ASME Journal of Mechanical Design
Online journal at:
https://asmedigitalcollection.asme.org/mechanicaldesign

A Cost-Aware Multi-Agent System for Black-Box Design Space Exploration

Siyu Chen
Walker Department of Mechanical Engineering,
University of Texas at Austin,
204 E Dean Keeton Street,
Austin, TX 78712
e-mail: siyu.chen@utexas.edu

Alparslan Emrah Bayrak
Department of Mechanical Engineering and Mechanics,
Lehigh University,
19 Memorial Dr W,
Bethlehem, PA 18015
e-mail: bayrak@lehigh.edu

Zhenghui Sha[1]
Walker Department of Mechanical Engineering,
University of Texas at Austin,
204 E Dean Keeton Street,
Austin, TX 78712
e-mail: zsha@austin.utexas.edu

Effective coordination of design teams must account for the influence of costs incurred while searching for the best design solutions. This article introduces a cost-aware multi-agent system (MAS), a theoretical model to (1) explain how individuals in a team should search, assuming that they are all rational utility-maximizing decision-makers and (2) study the impact of cost on the search performance of both individual agents and the system. First, we develop a new multi-agent Bayesian optimization framework accounting for information exchange among agents to support their decisions on where to sample in search. Second, we employ a reinforcement learning approach based on the multi-agent deep deterministic policy gradient for training MAS to identify where agents cannot sample due to design constraints. Third, we propose a new cost-aware stopping criterion for each agent to determine when costs outweigh potential gains in search as a criterion to stop. Our results indicate that cost has a more significant impact on MAS communication in complex design problems than in simple ones. For example, when searching in complex design spaces, some agents could initially have low-performance gains, thus stopping prematurely due to negative payoffs, even if those agents could perform better in the later stage of the search. Therefore, global-local communication becomes more critical in such situations for the entire system to converge. The proposed model can serve as a benchmark for empirical studies to quantitatively gauge how humans would rationally make design decisions in a team. [DOI: 10.1115/1.4065914]

Keywords: design space exploration, multi-agent Bayesian optimization (MABO), multi-agent reinforcement learning (MARL), black-box optimization, collaborative design

1 Introduction

Design space exploration (DSE) involves systematically searching and examining various design alternatives that meet specifications and constraints to find the best design [1]. This contrasts with gradient-based optimization algorithms, which require predefined objectives and utilize derivatives to guide their search. Unlike methods such as gradient descent [2], stochastic gradient descent [3], and Adam (Adaptive Moment Estimation) [4], which operate based on gradient information to find optimal solutions, DSE adopts a more sequential and heuristic approach. It incrementally develops knowledge of the design space by sequentially evaluating design alternatives due to the black-box nature of the design problem. This happens because, in many cases, the objective function is not clearly defined or too complex for an analytical description. Black-box optimization addresses this by sequentially and iteratively testing and learning about the design space through targeted sampling, even when the underlying functional relationships are not fully known [5]. Through this method, DSE aims to uncover the best possible solutions, navigating within defined constraints and specifications to find those that are optimal or as close to optimal as possible.

DSE for black-box problems exhibits three main characteristics. First, complex design problems often require a team of designers or computer agents to work together to solve them. The effectiveness and efficiency of a design team are related to the efforts of every team member and the coordination between them. This is particularly true when navigating complex design spaces with many local optima. Second, the unknown design space is often accompanied by unknown constraints, such as technical and environmental limits or specific project requirements, so that viable design alternatives are limited [6,7]. DSE for such problems requires agents to adapt and learn about those unknown constraints dynamically. Third, every sampling decision made in DSE comes with an intricate balance between cost management and team collaboration. Each sampling decision incurs specific costs, be it time or resources, necessitating efficient resource management to stay within budgetary limits, as highlighted by Panchal et al. [8]. The dynamic between cost and collaboration significantly influences the decision-making process in DSE. In previous studies [9–11], they indicate that budget constraints significantly affect group performance. Therefore,

[1]Corresponding author.

Contributed by the Design Automation Committee of ASME for publication in the Journal of Mechanical Design. Manuscript received December 18, 2023; final manuscript received July 1, 2024; published online August 21, 2024. Assoc. Editor: Mark Fuge.

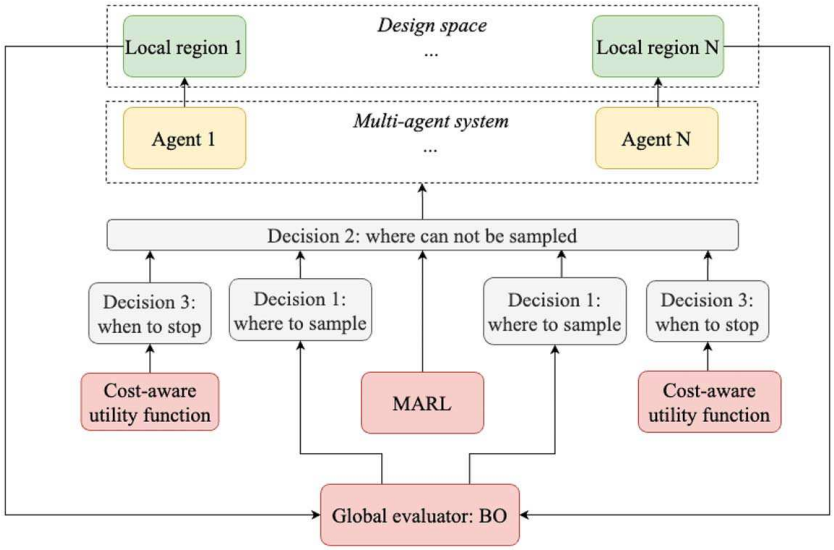


Fig. 1 Overview of the proposed multi-agent system for design space exploration

effective teamwork, in which members share information and resources, is crucial to optimize resource use and ensure that DSE meets budget constraints.

In this article, we quantitatively investigate the impact of cost on design decisions in teams for problems with unknown objective and constraints. We present a novel and unified framework that incorporates the three characteristics mathematically. Assuming idealized decision-making behaviors, we use this framework to define how design teams should work under different cost scenarios. This framework consists of a cost-aware multi-agent system (MAS) based on Bayesian optimization (BO) and reinforcement learning to model the sequential decision-making process of a rational design team in the exploration of complex design spaces with unknown constraints. Figure 1 shows an overview of this MAS. In particular, our study answers the following research question: *What impact would the cost-aware stopping criteria have on collaboration between multiple agents in design space exploration?*

The core contributions of this article revolve around three decision-making principles in a cost-aware MAS strategy to find the global optimum.

(1) *Where to sample*: Each agent in the MAS decides where to sample based on a multi-agent Bayesian optimization [12], a strategy to identify optimal solutions within complex design spaces by enabling information exchange between agents, which enhances the collective intelligence and effectiveness of an MAS. (2) *Where not to sample*: We address the issue of unknown constraints using MARL formulated based on multi-agent deep deterministic policy gradient (MADDPG) [13]. This approach enables agents to recognize and adapt to constraints autonomously. If the suggested points of the multi-agent Bayesian optimization (MABO) fall within those recognized constraints, agents sample around the infeasible regions. (3) *When to stop*: We develop a cost-aware stopping criterion for each agent based on two elements: information gain (IG) and performance gain (PG). IG is the potential improvement that the agent could get in the future, while PG is the performance that the agent has already received. By adjusting the parameters of IG and PG, the agent can balance between what it has already gained (PG) and potential new improvements (IG). Each agent is equipped with knowledge of its own stopping criterion based on the gains it has achieved in relation to the cost it has incurred.

The remainder of the article is structured as follows. Section 2 presents an overview of existing computational approaches in DSE and presents the contribution. Section 3 presents the technical background on BO and multi-agent reinforcement learning as preliminaries for the proposed model. Next, Sec. 4 delves into the technical details of the main proposed framework, presenting the problem formulation and how the MAS navigates in the design space. Section 5 presents the experimental settings and results under different design scenarios with varying complexity. Section 6 provides a discussion of the research findings and draws insight into the impact of cost on collaboration in design teams. Finally, Sec. 7 concludes the article with a summary of the findings and limitations that lead to the future work.

2 Bayesian Models of Design Decision-Making

DSE often presents itself as a black-box optimization challenge, particularly when the configuration space or the form of the function is unknown or not well defined. In such scenarios, the process of sampling subsequent design candidates for evaluation becomes a crucial decision-making task that is highly dependent on experiences. BO employs knowledge-based reasoning to navigate these unknown design spaces, using insights obtained from historical data and experiences [14], which offers a systematic approach to providing informed design recommendations that guide sampling decisions. In what follows, we discuss the three challenges that this study aims to address within the BO literature.

2.1 Team-Based Decisions. BO is a commonly employed methodology to model design search in black-box problems. Conventionally, BO treats each design experiment sequentially, with a new one proposed only after completing the previous one [6]. However, this method can be time consuming when dealing with complex design spaces due to the step-by-step approach to finding the optimum. Advancements in computational and communication technologies, as detailed by Kontar et al. [15], have made it possible for MAS as a design team to handle complex DSE based on BO. In another study, Peralta et al. [16] develop an MABO for multi-objective optimization, with the aim of enhancing the availability and affordability of water quality monitoring. Since solving such complex problems requires teams in practice, we introduce the MABO framework as a model of teamwork in our recent study [12]. This MABO framework can significantly improve convergence through the global-local communication strategy, enabling faster identification of optimal solutions in complex design spaces.

2.2 Design Constraints. In many practical design problems, certain regions of the space are infeasible due to design constraints

[7]. If the constraints are predefined, they can be integrated into the acquisition function (AF) to be maximized in the BO process. However, the cases where constraints are not previously predicted pose a greater challenge. In response to this challenge, many constraint-handling techniques have been developed. The augmented Lagrangian relaxation method is one such technique that integrates constraints into a Lagrangian function to be optimized, making them amenable to BO. Although originally developed for gradient-based optimization, it has been used in black-box optimization without requiring an explicit formulation of the constraints [17]. However, this approach involves nonstationary surrogate models that lead to modeling complexities, as highlighted in Ref. [18]. The second technique integrates AFs with a probability of feasibility such as constrained expected improvement (cEI) or constraint-weighted expected improvement (EI) [7,19]. However, this method requires the best current observation, which poses difficulties for noisy experiments. Letham et al. [20] addressed this by extending cEI to noisy observations, although it remains sensitive to highly constrained problems. Bernardo et al. [21] proposed an integrated expected conditional improvement AF, which defines an expected reduction in EI with limited satisfaction probability to allow infeasible regions to provide information. Despite existing methodologies that reduce the likelihood of sampling within infeasible regions, a significant degree of uncertainty persists. In this study, we present an approach that integrates MARL into the proposed MABO framework to prevent the agent from sampling in infeasible regions. In this approach, while direct sampling is prohibited in infeasible areas, agents still obtain information at the place that is closest to the infeasible regions as their best attempts of samples.

2.3 Cost of Design Search. Most BO processes work for a predefined number of iterations. This approach, while beneficial for comparative studies, does not realistically reflect the constraints in team decision-making scenarios, where the number of design evaluations is often bound by budget constraints [22,23]. In traditional BO, the next sample is chosen solely based on the maximization of an AF, such as EI [24,25], without considering the budget. However, a more practical alternative, which aligns with the purpose of the present article, is to incorporate the cost of sampling into the decision-making process [26]. By selecting the option with the highest net value, calculated as the maximum EI minus the cost of further evaluation, we address the practical challenge of limited resources. This strategy underscores the necessity of including sampling costs in the stopping criteria for each agent in BO. In this article, we develop a cost-aware stopping criterion aligned with the design team's decision-making process.

