:

$$\hat{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \left(-\log \left[p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\theta}) \right] \right), \tag{4}$$

with

$$-\log \left[p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\theta}) \right] = \frac{1}{2} (\boldsymbol{y} - \beta)^{\mathrm{T}} \boldsymbol{K}_0^{-1} (\boldsymbol{y} - \beta) + \frac{1}{2} \log \left[|\boldsymbol{K}_0| \right] + \frac{n}{2} \log \left[2\pi \right], \tag{5}$$

where $\boldsymbol{K}_0$ is a $n \times n$ covariance matrix with its (i, j)-th entry being $[\boldsymbol{K}_0]_{ij} = k_0(\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)})$. Eq. (4) can be solved by gradient-based optimization schemes since the derivatives of NLML in Eq. (5) with respect to $\boldsymbol{\theta}$ are analytically tractable.

Conditioning on the observations $(\boldsymbol{X}, \boldsymbol{y})$ and GP prior will give rise to a posterior distribution $\hat{g}_n$ of g. This distribution still follows a GP $\hat{g}_n(\boldsymbol{x}) \sim \mathcal{GP}(m_n(\boldsymbol{x}), k_n(\boldsymbol{x}, \boldsymbol{x}'))$, with the posterior mean and covariance functions as follows:

$$m_n(\boldsymbol{x}) = m_0(\boldsymbol{x}) + \boldsymbol{k}_0(\boldsymbol{x}, \boldsymbol{X}) \boldsymbol{K}_0^{-1} (\boldsymbol{y} - \boldsymbol{m}_0(\boldsymbol{X})),$$

$$k_n(\boldsymbol{x}, \boldsymbol{x}') = k_0(\boldsymbol{x}, \boldsymbol{x}') - \boldsymbol{k}_0(\boldsymbol{x}, \boldsymbol{X}) \boldsymbol{K}_0^{-1} \boldsymbol{k}_0(\boldsymbol{x}', \boldsymbol{X})^{\mathrm{T}},$$

where $\boldsymbol{k}_0(\boldsymbol{x}, \boldsymbol{X})$ is a $1 \times n$ covariance vector between $\boldsymbol{x}$ and $\boldsymbol{X}$, whose j-th element is $k_0(\boldsymbol{x}, \boldsymbol{x}^{(j)})$; $\boldsymbol{k}_0(\boldsymbol{x}', \boldsymbol{X})$ is similarly defined; $\boldsymbol{m}_0(\boldsymbol{X})$ is a $n \times 1$ mean vector, whose j-th element is $m_0(\boldsymbol{x}^{(j)})$. It is seen that via a Bayesian treatment a full predictive distribution $\hat{g}(\boldsymbol{x}) \sim \mathcal{N}(m_n(\boldsymbol{x}), \sigma_n^2(\boldsymbol{x}))$ is now available, where the posterior mean function $m_n(\boldsymbol{x})$ can be used as a predictor, while the posterior variance function $\sigma_n^2(\boldsymbol{x}) = k_n(\boldsymbol{x}, \boldsymbol{x})$ can measure the prediction uncertainty.

3.2. Proposed triple-engine pseudo expected improvement criterion

In order to make inference about the minimum and maximum of the g-function using as few function evaluations as possible, our main concern is to design an efficient infill sampling criterion that can effectively suggest future evaluation points based on the posterior GP (implicitly the past observations). In particular, we seek to identify a batch of **informative and diverse** points at each iteration. Hence, multiple evaluations of the g-function can be distributed on several cores simultaneously so as to reduce the overall wall-clock time. For convenience of illustration, we assume that the number of points we would like to select at

each iteration is a even number q in sequel, though it should not to be. Our purposes are achieved by generalizing the pseudo expected improvement (PEI) criterion [41], which has been recently developed in the field of global optimization, to an enhanced version, termed 'triple-engine pseudo expected improvement' (T-PEI) criterion. **The T-PEI criterion actually involves a set of three infill sampling criteria that we call them 'engines', as discussed below.**

3.2.1. Engine 1: PEI for minimum

The first engine is the PEI criterion originally developed in [41] for global minimization problems (denoted by PEI-MIN for convenience). In the present study, this criterion will be directly used to select q promising points for the propose of minimizing the g-function wherever applicable.

Let $y_{\min} = \min_{1\le j\le n} y^{(j)}$ indicate the minimum value of y observed so far. The improvement at point $\boldsymbol{x}$ over the current best solution $y_{\min}$ can be defined as [32]:

$$I_{\min}(\boldsymbol{x}) = \max\left(y_{\min} - \hat{g}_n(\boldsymbol{x}), 0\right) = \begin{cases} y_{\min} - \hat{g}_n(\boldsymbol{x}), \textbf{ if } \hat{g}_n(\boldsymbol{x}) < y_{\min} \\ 0, \textbf{ otherwise} \end{cases}, \tag{6}$$

which is a random variable at site $\boldsymbol{x}$. The so-called expected improvement (EI) over the current minimum $y_{\min}$ consists of taking expectation of $I_{\min}(\boldsymbol{x})$, and can be derived in a closed-form expression as [32]:

$$EI_{\min}(\boldsymbol{x}) = \mathbb{E}\left[I_{\min}(\boldsymbol{x})\right] = (y_{\min} - m_n(\boldsymbol{x}))\Phi\left(\frac{y_{\min} - m_n(\boldsymbol{x})}{\sigma_n(\boldsymbol{x})}\right) + \sigma_n(\boldsymbol{x})\phi\left(\frac{y_{\min} - m_n(\boldsymbol{x})}{\sigma_n(\boldsymbol{x})}\right), \tag{7}$$

where $\phi(\cdot)$ and $\Phi(\cdot)$ are the probability density function and cumulative distribution function of the standard normal variable, respectively. The next best point be acquired within the minimization process can be selected by maximizing $EI_{\min}(\boldsymbol{x})$, i.e.,

$$\boldsymbol{x}_{\min}^{(n+1)} = \arg\max_{\boldsymbol{x}\in\boldsymbol{x}^I} EI_{\min}(\boldsymbol{x}). \tag{8}$$

This criterion is referred to as EI-MIN for the sake of convenience. Note that the first term of $EI_{\min}(\boldsymbol{x})$ (see Eq. (7)) prefers the point whose prediction $m_n(\boldsymbol{x})$ is small, whereas the second

term prefers the point whose variance $\sigma_n^2(\boldsymbol{x})$ is large.Thus, the EI-MIN criterion gives an elegant balance between exploitation (i.e., local search) and exploration (i.e., global search). Despite this, the EI-MIN criterion can only produce one single point at each iteration, and hence not suitable for parallelization.

To overcome the limitation, the basic idea of the PEI-MIN criterion is to modify the initial EI function (Eq. (7)) sequentially, by multiplying it by an influence function. That is, the first updating point $\boldsymbol{x}_{\mathbf{min}}^{(n+1)}$ is still generated by using the initial EI-MIN criterion (Eq. (8)). Then, the second one $\boldsymbol{x}_{\mathbf{min}}^{(n+2)}$ can be identified by maximizing a modified EI function that considers the possible impact of the first updated point bringing to the EI function. In such a sequential way, a desired number of points can be obtained at each iteration without evaluating the g-function at any newly selected points. Thus, a good influence function should capture the real influence of the newly identified points on the initial EI function as much as possible, while remaining computationally tractable. The influence function proposed in [41] is motivated by the fact that the EI function (Eq. (7)) is zero at the sampled points, and positive in between. After $q-1$ points have been identified, the synthesized influence function for the q-th point is formulated as [41]:

$$\begin{aligned} IF(\boldsymbol{x};\boldsymbol{x}_{\mathbf{min}}^{(n+1)},\boldsymbol{x}_{\mathbf{min}}^{(n+2)},\cdots,\boldsymbol{x}_{\mathbf{min}}^{(n+q-1)}) &= \prod_{j=1}^{q-1}\left[1-\rho\left(\boldsymbol{x},\boldsymbol{x}_{\mathbf{min}}^{(n+j)}\right)\right] \\ &= \prod_{j=1}^{q-1}\left[1-\exp\left[-\frac{1}{2}\left(\boldsymbol{x}-\boldsymbol{x}_{\mathbf{min}}^{(n+j)}\right)\boldsymbol{\Sigma}^{-1}\left(\boldsymbol{x}-\boldsymbol{x}_{\mathbf{min}}^{(n+j)}\right)^{\mathbf{T}}\right]\right], \end{aligned} \tag{9}$$

where $\rho\left(\boldsymbol{x},\boldsymbol{x}_{\mathbf{min}}^{(n+j)}\right)$ is the correlation function between two points $\boldsymbol{x}$ and $\boldsymbol{x}_{\mathbf{min}}^{(n+j)}$. It should be noted that the influence function is zero at the $q-1$ newly selected points $\boldsymbol{x}_{\mathbf{min}}^{(n+1)},\boldsymbol{x}_{\mathbf{min}}^{(n+2)},\cdots,\boldsymbol{x}_{\mathbf{min}}^{(n+q-1)}$, and approaches to one when far away from these points. The PEI function for the q-th point can be defined as [41]:

$$PEI_{\mathbf{min}}(\boldsymbol{x};\boldsymbol{x}_{\mathbf{min}}^{(n+1)},\boldsymbol{x}_{\mathbf{min}}^{(n+2)},\cdots,\boldsymbol{x}_{\mathbf{min}}^{(n+q-1)}) = EI_{\mathbf{min}}(\boldsymbol{x}) \times IF(\boldsymbol{x};\boldsymbol{x}_{\mathbf{min}}^{(n+1)},\boldsymbol{x}_{\mathbf{min}}^{(n+2)},\cdots,\boldsymbol{x}_{\mathbf{min}}^{(n+q-1)}). \tag{10}$$

