Structural and Multidisciplinary Optimization (2019) 59:2131–2154
https://doi.org/10.1007/s00158-018-2182-1

RESEARCH PAPER

# Constrained mixed-integer Gaussian mixture Bayesian optimization and its applications in designing fractal and auxetic metamaterials

**Anh Tran**[1] · **Minh Tran**[1] · **Yan Wang**[1]

Received: 10 July 2018 / Revised: 13 November 2018 / Accepted: 11 December 2018 / Published online: 3 January 2019


**Abstract**
Bayesian optimization (BO) is a global optimization method that has the potential for design optimization. However, in classical BO algorithm, the variables are considered as continuous. In real-world engineering problems, both continuous and discrete variables are present. In this work, an efficient approach to incorporate discrete variables to BO is proposed. In the proposed constrained mixed-integer BO method, the sample set is decomposed into smaller clusters during sequential sampling, where each cluster corresponds to a unique ordered set of discrete variables, and a Gaussian process regression (GP) metamodel is constructed for each cluster. The model prediction is formed as the Gaussian mixture model, where the weights are computed based on the pair-wise Wasserstein distance between clusters and gradually converge to an independent GP as the optimization process advances. The definition of neighborhood can be flexibly and manually defined to account for independence between clusters, such as in the case of categorical variables. Theoretical results are provided in concert with two numerical and engineering examples, and two examples for metamaterial developments, including one fractal and one auxetic metamaterials, where the effective properties depend on both the geometry and the bulk material properties.

**Keywords** Bayesian optimization · Gaussian process · Constrained · Mixed-integer · Metamaterials

## 1 Introduction

Designing materials is to identify structures at micro- and nanoscales to achieve the desirable properties. The major process of design is to establish structure-property relationships, based on which design optimization can be performed. Simulation tools at multiple scales (from atomistic to continuum) have been developed to accelerate this process. Nevertheless, the major technical challenges of efficiency and accuracy still exist. The first one is searching in high-dimensional design space to find the global optimum of material compositions and structural configurations. The second one is the uncertainty associated with the high-dimensional structure-property relationships, which are usually constructed as surrogate models or metamodels. Particularly, aleatory uncertainty can be linked to natural randomness of materials (e.g., grain sizes and grain shapes in polycrystalline materials). Epistemic uncertainty is mainly due to approximations and numerical treatments in surrogates and simulation models. Methods of searching globally for optimal and robust solutions are needed.

Bayesian optimization (BO) is a metamodel-based methodology to seek for the global optimal solution under uncertainty in the search space with sequential sampling. Compared to other bio-inspired global optimization algorithms, such as ant colony systems, particle swarm, and genetic algorithm (GA), it has the advantage of maintaining the global search history by constructing a metamodel to approximate the objective function. Typically the metamodel is based on the Gaussian process (GP) method, and actively updated as more samples are collected. However, in current formulation of GP, input variables are restricted to be continuous. In real-world engineering problems, input design variables and parameters can be categorical or discrete. For example, binary variables can be used to enable or disable a design feature. The number of features has integer values. Therefore, extending BO method to accommodate

Responsible Editor: Byeng D Youn

✉ Yan Wang
yan.wang@me.gatech.edu

[1] George W. Woodruff School of Mechanical Engineering, Georgia Institute of Technology, Atlanta, USA

discrete variables is an important topic for solving real-world problems.

Another major issue that prohibits the BO and GP framework is its lack of scalability in searching the high-dimensional space when the number of input variables is large. The required number of sample points grows exponentially as $\mathcal{O}(s^d)$ with respect to the dimension of search space $d$, where $s$ is the number of sampling point for each dimension. The phenomenon is referred to as the curse-of-dimensionality in the literature. As a result, the size of the covariance matrix in GP also grows exponentially with respect to the dimensionality, creating the computational bottleneck in computing the inverse of the covariance matrix.

In this paper, a new BO method is proposed for constrained mixed-integer optimization problems to incorporate discrete design variables into the BO algorithm. In the proposed method, the large dataset of samples is decomposed into smaller clusters, where each cluster corresponds to a unique combination of discrete variable values, which is referred to as a discrete tuple. A GP is then constructed within each cluster. During the search and metamodel update processes, the mean and variance predictions are formulated as a Gaussian mixture model, where the weighted average predictions are combined from those of neighboring clusters, based on the pair-wise distance between the main and the neighboring clusters. The neighborhood of each cluster is constructed only once during the initialization.

Because of the decomposition approach, the number of sampling points to construct each cluster is significantly reduced compared to the whole dataset, and the GP thus is faster to construct for each cluster. This approach, however, leads to an undesirable effect of sparsity within each GP cluster. As a result, the posterior variance might be slightly overestimated. To circumvent the sparsity effect of the decomposition approach, a weighted average scheme is adapted to "borrow" the sampling points from neighboring clusters, where the discrete tuples of the neighbors slightly differ from the discrete tuple of the original cluster. The definition of neighborhood is completely controlled by users, and neighbors can be added or removed accordingly.

The unique advantage of the proposed method is that the optimization problem of both continuous and discrete variables and the acceleration of GP for high-dimensional problems are solved simultaneously. Theoretical results are provided and discussed in concert with computational metamaterials design applications.

In the remainder of the paper, Section 2 provides a literature review for BO methodology, its extension, such as constrained and mix-integer optimization problems, and its applications. Section 3 describes the proposed constrained mixed-integer BO algorithm using Gaussian mixture model, including theoretical analysis of algorithmic complexity as well as lower and upper bounds of the predictions. The methodology is demonstrated with applications in computational design of metamaterials. Metamaterials are an emerging class of engineered materials that exhibit interesting and desirable macroscopic properties, which can be tailored, because of their engineered geometric structures rather than the material composition.

In Section 4, the proposed method is verified using an analytical function that is modified based on a discrete version of the Rastrigin function, an engineering example of welded beam design, where the discrete variables encode the material selection and design configuration of the beam. In the first engineering example of Section 5.1, we focus on designing high-strength and low-weight fractal metamaterials, where the effective material properties, such as effective Young's modulus, are obtained using finite element method (FEM). In the second engineering example of Section 5.2, the method is demonstrated using an auxetic metamaterials for polymers, where the effective negative Poisson's ratio is optimized. Section 6 includes the discussion of the limitations in the proposed approach, and Section 7 concludes the paper, respectively.

## 2 Related work

Here, we conduct a literature review on related BO work and its design applications. In Section 2.1, the widely used acquisition functions for BO are introduced. The constrained optimization problem in BO is reviewed in Section 2.2. In Section 2.3, the mixed-integer optimization problem in BO and its related work is discussed. In Section 2.4, the applications of GP in design optimization is provided.

### 2.1 Acquisition function

BO is a metamodel-based optimization framework that uses GP as the metamodel. The major difference between BO- and GP-based optimization is the sampling strategy to construct the metamodel.

The significant extension of BO is the implementation of the so-called acquisition function that dictates the location of the next sampling design site. This acquisition function reconciles the trade-off between exploration (navigating to the most uncertain region) and exploitation (driving the solution to the optimum) in the optimization process.

Given the objective function $y = f(\boldsymbol{x})$, the acquisition function $a(\boldsymbol{x}; \{\boldsymbol{x}_i, y_i\}_{i=1}^N, \theta)$ depends on previous $N$ observations or samples $\{\boldsymbol{x}_i, y_i\}_{i=1}^N$ and GP hyperparameters $\theta$ and must be defined to strike a balance between exploration and exploitation. In exploration, the acquisition function $a$ would lead to the next sampling point in an unknown region where the posterior variance $\sigma^2(\boldsymbol{x})$ is large. In exploitation,

the acquisition function $a$ would result in the next sampling point where posterior mean $\mu(\boldsymbol{x})$ is large for a maximization problem (or small for minimization).

There are mainly three types of acquisition functions: probability of improvement (PI), expected improvement (EI), and upper confidence bound (UCB). They are defined as follows.

Let $\boldsymbol{x}_{\text{best}} = \arg\max_{\boldsymbol{x}_i} f(\boldsymbol{x}_i)$ be the best sample achieved so far during sequential sampling for a maximization problem, $\phi(\cdot)$ and $\Phi(\cdot)$ be the probability density function and cumulative distribution function of the standard normal distribution respectively. The PI acquisition function (Mockus and Mockus 1991) is defined as

$$a_{\text{PI}}(\boldsymbol{x}; \{\boldsymbol{x}_i, y_i\}_{i=1}^N, \theta) = \Phi(\gamma(\boldsymbol{x})), \tag{1}$$

where

$$\gamma(\boldsymbol{x}) = \frac{\mu(\boldsymbol{x}; \{\boldsymbol{x}_i, y_i\}_{i=1}^N, \theta) - f(\boldsymbol{x}_{\text{best}})}{\sigma(\boldsymbol{x}; \{\boldsymbol{x}_i, y_i\}_{i=1}^N, \theta)} \tag{2}$$

indicates the deviation away from the best sample.

The EI acquisition function (Mockus 1975; Huang et al. 2006) is mathematically expressed as

$$a_{\text{EI}}(\boldsymbol{x}; \{\boldsymbol{x}_i, y_i\}_{i=1}^N, \theta) = \sigma(\boldsymbol{x}; \{\boldsymbol{x}_i, y_i\}_{i=1}^N, \theta) \cdot (\gamma(\boldsymbol{x})\Phi(\gamma(\boldsymbol{x})) + \phi(\gamma(\boldsymbol{x})) \tag{3}$$

Recently, Srinivas et al. (2009, 2012) proposed a new form of UCB acquisition function,

$$a_{\text{UCB}}(\boldsymbol{x}; \{\boldsymbol{x}_i, y_i\}_{i=1}^N, \theta) = \mu(\boldsymbol{x}; \{\boldsymbol{x}_i, y_i\}_{i=1}^N, \theta) + \kappa\sigma(\boldsymbol{x}; \{\boldsymbol{x}_i, y_i\}_{i=1}^N, \theta), \tag{4}$$

where $\kappa$ is a hyperparameter describing the exploitation-exploration balance.

## 2.2 Constrained BO

Constrained BO is a natural and important extension of the classical BO method. Constrained optimization problems based on engineering model and simulation can be classified as two types: known and unknown constraints. The known constraints, or a priori constraints, are the ones known before the simulation, and thus can be evaluated independently without running simulations. On the other hand, the unknown constraints are the ones that are unpredictable without running the simulation, and thus can be only incorporated once the simulation is over, e.g., no solution because of numerical divergence. Generally speaking, the unknown constraints are more difficult to assess because it involves handling the classification problem, satisfied or violated, with respect to the optimization problem.

Digabel and Wild (2015) summarized and provided a systematic classification and taxonomy for constrained optimization problem. Gardner et al. (2014) proposed a penalized acquisition function approach to limit the searching space for the next sampling location. Gelbart et al. (2014) suggested an entropy search criterion to search for the next sampling point under the formulation of the EI acquisition function. Hernández-Lobato et al. (2015, 2016) introduced a predictive entropy search and predictive entropy search with constraints, respectively, which maximizes the expected information gained with respect to the global maximum. Rehman and Langelaar (2017) modeled constraints as a simple model and incorporated probability of feasibility measure to alternate the EI acquisition function. Li et al. (2018) proposed a sequential Monte Carlo approach with radial basis function as surrogate model to solve for the constrained optimization problem.

## 2.3 Mixed-integer Bayesian optimization

The BO extension to mixed-integer problems is rather limited, partly because mixed-integer problems carry difficulties from both discrete and continuous optimization problems. Another approach is that the discrete optimization can be converted to continuous optimization, using simple rounding operation. The approach is not mathematically rigorous, but is still widely accepted in practice. Here, we review several contributions in term of methodology to incorporate discrete variables.

Davis and Ierapetritou (2009) combined a branch-and-bound approach with BO method to solve the mixed-integer optimization problems.

Müller et al. (2013, 2014, 2016) introduced three algorithms, which are Surrogate Optimization-Mixed Integer (Müller et al. 2013), Surrogate Optimization-Integer (Müller et al. 2014), and Mixed-Integer Surrogate Optimization (Müller 2016), which differ in the perturbation sampling strategies and utilize GP as the surrogate model, to solve for the mixed-integer nonlinear problems. Hemker et al. (2008) compared the performance of a GA, the implicit filtering algorithm, and a branch-and-bound approach formulated on BO algorithm to solve for a set of constrained mix-integer problems in groundwater management.

For mixed-integer extension for GP, van Stein et al. (2015) proposed a distributed kriging approach, where the dataset is decomposed for continuous variables using $k$-mean algorithm, and the optimal weights are computed based on the inverse posterior variance of each cluster. Gramacy and Lee (2008a, b, 2010) developed a treed GP that is naturally extensible to handle discrete variables. In the case of discrete variables, the GP is one-hot encoded by the binary combination of the discrete variables. Storlie et al.

(2011) developed the adaptive component selection shrinkage operator (ACOSSO) method extended from Lin and Zhang (2006a, b), which uses the smoothing spline ANOVA decomposition to decompose the total variance to multivariate functions. Qian et al. (2008) and Zhou et al. (2011) approached the mixed-integer problem from the covariance kernel of GP, proposing the exchange correlation, the multiplicative correlation, and the unrestricted correlation functions to handle discrete variable that is reminiscent of categorical regression. Swiler et al. (2014) compared three above methods and concluded that GP with special correlation kernel (Qian et al. 2008; Zhou et al. 2011) performs most consistently among the test functions.

### 2.4 GP-based design optimization

GP, also known as kriging, has been widely applied in constructing surrogates or metamodels for design optimization. Simpson et al. (2001), Queipo et al. (2005), Martins and Lambe (2013), Sóbester et al. (2014), and Viana et al. (2014) provided comprehensive reviews on the use of kriging and other surrogate models for multi-disciplinary design optimization. More recently, Li et al. (2008) proposed a kriging metamodel assisted multi-objective GA to solve multi-objective optimization problems. Jang et al. (2014) used dynamic kriging to solve a design optimization in fluid-solid interaction. Zhang et al. (2014) also used kriging to approximate the pump performance and optimize two objective functions with respect to four design variables. Kim et al. (2017a) optimized and verified a fluid dynamic bearings simulation using kriging approach. Kim et al. (2017b) applied multi-fidelity kriging and optimized film-cooling hole arrangement. Liu et al. (2017) employed surrogate-based parallel optimization method to reduce the computational time for a computational fluid dynamics problem with six design variables. Song et al. (2017) used a gradient-enhanced hierarchical kriging to optimize drag on airfoils at a specified angle of attack. Zhou et al. (2017, 2018) developed a multi-fidelity kriging scheme to approximate the lift coefficient as a function of Mach number and angle of attack in airfoils with computational fluid dynamics analysis.

In the above work, design variables are all continuous. Compared to these GP-based optimization, BO formulation provides a more generic and robust searching procedure.

## 3 Proposed mixed-integer Bayesian optimization

The proposed mixed-integer BO based on distributed GP provides an efficient searching method for large scale design problems, where design variables can be either continuous or discrete. The discrete variables include both categorical and integer variables, regardless of the existence of order relations. Let $\boldsymbol{x} = (\boldsymbol{x}^{(d)}, \boldsymbol{x}^{(c)})$ be the design variables, where $\boldsymbol{x}^{(d)} \in \mathbb{D}$ are discrete variables in $n$-dimensional space $\mathbb{D}$ and $\boldsymbol{x}^{(c)} \in \mathbb{R}^{m-n}$ are continuous variables in $(m-n)$-dimensional space $\mathbb{R}^{m-n}$. Together, they form a vector of design variables in the $m$-dimensional space $\mathcal{X}$. Let $f(\boldsymbol{x})$ be the objective function. The design optimization problem solves the maximization problem

$$\boldsymbol{x}^* = \arg\max_{\boldsymbol{x}\in\mathcal{X}} f(\boldsymbol{x}), \tag{5}$$

subject to some inequality constraints

$$g_i(\boldsymbol{x}) \leq 0, i = 1, \cdots, i_c \tag{6}$$

where $i_c$ is the number of inequality constraints.