In this article, a cost-aware stopping criterion is represented by a net value or utility score. This score is calculated as the cumulative gain minus the total cost incurred so far. Therefore, evaluating the cumulative gain is essential for the decision-making process. In most BO studies, the decision on where to sample next is usually based on an AF, as mentioned previously. The rationale behind this approach is that the AF determines the potential net value or information gain that can be achieved in the subsequent step. However, focusing solely on AF overlooks a crucial aspect of the decision-making process: the actual performance achieved by the agent in previous iterations. This performance is often quantified as "regret," a metric used to determine when to terminate the BO process. A study by Lorenz et al. [27] suggests an approach based on the Euclidean distance (ED) as their regret. They recommend terminating the BO algorithm when the ED between the point of the most recent observation and the forthcoming observation falls below a certain threshold. A notable method proposed by Ref. McLeod et al. [28] involves the concept of regret, splitting it into local and global components as the stopping criterion for BO. However, it is plausible that even though regret falls below a certain threshold, indicating the performance gain in the system, the value of the AF, as a measure of potential gain, might still be large. In [29], it was found that when human designers make the decision to stop evaluating, they tend to look a few steps further after achieving the best design to reduce uncertainty in the process. In this study, we propose a similar cost-aware stopping criterion based on a combination of both actual performance gains and information gains (value of the AF).

3 Preliminaries

3.1 Bayesian Optimization. Commonly used as a global optimization method, BO searches for the optimum of a black-box objective function $f(\mathbf{x})$, where $\mathbf{x} \in A$; A is a domain of d dimensional design space, where $A \subseteq \mathbb{R}^d$. BO relies on two main components: (1) a statistical inference method, typically a Gaussian process (GP) regression, to model the unknown objective function value based on the collected data, and (2) an acquisition function (AF) to determine where to sample within the design space [30].

3.1.1 Gaussian Process. Gaussian process is a commonly used statistical inference model, which defines a distribution over possible unknown functions [31]. BO realizes the reasoning about $f(\mathbf{x})$ by choosing an appropriate Gaussian process prior:

$$\mathbf{f}(\mathbf{x}_{1:k}) \sim \mathcal{GP}\big(\boldsymbol{\mu}_0(\mathbf{x}_{1:k}), \boldsymbol{\Sigma}_0(\mathbf{x}_{1:k}, \mathbf{x}_{1:k})\big) \tag{1}$$

where the set of observations is $\mathcal{D} = (\mathbf{x}_{1:k}, \mathbf{f}(\mathbf{x}_{1:k}))$, $\mathbf{x}_{1:k} = [\mathbf{x}_1, \ldots, \mathbf{x}_k]$, $\mathbf{f}(\mathbf{x}_{1:k}) = [f(\mathbf{x}_1), \ldots, f(\mathbf{x}_k)]$, $\boldsymbol{\mu}_0(\mathbf{x}_{1:k}) = \big[\mu_0(\mathbf{x}_1), \ldots, \mu_0(\mathbf{x}_k)\big]$ is the mean vector by evaluating a mean function μ_0 at each $\mathbf{x}_1, \ldots, \mathbf{x}_k$, and $\boldsymbol{\Sigma}_0(\mathbf{x}_{1:k}, \mathbf{x}_{1:k}) = [\Sigma_0(\mathbf{x}_1, \mathbf{x}_1), \ldots, \Sigma_0(\mathbf{x}_1, \mathbf{x}_k); \ldots; \Sigma_0(\mathbf{x}_k, \mathbf{x}_1), \ldots, \Sigma_0(\mathbf{x}_k, \mathbf{x}_k)]$ is constructed by covariance $\Sigma_0(\cdot, \cdot)$ between each observation. Given the observation data $\mathcal{D}$, the posterior probability distribution is defined as follows [30]:

$$\begin{aligned} f(\mathbf{x}) \mid \mathbf{f}(\mathbf{x}_{1:k}) &\sim \mathcal{GP}\big(\mu(\mathbf{x}), \sigma^2(\mathbf{x})\big) \\ \mu(\mathbf{x}) &= \boldsymbol{\Sigma}_0(\mathbf{x}, \mathbf{x}_{1:k}) \boldsymbol{\Sigma}_0(\mathbf{x}_{1:k}, \mathbf{x}_{1:k})^{-1} \big(\mathbf{f}(\mathbf{x}_{1:k}) - \boldsymbol{\mu}_0(\mathbf{x}_{1:k})\big) + \mu_0(\mathbf{x}) \\ \sigma^2(\mathbf{x}) &= \Sigma_0(\mathbf{x}, \mathbf{x}) - \boldsymbol{\Sigma}_0(\mathbf{x}, \mathbf{x}_{1:k}) \boldsymbol{\Sigma}_0(\mathbf{x}_{1:k}, \mathbf{x}_{1:k})^{-1} \boldsymbol{\Sigma}_0(\mathbf{x}_{1:k}, \mathbf{x}) \end{aligned} \tag{2}$$

where $\mu(\mathbf{x})$ denotes the posterior mean and $\sigma^2(\mathbf{x})$ denotes the posterior variance.

3.1.2 Acquisition Function. AF is used to identify the next point to sample within the design space. This function utilizes the probabilistic surrogate model, which is described in Sec. 3.1.1 and approximates the objective function. When choosing the next observation, an AF is optimized [30]. AFs are developed to balance the exploration of new areas in the design space and the exploitation of areas already known to provide high-value results.

Chaudhari et al. [32] categorize AFs primarily into two groups: expected utility (EU) based and heuristic based. A widely used EU-based model is EI, while lower confidence bound (LCB) [33] is representative of a heuristic model. In Ref. [32], they show that the heuristic model provides the best descriptive model of the sequential information acquisition process. Furthermore, it is possible to create distinct sampling strategies for different agents by adjusting the LCB parameter, which is instrumental in managing the balance between exploitation and exploration. Thus, we adopted the LCB as a typical acquisition function of the heuristic model in this study. Its formulation is given as follows:

$$a_{\mathrm{LCB}}(\mathbf{x}) = \mu(\mathbf{x}) - \lambda\sigma(\mathbf{x}) \tag{3}$$

where μ represents the mean function of the posterior probability distribution for f and σ is its standard deviation. In addition, $\lambda > 0$ is a parameter that determines the trade-off between exploitation and exploration.

With AF a_{LCB} presented in Eq. (3), the next sampling point $\mathbf{x}$ is chosen as the one that minimizes this LCB. The main idea behind LCB is to find a balance between evaluating points where the function value (mean) is expected to be low (exploitation) and evaluating points where the uncertainty about the function value is high

(exploration). By subtracting a scaled version of the uncertainty from the mean, LCB encourages the algorithm to explore regions of the search space where there is a lot of uncertainty, but it also tends to exploit regions with low expected function values.

3.2 Multi-Agent Reinforcement Learning. In this article, we apply an MARL based on a MADDPG [13] to implement a constraint handling mechanism for MAS when sampling in a constrained design space autonomously. By this model, agents can find the shortest path to targets suggested by the global evaluator and avoid sampling in infeasible regions.

The MADDPG framework for training the MAS is shown in Fig. 2. The core idea of this framework is centralized training and decentralized execution. Within this framework, each agent in MAS has a specific actor-critic network, which is trained through the deep deterministic policy gradient algorithm [34]. During the training process, a centralized critic network for each agent Q_i is updated using shared observations o_i and actions a_i of all agents in the MAS. This network evaluates the efficacy of the actions a_i proposed by the actor network to optimize the policy π_i using a policy gradient methodology. In the execution process, each agent relies solely on its actor network to offer a deterministic policy π_i. This policy guides the updates of actions a_i based on environmental observations o_i.

Assume that N agents are set for exploration in the design space, $\boldsymbol{\theta} = [\theta_1, \theta_2, \ldots, \theta_N]$ are the parameters for deterministic policies for N agents $\boldsymbol{\mu} = [\mu_{\theta_1}, \mu_{\theta_2}, \ldots, \mu_{\theta_N}]$, then the policy gradient for agent i can be given as follows:

$$\nabla_{\theta_i} J(\pi_i) = \mathbb{E}_{\mathbf{x},a\sim \mathrm{D}_r}\big[\, \nabla_{\theta_i}\, \pi_i(a_i|o_i) \nabla_{a_i}\, Q_i^{\mu}(\mathbf{x}, a_1, \ldots, a_N)|_{a_i=\pi_i(o_i)}\big] \tag{4}$$

where $\mathbf{x} = [o_1, \ldots, o_N]$ denotes the state of MAS, Q_i^{μ} is the value function, a_i and o_i are the action and observation, respectively, of agent i. D_r represents the experience replay buffer containing a series of tuples $(\mathbf{x}, \mathbf{x}', a_1, \ldots, a_N, r_1, \ldots, r_N)$, where $\mathbf{x}'$ is the new state. The critic network Q_i^{μ} is updated by the loss function as follows:

$$L(\theta_i) = \mathbb{E}[(Q_i^{\mu}(\mathbf{x}, a_1, \ldots, a_N) - y)^2] \tag{5}$$

where $y = R_i + \gamma Q_i^{\mu'}(\mathbf{x}', a_1', \ldots, a_N')$, R_i is the reward for the agent i, designed by the tasks for the MAS in a specific scenario. The actor network is updated by minimizing the policy gradient as follows:

$$\nabla_{\theta_i} J \approx \frac{1}{S}\sum_{n} \nabla_{\theta_i}\pi_i(o_i^n)\, \nabla_{a_i}\, Q_i^{\mu}(\mathbf{x}^n, a_1^n, \ldots, a_N^n)|_{a_i=\pi_i(o_i^n)} \tag{6}$$

where S is a random minibatch size and n is its index.

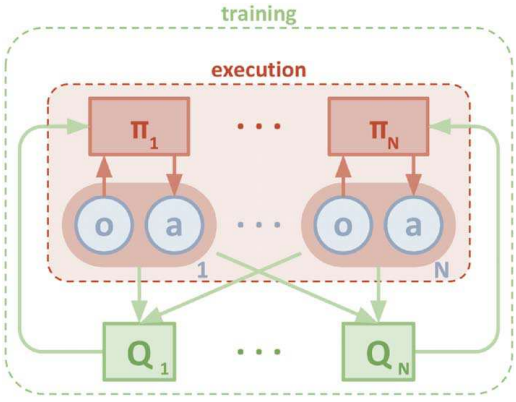


Fig. 2 MADDPG framework [13]

4 Methods

4.1 Problem Setup. In this section, we show a problem formulation to find the minimum black-box function in a d dimensional design space $A \subseteq \mathbb{R}^d$ with N agents in a team (i.e., an MAS). The goal of agent i, where $i \in \{1, 2, \ldots, N\}$, is to find the location of global minimum $\mathbf{x}^*$:

$$\mathbf{x}^* = \underset{\mathbf{x}\in A}{\operatorname{argmin}}\, f(\mathbf{x}) \tag{7}$$

where $f(\cdot)$ represents a black-box objective function and $\mathbf{x} = (x_1, x_2, \ldots, x_d) \in \mathbb{R}^d$ is an element of the set A.

We assume that the design space is partitioned into N local regions a priori, with each agent assigned to a unique region A_i within the design space A for the division of labor between agents in the MAS. In this search process, agents are not allowed to extend their search beyond their local regions. Instead, they communicate by sharing sampled local points with a global evaluator. There exist infeasible regions (constraints) in the design space, denoted as $D_i \subseteq A_i$.