The $PEI_{\mathbf{min}}$ function can be interpreted as an approximation of the 'real' $EI_{\mathbf{min}}$ function because it is constructed without evaluating the g-function at these $q-1$ points and updating

the GP model (i.e., re-evaluating the hyper-parameters). Besides, it reduces to the standard $EI_{\mathbf{min}}$ function when $q = 1$, and hence the standard $EI_{\mathbf{min}}$ function can be seen as a special case of the PEI function. The q-th point can be selected by maximizing the $PEI_{\mathbf{min}}$ function such that:

$$\boldsymbol{x}_{\mathbf{min}}^{(n+q)} = \arg\max_{\boldsymbol{x}\in\boldsymbol{x}^I} PEI_{\mathbf{min}}(\boldsymbol{x}; \boldsymbol{x}_{\mathbf{min}}^{(n+1)}, \boldsymbol{x}_{\mathbf{min}}^{(n+2)}, \cdots, \boldsymbol{x}_{\mathbf{min}}^{(n+q-1)}).$$

3.2.2. Engine 2: PEI for maximum

Inspired by the PEI-MIN criterion, we can also define a similar criterion to select q promising points for maximizing the g-function if needed. The resulting criterion is called PEI-MAX, which is regarded as the second engine.

Let $y_{\mathbf{max}} = \max_{1\le j\le n} y^{(j)}$ denote the maximum value of y among the past n observations. In analogy to Eq. (6), the improvement at point $\boldsymbol{x}$ beyond the current best solution $y_{\mathbf{max}}$ can be defined as:

$$I_{\max}(\boldsymbol{x}) = \max\left(\hat{g}_n(\boldsymbol{x}) - y_{\max}, 0\right) = \begin{cases} \hat{g}_n(\boldsymbol{x}) - y_{\max}, \textbf{if } \hat{g}_n(\boldsymbol{x}) > y_{\max} \\ 0, \ \textbf{otherwise} \end{cases}. \tag{11}$$

The EI for the maximum is analytically derived in closed form as follows:

$$EI_{\mathbf{max}}(\boldsymbol{x}) = \mathbb{E}\left[I_{\max}(\boldsymbol{x})\right] = (m_n(\boldsymbol{x}) - y_{\max})\Phi\left(\frac{m_n(\boldsymbol{x}) - y_{\max}}{\sigma_n(\boldsymbol{x})}\right) + \sigma_n(\boldsymbol{x})\phi\left(\frac{m_n(\boldsymbol{x}) - y_{\max}}{\sigma_n(\boldsymbol{x})}\right). \tag{12}$$

However, by maximizing the $EI_{\mathbf{max}}$ function (the EI-MAX criterion), only one point for maximization is produced. In order to obtain a batch of q points, the the first point $\boldsymbol{x}_{\mathbf{max}}^{(n+1)}$ can be identified by $\boldsymbol{x}_{\mathbf{max}}^{(n+1)} = \arg\max_{\boldsymbol{x}\in\boldsymbol{x}^I} EI_{\mathbf{max}}(\boldsymbol{x})$. The following $q-1$ points should be sequentially selected by using a modified $EI_{\mathbf{max}}$ function. In analogy to the $PEI_{\mathbf{min}}$ function (Eq. (10)), we can define the $PEI_{\mathbf{max}}$ function for the q-th point such that:

$$PEI_{\mathbf{max}}(\boldsymbol{x}; \boldsymbol{x}_{\mathbf{max}}^{(n+1)}, \boldsymbol{x}_{\mathbf{max}}^{(n+2)}, \cdots, \boldsymbol{x}_{\mathbf{max}}^{(n+q-1)}) = EI_{\mathbf{max}}(\boldsymbol{x}) \times IF(\boldsymbol{x}; \boldsymbol{x}_{\mathbf{max}}^{(n+1)}, \boldsymbol{x}_{\mathbf{max}}^{(n+2)}, \cdots, \boldsymbol{x}_{\mathbf{max}}^{(n+q-1)}), \tag{13}$$

where the $IF(\cdot)$ function is defined in Eq. (9). The q-th point $\boldsymbol{x}_{\mathbf{max}}^{(n+q)}$ is obtained by:

$$\boldsymbol{x}_{\mathbf{max}}^{(n+q)} = \arg\max_{\boldsymbol{x}\in\boldsymbol{x}^I} PEI_{\mathbf{max}}(\boldsymbol{x}; \boldsymbol{x}_{\mathbf{max}}^{(n+1)}, \boldsymbol{x}_{\mathbf{max}}^{(n+2)}, \cdots, \boldsymbol{x}_{\mathbf{max}}^{(n+q-1)}).$$

3.2.3. Engine 3: PEI for both minimum and maximum

As we would like to infer both the minimum and maximum simultaneously, rather than in a sequential order, promising points for both extrema should be identified within one iteration until some predefined criteria are satisfied. Based on the PEI-MIN and PEI-MAX criteria, a infill sampling criterion for both minimizing and maximizing the g-function can be developed. This criterion is denoted by PEI-MIN-MAX, and it is served as the third engine.

The proposed PEI-MIN-MAX criterion proceeds as follows. The first updating point is identified by $\boldsymbol{x}_{\mathbf{min}}^{(n+1)} = \arg\max_{\boldsymbol{x}\in\boldsymbol{x}^I} EI_{\mathbf{min}}(\boldsymbol{x})$, which is used for minimization. Likewise, the second one (the first point for maximization) is computed by maximizing the $PEI_{\mathbf{max}}(\boldsymbol{x})$ function, i.e., $\boldsymbol{x}_{\mathbf{max}}^{(n+2)} = \arg\max_{\boldsymbol{x}\in\boldsymbol{x}^I} PEI_{\mathbf{max}}(\boldsymbol{x}; \boldsymbol{x}_{\mathbf{min}}^{(n+1)})$. The third point (the second for minimization) is produced by maximizing the $PEI_{\mathbf{min}}$ function, i.e., $\boldsymbol{x}_{\mathbf{min}}^{(n+3)} = \arg\max_{\boldsymbol{x}\in\boldsymbol{x}^I} PEI_{\mathbf{min}}(\boldsymbol{x}; \boldsymbol{x}_{\mathbf{min}}^{(n+1)}, \boldsymbol{x}_{\mathbf{max}}^{(n+2)})$, and the fourth one (the second point for maximization) is determined by maximizing the $PEI_{\mathbf{max}}$ function, i.e., $\boldsymbol{x}_{\mathbf{max}}^{(n+4)} = PEI_{\mathbf{max}}(\boldsymbol{x}; \boldsymbol{x}_{\min}^{(n+1)}, \boldsymbol{x}_{\max}^{(n+2)}, \boldsymbol{x}_{\min}^{(n+3)})$. As the process goes on, a desired q (≥ 2) updating points can be obtained sequentially ahead of observing their g-function values. Note that one can also start the first point with $\boldsymbol{x}_{\mathbf{max}}^{(n+1)}$, and then generate a set of q points $(\boldsymbol{x}_{\max}^{(n+1)}, \boldsymbol{x}_{\min}^{(n+2)}, \boldsymbol{x}_{\max}^{(n+3)}, \cdots)$ following a similar procedure.

3.3. Proposed T-PBGO algorithm

Based on the GP model and T-PEI infill sampling criterion, we propose a T-PBGO algorithm for interval analysis. The numerical implementation procedure of the proposed T-PBGO algorithm, which is also illustrated in Fig. 1, includes the following main steps:

Step 1: Define the problem and initialize the optimization

Define the minimization and maximization problem to be solved in terms of its ojective function $g(\boldsymbol{x})$ and feasible region $\boldsymbol{x}^I$, as in Eqs. (2) and (3). Initialize the parameters of the proposed T-PBQO method, namely, the initial sample size n_0, and two thresholds $\varepsilon_{\min}$ and

$\varepsilon_{\max}$. **Details about these parameters and possible numerical values for them are discussed below.**

Step 2: Generate an initial training dataset

Generate an initial set of n_0 samples using Latin hypercube sampling (LHS) over $\boldsymbol{x}^I$, denoted by a $n_0 \times d$ matrix $\boldsymbol{X} = \left\{\boldsymbol{x}^{(j)}\right\}_{j=1}^{n_0}$. Observations of the g-function at these points can be computed in parallel, which are denoted by a $n_0 \times 1$ vector $\boldsymbol{y} = \left\{y^{(j)}\right\}_{j=1}^{n_0}$ with $\boldsymbol{y}^{(j)} = g(\boldsymbol{x}^{(j)})$. The initial training dataset is defined as $\boldsymbol{\mathcal{D}} = \{\boldsymbol{X}, \boldsymbol{y}\}$. Set $n = n_0$.

As we seek to enlarge the training dataset sequentially, the initial size n_0 should not be chosen too large and it is usually set as 5-10.

Step 3: Construct a GP model for the g-function

Construct a GP model $\mathcal{GP}(m_n(\boldsymbol{x}), k_n(\boldsymbol{x}, \boldsymbol{x}'))$ for $y = g(\boldsymbol{x})$ based on the training dataset $\boldsymbol{\mathcal{D}}$. This step mainly consists of specifying the hyper-parameters by using the maximum likelihood estimation. All the numerical examples in this study are performed with the *fitrgp* function in Matlab Statistics and Machine Learning Toolbox.