Here, the notation for the rest of the paper is as follows. $\mu_l(\boldsymbol{x})$ is used to denote the posterior mean of the $l$th cluster at the query point $\boldsymbol{x}$. $\hat{\mu}$ is the prediction formed by Gaussian mixture model of all the clusters. $\bar{\mu}_l$ is the mean of the $l$th cluster.

In the proposed mixed-integer BO, the large dataset of observations is decomposed into smaller local clusters, where each cluster is used to construct a local GP. Because the large dataset has been decomposed and the number of data points has reduced, the prediction within each cluster is not as accurate, and can be improved by "borrowing" from neighboring dataset under a weighted average scheme. The large dataset with continuous and discrete variables can be decomposed to finitely many clusters, according to the tuple of discrete variables. In each cluster, the data points share the same discrete variable values. The classical GP approach is then applied to the dataset in each cluster to construct a GP model.

Because of the decomposition scheme, the number of data points within each cluster is reduced, compared to the number of data points of the whole dataset. This leads to a sparser dataset within a cluster, and the posterior variance is enlarged. To improve the prediction, the datasets from neighboring clusters are initially "borrowed" to improve the prediction on the tuple of continuous variables $\boldsymbol{x}^{(c)} \in \mathbb{R}^{m-n}$, where the "borrowed" data points are gradually eliminated as the optimization process converges via the weight computation algorithm. On the other hand, the sparsity induced by the decomposition scheme reduces the cost of computing the inverse of the covariance matrix. In this weighted average scheme, the weights are computed and penalized based on the pair-wise Wasserstein distance between clusters, as well as the posterior variance of the cluster to obtain a more accurate predictions to aid in the convergence of the optimization process.

Figure 1 presents an overview of the workflow for the proposed mixed-integer BO method in this paper. First, initial samples, typically obtained from Monte Carlo or Latin hypercube sampling, are used to construct the metamodel, where a local GP is associated with each individual cluster. Next, a next sampling point is located within each cluster according to its acquisition functional value. Then, a global sampling point for all clusters is determined among the collection of all the next sampling points from each cluster. The objective function is then called to evaluate at the global sampling location. A local GP is updated at the cluster corresponding to the global sampling point. A new local sampling point is located within the same cluster, and the process repeats until some optimization criteria are met.

The following subsections are organized as follows. Section 3.1 briefly reviews the GP formulation. Section 3.2 discusses the enumeration algorithm for clusters and the discrete tuple. Section 3.3 describes the definition of cluster neighborhood that is used to form a Gaussian mixture model. Section 3.4 details the weight computations for each individual cluster in the Gaussian mixture model. Section 3.5 presents the computation of posterior mean and posterior variance of the Gaussian mixture model. Section 3.6 describes the penalized scheme to incorporate constraints into the acquisition function. Section 3.7 analyzes the theoretical bounds and computational cost of the proposed mixed-integer BO method.

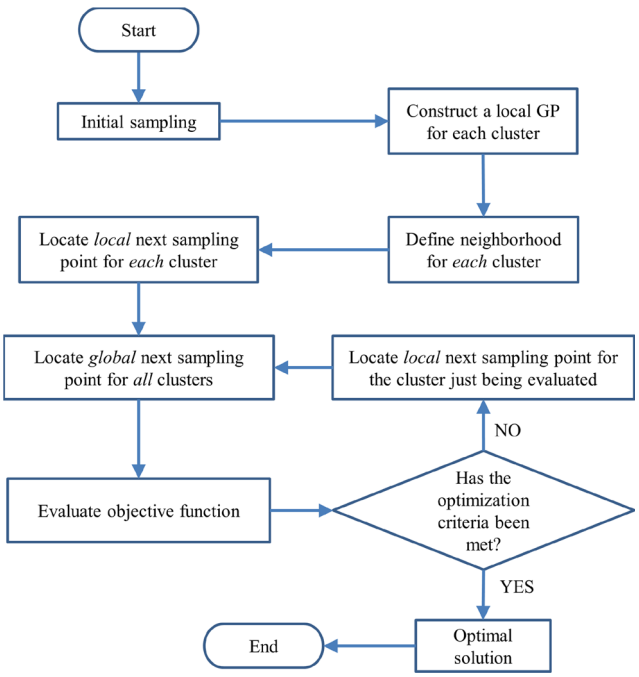



**Fig. 1** Overall workflow of the proposed mixed-integer Bayesian optimization

### 3.1 Gaussian process

We follow the notation introduced by Shahriari et al. (2016) to briefly introduce GP formulation for continuous variables. GP$(\mu_0, k)$ is a nonparametric model that is characterized by its prior mean $\mu_0 : \mathcal{X} \mapsto \mathbb{R}$ and its covariance kernel $k : \mathcal{X} \times \mathcal{X} \mapsto \mathbb{R}$. Define $f_i = f(\boldsymbol{x}_i)$ and $y_{1:N}$ as the unknown function values and noisy observations, respectively. In the GP formulation, it is assumed that the $\boldsymbol{f} = f_{1:N}$ are jointly Gaussian and $\boldsymbol{y} = y_{1:N}$ are normally distributed given $\boldsymbol{f}$; then, the prior distribution induced by the GP can be described as

$$\boldsymbol{f}|\boldsymbol{X} \sim \mathcal{N}(\boldsymbol{m}, \boldsymbol{K}), \quad \boldsymbol{y}|\boldsymbol{f}, \sigma^2 \sim \mathcal{N}(\boldsymbol{f}, \sigma^2 \boldsymbol{I}), \tag{7}$$

where the elements of mean vector and covariance matrix are described by $m_i := \mu_0(\boldsymbol{x}_i)$ and $K_{i,j} := k(\boldsymbol{x}_i, \boldsymbol{x}_j)$.

Equation 7 describes the prior distribution induced by the GP, where $\boldsymbol{X}$ is the sampling location, and $f$ is the objective function. In the GP formulation, $y$ is the noise-corrupted stochastic output of $f(\boldsymbol{x})$ with the variance of $\sigma^2$, at the sampling location $\boldsymbol{X}$. The objective function $f$ is assumed to be a multivariate normal distribution function with mean $\boldsymbol{m}(x)$ and covariance $\boldsymbol{K}(x)$.

Let $N$ be the number of sampling locations, and $\mathcal{D}_N = \{\boldsymbol{x}_i, y_i\}_{i=1}^N$ be the set of observations. The covariance kernel $k$ is a choice of modeling the correlation between input locations $\boldsymbol{x}_i$. Covariance functions where length-scale parameters can be inferred through maximum likelihood function is known as automatic relevance determination kernels. One of the most widely used kernels in this kernel family is the squared-exponential kernel,

$$\boldsymbol{K}_{i,j} = k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \theta_0^2 \exp\left(-\frac{r^2}{2}\right), \tag{8}$$

where $r^2 = (\boldsymbol{x} - \boldsymbol{x}')\boldsymbol{\Gamma}(\boldsymbol{x} - \boldsymbol{x}')$, $\boldsymbol{\Gamma}$ is a diagonal matrix of $(m-n) \times (m-n)$, and $\theta_i$ is the length scale parameter.

The posterior Gaussian for the sequential BO is characterized by the mean

$$\mu_{N+1}(\boldsymbol{x}) = \mu_0(\boldsymbol{x}) + \boldsymbol{k}(\boldsymbol{x})^T (\boldsymbol{K} + \sigma^2 \boldsymbol{I})^{-1} (\boldsymbol{y} - \boldsymbol{m}), \tag{9}$$

and the variance

$$\sigma^2_{N+1}(\boldsymbol{x}) = k(\boldsymbol{x}, \boldsymbol{x}) - \boldsymbol{k}(\boldsymbol{x})^T (\boldsymbol{K} + \sigma^2 \boldsymbol{I})^{-1} \boldsymbol{k}(\boldsymbol{x}), \tag{10}$$

where $\boldsymbol{k}(\boldsymbol{x})$ is the vector of covariance terms between $\boldsymbol{x}$ and $\boldsymbol{x}_{1:N}$.

### 3.2 Clustering and enumeration algorithm

Assuming that the discrete variables are independent of each other, a clustering and enumeration algorithm is devised to automatically decompose the large dataset to smaller clusters based on the discrete tuple and tag a cluster with a unique index from the enumeration scheme. For the case

when some discrete variables are dependent on others, the neighborhood can be manually changed to reflect the knowledge. The set of discrete variables for each cluster is represented as a discrete tuple where each element is a positive integer.

For an integer variable where order relation exists, the discrete variable can simply be represented as a positive integer, e.g., $1 \leq 2$. For a categorical variable where order relation does not exist, such as the type of cross section (square or circular), colors (red or blue), type of materials (aluminum or copper), and configuration settings, positive integers can still be used. The choice of using tuple of positive integers as a general representation does not affect the clustering and enumeration scheme, but would affect the construction of neighborhood for each cluster, depending on the nature of discrete variables.

Suppose that the input $\mathbf{x} = (\boldsymbol{x}^{(d)}, \boldsymbol{x}^{(c)}) = (x_1, \cdots, x_n, x_{n+1}, \cdots, x_m)$ includes $n$ discrete and $m-n$ continuous variables. If $p_i$ is denoted as the total number of possible values for discrete variable $x_i$, $1 \leq i \leq n$, then the number of clusters is $L = \prod_{i=1}^{n} p_i$. Due to the complexity of possible combinations, each cluster is assigned a unique index in such a way that the map between their discrete variables and cluster index is one-to-one. The index is calculated based on the total ordering of tuples. Without loss of generality, assume that each discrete variable $x_i$ is bounded by $1 \leq x_i \leq p_i$, i.e., $x_i \in \{1, \cdots, p_i\}$ for $1 \leq i \leq n$. Then, the relation of lexicographical order, denoted as $\prec$, can be defined for a pair of tuples on the set of all tuples as

$$(a_1, \cdots, a_n) \prec (b_1, \cdots, b_n), \tag{11}$$

if and only if $\exists k : 1 \leq k \leq n : (\forall j : 1 \leq j < k : a_i = b_i)$ and$a_k < b_k$, and $1 \leq a_i, b_i \leq p_i$ for all $i$. With the definition of lexicographical order $\prec$, the cluster index $l$ for the tuple $(a_1, \cdots, a_n)$ can now be calculated as

$$l = \sum_{i=1}^{n-1} (a_i - 1) \prod_{j=i+1}^{n} p_j + a_n. \tag{12}$$

Because the index of cluster is uniquely defined based on the tuple of discrete variables, the tuple describing the set of discrete variables can be reconstructed using the index of the cluster, with the quotient and remainder algorithm recursively shown in Algorithm 1. It describes how to construct the set of discrete variables from the cluster index $l$.

The implementation of Algorithm 1 can be based on existing functions such as MATLAB function ind2sub(). Equation 12, which is a reverse operation of Algorithm 1, can also be implemented using MATLAB function sub2ind().

**Algorithm 1** Reconstruct the tuple of discrete variables $(x_1, \cdots, x_n)$ from cluster index $l$

**Input:** cluster index $l$, tuple $(p_1, \cdots, p_n)$.
**Output:** tuple $(a_1, \cdots, a_n)$ of discrete variables

**for** $i \leftarrow 1, n$ **do**
  **if** $i \neq n$ **then**
    $q \prod_{j=i+1}^{n} p_j + r = l$ ▷ find quotient $q$, remainder $r$
    $l \leftarrow r$
    $a_i \leftarrow q + 1$ ▷ assign discrete variable in order
  **else**
    $a_n \leftarrow r$ ▷ assign last discrete variable
  **end if**
**end for**
**for** $i \leftarrow n, -1, 1$ **do** ▷ exception if $a_i = 0$
  **if** $a_i = 0$ **then**
    $a_i \leftarrow p_i, a_{i-1} \leftarrow a_{i-1} - 1$
  **end if**
**end for**

### 3.3 Construction of neighborhood

Consider a cluster with index $l$, with the tuple of discrete variables $(a_1, \cdots, a_n)$, the neighbors of the $l$th cluster $\mathcal{B}(l)$ is the collection of clusters that share most of similarity with the original cluster. Intuitively, the neighborhood is constructed based on the belief of whether there exists a relationship between two clusters.

For example, for integer variables, the discrete tuples of the neighboring clusters may differ in one or a few different integer variables compared to that of the original cluster. In the same manner, for categorical variables, the discrete tuples of the neighboring clusters may differ in one or a few categorical variables compared to that of the original cluster. Based on this description, a possible choice to define the neighborhood $\mathcal{B}(l)$ of the $l$th cluster can be mathematically expressed as

$$\mathcal{B}(l) = \{(a_1^*, \cdots, a_n^*) \mid d\left((a_1^*, \cdots, a_n^*), (a_1, \cdots, a_n)\right) \leq d_{\text{th}}\}, \tag{13}$$

where $d\left((a)_{i=1}^n, (a^*)_{i=1}^n\right)$ is some metric on a discrete topological space $\mathbb{D}$, and $d_{\text{th}}$ is a user-defined threshold. The metric $d(\cdot, \cdot)$ can be any $l_p$-norm, for example, Manhattan distance ($l_1$-norm), or a counting metric of how many discrete (integer and categorical) variables are different between two tuples. It is noted that the metric $d(\cdot, \cdot)$ does not have to strictly obey the definition of mathematical norm. In the special case when this metric is set to zero, i.e., $d\left((a)_{i=1}^n, (a^*)_{i=1}^n\right) = 0$, it means that all the clusters are considered to be completely independent of each other. The construction of neighborhood only occurs once during the initialization.

Furthermore, it should be emphasized that the neighboring list can be manually changed to reflect the physics-based knowledge from the users, or manually constructed to reflect the dependency of the discrete variables. In the case of categorical variables where independence is usually observed, one can simply remove the neighboring cluster from the corresponding categorical variable, as the neighborhood can be manually changed during the initialization phase of the optimization process.

It is recommended to define the neighborhood carefully, as the neighborhood definition has an impact on both convergence rate, and whether the optimization would be trapped at local optimum. The safest setting is to assign $d_{\text{th}} = 0$, where clusters are assumed to be completely independent of each other. Small values of $d_{\text{th}}$, e.g., $d_{\text{th}} = 1$ or $d_{\text{th}} = 2$, might be beneficial, depending on the specific applications. Large value is not recommended.

Figure 2 shows an example of constructing clusters for two discrete variables $(x_1, x_2)$, where $1 \le x_1 \le 4$ and $1 \le x_2 \le 3$. According to Algorithm 1, the tuple $p$ is (4, 3), cluster 1 is associated with (1,1), cluster 2 is associated with (1,2), cluster 4 is associated with (2,1), etc. The cluster index is denoted as an italic number on the top right corner of the square. Consider cluster 8, which is associated with the discrete tuple (3,2). If the Manhattan distance is chosen to define the neighborhood, then the choice of $d_{\text{th}} = 0$ in (13) would make every cluster the only neighbor of itself, e.g., the neighbor of cluster 8 is cluster 8. The choice of $d_{\text{th}} = 1$ would include clusters 5, 7, 8, 9, and 11 in cluster 8's neighborhood. Similarly, the choice of $d_{\text{th}} = 2$ would include clusters 2, 4, 5, 6, 7, 8, 9, 10, 11, and 12 in cluster 8's neighborhood.

### 3.4 Weight computation

The weight of each cluster's prediction is determined by the Wasserstein distance between the Gaussian posterior of the main cluster with that of the neighboring clusters. Combined together, they form a Gaussian mixture model to predict a response at a query point $\boldsymbol{x}$.