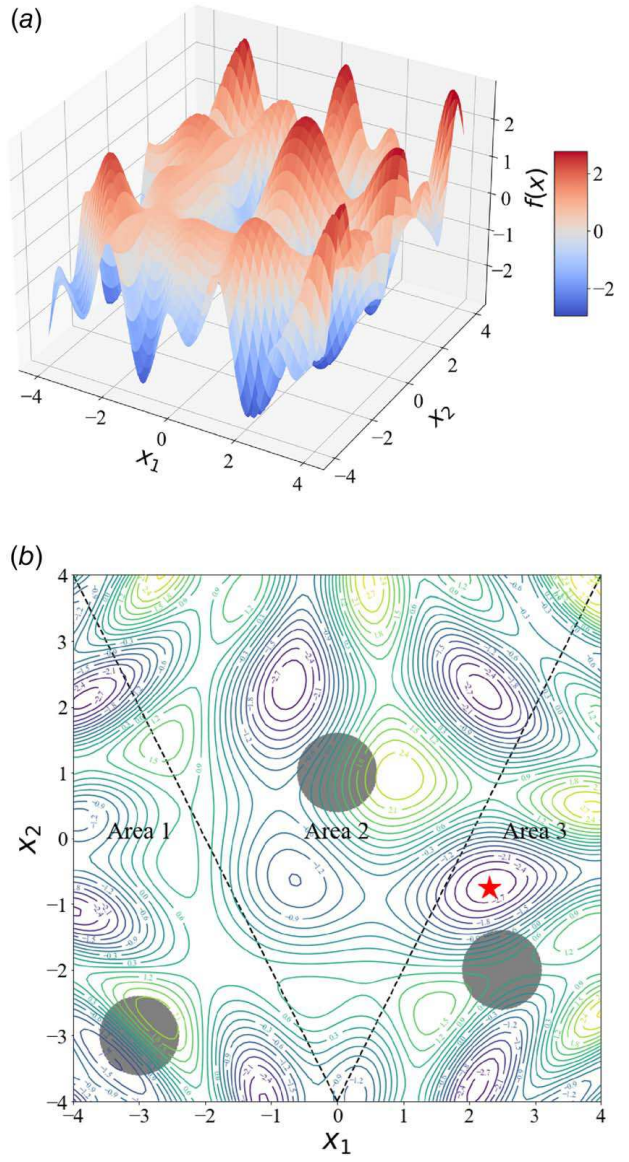


Fig. 3 Design space exploration with objective and constraints unknown to the agents. The star denotes the global minimum: (*a*) objective function and (*b*) contour plot.

Figure 3 shows an example of this design space exploration with unknown constraints based on an MAS with three agents. In this example, the objective function is shown in Fig. 3(*a*). The design space has been arbitrarily divided into three regions, as shown in Fig. 3(*b*). In each region, only one agent is responsible for searching the local space to find the global minimum (star in Area 3). Agents are not allowed to sample points within infeasible regions D_i (circles in Area 1, 2, and 3) in each local space A_i. It is worth noting that the impact of design space partitioning on MAS performance is not within the scope of this study. We adopt a particular strategy that divides the design space into N regions in every scenario to illustrate our approach and study the impact of cost on coordination between design agents.

4.2 Three Decisions for Design Space Exploration. The decision-making process of each agent in the proposed MAS involves three key decisions: *where to sample*, *where not to sample*, and *when to stop*. Initially, the MAS is trained via MARL, utilizing MARL to facilitate the identification of infeasible regions by the MAS in the decision of where not to sample. Subsequent to training, during implementation, MABO is adopted to determine where to sample next, and a proposed cost-aware stopping criterion guides each agent in deciding when to stop.

4.2.1 Decision 1: Where to Sample. In this study, we apply MABO to model each agent's decision-making process about where to sample within a team, aiming to optimize an unknown objective function by assigning local regions to each agent [12]. By enabling collaboration between agents, the system can search the complex design space more efficiently.

We use the MABO framework shown in Fig. 4 from our previous study [12]. First, the search space is divided into local regions. The number of these regions corresponds to the number of agents participating in exploration. Each region has an agent performing a local search on a specific segment of the objective function. To foster collaboration and information exchange among agents, a global-local communication strategy is enabled. This mechanism allows each local agent to share its sampling points with a global evaluator. This global evaluator consolidates data across all local searches, ensuring that the search process benefits from all the individual agents' information. When determining the next sample point, each agent works within its local region. However, rather than having access to the evaluation of the acquisition function across the entire design space, each agent is restricted to the evaluation of the acquisition function within its local region. Consequently, each agent makes its decision about the design that maximizes the value of the local acquisition function.

4.2.2 Decision 2: Where Not to Sample. Prior to implementing the MAS for DSE, it is initially trained using the MARL framework, as introduced in Sec. 3.2. After training, the objective of MARL is path planning, transitioning from one assigned sampling point to the next as directed by a global evaluator while avoiding infeasible areas during the sampling process.

MAS agents in the DSE need to learn how to take action $\mathbf{a}_i$ in a state $\mathbf{x}_i$ based on the system reward R they receive [35]. In this context, the state $\mathbf{x}_i$ is defined as the location of the agent in the design space, which is the value of the design variables. $\mathbf{a}_i$ indicates how much the design variable $\mathbf{x}_i$ needs to be increased or decreased.

In Sec. 3.2, we introduced that MADDPG contains a centralized training process and decentralized execution. This centralized training process generates optimal policies $\boldsymbol{\pi}_i$ for each agent through this framework. Every agent has the ability to perform actions $\mathbf{a}_i$ to reach a specific target $\mathbf{x}_i^*$ in its local region. In the training process, we define $\mathbf{x}_i^*$ as a random location in the design space that excludes unknown infeasible regions. In the implementation of MAS after training, the target $\mathbf{x}_i^*$ is set as the location where to sample next suggested by the global evaluator. After taking action $\mathbf{a}_i \in \mathbb{R}^d$, the agent updates its current state $\mathbf{x}_i^{'} = \mathbf{x}_i + \mathbf{a}_i$. During the decentralized execution in each episode, agents do not necessarily need to communicate or synchronize their actions with each other. Each agent independently decides its action based on its current state and the policy learned through the MADDPG algorithm.

This training process is described in Algorithm 1. During this process, the MAS undertakes the following *two tasks*:

(1) Target tracking, enabling agents to execute actions within the design space;
(2) Infeasible region detection, determining whether the targets are located in infeasible regions.

In order to complete the two tasks, a well-structured system reward R is required, which measures the quality of the action $\mathbf{a}_i$ for each agent. Therefore, we develop a reward mechanism specifically for target tracking and infeasible region detection as follows:

(1) Target tracking reward, R_i^o, is calculated according to the distance between the agent and the target $\mathbf{x}_i^*$, $R_i^o = -\sqrt{\|\mathbf{x}_i - \mathbf{x}_i^*\|^2}$, where $\mathbf{x}_i$ is the location of the agent i;
(2) Reward for infeasible region detection R_i^c, is calculated as $R_i^c = -N_c$, where N_c the number of collision times.

The rationale for not structuring the reward for detecting unknown infeasible regions as the distance between agents and these areas is that the agents need to sample points close to infeasible regions, which could be the potential global optimum.

Note that this model is configured in a cooperative setting. This means that each agent within the system is not only concerned with maximizing its own reward R_i but also contributes towards maximizing the total system reward R. The total system reward R is defined as the aggregate of individual rewards R_i obtained by each agent in the system.

Algorithm 1 MARL training process via MADDPG

Initialize the locations of constraints D_i $\forall i = \{1, 2, N\}$,
$MAX - episode$ and $MAX - step$
for $k = 1$ to $MAX - episode$ **do**
  Initialize $R_i^o \leftarrow 0$, $R_i^c \leftarrow 0$,
  for agent $i = 1$ to N **do**
    Receive initial state $\mathbf{x}_i$
    Random generate sampling point $\mathbf{x}_i^*$
    for $t = 1$ to $MAX - step$
      Select action a_i based on policy $\boldsymbol{\pi}_i$
      Update new state $\mathbf{x}_i \leftarrow \mathbf{x}_i + \mathbf{a}_i$
      Calculate $R_i^o = -\sqrt{\|x_i - x_i^*\|^2}$
      if Collide to the constraint **then**
        $R_i^c \leftarrow R_i^c - 1$
      else
        $R_i^c \leftarrow R_i^c - 0$ ▷ calculate the collision times N_c
      end if
      $R_i \leftarrow R_i^o + R_i^c$
    end for
    $R \leftarrow R + R_i$
    Return system reward R
    Store experience in replay buffer
    Sample a random minibatch of S in replay buffer
    Update the critic network by minimizing the loss in Eq. (5)
    Update the actor network by policy gradient in Eq. (6)
  end for
end for

4.2.3 Decision 3: When to Stop. To include cost considerations in the design process, we establish a stopping criterion for each agent based on a cost-aware utility function that we propose, presented as Eq. (8). Each agent stops the search if the utility score U is less than or equal to 0.

$$U = G - Kc \tag{8}$$

In this equation, the total number of iterations taken is represented by K, while c indicates the cost associated with each

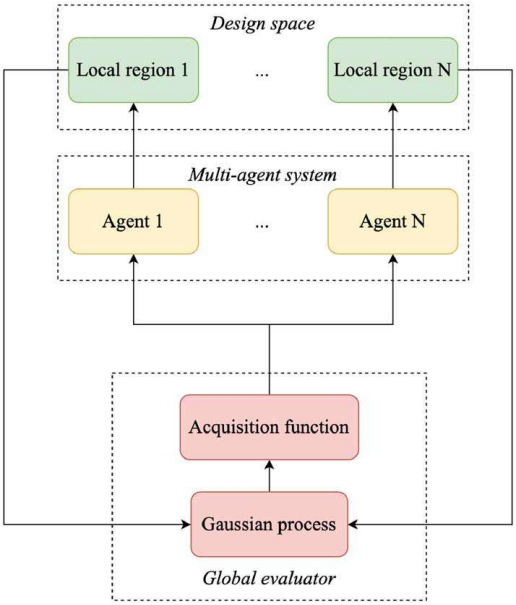


Fig. 4 MABO framework

sample to be defined. G refers to the cumulative gain achieved by the agent, defined as $G = \omega_{PG} PG + \omega_{IG} \text{IG}$. Here, IG is information gain, calculated as $\text{IG} = \sum_{k=1}^{K} a_{\text{LCB, k}}$. Here, $a_{\text{LCB, k}}$ represents the normalized value of the acquisition function at the kth step. Performance gain is represented by PG and calculated as $\text{PG} = -\sum_{k=1}^{K} (f_k^* - f_{k-1}^*)$. Here, f_k^* represents the normalized best value at step k. Intuitively, PG represents the cumulative performance each agent already achieves in each step, and IG indicates the potential gain it can achieve in the next sampling iteration. It is important to note that we use normalized values for both PG and IG to ensure a fair trade-off between performance gain and information gain. The tunable parameters ω_{PG} and ω_{IG} help to strike a balance between these two elements.

A key parameter in this article is the value of c, which has a significant impact on the agents' decision about when to stop. In particular, agents within the MAS exhibit varying sampling behaviors and influence overall system performance. Specifically, we experiment with both the same and agent-specific cost configurations. Furthermore, variations in the weights of IG and PG, represented by ω_{PG} and ω_{IG}, affect cumulative gains and the stopping criterion. Exploring cost settings in different ω_{PG} and ω_{IG} configurations plays an essential role in observing agent behaviors about when to stop in the MAS.

In this study, three cost-setting strategies are designed in this cost-aware approach:

- *Strategy A*: Different costs for each value of ω_{PG} and ω_{IG}, different costs for each agent;
- *Strategy B*: Same cost for each agent, different costs for each value of ω_{PG} and ω_{IG};
- *Strategy C*: Different costs for each agent, same costs for each value of ω_{PG} and ω_{IG}.

By adjusting the costs, we observe variations in the sampling behavior of individual agents and the interactions between agents within the MAS.

4.3 Cost-Aware Multi-Agent System for Design Space Exploration. In this section, we build the cost-aware MAS for design space exploration and integrate the process of making three decisions in this method. The structure of the design space exploration process with an MAS we propose is illustrated in Fig. 5. Algorithm 2 presents our cost-aware search strategy.

We begin by training the MAS to identify unknown constraints using the MADDPG framework, as outlined in Algorithm 1. After training, the agents can sample points within the design space. Before MAS starts sampling this design space, the design space A is divided into local areas A_i, and each is assigned to a specific agent i. Within their local region, each agent receives a prior Gaussian process given the initial observations $\mathcal{D}$.