Step 4: Check the predefined criteria and select the engine

In this stage, we first need to check whether the GP has achieved reasonable accuracy at both the minimum and maximum. If not, the GP should be then improved further, and this improvement means computing additional points. Thus, it should be clear what kind of additional points is still required, for minimization, maximization or both. Let $y_{\min} = \min_{1 \le j \le n} y^{(j)}$ and $y_{\max} = \max_{1 \le j \le n} y^{(j)}$ denote the minimum and maximum values of y observed so far, respectively. Compute the maxima of $EI_{\min}(\boldsymbol{x})$ and $EI_{\max}(\boldsymbol{x})$ by: $\delta y_1 = \max_{\boldsymbol{x} \in \boldsymbol{x}^I} EI_{\min}(\boldsymbol{x})$ and $\delta y_2 = \max_{\boldsymbol{x} \in \boldsymbol{x}^I} EI_{\max}(\boldsymbol{x})$. In this study, five criteria consisting in judging the ratios of the maximum expected improvements (i.e., δy_1 and δy_2) to the best current minimum and maximum (i.e., $y_{\min}$ and $y_{\max}$) respectively, are given as follows:

- ***Criterion 1 (Stopping criterion)***. If both $\frac{\delta y_1}{|y_{\min}|+\delta} < \varepsilon_{\min}$ and $\frac{\delta y_2}{|y_{\max}|+\delta} < \varepsilon_{\max}$ are satisfied for two successive iterations, go to **Step 7**; Else, check Criterion 2.

- ***Criterion 2***. If $\frac{\delta y_1}{|y_{\min}|+\delta} \ge \varepsilon_{\min}$ and $\frac{\delta y_2}{|y_{\max}|+\delta} \ge \varepsilon_{\max}$, this indicates that the GP could be still not

accurate enough for estimating both the minimum and maximum and one should go to **Step 5c**; Else, check Criterion 3.

- ***Criterion 3***. If $\frac{\delta y_1}{|y_{\min}|+\delta} < \varepsilon_{\min}$ and $\frac{\delta y_2}{|y_{\max}|+\delta} < \varepsilon_{\max}$, this indicates that the GP could be still not accurate enough for both estimating the minimum and maximum (due to potential fake convergence) and one should go to **Step 5c**; Else, check Criterion 4.

- ***Criterion 4***. If $\frac{\delta y_1}{|y_{\min}|+\delta} \geq \varepsilon_{\min}$ and $\frac{\delta y_2}{|y_{\max}|+\delta} < \varepsilon_{\max}$, this indicates that the GP could be still not accurate for estimating the minimum one should go to **Step 5a**; Else, check Criterion 5.

- ***Criterion 5***. If $\frac{\delta y_1}{|y_{\min}|+\delta} < \varepsilon_{\min}$ and $\frac{\delta y_2}{|y_{\max}|+\delta} \geq \varepsilon_{\max}$, this indicates that the GP could be still not accurate enough for estimating the maximum one should go to **Step 5b**.

In Criteria 1-5, δ is a small number to ensure that the denominators are always greater than zero, which is specified as 10^{-6} in this study. It should be noted that these two quantities $\frac{\delta y_1}{|y_{\min}|+\delta}$ and $\frac{\delta y_2}{|y_{\max}|+\delta}$ play a pivotal role for our decision-making. The first one represents the ratio of maximum expected improvement for the minimum to the current absolute minimum, while the second one is the ratio of maximum expected improvement for the maximum to the current absolute maximum, if δ is treated as zero. When the current GP model is relatively accurate for both the minimum and maximum, it is expected that these two ratios should be very small. Thus, it is appropriate to judge the convergence of the proposed method by monitoring these two ratios. According to our experience, $\varepsilon_{\min}$ and $\varepsilon_{\max}$ can be set in the order of 0.001.

Step 5a: Identify q updating points for minimization (Engine 1)

Identify q updating points for minimization by using the PEI-MIN criterion. The first point is selected by $\boldsymbol{x}_{\min}^{(n+1)} = \arg\max_{\boldsymbol{x}\in\boldsymbol{x}^I} EI_{\min}(\boldsymbol{x})$, the second one $\boldsymbol{x}_{\min}^{(n+2)} = \arg\max_{\boldsymbol{x}\in\boldsymbol{x}^I} PEI_{\min}(\boldsymbol{x}; \boldsymbol{x}_{\min}^{(n+1)})$, and the third one $\boldsymbol{x}_{\min}^{(n+3)} = \arg\max_{\boldsymbol{x}\in\boldsymbol{x}^I} PEI_{\min}(\boldsymbol{x}; \boldsymbol{x}_{\min}^{(n+1)}, \boldsymbol{x}_{\min}^{(n+2)})$, etc. The q updating points can be denoted by $\boldsymbol{X}_{\text{add}} = \left\{\boldsymbol{x}_{\min}^{(n+1)}, \boldsymbol{x}_{\min}^{(n+2)}, \cdots, \boldsymbol{x}_{\min}^{(n+q)}\right\}$. Then, go to **Step 6**.

Step 5b: Identify q updating points for maximization (Engine 2)

Identify q updating points for maximization by using the PEI-MAX criterion. The first point is selected by $\boldsymbol{x}_{\max}^{(n+1)} = \arg\max_{\boldsymbol{x}\in\boldsymbol{x}^I} EI_{\max}(\boldsymbol{x})$, the second one $\boldsymbol{x}_{\max}^{(n+2)} = \arg\max_{\boldsymbol{x}\in\boldsymbol{x}^I} PEI_{\max}(\boldsymbol{x}; \boldsymbol{x}_{\max}^{(n+1)})$, and the third one $\boldsymbol{x}_{\max}^{(n+3)} = \arg\max_{\boldsymbol{x}\in\boldsymbol{x}^I} PEI_{\max}(\boldsymbol{x}; \boldsymbol{x}_{\max}^{(n+1)}, \boldsymbol{x}_{\max}^{(n+2)})$, etc. The q updating points can be denoted by

$\boldsymbol{X}_{\text{add}} = \left\{\boldsymbol{x}_{\max}^{(n+1)}, \boldsymbol{x}_{\max}^{(n+2)}, \cdots, \boldsymbol{x}_{\max}^{(n+q)}\right\}$. Then, go to **Step 6**.

Step 5c: Identify q updating points for both minimization and maximization (Engine 3)

Identify q updating points for both minimization and maximization by using the PEI-MIN-MAX criterion. The first point is selected by $\boldsymbol{x}_{\min}^{(n+1)} = \arg\max_{\boldsymbol{x}\in\boldsymbol{x}^I} EI_{\min}(\boldsymbol{x})$, the second one $\boldsymbol{x}_{\max}^{(n+2)} = \arg\max_{\boldsymbol{x}\in\boldsymbol{x}^I} PEI_{\max}(\boldsymbol{x}; \boldsymbol{x}_{\min}^{(n+1)})$ and the third one $\boldsymbol{x}_{\min}^{(n+3)} = \arg\max_{\boldsymbol{x}\in\boldsymbol{x}^I} PEI_{\min}(\boldsymbol{x}; \boldsymbol{x}_{\min}^{(n+1)}, \boldsymbol{x}_{\max}^{(n+2)})$, etc. The q updating points can be denoted by $\boldsymbol{X}_{\text{add}} = \left\{\boldsymbol{x}_{\min}^{(n+1)}, \boldsymbol{x}_{\max}^{(n+2)}, \cdots, \boldsymbol{x}_{\max}^{(n+q)}\right\}$. Then, go to **Step 6**.

Step 6: Enrich the training dataset

The q updating points $\boldsymbol{X}_{\text{add}}$ are evaluated on the g-function in parallel, and the corresponding observations are denoted by $\boldsymbol{y}_{\text{add}} = \left\{y^{(n+1)}, y^{(n+2)}, \cdots, y^{(n+q)}\right\}$. The training dataset $\boldsymbol{\mathcal{D}}$ is enriched by $\boldsymbol{\mathcal{D}}_{\text{add}} = \{\boldsymbol{X}_{\text{add}}, \boldsymbol{y}_{\text{add}}\}$, i.e., $\boldsymbol{\mathcal{D}} = \boldsymbol{\mathcal{D}} \cup \boldsymbol{\mathcal{D}}_{\text{add}}$. Set $n = n + q$ and then go to **Step 2**.

Step 7: Record results and end the algorithm

Record $y_{\min} = \min_{1\le j\le n} y^{(j)}$ and $y_{\max} = \max_{1\le j\le n} y^{(j)}$ as approximate solutions to the lower and upper bounds of y^I respectively, and end the algorithm.

In Steps 4 and 5a-5c, the involved optimization problems are solved by a nature-inspired global optimizer, called Teaching–learning-based optimization (TLBO) [42], as they are usually much more cheaper compared to one call of the computational model. As the proposed method is rooted in the classical BGO method, its theoretical analysis may refer to, e,g., [43], which, however, is beyond the scope of the present study.