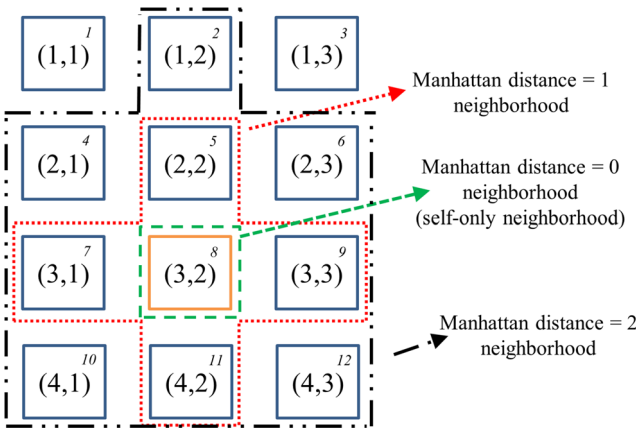



**Fig. 2** An example of cluster enumeration and neighborhood definition

Consider a query point $\boldsymbol{x}$ in the $l$th cluster, which has the continuous tuple $\boldsymbol{x}^{(c)} = (x_{n+1}, \cdots, x_m)$. Denote the neighborhood of the $l$th cluster as $B(l) = \{l^*\}$, where the cardinality of $|B(l)| = k$, i.e., there are $k$ neighbors in the $l$th cluster neighborhood.

Each of the neighboring cluster $l^*$ can form its own prediction $\mathcal{N}(\mu_{l^*}, \sigma^2_{l^*})$ from the continuous tuple, including $\mathcal{N}(\mu_l, \sigma^2_l)$ for $l$th cluster. However, the prediction must be adjusted by accounting for the bias, i.e., $\text{Bias}_{l^*}[\mu_{l^*}] = \mathbb{E}[\mu_{l^*} - \mu_l] = \bar{\mu}_{l^*} - \bar{\mu}_l$ as the difference between the posterior means of two clusters, and the variance $\sigma^2_{l^*}$.

The weight $w_{l^*}$ associated with the prediction from the $l^*$ cluster should be larger with smaller bias $(\bar{\mu}_{l^*} - \bar{\mu}_l)$ and smaller posterior variance $\sigma^2_{l^*}$. The necessity of bias correction is explained later in Theorem 4. The Wasserstein distance between two univariate Gaussian $\mathcal{N}(\mu_{l^*}, \sigma^2_{l^*})$ and $\mathcal{N}(\mu_l, \sigma^2_l)$ is provided by Givens et al. (1984) as

$$W_2\left(\mathcal{N}(\mu_{l^*}, \sigma^2_{l^*}), \mathcal{N}(\mu_l, \sigma^2_l)\right) = \|\mu_l - \mu_{l^*}\|^2 + \left\|\sqrt{\sigma^2_l} - \sqrt{\sigma^2_{l^*}}\right\|^2 \quad (14)$$

Here, we propose a deterministic way to compute the numerical weights based on the pair-wise Wasserstein distance, which eventually converges to an independent GP as the optimization process advances. It is easy to see that the $W_2$-distance of the $l$th cluster's prediction to itself is zero, as $W_2$ is a distance. The weights are computed according to an inverse $W_2$-distance with a term $\sigma^2_l$ from the $l$th cluster, as

$$w_{l^*} \propto \left[\sigma^2_l + W_2\left(\mathcal{N}(\mu_{l^*}, \sigma^2_{l^*}), \mathcal{N}(\mu_l, \sigma^2_l)\right)\right]^{-1}. \quad (15)$$

In (15), $w_{l^*}$ are computed based on two factors, the $W_2$-distance, and the $\sigma^2_l$ prediction of the $l$th cluster. As the optimization process advances, the posterior variance approaches zero, i.e., $\sigma^2_l \to 0$. As a result, the weight scheme converges to a single GP prediction of the corresponding $l$th cluster.

### 3.5 Prediction using weighted average of *k*-nearest neighboring clusters

We model the prediction of a query point using a Gaussian mixture distribution, where the weights are computed on the statistical Wasserstein distance. To predict an unknown query point $\boldsymbol{x} = (\boldsymbol{x}_d, \boldsymbol{x}_c) = (x_1, \cdots, x_n, x_{n+1}, \cdots, x_m)$, we first find the cluster in which $\boldsymbol{x}$ belongs to, and its neighboring clusters. Assume that $\boldsymbol{x}$ belongs to the $l$th cluster, and there are $k$-neighboring clusters.

The principle for weight computation is as follows. As the bias increases, the contributed weight of the prediction $w_{l^*}$ from the $l^*$th cluster to $l$th cluster is reduced to a smaller value. As the bias or the pair-wise distance between clusters increases, the contributed weights also decrease. The weight vector is normalized at every step, and eventually converges to a single GP prediction with the weight vector of $[0, \cdots, 1, \cdots, 0]$, where 1 is located as the $l$th cluster.

Since $\boldsymbol{x}$ is located within the $l$th cluster, the weight from the $l$th cluster is the highest, i.e., if $l^* = l$, then $\mu_{l^*} + \bar{\mu}_l - \bar{\mu}_{l^*} = \mu_l$, which is the GP prediction for the $l$th cluster. The posterior mean of the proposed method is written as

$$\hat{\mu} = \sum_{l^* \in \mathcal{B}(l)} w_{l^*} \left(\mu_{l^*} + \bar{\mu}_l - \bar{\mu}_{l^*}\right), \tag{16}$$

where the sum is taken over the list of neighboring cluster from the main cluster $l$th. $\bar{\mu}_l$ and $\bar{\mu}_{l^*}$ denote the means of the $l$-th and $l^*$-th clusters, respectively. $w^*$ denotes the weight corresponding to the $l^*$th cluster, which is computed once the discrete tuple $\boldsymbol{x}^{(d)}$ of the query point $\boldsymbol{x} = (\boldsymbol{x}^{(d)}, \boldsymbol{x}^{(c)})$ is determined. The posterior variance of the proposed method is calculated as

$$\hat{\sigma}^2 = \sum_{l^* \in \mathcal{B}(l)} w_{l^*}^2 \sigma_{l^*}^2, \tag{17}$$

where $\sigma_{l^*}^2$ denotes the posterior variance associated with the continuous tuple $\boldsymbol{x}^{(c)}$ of the query point $\boldsymbol{x} = (\boldsymbol{x}^{(d)}, \boldsymbol{x}^{(c)})$.

The prediction scheme for mean $\hat{\mu}(\boldsymbol{x})$ and variance $\hat{\sigma}^2(\boldsymbol{x})$ for an arbitrary location $\boldsymbol{x}$ using Gaussian mixture model can be summarized in Algorithm 2.

**Algorithm 2** Prediction using weighted average GP from nearest neighboring clusters

**Input:** location $x = (x_1, \cdots, x_n, x_{n+1}, \cdots, x_m)$, mean output of each cluster $\bar{\mu}_{(\cdot)}$

**Output:** Gaussian mixture posterior mean $\hat{\mu}$ and posterior variance $\sigma^2$

1: Find cluster index $l$ corresponding to $x^{(d)} = (x_1, \cdots, x_n)$ ▷ locate the $l$-th cluster
2: Construct a neighborhood $\mathcal{B}(\cdot)$ for each cluster ▷ query $x$ in all neighboring clusters
3: **for** $l^* \in \mathcal{B}(l)$ **do**
4: Compute GP posterior of the $l^*$-th cluster: $\hat{\mu}_{l^*}, \sigma_{l^*}^2$
5: **end for**
6: Compute weight $w_{l^*} \propto [\sigma_l^2 + W_2(\mathcal{N}(\mu_{l^*}, \sigma_{l^*}^2), \mathcal{N}(\mu_l, \sigma_l^2))]^{-1}$ ▷ pair-wise Wasserstein distance
7: $w_{l^*} \leftarrow \dfrac{w_{l^*}}{\sum_{l^* \in \mathcal{B}(l)} w_{l^*}}$ ▷ weight normalization
8: $\hat{\mu} \leftarrow \sum_{l^* \in \mathcal{B}(l)} w_{l^*} \left(\mu_{l^*} + \bar{\mu}_l - \bar{\mu}_{l^*}\right)$ ▷ Gaussian mixture posterior mean
9: $\hat{\sigma}^2 \leftarrow \sum_{l^* \in \mathcal{B}(l)} w_{l^*}^2 \sigma_{l^*}^2$ ▷ Gaussian mixture posterior variance
10: Update the average mean of the $l$-th cluster $\bar{\mu}_l$

### 3.6 Constrained acquisition function in mixed-integer Bayesian optimization

The acquisition function is adopted from Gardner et al. (2014) for inequality constraints, and further extended to accommodate discrete and continuous variables to solve for the constrained mixed-integer optimization problems.

First, the constraint is checked using an indicator function $\mathcal{I}(\boldsymbol{x})$ for all $i_c$ constrained inequalities, as

$$\mathcal{I}(\boldsymbol{x}) = \begin{cases} 1 \text{ if } \forall 1 \le i \le i_c : g_i(\boldsymbol{x}) \le 0, \\ 0 \text{ if } \exists 1 \le i \le i_c : 0 \le g_i(\boldsymbol{x}). \end{cases} \tag{18}$$

The constrained acquisition function can be considered as the product of the classical acquisition function. As a result, the acquisition function is assigned to have zero value for infeasible region. The penalized approach can be implemented directly into the auxiliary optimizer, which is used to maximize the acquisition function in BO.

In distributed GP, an input $\boldsymbol{x}_{\text{next}} = (x_1, \cdots, x_n, x_{n+1}, \cdots, x_m)$ is comprised of both discrete and continuous variables. For each cluster corresponding to a unique set of discrete tuple $(x_1, \cdots, x_n)$, a distinct next sampling point associated with each cluster is located by maximizing the acquisition function on the tuple of continuous variables $(x_{n+1}, \cdots, x_m)$ for each iteration, in the same manner as classical BO. These next sampling points are retained within the respective clusters. However, only the sampling point corresponding to the maximal value of acquisition function among all clusters is chosen, and a new sampling point within that cluster is located and updated for the corresponding cluster. The sampling procedure repeats until the optimization criterion is met. In other words, the next sampling point is chosen as

$$\boldsymbol{x}_{\text{next}} = \arg\max_{l^*} \arg\max_{(x_n, x_{n+1}, \cdots, x_m)} a_{l^*}(\boldsymbol{x}; \{\boldsymbol{x}_i, y_i\}_{i=1}^N, \theta) \cdot \mathcal{I}(\boldsymbol{x}), \tag{19}$$

where the $l^*$th cluster corresponds to the tuple of discrete variables $(x_1, \cdots, x_n)$, and $\mathcal{I}(\boldsymbol{x})$ is the constraint indicator function.

Equation 19, which describes the searching procedure for the next sampling point by maximizing the penalized acquisition function, is explained as follows. Two loops are constructed to search for the global sampling point. In the inner loop which searches for the local sampling point within each cluster, the penalized acquisition function is the objective function. Maximizing this penalized acquisition function using an auxiliary optimizer yields the local sampling point for each cluster. In the outer loop, the cluster with the maximized acquisition function value is determined. The discrete tuple corresponding to the cluster index, which contains the sampling point with the maximum value for the acquisition function, is reconstructed using Algorithm 1. In other words, the sampling location $\boldsymbol{x}$ is decomposed

to two parts: the inner loop searches for the continuous tuple, whereas the outer loop yields the discrete tuple. Theoretically, once the functional evaluation is over, only the cluster that contains the last sampling location needs to be updated. Practically, all the clusters need to update their corresponding sampling locations $\boldsymbol{x}_{\text{next}}$ after certain number of iterations, in order to avoid trapping in local optimum.

The tuple of continuous variables is found by maximizing the acquisition function, whereas the tuple of discrete variables is assigned according to the cluster index. For the EI and PI acquisition functions, $x_{\text{best}}$ is modified to be the best point achieved so far among all clusters. For the UCB acquisition function, no modification is needed, assuming the hyperparameter $\kappa$ is uniform for all clusters. It is noted that the balance between exploration and exploitation is preserved locally within each cluster and thus is also preserved globally for all the clusters.

### 3.7 Theoretical bounds and computational cost

Here, we provide the theoretical lower and upper bounds for predictions and algorithm complexity under the formulation of Gaussian mixture model in Theorem 1 and Theorem 2. Theorem 3 proves that under the formulation of the proposed method, the largest weight is associated with the main cluster. Theorem 4 explains the necessity of translation in mean prediction so that the expected value of the mean is the same with the expected mean in the main cluster.

**Theorem 1** *The Gaussian mixture posterior mean* $\hat{\mu} = \sum_{l^* \in \mathcal{B}(l)} w_{l^*} (\mu_{l^*} + \bar{\mu}_l - \bar{\mu}_{l^*})$ *is bounded by*

$$\min_{l^*} \left(\hat{\mu}_{l^*} + \bar{\mu}_l - \bar{\mu}_{l^*}\right) \leq \hat{\mu} \leq \max_{l^*} \left(\hat{\mu}_{l^*} + \bar{\mu}_l - \bar{\mu}_{l^*}\right) \tag{20}$$

*Proof* The proof for the posterior mean is straightforward, noting that $w_{l^*} \geq 0, \forall l^*$ and $\sum w_{l^*} = 1$. □

**Theorem 2** *The Gaussian mixture posterior variance* $\hat{\sigma}^2 = \sum_{l^* \in \mathcal{B}(l)} w_{l^*}^2 \sigma_{l^*}^2$ *is bounded by*

$$(\sum_{l^*} w_{l^*}^2 \sigma_{l^*})^2 \leq \hat{\sigma}^2 \leq \max_{l^*} \sigma_{l^*}^2 \tag{21}$$

*Proof* For the right-hand side of the variance inequality, observe that

$$\begin{aligned} \hat{\sigma}^2 &= \sum_{l^*} w_{l^*}^2 \sigma_{l^*}^2 \leq \sum_{l^*} w_{l^*} \sigma_{l^*}^2 \quad (\text{ because } w_{l^*}^2 \leq w_{l^*}) \\ &\leq (\sum_{l^*} w_{l^*}) \max_{l^*} \sigma_{l^*}^2 \\ &\leq \max_{l^*} \sigma_{l^*}^2 \quad (\text{ because } \sum_{l^*} w_{l^*} = 1) \end{aligned} \tag{22}$$

For the left-hand side of the variance inequality, recall the Jensen's inequality: $\rho\left(\frac{\sum_i a_i x_i}{\sum_i a_i}\right) \leq \frac{\sum_i a_i \rho(x_i)}{\sum_i a_i}$, where $\rho(\cdot)$ is a convex function. Substituting $w_{l^*}^2 \to a_i$, $\sigma_{l^*} \to x_i$ and $\rho(x) = x^2$ into the Jensen's inequality, we have

$$\begin{aligned} \left(\frac{\sum_{l^*} w_{l^*}^2 \sigma}{\sum_{l^*} w_{l^*}^2}\right)^2 \leq \frac{\sum_{l^*} w_{l^*}^2 \sigma_{l^*}^2}{\sum_{l^*} w_{l^*}^2} \quad \text{or} \quad \left(\sum_{l^*} w_{l^*}^2 \sigma_{l^*}\right)^2 \\ \leq \left(\sum_{l^*} w_{l^*}^2\right)\left(\sum_{l^*} w_{l^*}^2 \sigma_{l^*}^2\right) \end{aligned} \tag{23}$$

Now, note that $\sum_{l^*} w_{l^*}^2 \leq \sum_{l^*} w_{l^*} = 1$. We obtain the left-hand side of the inequality. □

**Theorem 3** *The largest weight is associated with the lth cluster.*

*Proof* Based on the weight formula,

$$w_{l^*} \propto \left[\sigma_l^2 + W_2\left(\mathcal{N}(\mu_{l^*}, \sigma_{l^*}^2), \mathcal{N}(\mu_l, \sigma_l^2)\right)\right]^{-1}, \tag{24}$$

it is easy to see that the Wasserstein distance between a cluster with itself is zero.