Our algorithm allows MAS agents to navigate a series of sampling iterations K. For each iteration, a global evaluator calculates the posterior mean and variance using all points sampled by local agents. The acquisition function is then calculated using the posterior mean and variance to guide the next sampling decisions for local agents. Each local agent only has access to the acquisition function evaluated in its local area. The agents select the design that offers the highest value of the acquisition function within this local space.

Next, we employ a cost-aware stopping criterion, which facilitates a balance between PG and IG, to quantify the utility scores

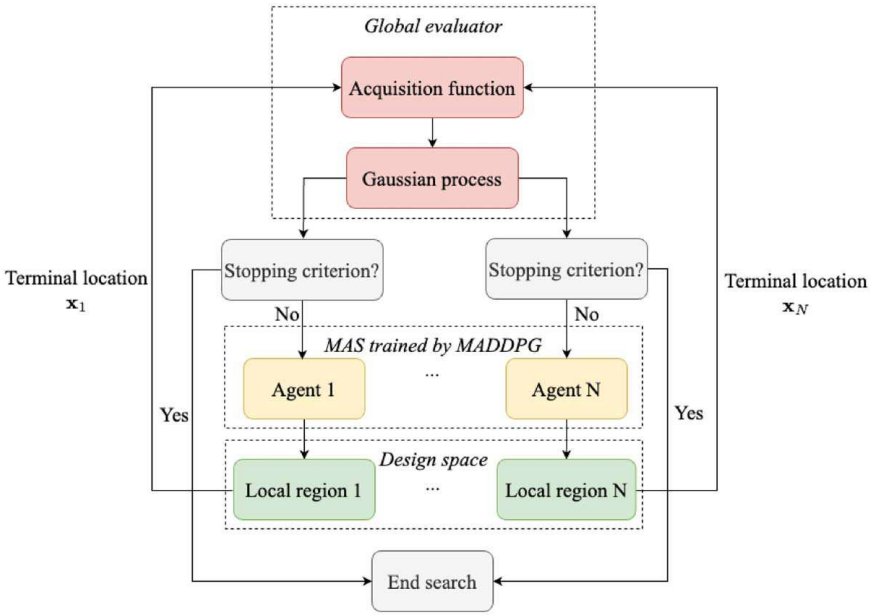


Fig. 5 Process flow for the cost-aware search

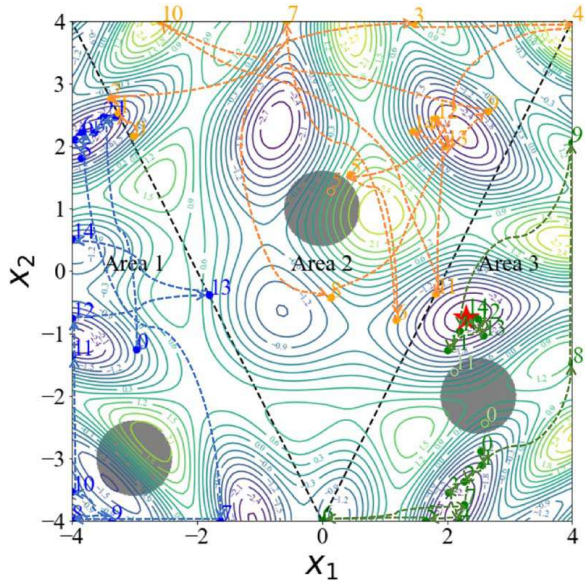


Fig. 6 An example of a sampling process involving three agents that possess global information, with each sampling point (filled points) labeled by its index. The dashed lines are the sampling trajectories of three agents. When the suggested points (hollow points, e.g., index 5 in area 2, index 0 and 11 in area 3) are located in the constraints (the gray circle areas), the agents can sample the corresponding points closest to the constraints in the feasible regions (filled points, index 5 in area 2, index 0 and 11 in area 3).

of points the agents sampled and decide when to stop. If the value of the utility function in Eq. (8) exceeds zero, the relevant $\mathbf{x}_i^*$ suggested by the global evaluator is communicated to the trained agent. This allows agents to take action within the design space, meaning that they can transition from the current points to the next suggested points and sample. In cases where the suggested points fall within infeasible regions, agents are restricted from sampling within those areas. However, they can still observe the function f at the destination $\mathbf{x}_i$ around the infeasible regions and send this information back to the global evaluator. Following the agents' actions, all associated data, including points and their associated function values, are collected into the global evaluator. If the value of the cost-aware utility function falls below or is equal to zero, which means that further sampling will not yield beneficial outcomes, the sampling process is terminated.

For a practical demonstration of Algorithm 2, refer to Fig. 3(*a*) that showcases the objective function, and the contour plot shown in Fig. 3(*b*). Figure 6 shows the corresponding sampling process under these constraints, involving three agents that are equipped with global information. These agents perform tasks in unique local regions, and dashed lines represent the paths of their operation. We present the sampling point in each iteration as a filled point, each of which is distinctly labeled by a numerical index. The interaction of the agents with their respective environments is critical to address the unknown constraints. For instance, when the agents get the points suggested by the global evaluator (represented as hollow points), like the point with index 5 in area 2, the points with index 0 and 11 in area 3, these points may sometimes fall within certain constraints, as shown by the gray circle. Accordingly, the agents have the capacity to sample points close to these constraints within feasible regions, like the filled points with index 5 in area 2, with index 0 and 11 in area 3, although the suggested points are located in the constraints. As a result, MAS has the ability to interact with the environment, allowing agents to address constraints without wasteful sampling steps and to send information about locations close to constraints to global evaluators.

Algorithm 2 Cost-aware search strategy

Training MAS via MADDPG framework as Algorithm 1
Initialize I, ω_{PG}, ω_{IG}, c in Eq. (8)
Set local region A_i for each agent in an unknown design space A
Place a Gaussian process prior $\mathcal{D}_i = (\mathbf{x}_{1:k}, \mathbf{f}(\mathbf{x}_{1:k}))$ in A_i for each agent
Observe f at the initial step in the design space A
for $k = 1$ to K **do**
  Update the posterior probability distribution with all available data on each local region A_i as Eq. (2)
  Update the acquisition function Eq. (3)
  for *agent* i, $i = 1$ to N **do**
    Let $\mathbf{x}_i^*$ be an optimizer of acquisition function Eq. (3) in the local region A_i
    Calculate cost-aware utility function as Eq. (8)
    If $U \geq 0$ **then**
      input $\mathbf{x}_i^*$ to the trained agent
      Agent take action a_i
      Agent sends back terminal state $\mathbf{x}_i$ and observes $f(\mathbf{x}_i)$
    end if
  end for
  Collect all data $(\mathbf{x}_i, f(\mathbf{x}_i))$ from agents
end for
Return a solution: the point $\mathbf{x}^*$ evaluated with the global optimum $f^*(\mathbf{x})$.

5 Experiments

5.1 Experimental Setups

5.1.1 MARL Settings. We use multi-agent particle environments to perform the experiments based on the environment presented by Ref. [13]. This environment is set in a two-dimensional space and is equipped with three agents and three infeasible regions, serving as a representation of a 2D unknown design space. During the training process, as detailed in Algorithm 1, we set specific parameters, $\mathrm{MAX-episode} = 100{,}000$ and $\mathrm{MAX-step} = 20$, in each episode.

5.1.2 MABO Settings. Three benchmark functions of varying complexity, from simple to complex, were evaluated: (1) the Cosines function, (2) the Michalewicz function, and (3) the Eggholder function, as illustrated in Table 1 and Fig. 7,to

Table 1 Three black-box functions

Name	Formula	Global minimum	Global domain A
Cosines	$f(\mathbf{x}) = 1 - (x_1^2 + x_2^2 - 0.3\cos(3\pi x_1) - 0.3\cos(3\pi x_2))$	$f(0.314, 0.303) = -1.596$	$\{\mathbf{x} \mid -1 \leq \mathbf{x} \leq 1\}$
Michalewicz	$f(\mathbf{x}) = -\sum_{i=1}^{2} \sin(x_i)\sin^{20}\left(\frac{ix_i^2}{\pi}\right)$	$f(2.201, 1.572) = -1.801$	$\{\mathbf{x} \mid -4 \leq \mathbf{x} \leq 4\}$
Eggholder	$f(\mathbf{x}) = -(x_2 + 47)\sin\left(\sqrt{\left\lvert x_2 + \frac{x_1}{2} + 47\right\rvert}\right) - x_1\sin\left(\sqrt{\lvert x_1 - (x_2 + 47)\rvert}\right)$	$f(512, 404.232) = -959.641$	$\{\mathbf{x} \mid -520 \leq \mathbf{x} \leq 520\}$

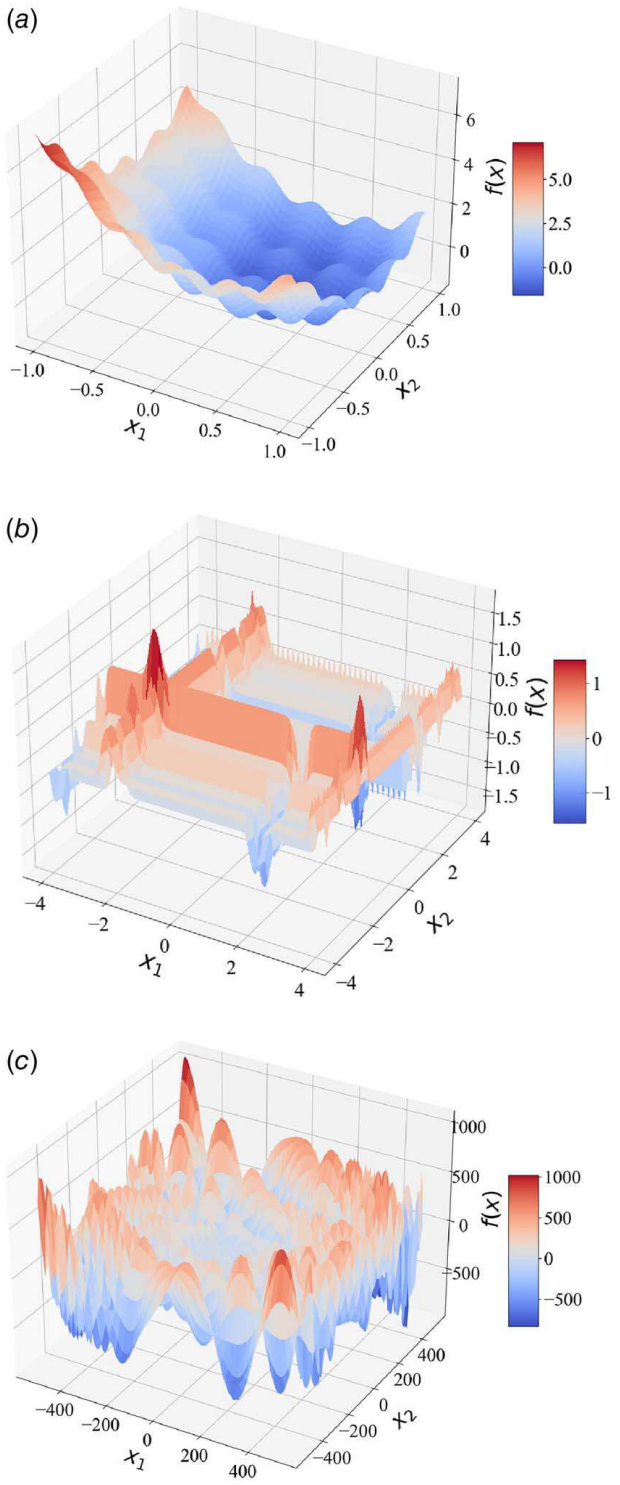


Fig. 7 Three benchmark functions. The Eggholder function is much more complex than the cosine and Michalewicz functions, with much more local optima: (*a*) cosines function, (*b*) Michalewicz function, and (c) Eggholder function.

understand the impact of the cost-aware stopping criterion on the collaborative search behaviors between agents. Specifically, the Eggholder function introduces greater complexity than the Cosines and Michalewicz functions because of its numerous local minima and maxima within the search space. These functions were selected on the basis of their distinct mathematical properties, which present various challenges to optimization algorithms. These functions serve as benchmarks in the BO literature and are widely recognized in the global optimization context [36–38].