The proposed method has four major advantages. First, the technique often requires relatively few g-function evaluations. This is possible because one can incorporate prior knowledge to explore the design space. Second, our method allows a high-level parallelization as the proposed T-PEI criterion is computationally tractable for selecting multiple informative and diverse points. This feature further makes the method time-saving when parallel computing is available. Third, the proposed method is derivative-free and directly works with black-boxes, and thus is easy to implement and widely applicable (e.g., no matter the g-function is linear or non-linear and how large the supports of the input intervals are). Fourth, accurate

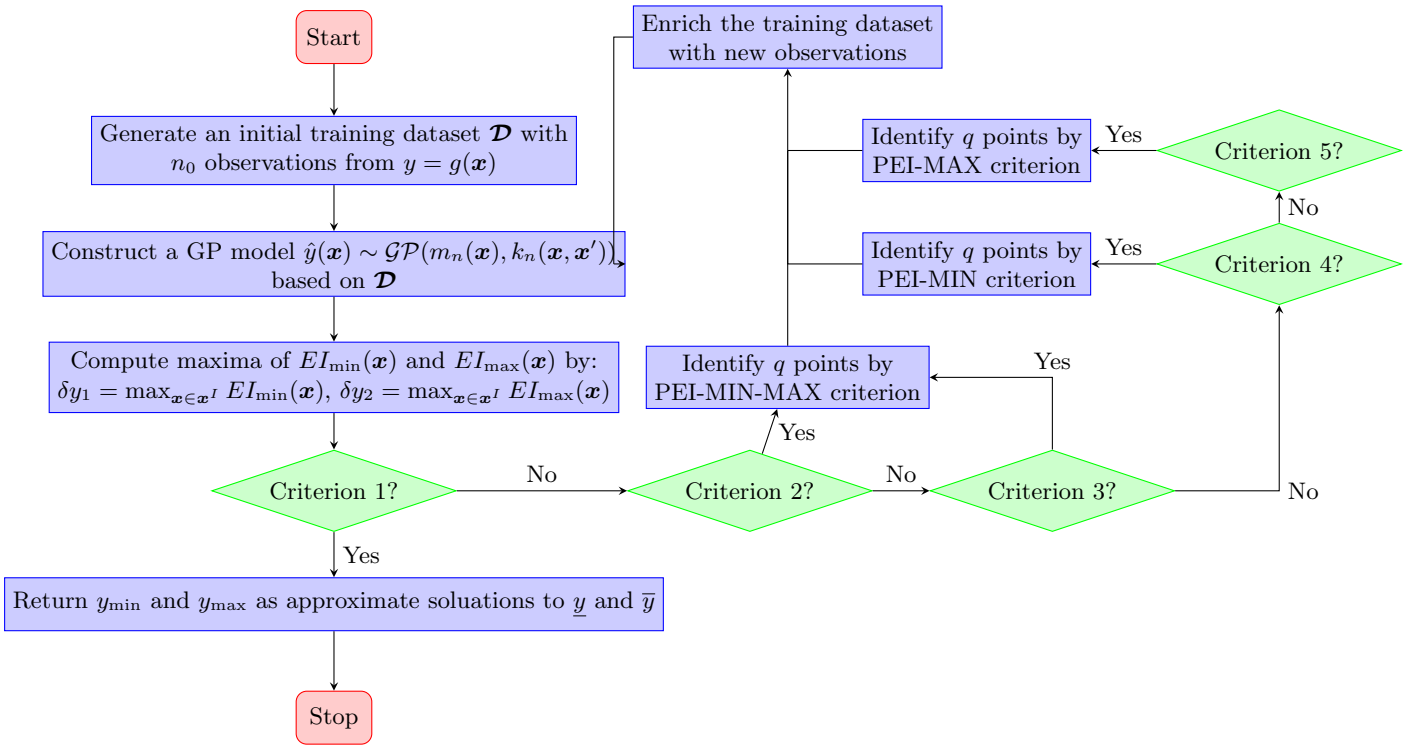

Figure 1: Flowchart of the proposed T-PBGO method.

approximate solutions to both lower and upper bounds of model response can be obtained with only one single run of the proposed algorithm.

3.4. Relationship to existing PBGO approaches

With the emergence of the classical BGO (originally called efficient global optimization) [32], there has been an growing interest to enable its capability of parallel processing. Representative works of PBGO include the q-EI criterion [44–46], multi-modal EI criterion [47, 48], PEI [41], Kriging Believer or Constant Liar strategy [45] and multiple surrogate assisted approach [49, 50], etc. The T-PEI criterion in the proposed T-PBGO method can be regarded as an improved PEI. The difference between the proposed method and the other PBGO methods is significant. The objective of the proposed method is to obtain both the minimum and maximum in one single run, while the other methods are only designed for minimum or maximum, not both.

4. Numerical examples

In order to illustrate and verify the proposed method, four numerical examples are studied in this section. These examples cover a wide range of types, from simple test problems to real-world applications. In all numerical examples, the proposed method is compared with several existing methods in terms of efficiency and accuracy. Besides, we propose a non-parallel BGO (N-PBGO) (given in Appendix A) as a potential competitor for the proposed method, which is also conducted for comparison.

4.1. Example 1: A one-dimensional test function

The first example consists of a test function with one interval:

$$y = g(x) = (2x-1)^2 \sin\left(4\pi x - \frac{\pi}{8}\right),$$

where $x \in [0, 1]$. As can also be seen in Fig. 2, the g-function is multi-modal and has multiple maxima and minima.

The lower and upper bounds of y are computed by the analytical method, vertex method, genetic algorithm, N-PBGO and proposed T-PBGO method ($n_0 = 5$ and $\varepsilon_{\min} = \varepsilon_{\max} = 0.002$). The results are summarized in Table 1 together with the total number of function evaluations N, and the number of iterations $N^\star$. Although the vertex method outperforms the other numerical methods in terms of both N and $N^\star$, it produces totally wrong estimates for the response bounds. The inaccuracy of the interval method is caused by its underlying assumption that y should be monotonic with respect to $\boldsymbol{x}$. As a representative of nature-inspired optimization algorithms, the genetic algorithm is able to yield accurate results, but at the expense of large computation cost. The N-PBGO method requires a relatively small number of function evaluations ($N = 16$), while still providing good results for both the lower and upper bounds. The N-PBGO method, however, is limited by its non-parallelism. On the contrary, the proposed T-PBGO method can overcome this limitation by taking advantage of the developed infill sampling criterion (i.e., T-PEI). Compared to N-PBGO, T-PBGO can significantly reduce the function evaluations in terms of $N^\star$, while still maintaining high accuracy. In addition, it also can be found that $N^\star$ gradually decreases with the increase of q, and remains the same when $q = 8, 10$, though N also increases non-monotonously.

Table 1: Interval analysis for Example 1 by different methods.

Method		Lower bound	Upper bound	N	$N^\star$	Reference
Exact solution		-0.7081	0.5197	-	-	-
Vertex method ($q = 2$)		-0.3827	-0.3827	2	1	[26]
Genetic algorithm ($q = 10$)		-0.7081	0.5197	$520 + 520 = 1040$	104	[51]
N-PBGO ($q = 1$)		-0.7081	0.5197	$5 + 6 + 5 = 16$	16	Appendix A
Proposed method (T-PBGO)	$q = 2$	-0.7081	0.5197	$5 + 8 = 13$	7	-
	$q = 4$	-0.7081	0.5197	$5 + 16 = 21$	6	-
	$q = 6$	-0.7081	0.5197	$5 + 24 = 29$	5	-
	$q = 8$	-0.7081	0.5197	$5 + 24 = 29$	4	-
	$q = 10$	-0.7081	0.5197	$5 + 30 = 35$	4	-

Note: N = the total number of g-function evaluations, and $N^\star$ = the number of iterations

To visually illustrate the proposed method, one special case is considered here (i.e., $q = 4$). It can be observed from Fig. 2 that the proposed method gradually approaches to the exact bounds as the iterative process goes on. Besides, these added points are more densely distributed around the global minimum and maximum, and thereby informative for our purposes.

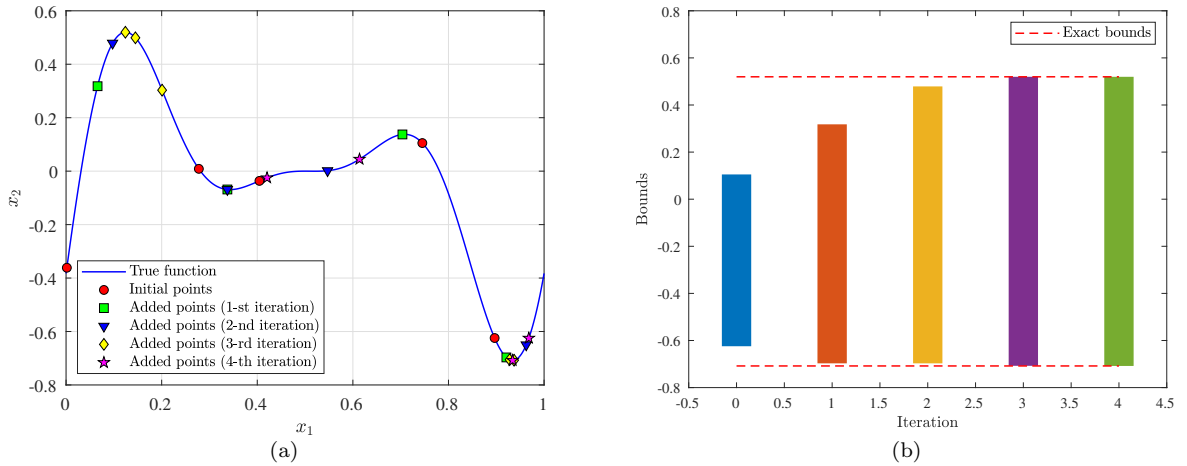

Figure 2: Illustration of the proposed method ($q = 4$) in Example 1: (a) True function, initial points and added points identified by T-PEI criterion; (b) Exact bounds and approximate bounds after each iteration.