Thus, the right-hand side is always less than $\sigma_l^2$, i.e.,

$$\sigma_l^2 + W_2\left(\mathcal{N}(\mu_{l^*}, \sigma_{l^*}^2), \mathcal{N}(\mu_l, \sigma_l^2)\right) \geq \sigma_l^2. \tag{25}$$

Inversing the last inequality completes the proof. The equality occurs when $l^* = l$. □

**Theorem 4** *The expectation of the posterior mean* $\hat{\mu} = \sum_{l^* \in \mathcal{B}(l)} w_{l^*} (\mu_{l^*} + \bar{\mu}_l - \bar{\mu}_{l^*})$ *is* $\bar{\mu}_l$*, i.e.,* $\mathbb{E}[\hat{\mu}] = \bar{\mu}_l$.

*Proof* Take the expectation of (9) for any $l$th cluster over the continuous domain, and note that $\mathbb{E}[\boldsymbol{y} - \boldsymbol{m}] = 0$, the mean of the posterior is recovered to the mean of the cluster, i.e.,

$$\mathbb{E}[\mu_l(\boldsymbol{x})] = \mu_0(\boldsymbol{x}) = \bar{\mu}_l(\boldsymbol{x}). \tag{26}$$

Equation 26 holds for any $l$th under the GP formulation. In the similar manner, taking the expectation of the posterior mean $\hat{\mu}$ from the proposed method over the continuous domain, we arrive at

$$\begin{aligned} \mathbb{E}[\hat{\mu}] &= \sum_{l^* \in \mathcal{B}(l)} w_{l^*} \mathbb{E}\left[\mu_{l^*} + \bar{\mu}_l - \bar{\mu}_{l^*}\right] \\ &= \sum_{l^* \in \mathcal{B}(l)} w_{l^*} \left[\mathbb{E}[\mu_{l^*}] + \mathbb{E}[\bar{\mu}_l] - \mathbb{E}[\bar{\mu}_{l^*}]\right] \\ &= \sum_{l^* \in \mathcal{B}(l)} w_{l^*} \left[\bar{\mu}_{l^*} + \mathbb{E}[\bar{\mu}_l] - \bar{\mu}_{l^*}\right] \\ &= \sum_{l^* \in \mathcal{B}(l)} w_{l^*} \left[\bar{\mu}_l\right] \\ &= \bar{\mu}_l, \end{aligned} \tag{27}$$

where the second equality is formed by distributing the expectation operator under linear combination rule. The

third equality follows (26) as described above. The fourth equality is formed by canceling two identical terms $\bar{\mu}_{I^*}$. □

A major problem of GP is its scalability, which originates from the computation of the inverse of correlation matrices. The dataset decomposition has a favorable computational aspect in which the scalability is alleviated. Here, we analyze the computational cost based on the assumption that the size of each cluster is roughly equal. Denote the number of data points for the whole dataset as $N$, and the number of clusters as $k$. The computational cost to compute all covariance matrices is reduced by a factor of $k^2$, as $k$ covariance matrices are involved, and each covariance matrix has the computational complexity $\mathcal{O}\left(\frac{N}{k}\right)^3$, thus resulting in the total cost of $k\mathcal{O}\left(\frac{N}{k}\right)^3 = \frac{1}{k^2}\mathcal{O}\left(N^3\right)$. Similarly, the cost of storing covariance matrices is also reduced by a factor of $k$, since $k\mathcal{O}\left(\frac{N}{k}\right)^2 = \frac{1}{k}\mathcal{O}(N^2)$. However, the computational cost of predicting the posterior mean $\mu$ and posterior variance $\sigma^2$ stays the same, since $k\mathcal{O}\left(\frac{N}{k}\right) = \mathcal{O}(N)$.

The decomposition approach in the proposed mixed-integer BO has a computational advantage to mitigate the scalability problem in GP, even though it is not completely eliminated.

## 4 Analytical examples

In this section, the proposed mixed-integer BO is compared with the genetic algorithm (GA) with various settings.

The settings for the GA are described as follows. To verify the robustness of the proposed method, three GA settings are chosen. In the first setting, the population size and the elite count parameters are set to be 50 and 3, respectively. In the second setting, the population size and the elite count parameters are set to be 150 and 10, respectively. In the third setting, they are 1500 and 10, respectively. Other parameters are left to be the default values in MATLAB function `ga()`.

In Section 4.1, a discrete modification of the multi-modal Rastrigin function is used as a benchmark function, where two variables are discrete and the other two are continuous. In Section 4.2, a welded beam design optimization with two discrete and four continuous variables is used to evaluate the performance of the proposed mixed-integer BO method where discrete variables come from the configuration and material of the beam. In Section 4.3, a pressure vessel design optimization with four continuous variables is benchmarked. In Section 4.4, a speed reducer design optimization function with one discrete and six continuous variables is utilized. In Section 4.5, a modification of discrete sphere function is devised to demonstrate the proposed mixed-integer BO method on high-dimensional optimization problems, with 5 discrete and 50 and 100 continuous variables.

### 4.1 Discrete Rastrigin function

In this example, the proposed method is applied on the discrete version of the Rastrigin function, which is an analytical function for testing different optimization methods. To evaluate the effectiveness of the proposed mixed-integer BO method, the optimization performance is compared against GA optimization performance.

#### 4.1.1 Problem statement

The DACE toolbox (Nielsen et al. 2002) for classical GP is extended to include the proposed distributed GP and Bayesian optimization. In this section, the hybrid Bayesian optimization is to find the global minimum on a tiled version of the Rastrigin function on 25 clusters, where each cluster corresponds to two discrete variables. The input $\boldsymbol{x} = (i, j, x, y)$ is comprised of four variables, in which the first two are discrete and the last two are continuous, as illustrated in Fig. 3. The original two-dimensional Rastrigin function is $f(x, y) = 20 + [x^2 - 10\cos(2\pi x) + y^2 - 10\cos(2\pi y)]$, where $-5.12 \leq x, y \leq 5.12$. The tiled Rastrigin function is constructed based on a tiled domain of the Rastrigin function, where each domain is characterized by a discrete tuple $(i, j)$, and the continuous domain is translated to $-0.75 \leq x_{\text{tiled}}, y_{\text{tiled}} \leq 0.75$ for all clusters. Figure 3 illustrates the construction of the tiled Rastrigin

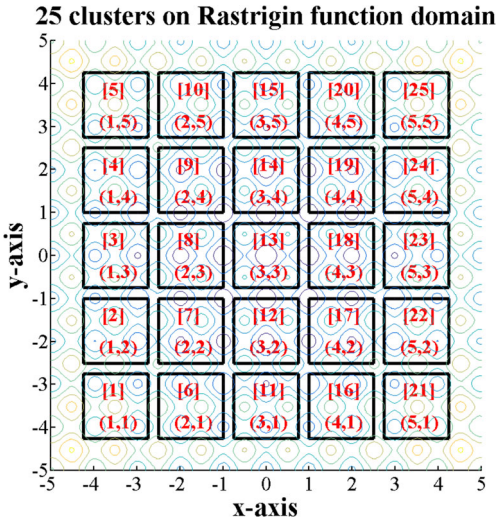



**Fig. 3** Tiled Rastrigin function comprising of 25 clusters, where each cluster corresponds to a square of dimension 1.50 × 1.50 and a tuple $(i, j)$. The cluster index is denoted within the square bracket $[\cdot]$, whereas the tuple is within the parenthesis $(\cdot, \cdot)$ in each square

function, and its relationship with the original Rastrigin function. The relationship between the tiled and original Rastrigin can simply be described by an affine function,

$$x_{\text{orig}} = -3.50 + 1.75(i-1) + x_{\text{tiled}};$$
$$y_{\text{orig}} = -3.50 + 1.75(j-1) + y_{\text{tiled}}, \quad (28)$$

where $-0.75 \leq x_{\text{tiled}}, y_{\text{tiled}} \leq 0.75$.

#### 4.1.2 Numerical results

In this example, to find the minimum of the Rastrigin function, we flip the sign of tiled Rastrigin and use the UCB acquisition function to locate the maximum of the negative tiled Rastrigin function. The covariance matrix adaptation evolution strategy (CMA-ES) (Hansen et al. 2003) method is employed to find the next sampling point within each cluster by locating the point with the maximum acquisition function. The parameters are set as follows: $\kappa = 5$, $d_{\text{penalty}} = 10^{-4}$, $N_{\text{shuffle}} = 15$, where $N_{\text{shuffle}}$ is the number of steps which CMA-ES is reactivated with different initial positions to search for the next sampling point on each local GP in order to avoid trapping in the local minima. To construct the initial GP response surface, 5 random data points are sampled from each cluster.

Because the global minimum of the original Rastrigin function is at $(x = 0, y = 0)$ with the functional evaluation $f(0, 0) = 0$, the hybrid Bayesian optimizer on the tiled Rastrigin function is expected to converge to cluster 13, as illustrated in Fig. 3. The neighbor list of cluster 13 includes clusters 8, 12, 13, 14, and 18. Figure 4 compares the numerical performance between the proposed mixed integer BO and the GA with three different settings.

Figure 4 presents the performance of the proposed method (solid line) with five different settings, and the GA method (dash line) with three different settings. For the proposed mixed-integer BO, the threshold distance $d_{\text{th}}$ is changed. The proposed mixed-integer BO performs best with small $d_{\text{th}}$ parameter, which measures the dissimilarity between discrete tuples.

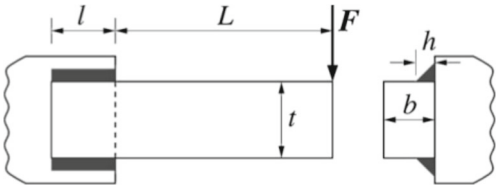



**Fig. 5** Welded beam design problem (Datta and Figueira 2011)

### 4.2 Welded beam design problem

To verify the result of the proposed method, an analytical engineering model for welded beam design is adapted from Deb and Goyal (1996), Gandomi and Yang (2011), Rao (2009), and Datta and Figueira (2011), as shown in Fig. 5, with some slight modifications.

#### 4.2.1 Problem statement

The low-carbon steel (C-1010) beam is welded to a rigid base to support a designated load $F$. The thickness of the weld $h$, the length of the welded joint $l$, the width of the beam $t$, and the thickness of the beam $b$ are the design continuous variables. Two different welding configurations can be used, four-sided welding and two-side welding (Deb and Goyal 1996). The bulk material of the beam can be steel, cast iron, aluminum, or brass, which is associated with different material properties. The stress, deflection, and buckling conditions are derived from Ravindran et al. (2006), where the constant parameters are as follows: $L = 14$inch, $\delta_{\max} = 0.25$ inch, and $F = 6,000$lb. The input $\boldsymbol{x}$ is comprised of $(w, m, h, l, t, b)$, where $w$ and $m$ are discrete variables and $h$, $l$, $t$, and $b$ are continuous variables. We

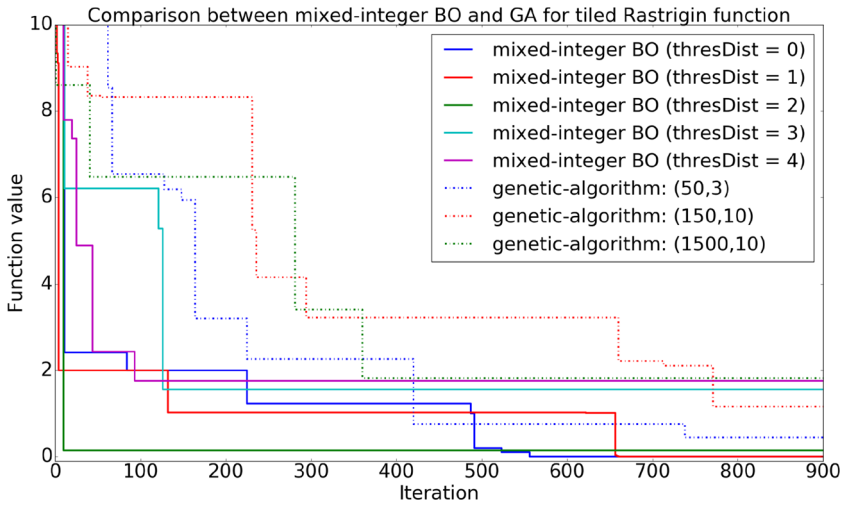



**Fig. 4** Performance comparison between the GA and the proposed mixed-integer BO for the tiled Rastrigin function

note that $h$, $t$, and $b$ are commonly considered as discrete variables in multiples of 0.0625 in, as well as continuous variables, bounded between lower and upper bounds.

Under this formulation, the objective is to minimize

$$f(w,m,h,l,t,b) = (1+C_1)(wt+l)h^2 + C_2 tb(L+l) \quad (29)$$

subject to the five inequality constraints:

$$\text{shear stress}(\tau) : g_1 = 0.577\sigma_d - \tau(\boldsymbol{x}) \geq 0 \quad (30a)$$

$$\text{bending stress in the beam}(\sigma) : g_2 = \sigma_d - \sigma(\boldsymbol{x}) \geq 0 \quad (30b)$$

$$\text{buckling load on the bar}(P_c) : g_3 = b - h \geq 0 \quad (30c)$$

$$\text{deflection of the beam} : g_4 = P_c(\boldsymbol{x}) - F \geq 0 \quad (30d)$$

$$\text{side constraints} : g_5 = \delta_{\max} - \delta(\boldsymbol{x}) \geq 0 \quad (30e)$$

where

$$\sigma(\boldsymbol{x}) = \frac{6FL}{t^2 b}, \delta(\boldsymbol{x}) = \frac{4FL^3}{Et^3 b}, P_c(\boldsymbol{x}) = \frac{4.013tb^3\sqrt{EG}}{6L^2}\left(1 - \frac{t}{4L}\sqrt{\frac{E}{G}}\right) \quad (31a)$$

$$\tau = \sqrt{(\tau')^2 + (\tau'')^2 + 2\tau'\tau''\cos\theta}, \tau' = \frac{F}{A}, \tau'' = \frac{F(L+0.5l)R}{J} \quad (31b)$$

$$w = 0 : \begin{cases} A = \sqrt{2}hl \\ J = \sqrt{2}hl\left[\frac{(h+t)^2}{4} + \frac{l^2}{12}\right] \\ R = \frac{1}{2}\sqrt{l^2 + (h+t)^2} \\ \cos\theta = \frac{l}{2R} \end{cases}, \quad (31c)$$

$$w = 1 : \begin{cases} A = \sqrt{2}h(t+l) \\ J = \sqrt{2}hl\left[\frac{(h+t)^2}{4} + \frac{l^2}{12}\right] + \sqrt{2}ht\left[\frac{(h+l)^2}{4} + \frac{t^2}{12}\right] \\ R = \max\left\{\frac{1}{2}\sqrt{l^2+(h+t)^2}, \frac{1}{2}\sqrt{t^2+(h+l)^2}\right\} \\ \cos\theta = \frac{l}{2R} \end{cases} \quad (31d)$$

where $w$ is the binary variable to model the type of weld, $w = 0$ is used for two-sided welding and $w = 1$ is used for four-sided welding. $C_1(m)$, $C_2(m)$, $\sigma_d(m)$, and $E(m)$, $G(m)$ are material-dependent parameters (Deb and Goyal 1996; Gandomi and Yang 2011) listed in Table 1. The lower and upper bounds of the problem are $0.0625 \leq h \leq 2$, $0.1 \leq l \leq 10$, $2.0 \leq t \leq 20.0$, and $0.0625 \leq b \leq 2.0$ (Datta and Figueira 2011).