For each scenario, we applied LCB as the acquisition function, represented by Eq. (3), and λ is fixed as 2.5 in each scenario. The initial number of samples for the Gaussian prior distribution, $\mathcal{D}_i = (\mathbf{x}_{1:k}, \mathbf{f}(\mathbf{x}_{1:k}))$ in A_i is set to $k = 5$ in Algorithm 2.

5.2 Experimental Results

5.2.1 Results Without a Stopping Criterion. By executing the proposed method with a predefined number of iterations without specifying any cost information for the agents, our objective is to understand how many steps it would take to achieve the convergence and therefore to identify the appropriate cost c to be set in different objective functions of varying complexities. Note that in a real-world design problem, the cost of design iterations for a designer is set externally by the problem context. However, in our theoretical (context-free) study, we need to choose a meaningful value for the cost to study its impact on team collaboration. Instead of choosing arbitrary values for the cost, we set the cost for sampling a new design based on a baseline study in which agents work together without any consideration of cost. On the basis of convergence results of this baseline study, we choose different values for c following the three strategies described earlier. In this context, the convergence of the MAS is characterized by the ability of the agents to achieve the global optimum. When the global optimum falls within the infeasible region, the convergence is then defined by the MAS's inability to achieve further improvement. The experimental results with fixed iterations of these objective functions are shown in Figs. 8–10.

(1) *Cosines function.* Each local region to which each agent is assigned and the infeasible region defined in each local region are shown in Table 2. The sampling process and the search trajectory (indicated by the number index) of each agent in its own region are displayed in Fig. 8(*a*). The best $f(\mathbf{x})$ (i.e., f^* in Eq. (3)) observed so far in each step shown in Fig. 8(*b*) describes the convergence speed for an MAS, and agent 2 reached the global minimum at the 13th step. Agent 1 and agent 3 found their local minima at the fourth and ninth steps, respectively.

(2) *Michalewicz function.* In this scenario, the division of the design space and the infeasible regions are set consistent with those of the Cosines function. As shown in Fig. 9(*a*), the global minimum of the Michalewicz function is located in area 2, denoted by $f(2.201, 1.572) = -1.801$. Figure 9(*b*) shows that agent 2 successfully reached this global minimum at the 18th iteration. Agents 1 and 3 cannot achieve better improvement after the 6th and 25th iterations, respectively.

(3) *Eggholder function.* Figure 10(*a*) illustrates the sampling process of MAS in the Eggholder function, and the global minimum is located in area 3, given by $f(512, 404.232) = -959.641$. The Eggholder function is more complex than the cosines and Michalewicz functions, making it harder to find the global minimum with MABO. Specifically, agent 3 achieves the global optimum after 27 iterations, as shown in Fig. 10(*b*). Agents 1 and 2 see no further improvement after the 30th and 40th iterations, respectively.

The trends of cumulative gains G defined in Eq. (8) during search are shown in Figs. 8(*c*), 9(*c*), and 10(*c*). Specifically, the cosines function shows a linear increase in cumulative gain across iterations (see Fig. 8(*c*)). On the other hand, the Michalewicz function follows a convex trend (see Fig. 9(*c*)), indicating an increase in growth with a decreasing rate in each iteration. Meanwhile, the Eggholder function exhibits a stepwise behavior (see Fig. 10(*c*)), with significant gains at certain iterations and minimal gains at others. These varying cumulative gain patterns can affect individual agents in the system in deciding when to

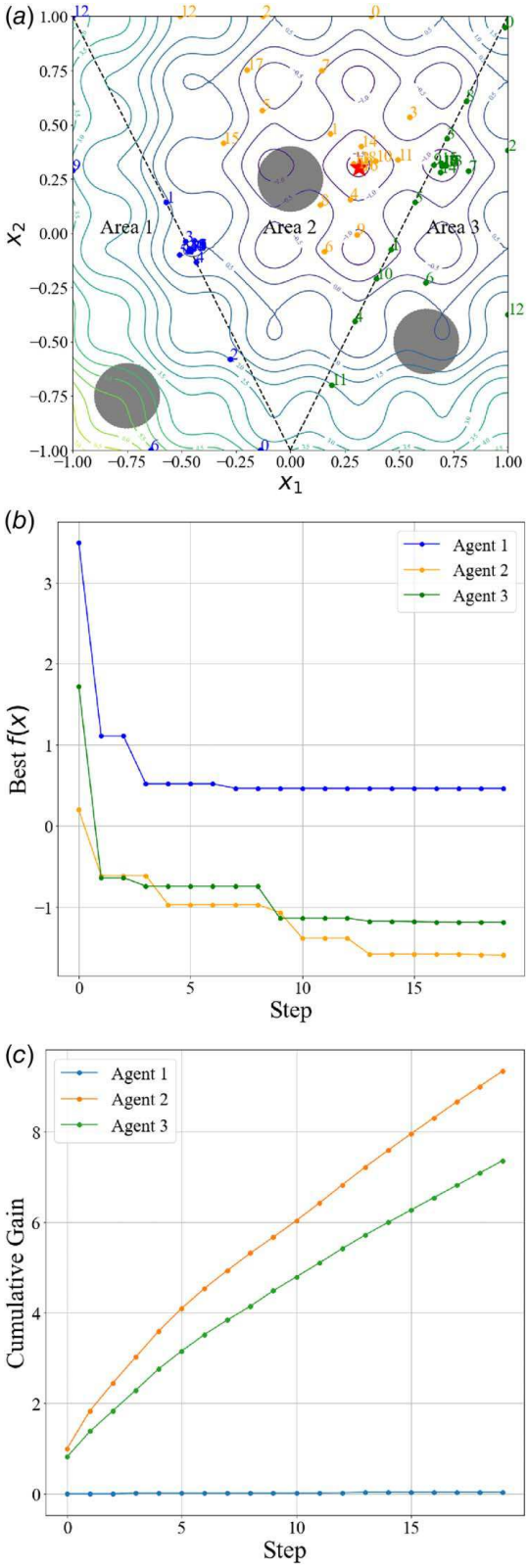


Fig. 8 Results for cosines function after 20 sampling steps with agents converging at steps 3, 13, and 9, where $\omega_{PG} = 0$, $\omega_{IG} = 1$: (*a*) sampling process, (*b*) convergence speed, and (c) cumulative gain

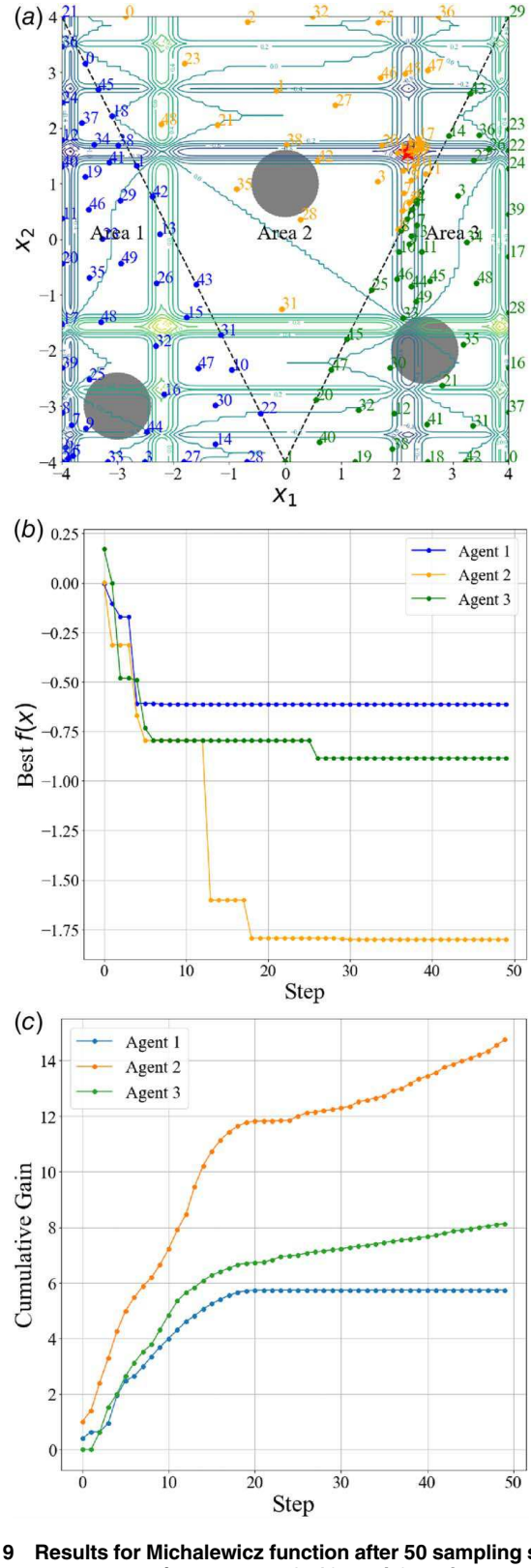


Fig. 9 Results for Michalewicz function after 50 sampling steps with agents converging at steps 6, 18, and 25, where $\omega_{PG} = 0$, $\omega_{IG} = 1$: (*a*) sampling process, (*b*) convergence speed, and (*c*) cumulative gain

stop and the convergence in the MAS. The unique trend of each cumulative gain exhibited in these three tested objective functions is another reason we chose them in this study, besides the varying complexity of the three functions.

5.2.2 Results With Stopping Criterion. We evaluated three cost-aware strategies, introduced in Sec. 4.2.3, on three objective functions with varying complexities. Table 3 presents detailed strategies along with their corresponding tables for three benchmark functions.

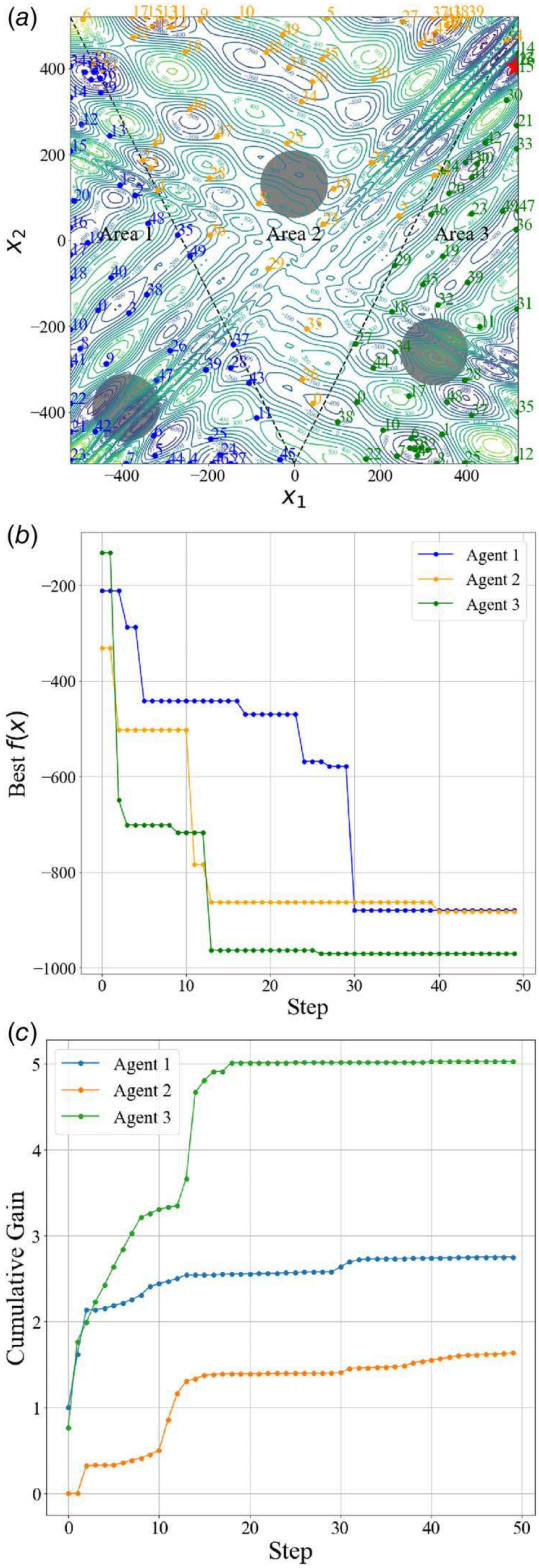


Fig. 10 Results for Eggholder function after 50 sampling steps with agents converging at steps 30, 40, and 27, where $\omega_{PG} = 0$, $\omega_{IG} = 1$: (*a*) sampling process, (*b*) convergence speed, and (*c*) cumulative gain