4.2. Example 2: A two-dimensional test function

The second example takes a test function with two intervals [21]:

$$y = g(\boldsymbol{x}) = (1.5x_1 - 2)^2 - (x_2 - 3)^2 + x_1 x_2 + 10\sin(2\pi x_1) + 10\sin(2\pi x_2),$$

where $x_1, x_2 \in [2, 5]$. As shown in Fig. 3, the test function is highly nonlinear and has several local optima over the prescribed region.

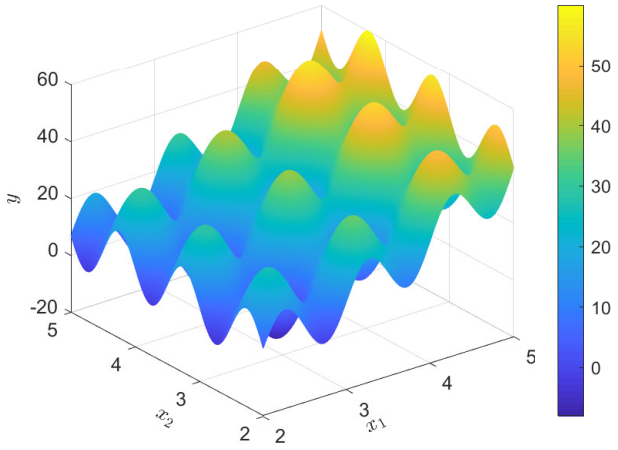

Figure 3: Plot of the two-dimensional test function in Example 2.

The lower and upper bounds of y are computed by several methods, as listed in Table 2. The exact response bounds of y are obtained as -8.10 and 59.95. The genetic algorithm can yield accurate results, but at the expense of 4000 g-function evaluations. Although the classical vertex method requires the minimum number of g-function evaluations among all the numerical methods, it gives completely wrong estimates for the lower and upper bounds. At the cost of 6912 g-function calls (the largest among all the numerical methods), the subinterval method is able to produce acceptable results. The subinterval decomposition analysis method yields close results to these of the subinterval method, while requires significantly less g-function evaluations. For the N-PGBO method, faily good results can be produced using a total number of 74 g-function evaluations, and 65 iterations. The proposed T-PBGO method

($n_0 = 10$, $\varepsilon_{\min} = 0.002$ and $\varepsilon_{\max} = 0.001$) is capable of generating quite accurate lower and upper bounds, while reducing the number of iterations down to 9 when $q = 8$.

Table 2: Interval analysis results for Example 2 by different methods.

Method		Lower bound	Upper bound	N	$N^\star$	Reference
Exact solution		-8.10	59.95	-	-	-
Genetic algorithm		-8.10	59.95	4000	-	Tab. 7 in [21]
Vertex method ($q = 4$)		4.00	51.25	4	1	[26]
Subinterval method		-8.70	60.39	6912	-	Tab. 7 in [21]
Subinterval decomposition analysis		-8.55	58.81	97	-	Tab. 7 in [21]
N-PBGO ($q = 1$)		-8.01	59.92	10 + 42 + 22 = 74	65	Appendix A
Proposed method (T-PBGO)	$q = 2$	-8.08	59.94	10 + 58 = 68	30	-
	$q = 4$	-8.08	59.94	10 + 72 = 82	19	-
	$q = 6$	-8.09	59.93	10 + 72 = 82	13	-
	$q = 8$	-8.10	59.94	10 + 80 = 90	9	-
	$q = 10$	-8.10	59.93	10 + 90 = 100	10	-

4.3. Example 3: A transmission tower subjected to wind loads

This example consists of a transmission tower subjected to wind loads (shown in Fig. 4), which is modified from Ref. [52]. The tower is modelled as a three-dimensional (3D) truss structure with 24 joints and 80 elements in OpenSees. Three kinds of members, i.e., columns, diagonal members and horizontal members, are included in the model, the cross-sectional area of which are denoted as A_1, A_2 and A_3, respectively. The geometry of the model is shown in Fig. 4(a). The wind effect acting on the tower is simplified to four equivalent static loads at the top four nodes, and inclined by θ° relative to the x-axis (Fig. 4(b)). The constitutive law of the steel material adopts the bi-linear model, as depicted in Fig. 4(c). Eight interval variables are included in the 3D truss model, which are descried in Table 3. The response of interest is

defined as the horizontal displacement of node A, i.e.,

$$y = g(P, \theta, F_y, E, b, A_1, A_2, A_3) = \sqrt{u_{\mathrm{A},x}^2 + u_{\mathrm{A},y}^2},$$

where $u_{\mathrm{A},x}$ and $u_{\mathrm{A},y}$ denote the displacements of node A in x and y directions, respectively.

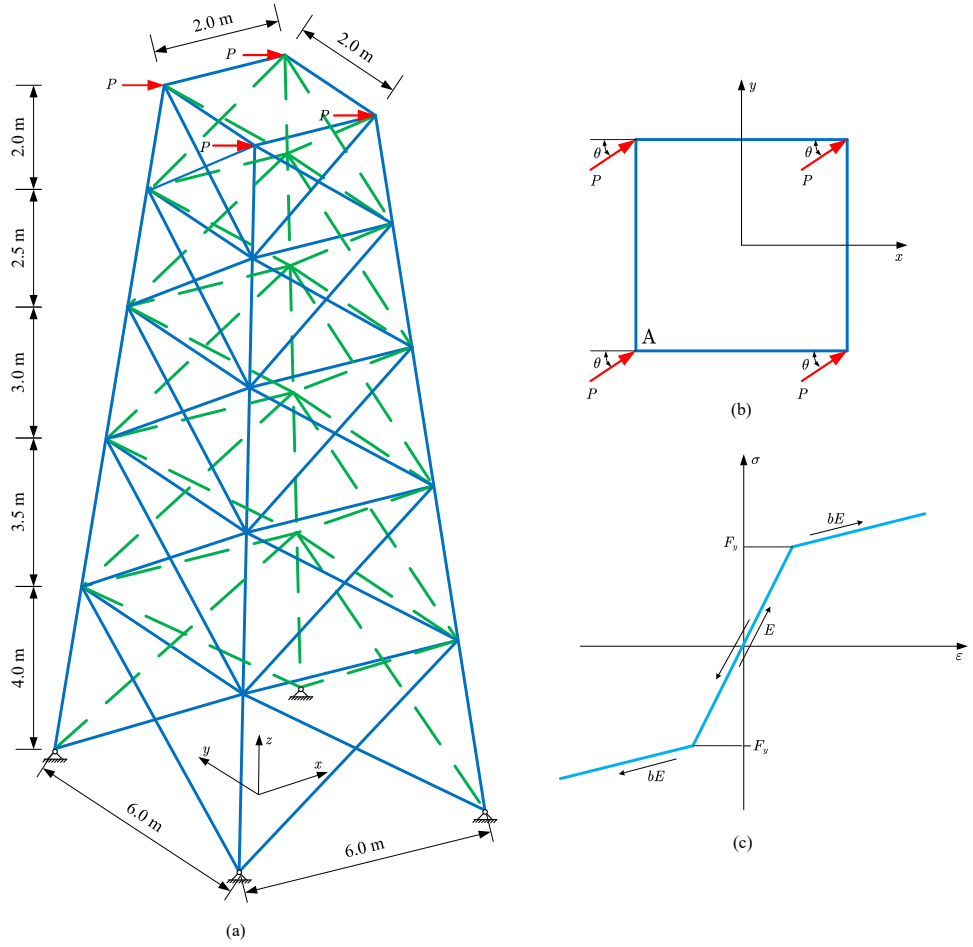

Figure 4: A transmission tower subject to wind loads: (a) 3D truss model; (b) loading at the top of tower; (c) bi-linear constitutive model.

The bounds of y are solved by several methods, and the results are summarized in Table 4. The particle swarm optimization ($q = 10$) is used to provide reference results for the bounds. For the proposed T-PBGO method, we set the user-specified parameters as: $n_0 = 10$, $\varepsilon_{\min} = 0.002$ and $\varepsilon_{\max} = 0.001$. The vertex method requires 256 g-function calls, which, however, greatly underestimates the upper bound. Both N-PBGO and T-PBGO can give close results to these of particle swarm optimization. The N-PBGO method is

Table 3: Interval variables for Example 3.

Variable	Description	Interval	Unit
P	Wind load	$[100, 200]$	kN
θ	Angle between the load direction and the x-axis	$[-45, 45]$	$^\circ$
F_y	Yield strength of steel	$[300, 400]$	MPa
E	Young's modulus of steel	$[1.8, 2.4] \times 10^5$	MPa
b	Strain hardening ratio	$[0.015, 0.025]$	-
A_1	Cross-sectional area of the column members	$[4000, 5000]$	mm^2
A_2	Cross-sectional area of the diagonal members	$[3000, 4000]$	mm^2
A_3	Cross-sectional area of the horizontal members	$[2000, 3000]$	mm^2

computationally advantageous in terms of N among all methods, while the proposed T-PBGO can further reduce $N^\star$ by taking advantage of its parallelism.

Table 4: Interval analysis results for Example 3 by different methods.