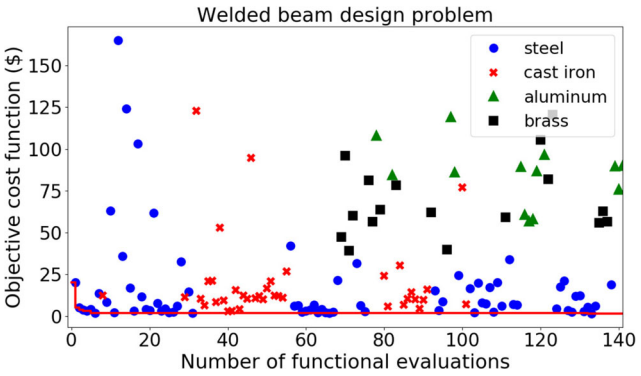



**Fig. 6** Convergence plot of the cost function in the welded beam design, with all clusters are neighbors, showing different combinatorial of discrete and categorical variables are attempted

#### 4.2.2 Numerical results

Here, the input vector is encoded as $\boldsymbol{x} = (w, m, h, l, t, b)$, where $w \in \{0, 1\}$, where $w = 0$ and $w = 1$ correspond to the two-sided and four-sided welding, respectively; $m \in \{1, 2, 3, 4\}$ corresponds to steel, cast iron, aluminum, and brass, respectively.

In this example, there are 8 clusters, because there are two choices for $w$ and four choices for $m$. The neighborhood $\mathcal{B}(\cdot)$ is considered as universal, i.e., the neighborhood for each cluster includes all clusters, such that they are all aware of others.

The bounds for hyperparameters $\theta$ for the GP in each cluster are set as follows: $\underline{\theta} = (0.1, 0.1, 0.1, 0.1)$, $\overline{\theta} = (20.0, 20.0, 20.0, 20.0)$. Every four iterations, the sampling point location in each cluster is computed again to avoid trapping in local minima. CMA-ES (Hansen et al. 2003) is used as an auxiliary optimizer for maximizing the acquisition function. There are two random sampling points in each cluster to initialize the GP construction. The EI acquisition function is used.

Figure 6 shows the convergence plot of the cost function in the welded beam design, where the circle, cross, triangle, and square corresponds to steel, cast iron, aluminum, brass, respectively. The optimal cost value $f(\boldsymbol{x})$ evolves at iterations 0, 1, 2, 3, 5, and 132, with the values of 20.1995, 5.0605, 3.7949, 3.2436, 1.7420, and 1.6297, respectively, with the last one being four-sided welded.

**Table 1** Material-dependent parameters and constants in the welded beam design problem

| Constants | Description | Steel | Cast iron | Aluminum | Brass |
|---|---|---|---|---|---|
| $C_1$ | Cost per volume of the welded material ($/in$^3$) | 0.1047 | 0.0489 | 0.5235 | 0.5584 |
| $C_2$ | Cost per volume of the bar stock ($/in$^3$) | 0.0481 | 0.0224 | 0.2405 | 0.2566 |
| $\sigma_d$ | Design normal stress of the bar material (psi) | $30 \cdot 10^3$ | $8 \cdot 10^3$ | $5 \cdot 10^3$ | $8 \cdot 10^3$ |
| $E$ | Young's modulus of bar stock (psi) | $30 \cdot 10^6$ | $14 \cdot 10^6$ | $10 \cdot 10^6$ | $16 \cdot 10^6$ |
| $G$ | Shear modulus of bar stock (psi) | $12 \cdot 10^6$ | $6 \cdot 10^6$ | $4 \cdot 10^6$ | $6 \cdot 10^6$ |

Compared to Datta and Figueira (2011), where the optimal value is $f(\boldsymbol{x}) = 1.9553$, our obtained result $f(\boldsymbol{x}) = 1.6297$ is smaller, because in our formulation $h$, $t$, and $b$ are continuous variables, in contrast to Datta and Figueira (2011) with $h$, $t$, and $b$ as discrete variables. Furthermore, the convergence occurs relatively fast, as the optimization algorithm exploits the most promising cluster by maximizing the acquisition function. This behavior can be explained by the fact that in this welded beam design example, different materials have significantly different cost objective functional value, which aids the optimization convergence.

To further demonstrate the effectiveness of the proposed method, we compare with GA. Two versions of the proposed method are used. In the first version, every cluster are considered as independent, leaving no neighbor in the neighborhood, whereas in the second version, all the clusters are considered as neighbors.

The performance comparison is presented in Fig. 7, showing that both variants of the mixed-integer BO clearly outperforms the GA in the welded beam design problem. The solution obtained from the GA is [0, 1, 0.24920115, 5.30060037, 7.12520087, 0.25345267], where the objective function is evaluated at 2.04016262. On the other hand, from the first variant (none is neighbor) of the proposed method, the solution obtained is [1, 1, 0.16934934, 5.61720010, 4.90884889, 0.27985016], where the objective function is evaluated at 1.68206763. From the second variant (all are neighbors) of the proposed method, the solution obtained is [1, 1, 0.16934934, 5.61720010, 4.90884889, 0.27985016], where the objective function is evaluated at 1.66457625. The convergence plots of these two variants are very similar. The asymptotic value using the second variant is slightly better than that using the first variant. However, we note that as the optimization process advances, the prediction converges to a single GP prediction, and thus both variants are similar at the later stage of search.

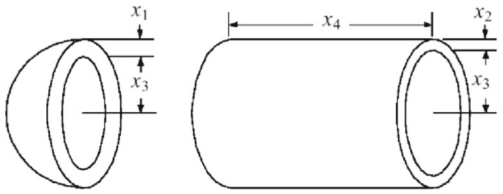



**Fig. 8** Pressure vessel design optimization problem (Cagnina et al. 2008)

The proposed mixed-integer method clearly outperforms the GA in all settings.

### 4.3 Pressure vessel design problem

Here, the proposed mixed-integer BO method is applied to solve the pressure vessel design optimization problem. The objective of this problem is to minimize the cost of a storage tank with $3{\cdot}10^3$ psi internal pressure shown in Fig. 8, where the minimum volume is 750 ft$^3$. The shell is made by joining two hemispheres and forming the longitudinal cylinder with another weld. The design variables are listed as follows: $x_1$ is the thickness of the hemisphere, $x_2$ is the shell thickness, $x_3$ is the inner radius of the hemisphere, $x_4$ is the length of the cylinder.

The objective function that accounts for the cost is

$$f(\boldsymbol{x}) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3, \tag{32}$$

where the imposed constraints are

$$g_1(\boldsymbol{x}) = -x_1 + 0.0193x_3 \leq 0,$$
$$g_2(\boldsymbol{x}) = -x_2 + 0.009541x_3 \leq 0, \tag{33a}$$
$$g_3(\boldsymbol{x}) = -\pi x_3^2x_4^2 - \frac{4}{3}x_3^3 + 1296000 \leq 0,$$
$$g_4(\boldsymbol{x}) = x_4 - 240 \leq 0, \tag{33b}$$

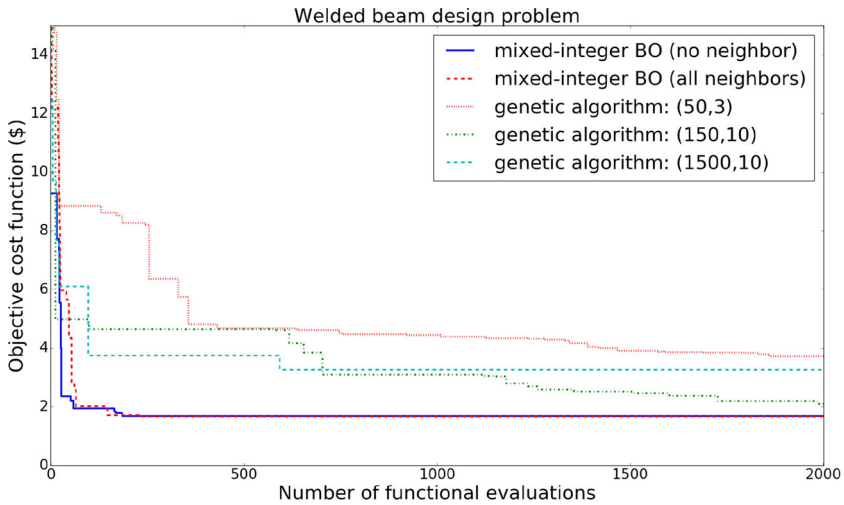



**Fig. 7** Performance comparison between the GA and the proposed mixed-integer BO for the welded beam design

and $0.00625 \leq x_1, x_2 \leq 0.61875$, $10.0 \leq x_3, x_4 \leq 200.0$. All variables are considered as continuous in this example.

Figure 9 shows the performance comparison between the proposed mixed-integer BO and the GA with various settings in terms of number of functional evaluations. Again, the BO clearly shows its advantage in term of convergence speed for continuous variables. The optimal input is [0.193114320, 0.0954997100, 10, 76.2478356], where the corresponding objective functional value is 125.02822748.

### 4.4 Speed reducer design problem

Figure 10 shows the design optimization problem of a speed reducer (Cagnina et al. 2008). Seven design variables are described as follows: $x_1$ is the face width, $x_2$ is the module of teeth, $x_3$ is the number of teeth on pinion, $x_4$ is the length of the first shaft between bearings, $x_5$ is the length of the second shaft between bearings, $x_6$ is the diameter of the first shaft, $x_7$ is the diameter of the second shaft. $x_3$ is the discrete variable, whereas the rest of the variables are continuous. The problem is 7-dimensional, one discrete and six continuous. With the formulation of the problem, there are 12 local GPs corresponding to 12 discrete values of $x_3$.

In iteration 148, the mixed-integer BO converges to the global minimum of $f(\boldsymbol{x}^*) = 2996.29614837$, where $\boldsymbol{x}^* = [3.50000447, 0.7, 17, 7.30566156, 7.8, 3.35022572, 5.28668406]$. The result is comparable with Cagnina et al. (2008), where particle swarm optimization is employed, yielding the optimal $f(\boldsymbol{x}^*) = 2996.348165$, where $\boldsymbol{x}^* = [3.5, 0.7, 17, 7.3, 7.8, 3.350214, 5.286683]$.

To evaluate the effect of initial sample size, the mixed-integer BO is performed with different number of initial samples. Figure 11 shows the convergence plot of the GA and the mixed-integer BO, each with various settings. In terms of the number of functional evaluations, the mixed-integer BO clearly shows the advantages with faster convergence, compared to the GA. The effect of initial samples is also shown in Fig. 11. It is observed that the proposed mixed-integer BO converges relatively fast after the initial sampling stage. Thus, for low-dimensional problems, it may not be necessary to sample extensively at the initial sampling stage. The balance between exploration and exploitation is well-tuned by the acquisition function, which is GP-UCB (Srinivas et al. 2012) in this case.

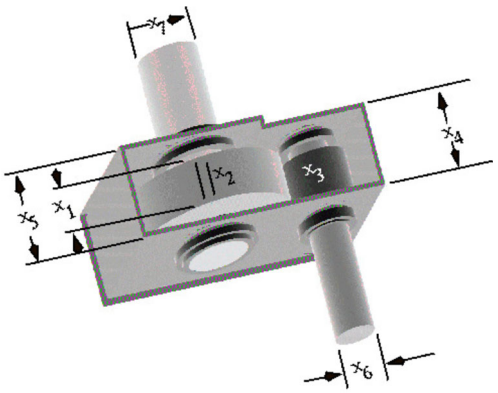



**Fig. 10** Speed reducer design optimization problem (Cagnina et al. 2008) from NASA

### 4.5 High-dimensional discrete sphere function

To evaluate the performance of the proposed mixed-integer BO in high-dimensional problems, two discrete sphere functions with 5-dimensional discrete variables and 50-dimensional and 100-dimensional continuous variables, respectively, are used to benchmark. The discrete sphere function is

$$f(\boldsymbol{x}^{(d)}, \boldsymbol{x}^{(c)}) = f(x_1, \cdots, x_n, x_{n+1}, \cdots, x_m) = \prod_{i=1}^{n} |x_i| \left( \sum_{j=n+1}^{m} x_j^2 \right) \tag{34}$$

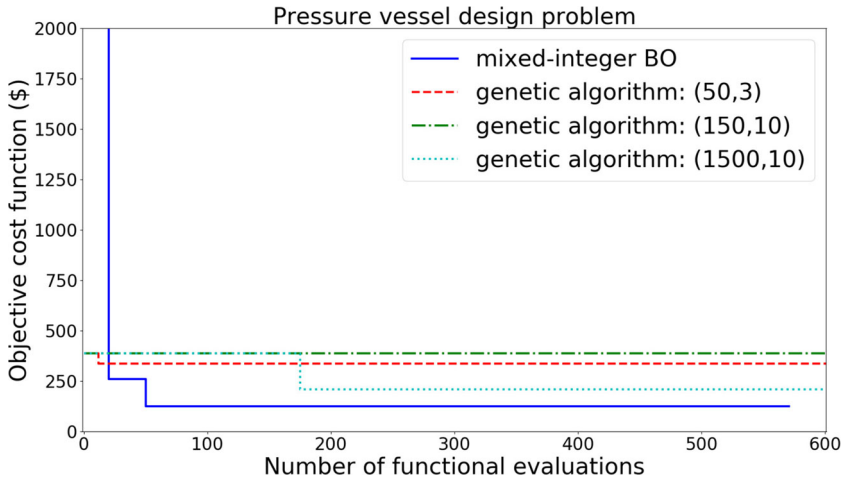



**Fig. 9** Performance comparison between the GA and the proposed mixed-integer BO for the pressure vessel design

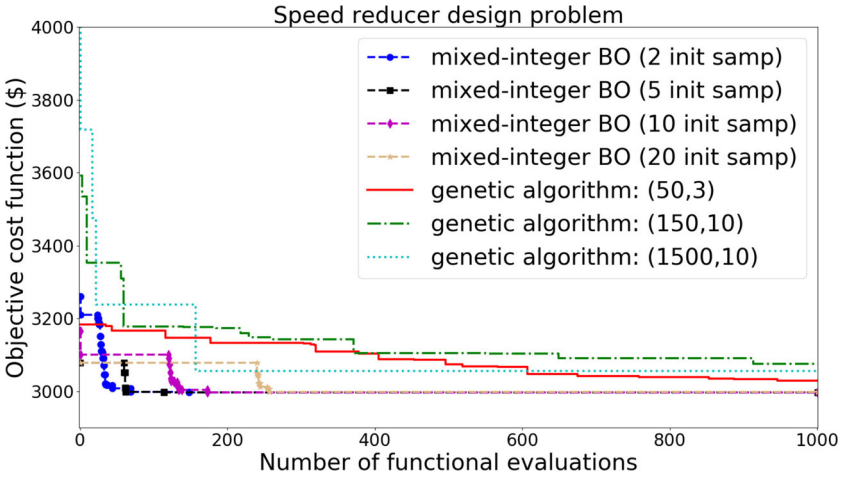



**Fig. 11** Performance comparison between the GA and the proposed mixed-integer BO with different initial samples for the speed reducer design

where $1 \leq x_i \leq 2 (1 \leq i \leq n)$ are $n$ integer variables and $-5.12 \leq x_j \leq 5.12 (n+1 \leq j \leq m)$ are $m - n$ continuous variables. Again, GA is used to compare against the proposed mixed-integer BO method. The global optimal of this function is $f(\boldsymbol{x}^*) = 0$, where $\boldsymbol{x}^* = [1, 1, 1, 1, 1, 0, \ldots, 0]$. The number of clusters in this example is $2 \times 2 \times 2 \times 2 \times 2 = 32$, where each cluster corresponds to a local GP. Figure 12 shows the convergence plot of the proposed mixed-integer BO with different number of initial samples and GA with different settings for the (50+5)D discrete spherical function, where 5 variables are discrete and 50 variables are continuous.

As seen in Fig. 12, the proposed mixed-integer BO quickly identifies the discrete tuple (1, 1, 1, 1, 1) that corresponds to the minimal response, with respect to the discrete tuple. The rest of the convergence plot focuses on the optimization of the continuous variables. The GA with population size of 50 and elite count of 3 performs on par with the proposed mixed-integer BO, whereas other GA settings converge much slower. The mixed-integer BO with 2 initial samples converges relatively fast at the beginning. However, the convergence at the later stage stagnates over a long period. On the contrary, the mixed-integer with 20 initial samples converge very fast right after the initial sampling stage. One of the reasons is that the local GP is able to approximate the objective function more accurately with more initial samples, compared to the one with less initial samples.