Each table details the various parameter settings and performance of MAS tested on the cosines (Tables 4–6), Michalewicz (Tables 7–9), and Eggholder functions (Tables 10–12). Columns ω_{PG} and ω_{IG} indicate the values of the complementary parameters of PG and IG in Eq. (8)—as the weight of PG decreases from 1 to 0 in 0.1 steps, the weight of IG increases in the same increment. Columns Cost 1, Cost 2, and Cost 3 represent the cost c in Eq. (8) for each of the three agents in these settings. The number of iterations taken by each agent during the search process is shown in the *agent 1 Iter, 2 Iter, and 3 Iter* columns. The column Global optimum? indicates whether the global optimum is reached, and if so, the number of iterations at which the global optimum was achieved is reported. In Figs. 11–13, we demonstrate a few examples of convergence speed of the cosines, Michalewicz, and Eggholder functions with a particular stopping criterion configuration, $\omega_{PG} = 0.5$ and $\omega_{IG} = 0.5$.

(1) Cosines function

Strategy A. Agents have different costs, and the costs are different in each setting of the parameters, ω_{PG} and ω_{IG}. We determine the appropriate cost values (e.g., not too high or too low) using the average gain observed when an agent converges in the scenario without stopping criteria, as detailed in Sec. 5.2.1. For example, in the results of the cosines function (Fig. 8), agents identify their local optima at the 3th, 13th, and 9th steps, respectively. At these steps, the corresponding cumulative gains G are 0.016, 7.35, and 0.461 (Fig. 8(*c*)) under the conditions of $\omega_{PG} = 0$ and $\omega_{IG} = 1$. The average gains derived from these are 0.004, 0.525, and 0.461. It should be noted that varying the values of ω_{PG} and ω_{IG} can lead to different cumulative gains and, consequently, different cost settings, as presented in Table 4.

We summarize three key observations from Table 4. First, we pinpointed an optimal approach to setting cost values. By adopting the first cost-aware strategy, we achieved consistent convergence, and the convergence speed of the global optimum was stable around 11 steps in all parameter settings of ω_{PG} and ω_{IG}. Second, we observed that the agents did not stop sampling as expected in the scenario without a stopping criterion, although we set the cost as the average gain when the agents converged. For example, agents did not stop at the 3rd, 13th, and 9th iterations. This behavior links to agent 1 stopping early in the sampling process, which left agent 2 and agent 3 without getting essential information from agent 1's local region for the search of global minimum. As a result, these agents extended their sampling, as IG from their own local region increased and costs were relatively low. Third, by setting the cost according to the average gain upon agents' convergence and noting the cost's increase with the growing weight of IG, it is clear that IG takes precedence over PG.

Strategy B. Agents have the same cost, but the costs are different in each parameter setting, ω_{PG} and ω_{IG}. In Table 4, we assign varying costs to each agent. Taking the average of these costs, we set the same cost for all three agents, as shown in Table 5. In the initial three settings, agent 2 and agent 3 registered small PG, resulting in limited sampling due to higher cost and therefore negative utility scores. Consequently, they ended their search in their local region, hindering the convergence of MAS to find the global optimum. As the weight of IG increases, from the fourth setting, agent 2 and agent 3 can achieve a higher cumulative gain and, therefore, a positive utility score, and the search process can continue until the global optimum is found.

Strategy C. Agents have different costs, but the costs are the same in each setting of the parameters, ω_{PG} and ω_{IG}. In Table 6, we applied the same cost for each setting of ω_{PG} and ω_{IG} but kept different costs for each agent. These cost values were derived from the averages in Table 4. For instance, we used the average of cost 1 across all settings of ω_{PG} and ω_{IG} from Table 4 for the value of cost 1 in Table 6. Therefore, we obtain the costs for each agent as 0.159, 0.288, and 0.288, respectively. With this cost-setting strategy, the system did not achieve convergence in the first five settings. However, it succeeded in the next five. This is due to the same reason observed in Table 5 that, as the weight of IG increased, agent 2 and agent 3 obtained higher gains in their local regions. This led to positive utility scores; thus, the agents will not stop sampling before reaching the global optimum.

Table 2 Cosines function

MAS	Local domain A_i	Infeasible area D_i
Agent 1	$\{\mathbf{x}\| -2x_1 + x_2 + 2 \le 0\}$	$\{\mathbf{x}\|(x_1 + 0.75)^2 + (x_2 + 0.75)^2 \le 0.15^2\}$
Agent 2	$\{\mathbf{x}\| -2x_1 - x_2 - 2 \le 0, 2x_1 - x_2 - 2 \le 0\}$	$\{\mathbf{x}\|x_1^2 + (x_2 - 0.25)^2 \le 0.15^2\}$
Agent 3	$\{\mathbf{x}\| -2x_1 + x_2 + 2 \le 0\}$	$\{\mathbf{x}\|(x_1 - 0.6)^2 + (x_2 + 0.5)^2 \le 0.15^2\}$

Note: Local design space domain and infeasible areas for three agents.

Table 3 Three cost-setting strategies and associated results under three benchmark functions

Strategies	Unit costs are different for each agent	Unit costs are different for each value of ω_{PG} and ω_{IG}	Associated tables
A	Yes	Yes	Cosines: 4, Michalewicz: 7, Eggholder: 10
B	No	Yes	Cosines: 5, Michalewicz: 8, Eggholder: 11
C	Yes	No	Cosines: 6, Michalewicz: 9, Eggholder: 12

Table 4 Cosines function

ω_{PG}	ω_{IG}	Cost 1	Cost 2	Cost 3	Agent 1 Iter	Agent 2 Iter	Agent 3 Iter	Global optimum?
1	0	0.313	0.051	0.115	6	16	12	12
0.9	0.1	0.282	0.098	0.149	5	16	12	11
0.8	0.2	0.251	0.146	0.184	5	16	12	11
0.7	0.3	0.22	0.193	0.218	5	16	12	11
0.6	0.4	0.19	0.241	0.253	5	16	12	11
0.5	0.5	0.159	0.288	0.288	4	15	11	11
0.4	0.6	0.128	0.335	0.322	4	15	11	11
0.3	0.7	0.097	0.382	0.357	4	15	11	11
0.2	0.8	0.066	0.43	0.392	5	15	11	11
0.1	0.9	0.035	0.478	0.426	5	16	12	11
0	1	0.004	0.525	0.461	6	16	12	12

Note: Strategy A: Agents have different costs, and the costs are different in each setting of the parameters, ω_{PG} and ω_{IG}. Columns ω_{PG} and ω_{IG} indicate the values of PG and IG in Eq. (8). Columns Cost 1, Cost 2, and Cost 3 represent cost c in Eq. (8) for each of the three agents. The number of iterations taken by each agent during the search process is shown in the agent 1 Iter, 2 Iter, and 3 Iter columns. The column Global optimum? indicates if the global optimum is reached, and if so, the number of iterations at which the global optimum was achieved is reported. The same notation applies to Tables 5–12.

Table 5 Cosines function

ω_{PG}	ω_{IG}	Cost	Agent 1 Iter	Agent 2 Iter	Agent 3 Iter	Global optimum?
1	0	0.16	8	3	7	No
0.9	0.1	0.176	7	5	8	No
0.8	0.2	0.193	7	9	10	No
0.7	0.3	0.21	5	12	13	12
0.6	0.4	0.228	5	17	14	11
0.5	0.5	0.245	5	22	15	11
0.4	0.6	0.262	5	24	17	11
0.3	0.7	0.279	5	27	18	11
0.2	0.8	0.296	5	30	19	11
0.1	0.9	0.313	5	32	20	11
0	1	0.33	5	34	21	11

Note: Strategy B: Agents have the same cost, but the costs are different in each setting of the parameters, ω_{PG} and ω_{IG}.

(2) Michalewicz function

Strategy A. Agents have different costs, and the costs are different in each setting of the parameters, ω_{PG} and ω_{IG}. We followed a similar approach in cosines function to determine the cost, using the average gain when the agent reaches a local optimum, which is 6th, 18th, and 25th, as shown in Fig. 9. With this cost-setting strategy, the MAS consistently reached its convergence at the 13th step for all values of ω_{PG} and ω_{IG}, as shown in Table 7. In this design space, we noticed sampling patterns and collaboration between agents similar to those in the cosine function. A significant observation from our research involved some agents stopping their sampling earlier than anticipated. For example, agent 1 stopped sampling around the 8th step. However, this early stopping did not hinder the system's performance. Interestingly, MAS converged faster, reaching its goal at the 13th step compared to the 18th step in the scenario without the stopping criterion. This indicates not only potential savings in sampling costs relative to strategies without a stopping criterion but also a more efficient route to convergence with fewer sampling iterations.

Strategy B. Agents have the same cost, but the costs are different in each setting of the parameters, ω_{PG} and ω_{IG}. To determine the costs, we utilized an approach in which each agent was assigned the same cost. This method appears beneficial, as the MAS consistently achieves convergence for every combination of ω_{PG} and ω_{IG},

Table 6 Cosines function

ω_{PG}	ω_{IG}	Cost 1	Cost 2	Cost 3	Agent 1 Iter	Agent 2 Iter	Agent 3 Iter	Global optimum?
1	0	0.159	0.288	0.288	8	2	4	No
0.9	0.1	0.159	0.288	0.288	7	2	4	No
0.8	0.2	0.159	0.288	0.288	6	3	5	No
0.7	0.3	0.159	0.288	0.288	6	5	6	No
0.6	0.4	0.159	0.288	0.288	5	8	7	No
0.5	0.5	0.159	0.288	0.288	5	13	9	11
0.4	0.6	0.159	0.288	0.288	5	18	12	11
0.3	0.7	0.159	0.288	0.288	5	23	15	11
0.2	0.8	0.159	0.288	0.288	5	27	17	11
0.1	0.9	0.159	0.288	0.288	5	31	20	11
0	1	0.159	0.288	0.288	5	36	22	11

Note: Strategy C: Agents have different costs, but the costs are the same in each setting of the parameters, ω_{PG} and ω_{IG}.

Table 7 Michalewicz function

ω_{PG}	ω_{IG}	Cost 1	Cost 2	Cost 3	Agent 1 Iter	Agent 2 Iter	Agent 3 Iter	Global optimum?
1	0	0.227	0.167	0.088	8	20	27	13
0.9	0.1	0.245	0.214	0.107	8	20	26	13
0.8	0.2	0.264	0.261	0.126	8	20	26	13
0.7	0.3	0.283	0.307	0.145	8	20	26	13
0.6	0.4	0.301	0.354	0.164	8	20	26	13
0.5	0.5	0.32	0.401	0.183	8	20	26	13
0.4	0.6	0.339	0.448	0.203	8	20	26	13
0.3	0.7	0.357	0.495	0.222	8	20	26	13
0.2	0.8	0.376	0.542	0.241	9	20	29	13
0.1	0.9	0.395	0.588	0.26	9	20	29	13
0	1	0.413	0.635	0.279	10	20	30	13

Note: Strategy A: Agents have different costs, and the costs are different in each setting of the parameters, ω_{PG} and ω_{IG}.