Method		Lower bound/mm	Upper bound/mm	N	$N^\star$	Reference
Particle swarm optimization ($q = 10$)		11.9592	57.2421	1920 + 3840 = 5760	576	[51]
Vertex method ($q = 10$)		11.9592	44.3887	256	25.60	[26]
N-PBGO ($q = 1$)		11.9592	57.2421	10 + 9 + 5 = 24	24	Appendix A
Proposed method (T-PBGO)	$q = 2$	11.9592	57.2403	10 + 22 = 32	16	-
	$q = 4$	11.9592	57.2421	10 + 28 = 38	10	-
	$q = 6$	11.9592	57.2372	10 + 36 = 46	8	-
	$q = 8$	11.9592	57.2421	10 + 40 = 50	7	-
	$q = 10$	11.9760	57.2388	10 + 60 = 70	7	-

4.4. Example 4: A spatial frame with viscous dampers subjected to earthquake

The last example considers a spatial frame with viscous dampers subjected to earthquake, as shown in Fig. 5. The 3-D finite element model is developed in OpenSees, the geometry of which can be found in

Fig. 5(a). Each beam/column member is modelled with an elastic beam-column element with cross section IPE270/IPB300 (Fig. 5(b)/(c)). For each viscous damper (see Fig. 5(d)), a two-node link element is used with the viscous damper material. We only consider the self weight as the mass source for the columns, while for beams the mass source is determined based on "self weight + dead load + 0.2 live load". The structure is subjected to a base acceleration corresponding to the N-S component of the El-Centro 1940 earthquake, as shown in 5(e). The ground motion is applied along the direction with a rotation angle θ° with respect to the y-axis (Fig. 5(a)). As summarized in Table 5, eleven interval variables are involved in this example. Of interest is the maximum horizontal displacement of node A, i.e.,

$$y = g(\theta, AF, DL, LL, K_{\mathrm{D}}, C_{\mathrm{D}}, \alpha, \rho, E, v, \zeta) = \max_{t} \sqrt{u_{\mathrm{A},x}^2(t) + u_{\mathrm{A},y}^2(t)},$$

where $u_{\mathrm{A},x}(t)$ and $u_{\mathrm{A},y}(t)$ denote the displacements of node A in x and y directions, respectively.

Table 5: Interval variables for Example 4.

Variable	Description	Interval	Unit
θ	Angle between the earthquake direction and the y-axis	$[-45, 45]$	$^{\circ}$
AF	Amplification factor of the earthquake ground motion	$[0.5, 1.5]$	-
DL	Floor dead load	$[4, 5]$	kN/m^2
LL	Floor live load	$[2, 3]$	kN/m^2
K_{D}	Axial Stiffness of the viscous damper	$[3, 4] \times 10^4$	kN/m
C_{D}	Damping coefficient of the viscous damper	$[20, 30]$	kN(s/m)$^{\alpha}$
α	Velocity exponent	$[0.2, 0.4]$	-
ρ	Density of steel	$[7.8, 7.9] \times 10^3$	kg/m^3
E	Young's modulus of steel	$[1.8, 2.2] \times 10^5$	MPa
v	Poisson's ratio	$[0.25, 0.30]$	-
ζ	Damping ratio	$[0.02, 0.04]$	-

The bounds of the model response y are computed by the particle swarm optimization, vertex method, N-PBGO and T-PBGO ($n_0 = 10$, $\varepsilon_{\min} = 0.002$ and $\varepsilon_{\max} = 0.001$), and the results are summarized in Table 6. The reference solution is taken from the particle swarm optimization method. The vertex method is able

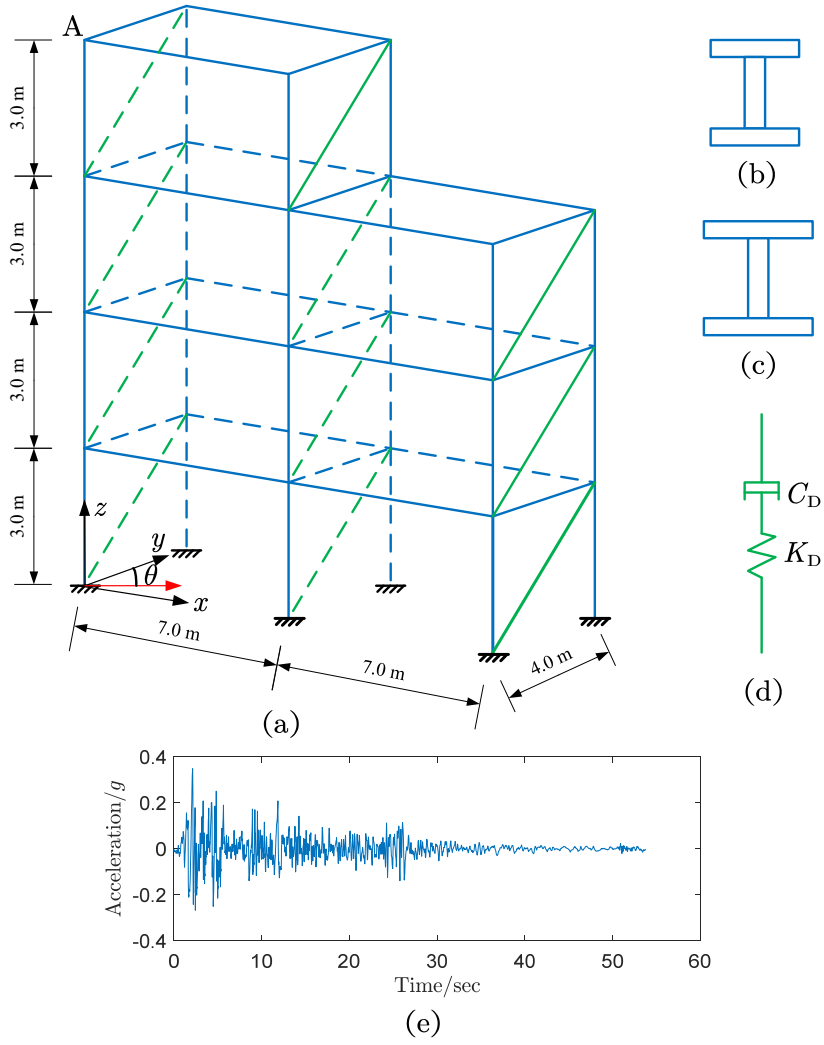

Figure 5: A spatial frame with viscous dampers subject to earthquake: (a) 3D frame model; (b) IPE270 for beams; (c) IPB300 for columns; (4) Viscous Damper; (e) N-S component of El Centro earthquake (1940)

to produce good estimates, but requires a large number of g-function evaluations ($N = 2048$ and $N^\star = 204.8$) in this example. Compared to the N-PBGO method and vertex method, the proposed T-PBGO method can significantly reduce the number of g-function calls per core, though the total number of g-function calls may increase (e.g., $q = 4, 8$) relative to the N-PBGO method. Besides, the proposed method still gives desirable results for the response bounds. It should be emphasized that $N^\star$ does not decrease monotonically as q increases. This means that there may be an optimal parallelization level q that minimizes $N^\star$, e.g., $q = 6$ in the example.

Table 6: Interval analysis results for Example 4 by different methods.

Method		Lower bound/mm	Upper bound/mm	N	$N^\star$	Reference
Particle swarm optimization ($q = 10$)		11.9762	137.4651	3000 + 2400 = 5400	540	[51]
Vertex method ($q = 10$)		12.0084	137.4651	2048	204.8	[26]
N-PBGO ($q = 1$)		12.0929	137.3746	10 + 14 + 4 = 28	28	Appendix A
Proposed method (T-PBGO)	$q = 2$	12.0483	137.4651	10 + 14 = 24	12	-
	$q = 4$	12.0084	137.2062	10 + 24 = 34	9	-
	$q = 6$	12.0489	137.4651	10 + 18 = 28	5	-
	$q = 8$	12.0063	137.4651	10 + 32 = 42	6	-

4.5. Finial remarks

In practical applications, the g-function can be rather expensive-to-evaluate and the computational budget is limited. In such cases, one may need to prespecify optimal values for the parameters n_0, q, $\varepsilon_{\min}$ and $\varepsilon_{\max}$ before running the proposed method in order to save the computational time, while remaining a desired level of accuracy. As a rule of thumb, the initial sample size n_0 can be set as 10. As observed in the four numerical examples, the number of iterations $N^\star$ does not decrease monotonically with q and takes its minimum value when $q = 8$ in most cases. Therefore, $q = 8$ is recommended in case that at least 8 cores are available. The two thresholds $\varepsilon_{\min}$ and $\varepsilon_{\max}$ not only influence the the efficiency of the proposed method, but also the accuracy, The smaller $\varepsilon_{\min}$ and $\varepsilon_{\max}$ are, the proposed method usually requires more iterations and more accurate results can be obtained. According to our experience, $\varepsilon_{\min} = 0.002$ and $\varepsilon_{\max} = 0.002$ can be adopted.

5. Conclusions

In this study, a triple-engine parallel Bayesian global optimization (T-PBGO) method is proposed for efficient interval numerical analysis, **especially when the computational model is a expensive-to-evaluate black box.** The advancement of the proposed method lies in utilizing the Gaussian process (GP,

also known as Kriging) prior for the expensive black-box g-function and an acquisition function (or infill sampling criterion) that can suggest promising points to be evaluated next. To order to make full use of prior knowledge and parallel computing, the main contribution of this paper is the development of a multi-points selection strategy, called 'triple-engine pseudo expected improvement' (T-PEI), which can select a batch of informative and diversity points for minimization and/or maximization at each iteration. Four numerical examples are investigated to demonstrate the proposed method. The main advantages of T-PBGO can be summarized as follows:

(i) The proposed method usually requires less g-function evaluations to achieve the same accuracy compared to non-Bayesian methods, due to its ability to exploit prior knowledge;

(ii) Compared to N-PBGO, T-PBGO allows for identifying multiple points at each iteration, and hence could be more efficient when parallel computing is available;

(iii) The developed method is non-intrusive in nature (directly works with black-box problems), and therefore easy-to-implement and broadly applicable;

(iv) Both lower and upper bounds can be obtained with one single run of the proposed method.