Similarly, Fig. 13 shows the convergence plot of the proposed mixed-integer BO with a different number of initial samples and GA with different settings for (100+5)D discrete spherical function, where 5 variables are discrete. The mixed-integer with 2 initial samples converges poorly, whereas other variants perform better. One of the reasons is that with the low initial sample size, the discrete tuple is incorrectly identified as (1,1,1,1,2), as opposed to (1,1,1,1,1). The other variants of the proposed mixed-integer BO are able to identify the correct tuple immediately after the initial sampling stage. Thus, it may be beneficial to have sufficient number of initial samples.

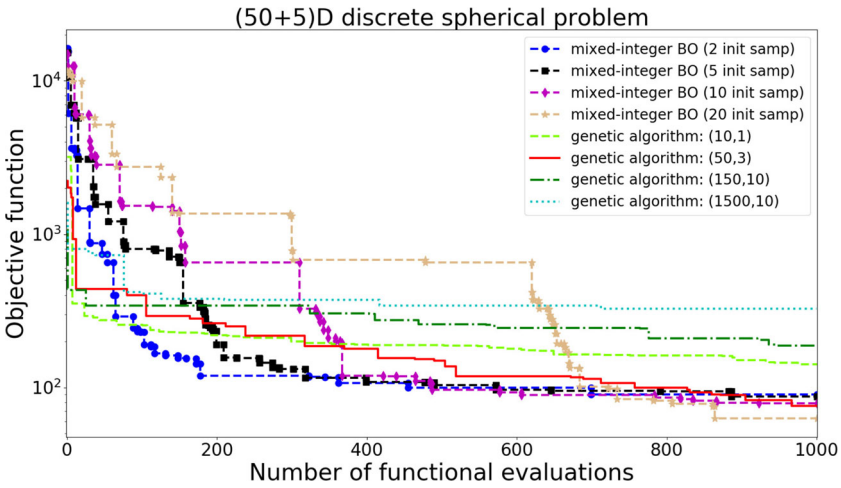



**Fig. 12** Performance comparison between the GA and the proposed mixed-integer BO with different initial samples for (50+5)D discrete spherical function

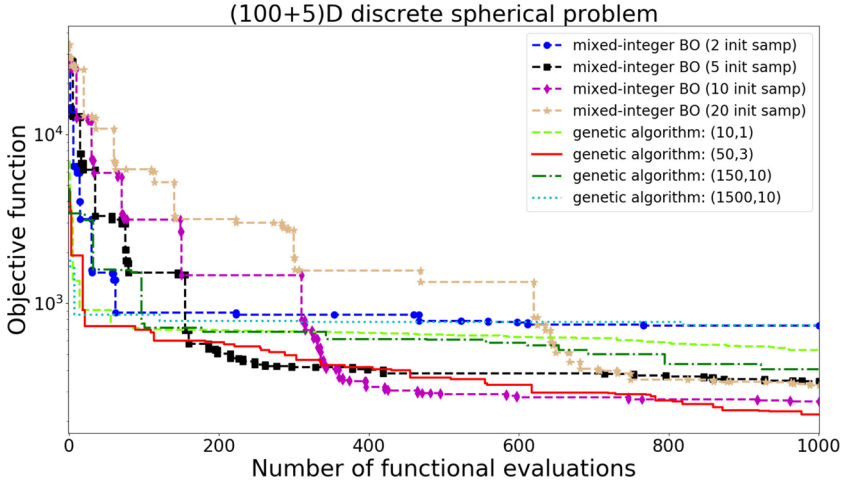



**Fig. 13** Performance comparison between the GA and the proposed mixed-integer BO with different initial samples for (100+5)D discrete spherical function

## 5 Metamaterials design examples

In this section, we demonstrate the applicability of the proposed method to the design of metamaterials, in which properties can be tailored depending on the geometric design of the structures. In Section 5.1, a mechanical metamaterial is considered, where the objective is to design a low-weight and high-strength unit cell. In Section 5.2, an auxetic metamaterial unit cell is considered. The proposed BO method is applied to minimize the negative Poisson's ratio.

### 5.1 An example of designing high-strength low-weight fractal metamaterials

Motivated by the recent experimental work of Meza et al. (2014) in designing high-strength and low-weight metamaterials at nano-scale for ceramic systems where the effective mechanical strength can be enhanced by hierarchical structure. We demonstrate the proposed methodology in searching for high-strength and low-weight metamaterials for multiple classes of materials.

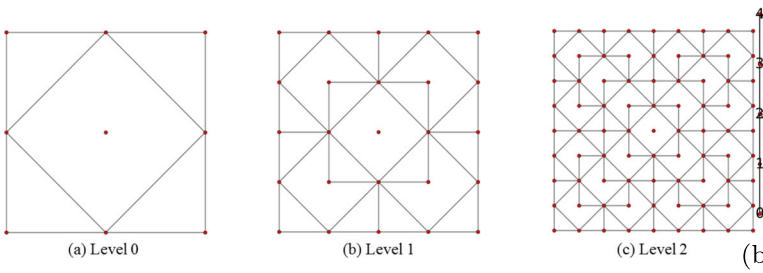
Particularly, our metamaterials are constructed with fractal geometry. Fractal geometry has the special property of self-similarity at different length scales. A parametric design and optimization approach for fractal metamaterials is demonstrated here. In this example, the goal is to maximize the effective strength of the structure.

The effective strength is defined as the ratio between the effective Young modulus and the volume of material with the assumption of homogenized material for the bulk properties. The material selection, including Ashby chart, is formulated as an inequality constraint to limit the searching space of materials.

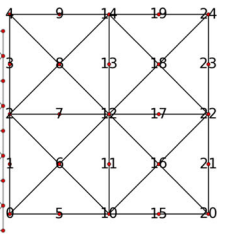
#### 5.1.1 Parametric design of fractal truss structures

Mathematically, fractals can be constructed iteratively using the so-called iterated function systems (IFSs). An IFS is a finite set of contraction mappings $\{f_i\}_{i=1}^N$ on a complete metric space $X$ (Barnsley 2014). Starting from an initial set $\mathcal{P}_0$, the fractal can be constructed iteratively as $\mathcal{P}_{k+1} = \cup_{i=1}^N f_i(\mathcal{P}_k)$. Geometrically, the IFSs $f_i$ can be expressed in terms of rotation, translation, scaling, and other set topological operations, such as complement, union, or intersect.

In this example, the fractal truss structures are constructed from the 2D profiles shown in Fig. 14c. They are based on the square shape, even though in principle they can be constructed from any arbitrary polygon such as triangle and hexagon. Figure 14c presents the first three levels of IFS construction. The IFSs are inspired by the projection




(a) Iterated function systems of truss designs on unit square.




(b) Truss options on fractal level 0 unit square.

**Fig. 14** Truss design parameters on the unit square: **a**–**c** Iterated function systems of truss designs on unit square, levels 0–2, respectively. **d** Truss options on fractal level 0 unit square

of Keplerian 3D fractals onto its corresponding 2D plane. Here, the IFS operators include the translation matrix $T = \text{diag}\{\pm d/2, \pm d/2, 1\}$ and the scaling matrix $S = \text{diag}\{1/2, 1/2, 1\}$. The rotation is not considered. Physically, the first four IFSs simply scale the design of previous fractal level by $1/2$, and translate them to the northwest, northeast, southwest, and southeast, respectively. The fifth IFS scales the design of previous fractal level by one half and deletes other features that overlap within the region.

Figure 14 illustrates the square basis with three design options: (1) diagonal truss, (2) inner square truss, and (3) perpendicular truss. The diagonal truss option enables edges connecting nodes 4, 8, 12, 16, and 20 and nodes 0, 6, 12, 18, and 24. The inner square truss option enables edges connecting nodes 2, 6, 10, 15, 22, 18, 14, and 8. The perpendicular truss option enables edges connecting nodes 2, 7, 12, 17, and 22 and nodes 10, 11, 12, 13, and 14. In the example of Fig. 14c, only the inner square truss option is enabled. In the construction process, the options are enabled by setting the truss control parameters to 0 or 1, respectively. The fundamental adjacency matrix of fractal level 0 is built to indicate whether a pair of nodes are connected. With the design of level 0 unit cell, the IFSs are applied recursively to create the more complicated geometry at the desired level. Once the profile is constructed, additional offset operations are applied to generate thickness of the 2D truss elements for a full 3D structure. Figure 15a shows a complete 2D fractal face. With the square face defined, a complete 3D fractal unit cell is built with six of the faces, as shown in Fig. 15b.

#### 5.1.2 Constitutive material model and the finite element analysis

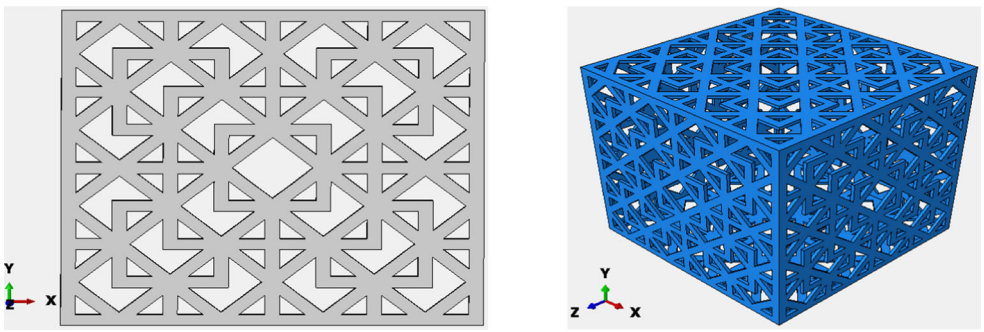
A general anisotropic material has 21 independent elastic constants to describe the stress-strain ($\sigma$-$\varepsilon$) relationship. To simplify the materials constitutive model, we assume isotropic and linear elastic materials behavior at small strain regime, where $\sigma$-$\varepsilon$ relationship for bulk material properties can be obtained via Young's modulus $E$ and Poisson's ratio $\nu$, i.e.,

$$\sigma_{ij} = \frac{E}{1+\nu}\left(\varepsilon_{ij} + \frac{\nu}{1-2\nu}\varepsilon_{kk}\delta_{ij}\right), \tag{35}$$

where $i$ and $j$ can be either $x, y$, or $z$, and $\delta_{ij}$ is the Kronecker delta of $i$ and $j$. The material properties $E$ and $\nu$, as well as material $\rho$, are taken as inputs to describe the linear elastic regime in the FEM simulation to obtain stress.

In simulations, we are concerned with an uniaxial compression. Therefore, to simplify the terminology, we refer to the component of effective stiffness tensor in the loading direction as effective Young's modulus. It is noteworthy that the effective stiffness tensor of the designed fractal truss structure is not the same as the bulk material stiffness tensor. Two displacement boundary conditions are imposed on the unit cube. One is the fixed boundary condition for both translation and rotation, and the other is the constant displacement on the opposite side of the cube. The stress is obtained by taking the maximum nodal stress in the active direction. The effective Young's modulus is calculated as the ratio of the maximal nodal stress $\sigma_{33}$ at the designated engineering strain $\varepsilon = 0.01$. The quadratic tetrahedral element (C3D10 in ABAQUS) is utilized for the FEM simulation. The total number of elements is between 5000 and 10,000. The exact number varies with respect to the finite element simulation. The size of the cube is around 1 mm ($10^{-3}$ m).

The dimension of the design space is 9, in which 4 discrete and 5 continuous variables are combined to create an input $\boldsymbol{x} = (x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9)$. The discrete variables include fractal level, the diagonal, inner square, and perpendicular truss options. The fractal level $x_1$ is an integer of either 0, 1, or 2, whereas each of the truss options $x_2$, $x_3$, and $x_4$ is a binary variable from design space, taking a value of 0 or 1. The continuous variables include thickness $x_5 = t$ of the truss, the extrusion depth $x_6 = et$ of the unit face, the material bulk density $x_7 = \rho$, bulk elastic Young's modulus $x_8 = E$, and bulk Poisson's ratio $x_9 = \nu$.




(a) 2D fractal face (b) The unit cube

**Fig. 15** Design of fractal unit cube. **a** The 2D fractal profile with a fractal level of 2 and only inner square truss option enabled. **b** The unit cube is composed of six identical fractal faces, and each face is designed by truss options, thickness, and extrusion depth

Three constraints are imposed as follows. Thickness and extrusion depth are limited to a constant that is related to the fractal level to preserve the fractal geometry of the structure. The higher the fractal level is, the smaller is the constant. Similarly, the material bulk density, Young's modulus, and Poisson's ratio are bounded within a physical limit, where values are taken from Table 3.1 of Bower (2011) for woods, copper, tungsten carbide, silica glass, and alloys. As a result, the imposed constraints are

$$\underline{T} \le x_5 \le \overline{T}, \quad x_6 \ge \overline{T}, \tag{36a}$$

$$x_5 \le 7 \cdot x_6, \quad x_6 \le 7 \cdot x_5, \tag{36b}$$

where $\underline{T} = 10^{-6}$ is the threshold for manufacturability and $\overline{T}$ is the threshold for the truss thickness as

$$\overline{T} = \begin{cases} \frac{1}{2 \cdot 2^{x_1+1}}, & \text{if} x_3 = x_4 = 2, \\ \frac{1}{2 \cdot 2^{x_1}}, & \text{otherwise.} \end{cases} \tag{37}$$

We expect the simulations to converge on the high-strength and low-density type of materials. However, Ashby chart indicates a high correlation between compressive strength and density among all types of materials. To circumvent this problem, another constraint is introduced to limit the search region, based on the upper bound of longitudinal wave speed as $\sqrt{E/\rho} = \sqrt{x_8/x_7} \le 10^{4.25}$ m/s.

#### 5.1.3 Simulation and results

Figure 16 shows an example of von Mises stress during the uniaxial compression of the architected metamaterial cell, as described in Section 5.1.2. In the simulation settings and its post-process, only $\sigma_{zz}$ is concerned.

The lower bounds of continuous variables ($x_5, x_6, x_7, x_8, x_9$) are ($2 \cdot 10^{-6}, 2 \cdot 10^{-6}, 0.4 \cdot 10^{+3}, 9 \cdot 10^{+9}, 0.16$). The lower bounds of $x_7, x_8$, and $x_9$ correspond to the density of wood, bulk Young's modulus of wood, and Poisson's ratio of silica glass, respectively. The upper bounds of continuous variables ($x_5, x_6, x_7, x_8, x_9$) are ($0.5 \cdot 10^{-3}, 0.5 \cdot 10^{-3}, 8.9 \cdot 10^{+3}, 650 \cdot 10^{+9}, 0.35$). The upper bounds of $x_7, x_8$, and $x_9$ correspond to the density of copper, bulk Young's modulus of tungsten carbide, and Poisson's ratio of a general alloy, respectively.

To initialize the optimization process, two random inputs are sampled to construct the GP model for each cluster. The number of clusters in this example is $2 \times 2 \times 2 \times 3 = 24$. The EI acquisition is used to locate the next sampling location $\boldsymbol{x}$. The CMA-ES (Hansen et al. 2003) is used as an auxiliary optimizer to maximize the penalized acquisition function. The optimization process is carried out for 170 iterations, as shown in Fig. 17. At iterations 0, 1, 2, 11, 14, 26, and 148, better objective function values of 1.9723, 2.7827, 10.4725, 12.1207, 22.1071, 23.3766, and 36.8316 $\cdot 10^6$ GPa/kg are identified, respectively. The relatively fast convergence plot demonstrates the effectiveness of the proposed BO method for the mix-integer optimization problems. Due to the expensive computational cost of the FEM simulation, the number of iterations is limited to 200.