Table 8 Michalewicz function

ω_{PG}	ω_{IG}	Cost	Agent 1 Iter	Agent 2 Iter	Agent 3 Iter	Global optimum?
1	0	0.161	10	21	15	18
0.9	0.1	0.189	10	23	15	18
0.8	0.2	0.217	10	24	15	18
0.7	0.3	0.245	10	25	15	18
0.6	0.4	0.273	9	25	17	13
0.5	0.5	0.301	9	26	17	13
0.4	0.6	0.33	9	26	17	13
0.3	0.7	0.358	8	27	16	13
0.2	0.8	0.386	8	28	16	13
0.1	0.9	0.414	8	28	16	13
0	1	0.442	7	28	15	13

Note: Strategy B: Agents have the same cost, but the costs are different in each setting of the parameters, ω_{PG} and ω_{IG}.

Table 9 Michalewicz function

ω_{PG}	ω_{IG}	Cost 1	Cost 2	Cost 3	Agent 1 Iter	Agent 2 Iter	Agent 3 Iter	Global optimum?
1	0	0.32	0.402	0.183	6	6	14	No
0.9	0.1	0.32	0.402	0.183	6	8	16	No
0.8	0.2	0.32	0.402	0.183	6	9	21	No
0.7	0.3	0.32	0.402	0.183	7	10	30	No
0.6	0.4	0.32	0.402	0.183	7	12	35	No
0.5	0.5	0.32	0.402	0.183	8	20	26	13
0.4	0.6	0.32	0.402	0.183	9	22	32	13
0.3	0.7	0.32	0.402	0.183	12	25	31	13
0.2	0.8	0.32	0.402	0.183	14	27	34	13
0.1	0.9	0.32	0.402	0.183	17	30	37	13
0	1	0.32	0.402	0.183	19	32	40	13

Note: Strategy C: Agents have different costs, but the costs are the same in each setting of the parameters, ω_{PG} and ω_{IG}.

Table 10 Eggholder function

ω_{PG}	ω_{IG}	Cost 1	Cost 2	Cost 3	Agent 1 Iter	Agent 2 Iter	Agent 3 Iter	Global optimum?
1	0	0.025	0.034	0.06	2	2	2	No
0.9	0.1	0.03	0.036	0.073	53	11	29	No
0.8	0.2	0.036	0.037	0.085	50	11	30	No
0.7	0.3	0.042	0.038	0.1	42	32	28	27
0.6	0.4	0.049	0.039	0.11	40	32	28	27
0.5	0.5	0.055	0.041	0.123	38	33	28	27
0.4	0.6	0.061	0.042	0.135	37	33	28	27
0.3	0.7	0.067	0.043	0.148	36	33	29	27
0.2	0.8	0.073	0.044	0.161	35	33	28	27
0.1	0.9	0.08	0.045	0.173	34	33	28	27
0	1	0.086	0.047	0.186	33	33	28	27

Note: Strategy A: Agents have different costs, and the costs are different in each setting of the parameters, ω_{PG} and ω_{IG}.

as shown in Table 8. Given that the global optimum resides in area 2, agent 2 tends to accumulate substantial gains until it locates this optimum. Using the average cost in each setting as described in Table 7, the cost of agent 2 becomes relatively lower. Consequently, agent 2 consistently registers a positive utility, enabling it to perform more iterations and discover the global optimum.

Strategy C. Agents have different costs, but the costs are the same in each setting of the parameters, ω_{PG} and ω_{IG}. In this cost-setting strategy, the search performance of the MAS is similar to that of the cosines function, as shown in Table 9. Using this method, the system could not achieve convergence in the first five settings of ω_{PG} and ω_{IG}, but it succeeded in the following five. As we increased the weight of IG, they achieved better gains locally, leading to positive utility and increased iterations of sampling.

(3) Eggholder function

Strategy A. Agents have different costs, and the costs are different in each setting of the parameters, ω_{PG} and ω_{IG}. In the experiment based on the Eggholder function, we set different costs for each agent based on their steps to reach their local optima: 30th, 40th, and 27th steps, respectively. The results based on the first cost setting are shown in Table 10. Compared to simple objective functions (cosines and Michalewicz functions), the agent stopping early would have a great impact on the convergence in a complex objective function. We can imagine that, for this complex design space, agents need to sample more for finding the optimal design compared to the simple case, and the MAS needs more information from local regions to better understand the entire design space. For example, in the first three settings with different ω_{PG} and ω_{IG}, the MAS is unable to achieve convergence. It is due to agent 2 stopping searching within the first ten steps. We can see from the cumulative gain without setting a stopping criterion (Fig. 10(*c*)) that agent 2, although initially showing low gains, had the potential for greater gains as the process continued. This means that if we establish the average gain of the agent achieved

Table 11 Eggholder function

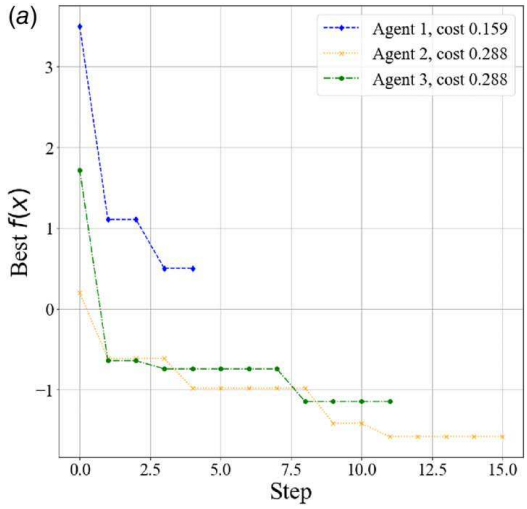
ω_{PG}	ω_{IG}	Cost	Agent 1 Iter	Agent 2 Iter	Agent 3 Iter	Global optimum?
1	0	0.04	2	2	2	No
0.9	0.1	0.046	19	9	45	No
0.8	0.2	0.053	27	8	31	No
0.7	0.3	0.06	34	8	86	No
0.6	0.4	0.066	39	8	100	No
0.5	0.5	0.073	24	6	100	79
0.4	0.6	0.079	26	6	93	55
0.3	0.7	0.086	32	5	38	No
0.2	0.8	0.093	33	5	39	No
0.1	0.9	0.099	33	5	44	No
0	1	0.106	33	3	44	No

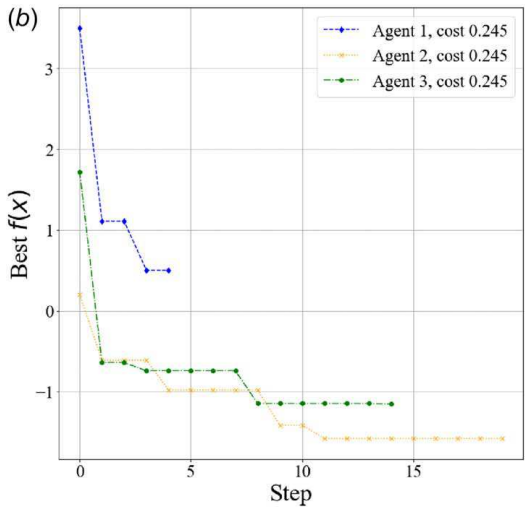
Note: Strategy B: Agents have the same cost, but the costs are different in each setting of the parameters, ω_{PG} and ω_{IG}.

Table 12 Eggholder function

ω_{PG}	ω_{IG}	Cost 1	Cost 2	Cost 3	Agent 1 Iter	Agent 2 Iter	Agent 3 Iter	Global optimum?
1	0	0.055	0.041	0.123	2	2	2	No
0.9	0.1	0.055	0.041	0.123	23	10	12	No
0.8	0.2	0.055	0.041	0.123	29	10	14	No
0.7	0.3	0.055	0.041	0.123	35	10	16	No
0.6	0.4	0.055	0.041	0.123	38	11	27	No
0.5	0.5	0.055	0.041	0.123	38	32	28	27
0.4	0.6	0.055	0.041	0.123	41	33	31	27
0.3	0.7	0.055	0.041	0.123	43	34	34	27
0.2	0.8	0.055	0.041	0.123	46	35	37	27
0.1	0.9	0.055	0.041	0.123	49	36	40	27
0	1	0.055	0.041	0.123	51	51	42	27

Note: Strategy C: Agents have different costs, but the costs are the same in each setting of the parameters, ω_{PG} and ω_{IG}.





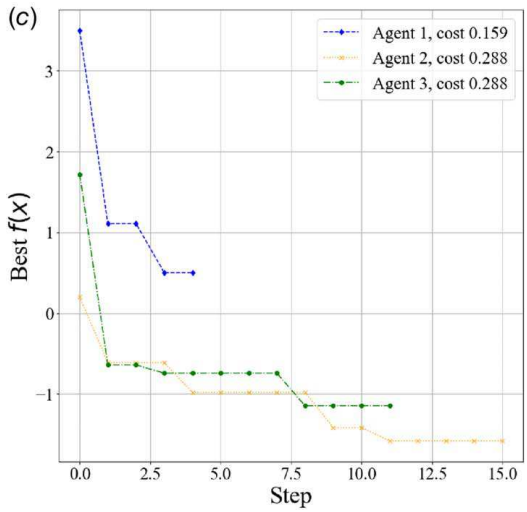
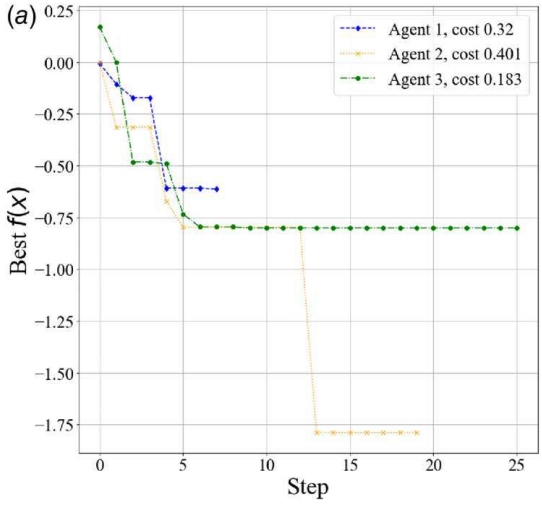
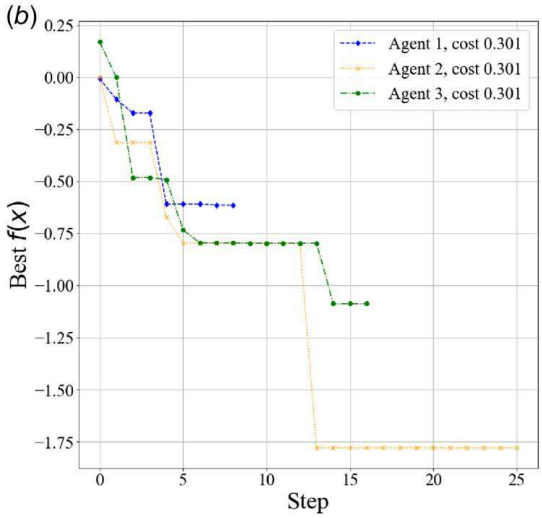


Fig. 11 Convergence speed based on the cosines function, demonstrating results with a stopping criterion where $\omega_{PG} = 0.5$, $\omega_{IG} = 0.5$: (*a*) Strategy A: agents have different costs, and the costs are different in each setting of the parameters, ω_{PG} and ω_{IG}. Costs for each agent in each iteration are 0.159, 0.288, and 0.288, respectively. (*b*) Strategy B: Agents have the same cost, but the costs are different in each setting of the parameters, ω_{PG} and ω_{IG}. The cost for each agent in each iteration is 0.245. (*c*) Strategy C: Agents have different costs, but the costs are the same in each setting of the parameters, ω_{PG} and ω_{IG}. Costs for each agent in each iteration are 0.159, 0.288, and 0.288, respectively.