However, the proposed method still has several major limitations. First, T-PBGO typically works only well in low dimensions (typically, $d < 20$), and for high-dimensional problems new developments are needed. Second, as the parallelization level q and the size of training dataset increase, optimizing the T-PEI criterion can be time-consuming. Third, only the bounds of a single model response can be captured by the proposed method in its current form. Future works can be done along these directions.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

Chao Dang is mainly supported by China Scholarship Council (CSC). Pengfei Wei is grateful to the support from the National Natural Science Foundation of China (grant no. 51905430 and 72171194). Marcos Valdebenito acknowledges the support by ANID (National Agency for Research and Development, Chile) under its program FONDECYT, grant number 1180271. Chao Dang, Pengfei Wei and Michael Beer also would like to appreciate the support of Sino-German Mobility Program under grant number M-0175.

Appendix A. Non-parallel Bayesian global optimization

The traditional Bayesian global optimization is sequential in nature, which means that only one update point is identified at each iteration. Therefore, it cannot take advantage of parallelism. Besides, finding the minimum and maximum of a function is typically treated as two separate optimization problems. However, this is not advisable when computational efficiency is of great concern. That is because that the observations obtained when searching the minimum can be reused to speed up searching the maximum, and vice versa. This strategy is adopted in this study as a potential competitor to the proposed method, and we simply call it non-parallel Bayesian global optimization (N-PBGO). The main procedure of N-PBGO is summarized as follows:

Step A.1: Generate an initial training dataset

Generate an initial set of n_0 samples using LHS over $\boldsymbol{x}^I$, denoted by a $n_0 \times d$ matrix $\boldsymbol{X} = \left\{\boldsymbol{x}^{(j)}\right\}_{j=1}^{n_0}$. Observations of the g-function at these points can be computed in parallel, which are denoted by a $n_0 \times 1$ vector $\boldsymbol{y} = \left\{y^{(j)}\right\}_{j=1}^{n_0}$ with $\boldsymbol{y}^{(j)} = g(\boldsymbol{x}^{(j)})$. The initial training dataset can be written as $\boldsymbol{\mathcal{D}} = \{\boldsymbol{X}, \boldsymbol{y}\}$. Set $n = n_0$.

Step A.2: Construct a GP model

Construct a GP model $\mathcal{GP}(m_n(\boldsymbol{x}), k_n(\boldsymbol{x}, \boldsymbol{x}'))$ based on the initial training dataset $\boldsymbol{\mathcal{D}}$. This step mainly consists of choosing the hyper-parameters by using the maximum likelihood estimation. All the numerical

examples in this study are performed with the *fitrgp* function in Matlab Statistics and Machine Learning Toolbox.

Step A.3: Compute maximum of $EI_{\mathbf{min}}(\boldsymbol{x})$

Let $y_{\min} = \min_{1\le j\le n} y^{(j)}$ denote the minimum value of y observed so far, respectively. Compute the maximum of $EI_{\min}(\boldsymbol{x})$ by $\delta y_1 = \max_{\boldsymbol{x}\in\boldsymbol{x}^I} EI_{\min}(\boldsymbol{x})$.

Step A.4: Check stopping criterion for minimization

if $\frac{\delta y_1}{|y_{\min}|+\delta} < \varepsilon_{\min}$ is satisfied for two successive times, go to **Step A.7**; Otherwise, go to **Step A.5**.

Step A.5: Identify one point by EI-MIN criterion

Identify the next point to evaluate by $\boldsymbol{x}_{\min}^{(n+1)} = \arg\max_{\boldsymbol{x}\in\boldsymbol{x}^I} EI_{\min}(\boldsymbol{x})$.

Step A.6: Enrich the training dataset

Compute the corresponding g-function value at the identified point at $\boldsymbol{x}_{\min}^{(n+1)}$, i.e., $y^{(n+1)} = g(\boldsymbol{x}_{\min}^{(n+1)})$. Enrich the training dataset $\boldsymbol{\mathcal{D}}$ with $(\boldsymbol{x}_{\min}^{(n+1)}, y^{(n+1)})$. Set $n = n+1$, and go to **Step A.2**.

Step A.7: Compute maximum of $EI_{\mathbf{max}}(\boldsymbol{x})$

Let $y_{\max} = \max_{1\le j\le n} y^{(j)}$ denote the maximum value of y observed so far, respectively. Compute the maxima of $EI_{\max}(\boldsymbol{x})$ by $\delta y_2 = \max_{\boldsymbol{x}\in\boldsymbol{x}^I} EI_{\max}(\boldsymbol{x})$.

Step A.8: Check stopping criterion for maximization

if $\frac{\mu_{\max}^{\max}}{|y_{\max}|+\delta} < \varepsilon_{\max}$ is satisfied for two successive times, go to **Step A.12**; Otherwise, go to **Step A.9**.

Step A.9: Identify one point by EI-MAX criterion

Identify the next point to evaluate by $\boldsymbol{x}_{\max}^{(n+1)} = \arg\max_{\boldsymbol{x}\in\boldsymbol{x}^I} EI_{\max}(\boldsymbol{x})$.

Step A.10: Enrich the training dataset

Compute the corresponding g-function value at the identified point at $\boldsymbol{x}_{\max}^{(n+1)}$, i.e., $y^{(n+1)} = g(\boldsymbol{x}_{\max}^{(n+1)})$. Enrich the training dataset $\boldsymbol{\mathcal{D}}$ with $(\boldsymbol{x}_{\max}^{(n+1)}, y^{(n+1)})$. Set $n = n+1$.

Step A.11: Construct a GP model

Construct a GP model $\mathcal{GP}(m_n(\boldsymbol{x}), k_n(\boldsymbol{x}, \boldsymbol{x}'))$ based on the initial training dataset $\boldsymbol{\mathcal{D}}$, and go to **Step A.7**.

Step A.12: End the algorithm

Take $y_{\min} = \min_{1\leq j\leq n} y^{(j)}$ and $y_{\max} = \max_{1\leq j\leq n} y^{(j)}$ as approximate solutions to the lower and upper bounds of y respectively, and end the algorithm.

In the above steps, TLBO is used for all optimization problems. Besides, for fair comparison the user-specified parameters (n_0, δ, $\varepsilon_{\min}$ and $\varepsilon_{\max}$) are set according to the proposed method in all numerical examples.

References

[1] J. Stoer, R. Bulirsch, Introduction to numerical analysis, Vol. 12, Springer Science & Business Media, 2013.

[2] J. C. Helton, J. D. Johnson, W. L. Oberkampf, An exploration of alternative approaches to the representation of uncertainty in model predictions, Reliability Engineering & System Safety 85 (1-3) (2004) 39–71. doi:https://doi.org/10.1016/j.ress.2004.03.025.

[3] M. Beer, S. Ferson, V. Kreinovich, Imprecise probabilities in engineering analyses, Mechanical Systems and Signal Processing 37 (1-2) (2013) 4–29. doi:https://doi.org/10.1016/j.ymssp.2013.01.024.

[4] T. Augustin, F. P. Coolen, G. De Cooman, M. C. Troffaes, Introduction to imprecise probabilities, John Wiley & Sons, 2014.

[5] M. Faes, D. Moens, Recent trends in the modeling and quantification of non-probabilistic uncertainty, Archives of Computational Methods in Engineering 27 (3) (2020) 633–671. doi:https://doi.org/10.1007/s11831-019-09327-x.

[6] C. Jiang, X. Han, G. Lu, J. Liu, Z. Zhang, Y. Bai, Correlation analysis of non-probabilistic convex model and corresponding structural reliability technique, Computer Methods in Applied Mechanics and Engineering 200 (33-36) (2011) 2528–2546. doi:https://doi.org/10.1016/j.cma.2011.04.007.

[7] R. E. Moore, Methods and applications of interval analysis, SIAM, 1979.

[8] C. Jiang, Q. Zhang, X. Han, J. Liu, D. Hu, Multidimensional parallelepiped model—a new type of non-probabilistic convex model for structural uncertainty analysis, International Journal for Numerical Methods in Engineering 103 (1) (2015) 31–59. doi:https://doi.org/10.1002/nme.4877.

[9] B. Ni, C. Jiang, X. Han, An improved multidimensional parallelepiped non-probabilistic model for structural uncertainty analysis, Applied Mathematical Modelling 40 (7-8) (2016) 4727–4745.

[10] C. Jiang, C.-M. Fu, B.-Y. Ni, X. Han, Interval arithmetic operations for uncertainty analysis with correlated interval variables, Acta Mechanica Sinica 32 (4) (2016) 743–752. doi:https://doi.org/10.1007/s10409-015-0525-3.