### 5.2 Design optimization of fractal auxetic metamaterials

In the second example, we study the auxetic metamaterial with application in flexible and stretchable devices. Inspired by the experimental work of Cho et al. (2014) in designing auxetic metamaterials using fractal cut, and its subsequent numerical and experimental work by Tang and Yin (2017) in developing shape-programmable materials, we use auxetic metamaterials to demonstrate the proposed BO methodology. The goal of this example is to minimize the effective Poisson's ratio, which is negative and evaluated through a FEM simulation.

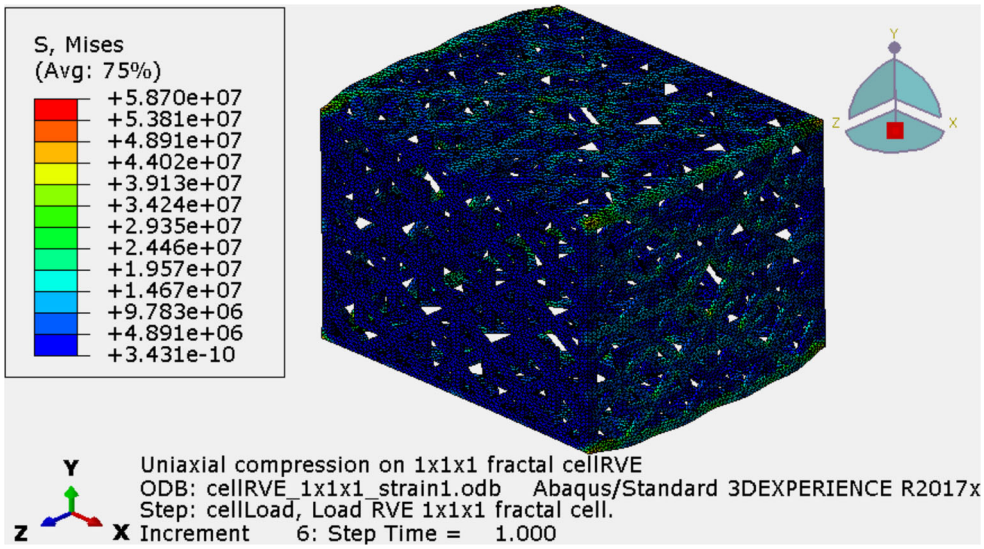



**Fig. 16** An example of von Mises stress of the structure under loading condition

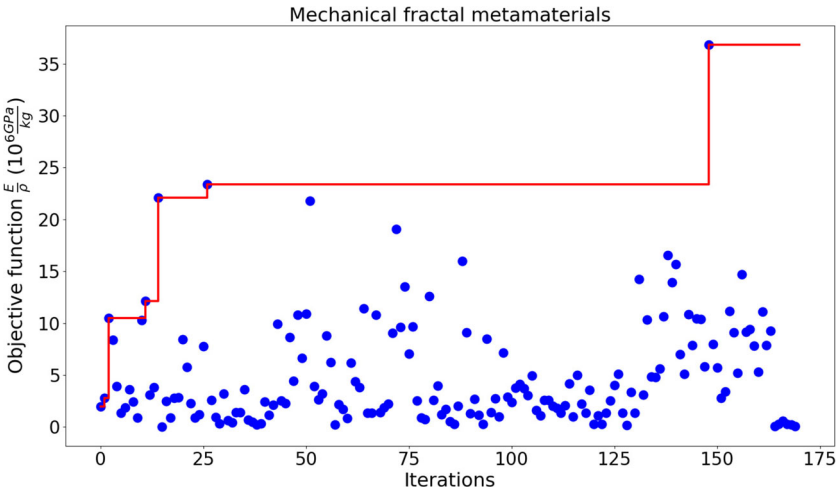



**Fig. 17** Convergence plot of the objective function, which is the ratio between the effective Young's modulus and the weight of the cell, i.e., $E_{\text{eff}}/m$

#### 5.2.1 Parametric design of auxetic metamaterials

Here, a parametric design of the unit cell, where the fractal level is fixed at 2, is devised. The cut motif $\alpha$ and $\beta$ for one level of the auxetic cell is shown in Fig. 18. Basically, this cut motif controls the free rotational hinges of the architected structure, such that the deformation energy dissipates through rotational motion, rather than translational motion. The principle of cut design is based on the connectivity of the rotating units, where the connectivity depends on the cut patterns, which in turn determines the maximum stretchability of the designed specimen. For further details about the fractal cut and its rotating mechanisms, readers are referred to the work of Cho et al. (2014) and Tang and Yin (2017). To create a fractal cut, a simple IFS is imposed on the cut to create subsequent level, with the scaling ratio of 1/2, and is then translated to four corners.

To tailor the negative Poisson's ratio, the shape of the cut is modeled as splines, where the coordinates of the control points are considered as inputs. The choice of $\alpha$ and $\beta$ cut is formulated using discrete variables. The dimension of this problem is 18, in which 2 discrete and 16 continuous variables are used. The parametric input $\boldsymbol{x}$ includes $x_1$ and $x_2$ as discrete variables, which takes the value of either 1 ($\alpha$-motif) or 2 ($\beta$-motif) for level 1 and level 2 cuts, respectively. The first 4 continuous variables $x_3$, $x_4$, $x_5$, and $x_6$ are used to describe the shape of the large center cut of level 1. The next 4 continuous variables $x_7$, $x_8$, $x_9$, and $x_{10}$ describe the shape of two small side cuts of level 1. In the same manner, the last 8 continuous variables are used to model the large center cut and two small side cuts of level 2. Figure 19 shows an example of the parametric design implementation of the designed auxetic metamaterials in the ABAQUS environment. The solid dots represent the control points of the cut. (Color is available on the electronic version. The blue solid dots denote the level 1 control points, whereas the red solid dots denote the level 2 control points.)

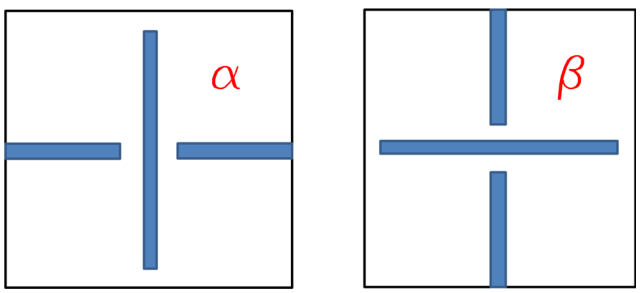



**Fig. 18** Cut motif $\alpha$ and $\beta$ in designing auxetic metamaterials by fractal cuts

#### 5.2.2 Constitutive material model and the finite element analysis

The study of Tang and Yin (2017) has demonstrated that the effective Poisson's ratio $\nu_{\text{eff}}$ is indeed a function of strain $\varepsilon$. In this work, we assume that the base material is natural rubber reinforced by carbon black. The Mooney-Rivlin constitutive model is used to describe the hyperelastic material behavior, where the suitable energy function $W$ is expressed as

$$W = C_{10}(\overline{I}_1 - 3) + C_{01}(\overline{I}_2 - 3) + \frac{1}{D_1}(J-1)^2, \qquad (38)$$

where $J$ is the elastic volume ratio and $I_1$, $I_2$, and $I_3$ are the three invariants of Green deformation tensor defined in term of principal stretch ratios $\lambda_1, \lambda_2, \lambda_3$, i.e.,

$$I_1 = \sum_{i=1}^{3} \lambda_i^2, \quad I_2 = \sum_{i,j=1;i\neq j} \lambda_i \lambda_j, \quad I_3 = \prod_{i=1}^{3} \lambda_i, \qquad (39)$$

and $\overline{I}_1 = I_1 J^{-2/3}$, $\overline{I}_2 = I_2 J^{-4/3}$. The materials parameter is adopted from Shahzad et al. (2015), where $C_{10} = 0.3339$ MPa, $C_{01} = -3.37 \cdot 10^{-}4$, and $D_1 = 1.5828 \cdot 10^{-3}$.

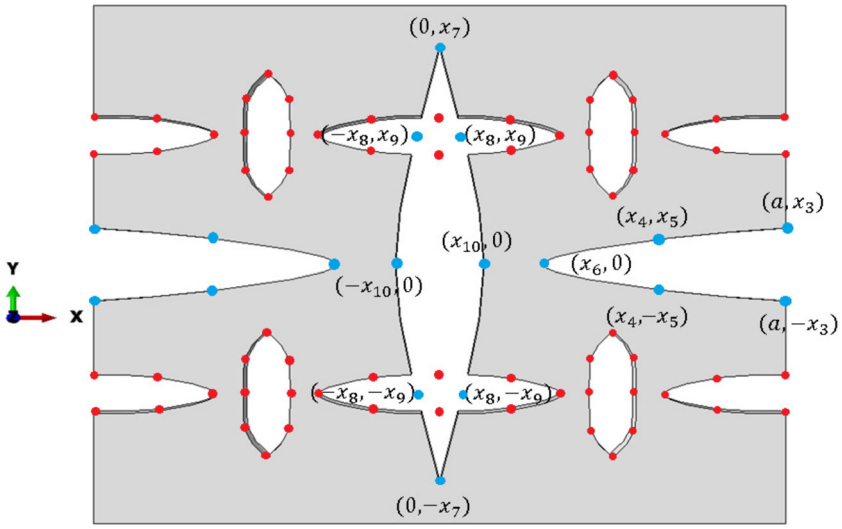



**Fig. 19** An implemented example of auxetic metamaterials by fractal cuts. The solid dots present the control points of the cut. (Color is available on the electronic version. Blue dots correspond to level 1, whereas red dots correspond to level 2)

The initial size of the square is 20 cm $\times$ 20 cm, and the thickness of the specimen is 1 mm. The specimen is then deformed in a uniaxial tension configuration in $y$-direction, where the displacement is fixed at 10 cm in one direction. The configuration for the simulation is plane-strain configuration, where displacement in the extrusion direction ($z$-direction) is fixed as zero.

In the deformed configuration, we extract the displacement in $x$-direction to infer the engineering transverse strain, and compute the effective Poisson's ratio as the ratio between transverse and longitudinal engineering strains.

The element used in this FEM simulation is the eight-node brick element (C3D8R, C3D6, and C3D4). The FEM is developed in the ABAQUS environment. The number of elements for each simulation is approximately 5000.

In this example, several constraints are imposed on the design variables, which are

$$x_5 \leq 0.010 - \underline{t}, \quad x_8 \leq x_4 - \underline{t}, \quad x_{16} \leq x_{12} - \underline{t} \tag{40a}$$

$$0 \leq x_6 \leq x_8, \quad 0 \leq x_7 \leq x_5, \quad x_4 \leq x_2 \leq 0.010, \quad 0 \leq x_3 \leq x_1 \tag{40b}$$

where $\underline{t} = 0.0015$m is the smallest thickness of the specimen. Two other constraints include the implementation of convexity for the large center cut of level 1 and level 2. Figure 20 presents an example of deformed configuration after the simulation converges.

### 5.2.3 Simulation and results

The lower bounds of the continuous variables are (0.25; 3.5; 0.50; 1.75; 8.0; 0.25; 4.0; 0.50; 0.25; 3.5; 0.50; 1.75; 4.0; 0.25; 3.0; 0.50)$\cdot 10^{-3}$. The upper bounds of the continuous variables are (2.00; 6.5; 1.75; 3.00; 9.5; 1.50; 8.0; 1.75; 2.00; 6.5; 1.75; 3.00; 5.5; 1.50; 4.0; 1.75)$\cdot 10^{-3}$.

Two random initial sampling points are created within each cluster. Because the fractal level is fixed at 2, where each fractal level corresponds to one cut motif $\alpha$ or $\beta$, 4 clusters are created during the initialization. The initial hyperparameters $\theta_i$ for all $i$ are set at 0.2. The lower and upper bounds for the hyperparameters $\theta_i$ for all $i$ are (0.01, 20).

The optimization process is carried out for 790 iterations. Figure 21 shows the convergence plot of the optimization process, where the best objective function value $\nu_{\text{eff}}$ is updated in iterations 0, 4, 24, 26, 30, 45, 63, 66, 69, 78, 81, 84, 513, 582, and 647, with the value of $-0.6603$, $-0.6605$, $-0.6628$, $-0.6628$, $-0.6902$, $-0.6941$, $-0.7143$, $-0.7410$, $-0.7517$, $-0.7576$, $-0.7627$, $-0.7784$, $-0.7785$, $-0.7802$, and $-0.7804$, respectively. The proposed BO shows relatively fast convergence for mid-level dimensionality $d = 16$, thus demonstrating the effectiveness in tackling mix-integer nonlinear optimization problems.

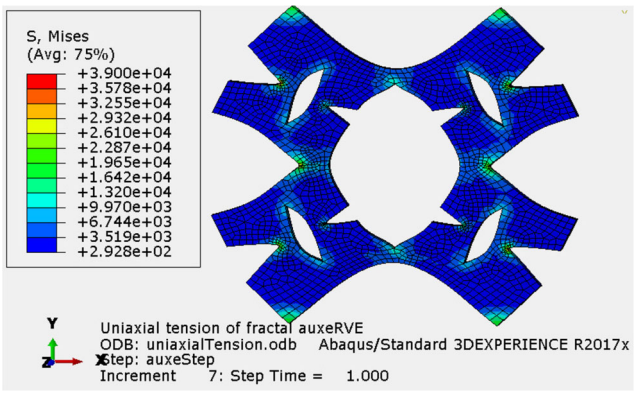



**Fig. 20** An example of uniaxial tension simulation of plane-strain configuration in designing auxetic metamaterials using fractal cut

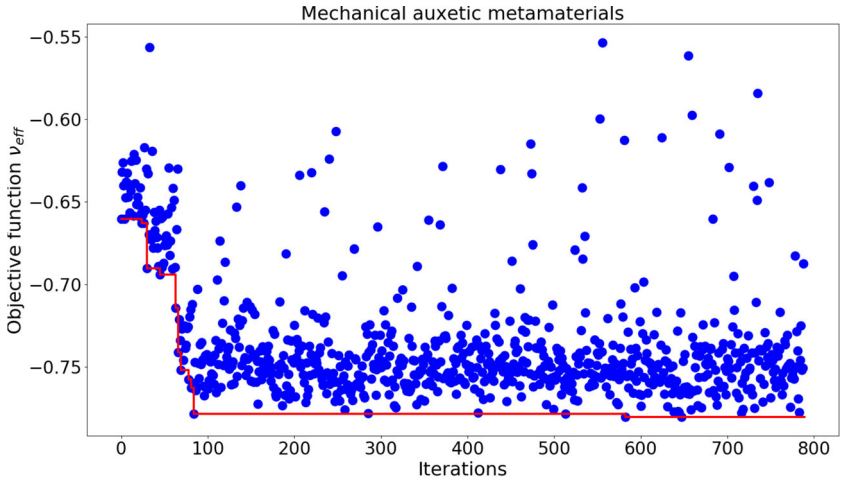



**Fig. 21** Convergence plot of the objective function, which is the effective Poisson's ratio $\nu_{\text{eff}}$. The best objective function value is updated at iterations 0, 4, 24, 26, 30, 45, 63, 66, 69, 78, 81, 84, 513, 582, 647, sequentially

## 6 Discussion

One of the advantages of the proposed BO algorithm is its extension to incorporate discrete variables for nonlinear mixed-integer optimization problems. The discrete variables include both categorical and integer variables, thus can be applied with or without the notion of order. The neighborhood of each cluster is built once during the initialization of the process and can be customized to adapt to specific user-defined requirements. Additionally, because the neighborhood can be modified and/or defined manually, the independence between clusters can be achieved by removing the corresponding clusters. Such independence is quite common in the case of categorical variables. However, the optimization performance of the proposed method does not depend on the enumeration of the clusters. We emphasize that if the cluster is ceased to exist, then it can be manually removed, and the cluster indices can be reenumerated manually by a slight modification of (12) and Algorithm 1.