[11] M. Faes, D. Moens, Multivariate dependent interval finite element analysis via convex hull pair constructions and the

519 extended transformation method, Computer Methods in Applied Mechanics and Engineering 347 (2019) 85–102. doi:
520 https://doi.org/10.1016/j.cma.2018.12.021.
521 [12] M. Faes, D. Moens, On auto-and cross-interdependence in interval field finite element analysis, International Journal for
522 Numerical Methods in Engineering 121 (9) (2020) 2033–2050. doi:https://doi.org/10.1002/nme.6297.
523 [13] D. Wu, W. Gao, Hybrid uncertain static analysis with random and interval fields, Computer Methods in Applied Mechanics
524 and Engineering 315 (2017) 222–246. doi:https://doi.org/10.1016/j.cma.2016.10.047.
525 [14] A. Sofi, E. Romeo, O. Barrera, A. Cocks, An interval finite element method for the analysis of structures with spatially
526 varying uncertainties, Advances in Engineering Software 128 (2019) 1–19. doi:https://doi.org/10.1016/j.advengsoft.
527 2018.11.001.
528 [15] I. Elishakoff, Y. Miglis, Novel parameterized intervals may lead to sharp bounds, Mechanics Research Communications 44
529 (2012) 1–8. doi:https://doi.org/10.1016/j.mechrescom.2012.04.004.
530 [16] I. Elishakoff, K. Thakkar, Overcoming overestimation characteristic to classical interval analysis, AIAA Journal 52 (9)
531 (2014) 2093–2097. doi:https://doi.org/10.2514/1.J053152.
532 [17] G. Manson, Calculating frequency response functions for uncertain systems using complex affine analysis, Journal of Sound
533 and Vibration 288 (3) (2005) 487–521. doi:https://doi.org/10.1016/j.jsv.2005.07.004.
534 [18] G. Muscolino, A. Sofi, Stochastic analysis of structures with uncertain-but-bounded parameters via improved interval
535 analysis, Probabilistic Engineering Mechanics 28 (2012) 152–163. doi:https://doi.org/10.1016/j.probengmech.2011.
536 08.011.
537 [19] Z. Qiu, L. Ma, X. Wang, Non-probabilistic interval analysis method for dynamic response analysis of nonlinear systems
538 with uncertainty, Journal of Sound and Vibration 319 (1-2) (2009) 531–540. doi:https://doi.org/10.1016/j.jsv.2008.
539 06.006.
540 [20] Z. Deng, Z. Guo, X. Zhang, Non-probabilistic set-theoretic models for transient heat conduction of thermal protection
541 systems with uncertain parameters, Applied Thermal Engineering 95 (2016) 10–17. doi:https://doi.org/10.1016/j.
542 applthermaleng.2015.10.152.
543 [21] C. Fu, L. Cao, J. Tang, X. Long, A subinterval decomposition analysis method for uncertain structures with large
544 uncertainty parameters, Computers & Structures 197 (2018) 58–69. doi:https://doi.org/10.1016/j.compstruc.2017.
545 12.001.
546 [22] M. Valdebenito, C. Pérez, H. Jensen, M. Beer, Approximate fuzzy analysis of linear structural systems applying intervening
547 variables, Computers & Structures 162 (2016) 116–129. doi:https://doi.org/10.1016/j.compstruc.2015.08.020.
548 [23] M. A. Valdebenito, H. A. Jensen, P. Wei, M. Beer, A. T. Beck, Application of a reduced order model for fuzzy analysis
549 of linear static systems, ASCE-ASME Journal of Risk and Uncertainty in Engineering Systems, Part B: Mechanical
550 Engineering 7 (2) (2021) 020904.
551 [24] J. Wu, Z. Luo, Y. Zhang, N. Zhang, L. Chen, Interval uncertain method for multibody mechanical systems using Chebyshev

inclusion functions, International Journal for Numerical Methods in Engineering 95 (7) (2013) 608–630. doi:https://doi.org/10.1002/nme.4525.

[25] J. Wu, Y. Zhang, L. Chen, Z. Luo, A Chebyshev interval method for nonlinear dynamic systems under uncertainty, Applied Mathematical Modelling 37 (6) (2013) 4578–4591. doi:https://doi.org/10.1016/j.apm.2012.09.073.

[26] W. Dong, H. C. Shah, Vertex method for computing functions of fuzzy variables, Fuzzy sets and Systems 24 (1) (1987) 65–78. doi:https://doi.org/10.1016/0165-0114(87)90114-X.

[27] Z. Qiu, Y. Xia, J. Yang, The static displacement and the stress analysis of structures with bounded uncertainties using the vertex solution theorem, Computer Methods in Applied Mechanics and Engineering 196 (49-52) (2007) 4965–4984. doi:https://doi.org/10.1016/j.cma.2007.06.022.

[28] R. R. Callens, M. G. Faes, D. Moens, Interval analysis using multilevel quasi-monte carlo, in: International Workshop on Reliable Engineering Computing (REC2021), Vol. 9, International Workshop on Reliable Engineering Computing, 2021, pp. 53–67.

[29] R. R. Callens, M. G. Faes, D. Moens, Multilevel quasi-monte carlo for interval analysis, International Journal for Uncertainty Quantification (Accepted for Publication).

[30] F. Biondini, F. Bontempi, P. G. Malerba, Fuzzy reliability analysis of concrete structures, Computers & structures 82 (13-14) (2004) 1033–1052. doi:https://doi.org/10.1016/j.compstruc.2004.03.011.

[31] L. Catallo, Genetic anti-optimization for reliability structural assessment of precast concrete structures, Computers & Structures 82 (13-14) (2004) 1053–1065. doi:https://doi.org/10.1016/j.compstruc.2004.03.018.

[32] D. R. Jones, M. Schonlau, W. J. Welch, Efficient global optimization of expensive black-box functions, Journal of Global Optimization 13 (4) (1998) 455–492. doi:https://doi.org/10.1023/A:1008306431147.

[33] M. De Munck, D. Moens, W. Desmet, D. Vandepitte, An efficient response surface based optimisation method for non-deterministic harmonic and transient dynamic analysis, Computer Modeling in Engineering & Sciences 47 (2) (2009) 119–166. doi:https://doi.org/10.3970/cmes.2009.047.119.

[34] Y. Liu, X. Wang, L. Wang, Z. Lv, A bayesian collocation method for static analysis of structures with unknown-but-bounded uncertainties, Computer Methods in Applied Mechanics and Engineering 346 (2019) 727–745. doi:https://doi.org/10.1016/j.cma.2018.08.043.

[35] H.-P. Wan, Y.-Q. Ni, A new approach for interval dynamic analysis of train-bridge system based on bayesian optimization, Journal of Engineering Mechanics 146 (5) (2020) 04020029. doi:https://doi.org/10.1061/(ASCE)EM.1943-7889.0001735.

[36] A. Cicirello, F. Giunta, Machine learning based optimization for interval uncertainty propagation, Mechanical Systems and Signal Processing 170 (2022) 108619. doi:https://doi.org/10.1016/j.ymssp.2021.108619.

[37] Z. Deng, Z. Guo, Interval identification of structural parameters using interval overlap ratio and monte carlo simulation, Advances in Engineering Software 121 (2018) 120–130.

[38] M. Imholz, M. Faes, D. Vandepitte, D. Moens, Robust uncertainty quantification in structural dynamics under scarse

experimental modal data: A bayesian-interval approach, Journal of Sound and Vibration 467 (2020) 114983.

[39] C. Jiang, Z. Zhang, Q. Zhang, X. Han, H. Xie, J. Liu, A new nonlinear interval programming method for uncertain problems with dependent interval variables, European Journal of Operational Research 238 (1) (2014) 245–253.

[40] C. K. Williams, C. E. Rasmussen, Gaussian processes for machine learning, Vol. 2, MIT press Cambridge, MA, 2006.

[41] D. Zhan, J. Qian, Y. Cheng, Pseudo expected improvement criterion for parallel EGO algorithm, Journal of Global Optimization 68 (3) (2017) 641–662. doi:https://doi.org/10.1007/s10898-016-0484-7.

[42] R. V. Rao, V. J. Savsani, D. Vakharia, Teaching–learning-based optimization: a novel method for constrained mechanical design optimization problems, Computer-Aided Design 43 (3) (2011) 303–315. doi:https://doi.org/10.1016/j.cad.2010.12.015.

[43] A. S. Di Perrotolo, A theoretical framework for bayesian optimization convergence, Master's thesis, KTH Royal Institute of Technology (2018).

[44] M. Schonlau, Computer experiments and global optimization, Ph.D. thesis, University of Waterloo (1997).

[45] D. Ginsbourger, R. L. Riche, L. Carraro, Kriging is well-suited to parallelize optimization, in: Computational intelligence in expensive optimization problems, Springer, 2010, pp. 131–162.

[46] C. Chevalier, D. Ginsbourger, Fast computation of the multi-points expected improvement with applications in batch selection, in: International Conference on Learning and Intelligent Optimization, Springer, 2013, pp. 59–69.

[47] A. Sóbester, S. J. Leary, A. J. Keane, A parallel updating scheme for approximating and optimizing high fidelity computer simulations, Structural and multidisciplinary optimization 27 (5) (2004) 371–383.

[48] D. Zhan, J. Qian, Y. Cheng, Balancing global and local search in parallel efficient global optimization algorithms, Journal of Global Optimization 67 (4) (2017) 873–892.

[49] F. A. Viana, R. T. Haftka, L. T. Watson, Efficient global optimization algorithm assisted by multiple surrogate techniques, Journal of Global Optimization 56 (2) (2013) 669–689.

[50] J. C. García-García, R. García-Ródenas, E. Codina, A surrogate-based cooperative optimization framework for computationally expensive black-box problems, Optimization and Engineering 21 (3) (2020) 1053–1093.

[51] MATLAB and Global Optimization Toolbox Release 2018a, The MathWorks Inc., Natick, Massachusetts, United States, 2018.

[52] Q. Gu, M. Barbato, J. P. Conte, P. E. Gill, F. McKenna, OpenSees-SNOPT framework for finite-element-based optimization of structural and geotechnical systems, Journal of Structural Engineering 138 (6) (2012) 822–834. doi:https://doi.org/10.1061/(ASCE)ST.1943-541X.0000511.