The weight computation scheme is devised in such a way that asymptotically, the weight prediction converges to a single GP prediction, by imposing a weight vector which has 0 everywhere, except for a single 1 that corresponds to the corresponding cluster.

It is recommended to choose the neighbors carefully. One way to do so is to set a small threshold discrete distance $d_{\text{th}}$, which measures the dissimilarity between clusters based on the discrete tuples, e.g., $d_{\text{th}} \leq 1$, and manually remove clusters that are known to be independent beforehand at the end of initialization. The safest setting is $d_{\text{th}} = 0$, which assumes clusters are completely independent of each other. This setting has some negative effect on the convergence rate, but would eventually reach the global optimal solution, and would not be trapped at local optima.

The initial sample size plays a role in the performance of the proposed mixed-integer BO method. It has been shown that for some low-dimensional problems, the initial sample size does not affect the optimization performance. However, for high-dimensional problems, the initial sample size does impact the optimization performance. Too many initial samples at the beginning would prevent the optimization from quick convergence. However, with moderate amount of initial samples, and thus a more accurate local GP, the mixed-integer BO converges faster, compared with fewer initial samples. As a general rule of thumb, the total initial sample size is recommended at between $5d$ and $10d$, where $d$ is the dimension of the problem, including both discrete and continuous variables.

Here, the scalability of GP for high-dimensional problems is alleviated, but not completely eliminated. It is noted that the decomposition approach and weighted average approach have been adopted (Nguyen-Tuong and Peters 2008; Nguyen-Tuong et al. 2009a, b, 2010; van Stein et al. 2015; Tran et al. 2018) for continuous variables. The decomposition method for continuous variables is typically referred to as local GP. This approach is promising in tackling the scalability problem. Particularly, in one of our previous studies (Tran et al. 2018), we have shown that the local GP is computationally one-order cheaper, compared to the classical GP, while maintaining a reasonable approximation error.

Nevertheless, further research is required to develop an efficient and robust decomposition scheme for both discrete and continuous variables.

One of the limitations in the proposed approach is the scalability with respect to discrete variables. Because of the decomposition scheme, the number of the clusters is the number of the combinatorial possibilities, i.e., the product of the number of choices for each discrete variable, and thus resulting in the sparsity problem in each cluster. To mitigate the undesirable sparsity effect, a Gaussian mixture model that combines all the predictions from neighboring clusters is used to exploit some useful information from the

neighborhood. As mentioned previously, the mixed-integer optimization problem, in general, is difficult, because it combines the difficulties for both discrete and continuous optimization. Particularly, some discrete and combinatorial optimization problems are NP-complete, such as the traveling salesman problem, knapsack problem, and graph coloring problem, to name a few. Another extension is to model the weights as stochastic variables, so that the metaheuristic methodologies can be applied (Bianchi et al. 2009).

The clustering and enumeration algorithm described in Algorithm 1 is based on the assumption of the independence of discrete variables. Algorithm 1 does not work if the discrete variables are dependent. However, in the case that discrete variables are dependent on each other, manual neighborhood definition of clusters can be introduced manually, and the proposed BO algorithm is functional with the demonstrated efficiency. However, the users must declare the neighborhood of each cluster manually. Strictly speaking, the computational efficiency of the proposed algorithm only depends on the number of clusters, not on the number of discrete variables. If all discrete variables are completely independent of each other, as demonstrated in the above examples, then the number of clusters is equal to the product of the number of choices for each discrete variable, i.e., $L = \prod p_i$.

Another practical limitation for the proposed BO algorithm for engineering models and simulations is its sequential nature of sampling and search. Each run of simulations usually demands a considerable amount of computational time. In practice, for high-fidelity and dedicated simulations, one should resort to multi-fidelity or batch-parallel BO for further improvement.

## 7 Conclusion and future work

In this paper, we propose a new BO algorithm to solve the nonlinear constrained mixed-integer design optimization problems. In this algorithm, the large dataset is decomposed according to the discrete tuples, in which each discrete tuple corresponds to a unique GP model. The prediction for mean and variance is formulated as a Gaussian mixture model, in which the weights are computed based on the pairwise Wasserstein distance between clusters. Constraints, which are formulated as a set of inequalities, are included during the optimization process. Theoretical bounds and algorithmic complexity are provided to demonstrate the computational efficiency compared to the classical GP.

The proposed algorithm is demonstrated with two fractal metamaterial design examples, where the mechanical properties are tailored by the hierarchically designed architect. In the first example, the algorithm is used to search for the fractal metamaterial with high-strength and low-density properties, where material selection is considered. In the second example, the algorithm is utilized to design an auxetic metamaterial for flexible and stretchable devices, where the effective Poisson's ratio is chosen as the objective function. For both computational material design examples, constraints are imposed to limit the design space. The proposed algorithm shows a promising performance in solving engineering problems, where high dimensionality is often an issue.

While several limitations exist, such as scalability for discrete and continuous variables, further research extensions can be made to improve the current methodology, including metaheuristic methodologies for stochastic combinatorial optimization.

**Acknowledgments** Authors thank Prof. Hongyuan Zha (Georgia Tech) for numerous helpful conversations about Bayesian optimization. The authors are grateful to two anonymous reviewers for their constructive feedback.

**Funding information** The research was supported in part by the National Science Foundation under grant number CMMI-1306996. Also, this research was supported in part through research cyberinfrastructure resources and services provided by the Partnership for an Advanced Computing Environment (PACE) at the Georgia Institute of Technology, Atlanta, GA, USA.



## References

Barnsley MF (2014) Fractals everywhere. Academic Press, New York

Bianchi L, Dorigo M, Gambardella LM, Gutjahr WJ (2009) A survey on metaheuristics for stochastic combinatorial optimization. Nat Comput 8(2):239–287

Bower AF (2011) Applied mechanics of solids. CRC Press, Boca Raton

Cagnina LC, Esquivel SC, Coello CAC (2008) Solving engineering optimization problems with the simple constrained particle swarm optimizer. Informatica 32(3):319–326

Cho Y, Shin JH, Costa A, Kim TA, Kunin V, Li J, Lee S, Yang S, Han HN, Choi IS et al (2014) Engineering the shape and structure of materials by fractal cut. Proc Natl Acad Sci 111(49):17390–17395

Datta D, Figueira JR (2011) A real-integer-discrete-coded particle swarm optimization for design problems. Appl Soft Comput 11(4):3625–3633

Davis E, Ierapetritou M (2009) A kriging based method for the solution of mixed-integer nonlinear programs containing black-box functions. J Glob Optim 43(2-3):191–205

Deb K, Goyal M (1996) A combined genetic adaptive search (geneAS) for engineering design. Comput Sci Inf 26:30–45

Digabel SL, Wild SM (2015) A taxonomy of constraints in simulation-based optimization. arXiv:1505.07881

Gandomi AH, Yang XS (2011) Benchmark problems in structural optimization. In: Computational optimization, methods and algorithms. Springer, pp 259–281

Gardner JR, Kusner MJ, Xu ZE, Weinberger KQ, Cunningham JP (2014) Bayesian optimization with inequality constraints. In: ICML, pp 937–945

Gelbart M, Snoek J, Adams R (2014) Bayesian optimization with unknown constraints. arXiv:1403.5607
Givens CR, Shortt RM et al (1984) A class of Wasserstein metrics for probability distributions. Mich Math J 31(2):231–240
Gramacy RB, Lee H (2008a) Gaussian processes and limiting linear models. Comput Stat Data Anal 53(1):123–136
Gramacy RB, Lee HKH (2008b) Bayesian treed Gaussian process models with an application to computer modeling. J Am Stat Assoc 103(483):1119–1130
Gramacy RB, Taddy M et al (2010) Categorical inputs, sensitivity analysis, optimization and importance tempering with tgp version 2, an R package for treed Gaussian process models. J Stat Softw 33(6):1–48
Hansen N, Müller SD, Koumoutsakos P (2003) Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES). Evol Comput 11(1):1–18
Hemker T, Fowler KR, Farthing MW, von Stryk O (2008) A mixed-integer simulation-based optimization approach with surrogate functions in water resources management. Optim Eng 9(4):341–360
Hernández-Lobato JM, Gelbart M, Hoffman M, Adams R, Ghahramani Z (2015) Predictive entropy search for Bayesian optimization with unknown constraints. In: International conference on machine learning, pp 1699–1707
Hernández-Lobato JM, Gelbart M, Adams R, Hoffman MW, Ghahramani Z (2016) A general framework for constrained Bayesian optimization using information-based search. Journal of Machine Learning Research
Huang D, Allen TT, Notz WI, Zeng N (2006) Global optimization of stochastic black-box systems via sequential kriging meta-models. J Glob Optim 34(3):441–466
Jang HL, Cho H, Choi KK, Cho S (2014) Reliability-based design optimization of fluid–solid interaction problems. Proc Inst Mech Eng Part C: J Mech Eng Sci 228(10):1724–1742
Kim K, Lee M, Lee S, Jang G (2017a) Optimal design and experimental verification of fluid dynamic bearings with high load capacity applied to an integrated motor propulsor in unmanned underwater vehicles. Tribol Int 114:221–233
Kim Y, Lee S, Yee K, Rhee DH (2017b) High-to-low initial sample ratio of hierarchical kriging for film hole array optimization. Journal of Propulsion and Power
Li M, Li G, Azarm S (2008) A kriging metamodel assisted multi-objective genetic algorithm for design optimization. J Mech Des 130(3):031401
Li X, Gong C, Gu L, Jing Z, Fang H, Gao R (2018) A reliability-based optimization method using sequential surrogate model and Monte Carlo simulation. Struct Multidiscip Optim:1–22
Lin Y, Zhang HH (2006a) Component selection and smoothing in multivariate nonparametric regression. Ann Stat 34(5):2272–2297
Lin Y, Zhang HH (2006b) Component selection and smoothing in smoothing spline analysis of variance models. Ann Stat 34(5):2272–2297
Liu J, Song WP, Han ZH, Zhang Y (2017) Efficient aerodynamic shape optimization of transonic wings using a parallel infilling strategy and surrogate models. Struct Multidiscip Optim 55(3):925–943
Martins JR, Lambe AB (2013) Multidisciplinary design optimization: a survey of architectures. AIAA J 51(9):2049–2075
Meza LR, Das S, Greer JR (2014) Strong, lightweight, and recoverable three-dimensional ceramic nanolattices. Science 345(6202):1322–1326
Mockus J (1975) On Bayesian methods for seeking the extremum. In: Optimization techniques IFIP technical conference. Springer, pp 400–404
Mockus JB, Mockus LJ (1991) Bayesian approach to global optimization and application to multiobjective and constrained problems. J Optim Theory Appl 70(1):157–172
Müller J, Shoemaker CA, Piché R (2013) SO-MI: a surrogate model algorithm for computationally expensive nonlinear mixed-integer black-box global optimization problems. Comput Oper Res 40(5):1383–1400
Müller J (2016) MISO: mixed-integer surrogate optimization framework. Optim Eng 17(1):177–203
Müller J, Shoemaker CA, Piché R (2014) SO-I: a surrogate model algorithm for expensive nonlinear integer programming problems including global optimization applications. J Glob Optim 59(4):865–889
Nguyen-Tuong D, Peters J (2008) Local Gaussian process regression for real-time model-based robot control. In: 2008. IROS 2008. IEEE/RSJ international conference on intelligent robots and systems. IEEE, pp 380–385
Nguyen-Tuong D, Peters J, Seeger M (2009a) Local Gaussian process regression for real time online model learning. In: Advances in neural information processing systems, pp. 1193–1200
Nguyen-Tuong D, Seeger M, Peters J (2009b) Model learning with local Gaussian process regression. Adv Robot 23(15):2015–2034
Nguyen-Tuong D, Seeger M, Peters J (2010) Real-time local Gaussian process model learning. In: From motor learning to interaction learning in robots. Springer, Berlin, pp 193–207
Nielsen HB, Lophaven SN, Søndergaard J (2002) DACE, a MATLAB kriging toolbox, vol 2. Citeseer, Princeton
Qian PZG, Wu H, Wu CJ (2008) Gaussian process models for computer experiments with qualitative and quantitative factors. Technometrics 50(3):383–396
Queipo NV, Haftka RT, Shyy W, Goel T, Vaidyanathan R, Tucker PK (2005) Surrogate-based analysis and optimization. Prog Aerosp Sci 41(1):1–28
Rao SS (2009) Engineering optimization: theory and practice. Wiley, New York
Ravindran A, Reklaitis GV, Ragsdell KM (2006) Engineering optimization: methods and applications. Wiley, New York
Rehman SU, Langelaar M (2017) Expected improvement based infill sampling for global robust optimization of constrained problems. Optim Eng 18(3):723–753
Shahriari B, Swersky K, Wang Z, Adams R, de Freitas N (2016) Taking the human out of the loop: a review of Bayesian optimization. Proc IEEE 104(1):148–175
Shahzad M, Kamran A, Siddiqui MZ, Farhan M (2015) Mechanical characterization and FE modelling of a hyperelastic material. Mater Res 18(5):918–924
Simpson TW, Mauery TM, Korte JJ, Mistree F (2001) Kriging models for global approximation in simulation-based multidisciplinary design optimization. AIAA J 39(12):2233–2241
Sóbester A., Forrester AI, Toal DJ, Tresidder E, Tucker S (2014) Engineering design applications of surrogate-assisted optimization techniques. Optim Eng 15(1):243–265
Song C, Song W, Yang X (2017) Gradient-enhanced hierarchical kriging model for aerodynamic design optimization. J Aerosp Eng 30(6):04017072
Srinivas N, Krause A, Kakade SM, Seeger M (2009) Gaussian process optimization in the bandit setting: no regret and experimental design. arXiv:0912.3995
Srinivas N, Krause A, Kakade SM, Seeger MW (2012) Information-theoretic regret bounds for Gaussian process optimization in the bandit setting. IEEE Trans Inf Theory 58(5):3250–3265
van Stein B, Wang H, Kowalczyk W, Bäck T, Emmerich M (2015) Optimally weighted cluster kriging for big data regression. In:

International symposium on intelligent data analysis. Springer, pp 310–321

Storlie C, Bondell HD, Reich BJ, Zhang HH (2011) Surface estimation, variable selection, and the nonparametric oracle property. Stat Sin 21(2):679

Swiler LP, Hough PD, Qian P, Xu X, Storlie C, Lee H (2014) Surrogate models for mixed discrete-continuous variables. In: Constraint programming and decision making. Springer, pp 181–202

Tang Y, Yin J (2017) Design of cut unit geometry in hierarchical kirigami-based auxetic metamaterials for high stretchability and compressibility. Extreme Mech Lett 12:77–85

Tran A, He L, Wang Y (2018) An efficient first-principles saddle point searching method based on distributed kriging metamodels. ASCE-ASME J Risk Uncertain Eng Syst Part B: Mech Eng 4(1):011006

Viana FA, Simpson TW, Balabanov V, Toropov V (2014) Special section on multidisciplinary design optimization: metamodeling in multidisciplinary design optimization: How far have we really come? AIAA J 52(4):670–690

Zhang Y, Hu S, Wu J, Zhang Y, Chen L (2014) Multi-objective optimization of double suction centrifugal pump using kriging metamodels. Adv Eng Softw 74:16–26

Zhou Q, Qian PZ, Zhou S (2011) A simple approach to emulation for computer models with qualitative and quantitative factors. Technometrics 53(3):266–273

Zhou Q, Wang Y, Choi SK, Jiang P, Shao X, Hu J (2017) A sequential multi-fidelity metamodeling approach for data regression. Knowledge-Based Systems

Zhou Q, Wang Y, Choi S-K, Jiang P, Shao X, Hu J, Shu L (2018) A robust optimization approach based on multi-fidelity metamodel. Struct Multidiscip Optim 57(2):1–23