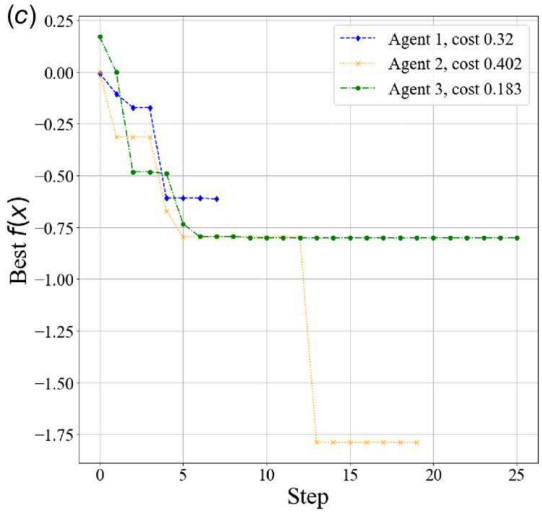


Fig. 12 Convergence speed based on the Michalewicz function, demonstrating results with a stopping criterion where $\omega_{PG} = 0.5$, $\omega_{IG} = 0.5$: (*a*) Strategy A: Agents have different costs, and the costs are different in each setting of the parameters, ω_{PG} and ω_{IG}. Costs for each agent in each iteration are 0.32, 0.401, and 0.183, respectively. (*b*) Strategy B: Agents have the same cost, but the costs are different in each setting of the parameters, ω_{PG} and ω_{IG}. The cost for each agent in each iteration is 0.301. (*c*) Strategy C: Agents have different costs, but the costs are the same in each setting of the parameters, ω_{PG} and ω_{IG}. The costs for each agent in each iteration are 0.32, 0.402, and 0.183, respectively.

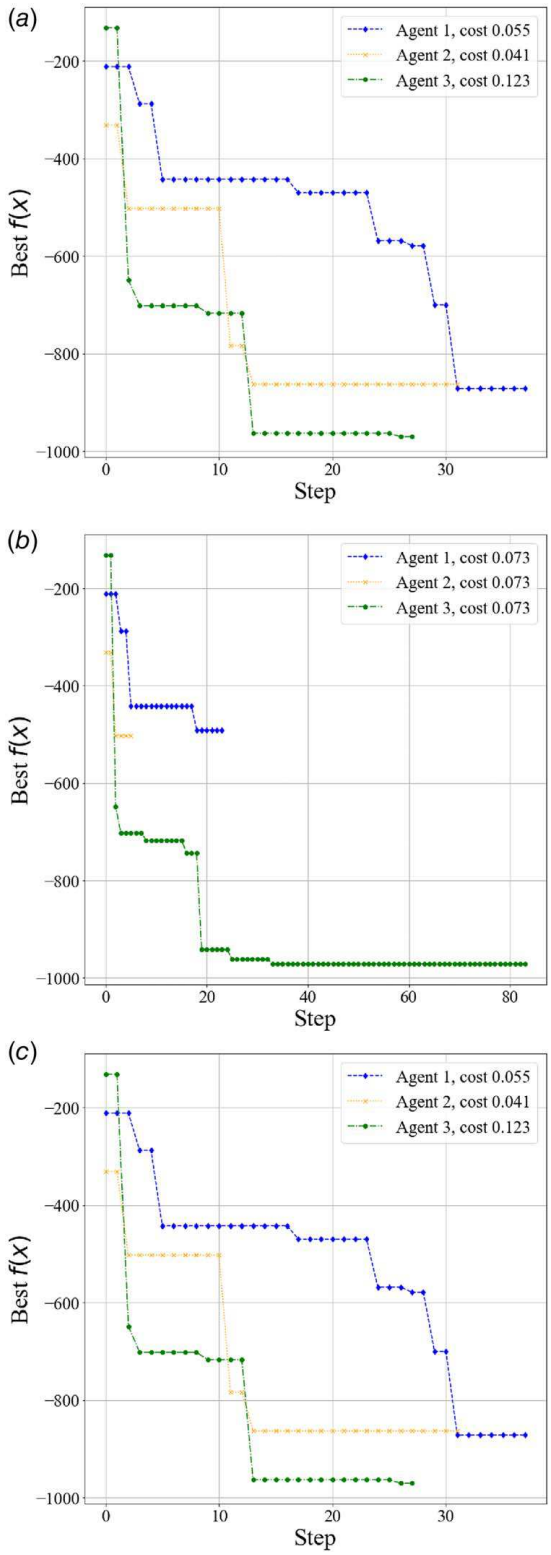


Fig. 13 Convergence speed based on the Eggholder function, demonstrating results with a stopping criterion where $\omega_{PG} = 0.5$, $\omega_{IG} = 0.5$: (*a*) Strategy A: Agents have different costs, and the costs are different in each setting of the parameters, ω_{PG} and ω_{IG}. Costs for each agent in each iteration are 0.055, 0.041, and 0.123, respectively. (*b*) Strategy B: Agents have the same cost, but the costs are different in each setting of the parameters, ω_{PG} and ω_{IG}. The cost for each agent in each iteration is 0.073. (*c*) Strategy C: Agents have different costs, but the costs are the same in each setting of the parameters, ω_{PG} and ω_{IG}. The costs for each agent in each iteration are 0.055, 0.041, and 0.123, respectively.

upon reaching the local optimum as the cost for each sample, the cost would be too high for agent 2 at the early stage of the sampling process. However, a turning point was observed in the fourth setting. Here, the system can identify the global optimum, mainly because all agents, especially agent 3, where the optimum is located, continue to sample until the stopping criterion is satisfied, and all agents can obtain the information from other regions consistently during this process. This phenomenon illustrates that information from all agents is crucial for identifying the global optimum in the MAS, which shows the importance of collaboration between agents for convergence in the complex objective function with many local optima.

Strategy B. Agents have the same cost, but the costs are different in each setting of the parameters, ω_{PG} and ω_{IG}. According to Table 8, the second cost-setting strategy seems ineffective for the convergence of the system in the Eggholder function, as agents often fail to identify the global optimum. This outcome is mainly due to the inconsistent gains realized by different agents. When the same cost was applied to all agents, those with low gains and high sampling costs, resulting in positive utility, had a propensity to stop searching. This early stop adversely impacted the MAS's capability to converge to the global optimum. Such findings emphasize the crucial role of collaboration between agents in complex design space exploration.

Strategy C. Agents have different costs, but the costs are the same in each setting of the parameters, ω_{PG} and ω_{IG}. Referring to Table 12, with this cost-setting strategy, the system does not converge in the first five settings. However, it achieves convergence in the subsequent five settings. A side-by-side examination of Tables 11 and 12 indicates the important role of agent 2's information in system convergence. In Table 11, agent 2 stops its operation early, sampling no more than ten steps. This early termination of agent 2 impedes the MAS's convergence. On the contrary, in the last five settings of ω_{PG} and ω_{IG} in Table 12, the system achieves convergence due to agent 2 sampling until the later stages.

6 Discussion

In this article, our primary research question is: *What impact would the cost-aware stopping criteria have on the collaboration between multiple agents in design space exploration?* On the basis of our experimental results, we have identified several key observations that answer this RQ. In the following section, we will dive into these observations, providing insight and drawing takeaways from their practical scenarios.

(1) Our results confirm the findings of our previous study in Ref. [12] that communication or information-sharing in MABO can greatly influence the individual agent's behavior and the team performance. We obtain these results with the assumption that the design task is divided into subtasks assigned to each member, and each design iteration incurs some cost. For instance, in our first test without stopping criteria, the cumulative gain looks like a step function (Fig. 10(*c*)). If the cost was high, as the value was set based on the average gain, some agents could stop early because they could not achieve much improvement in the initial search, even if they could make large improvements later. However, stopping early affects other agents' search performance, as they could not continuously receive information from those who stopped. Although certain regions of the design space do not yield promising performance results at the beginning of the search, spending additional resources to further explore those regions could provide valuable insight that could help other members of the design team achieve the global optimum. As a solution, we can consider creating an additional parameter that serves as an initial bet (I) for all agents. For instance, the utility score, as defined in Eq. (8), can be reformulated as $U = (I + G) - Kc$. This can prevent certain "promising" or "high-potential" agents, who may have modest gains in the beginning but are likely to see significant improvement with continued exploration, from stopping

searching too early. Even if this may accompany the waste of additional costs on exploring agents' local regions, this approach could effectively guide the system toward convergence. Consequently, there exists a trade-off between convergence/optimum achievability and search cost incurred.

(2) Our results show that the impact of communication is more profound in problem with higher complexity. Specifically, the complexity of the objective function plays a crucial role on when agents stop searching. For relatively simple functions, early stops in a few local regions do not significantly influence other agents' performance. In particular, in the area where the global optimum is located, if the agent in that area does not stop early, it can continue the search of its local region even with less information acquired from other areas where agents stop early, thus successfully converging to the global optimum. However, when dealing with more complex objective functions, the contribution of each agent becomes indispensable to effectiveness and the possibility of achieving the global optimum.

As an illustration, in both the cosines and Michalewicz functions, it appears that agent 1's information is neglectable. When this agent stops sampling, not only is the convergence unaffected but also it enhances the convergence speed in certain cases. This can be observed with the cosines function, in which the convergence speed is improved from the 13th step to the 11th step, as presented in Table 4. Similarly, the convergence in the Michalewicz function is improved from the 18th step to the 13th, as shown in Table 7. In contrast, the case of the Eggholder function with many local optima demonstrates how vital every agent's information is for convergence. Specifically, if agent 2 stops within the first ten steps, the system fails to converge, as highlighted in Tables 10 and 12. However, when agent 2 continues its search, sharing its information with other agents, the system can achieve convergence (see Table 12). These observations shed light on the formation of a design team. For a design team, communication mechanisms and incentive structures for solution search shall be designed and tailored according to the complexity of the problem to be solved.

(3) Our results also indicate the delicate balance between the value of design space exploration and cost of design iterations. In practical applications of design optimization, the concepts of "gain" and "cost" are pivotal. "Gain" typically refers to the benefits or improvements achieved through the design process. This could be in terms of performance, efficiency, utility, or any other metric of value in a specific context. For instance, in architectural design optimization, the "gain" might be maximized living space, energy efficiency, or aesthetic appeal. On the other hand, the "cost" in design optimization encapsulates the resources expended to achieve these gains. Beyond just monetary expenditures, the cost could represent the time taken, the manpower used, the environmental impact, or any other resource consumed in the process. In our earlier architectural example, the "cost" could be construction time, materials used, or even the environmental toll of sourcing those materials. Understanding and measuring gain and cost within the same unit or system is crucial. This ensures that we are making evaluations and decisions based on a consistent frame of reference. In real-world scenarios, this consistency assists stakeholders in making informed choices, prioritizing where to allocate resources, and understanding the trade-offs involved in pursuing specific design objectives.

7 Conclusions

This article develops a model centered around a cost-aware MAS to study the impact of search cost on collaboration between multiple rational agents in design space exploration. This process involves three key decisions: where to sample, where not to sample, and when to stop. We leveraged MABO to determine optimal solutions while enabling a global-local communication mechanism for information exchange among agents, thus accelerating the speed of finding the best solutions. Additionally, multi-agent reinforcement learning (MARL) allows agents to recognize and adapt to unfeasible regions autonomously during the sampling process. We also propose a cost-aware stopping criterion, constructing each agent's utility (also known as payoff) that incorporates PG and IG. The experimental findings reveal that cost significantly impacts the communication and performance of MAS in complex design problems more than in simpler ones. In complex scenarios, agents may exhibit low performance initially, leading to minimal gains and potentially terminating search efforts due to negative payoffs. These early terminations, driven by initial negative outcomes, can substantially affect the overall system performance. Consequently, for a design team, it becomes essential to design and customize incentive structures to find solutions according to the specifics of the design problem to be solved. This insight, derived from a theoretical model of how rational design agents should work with a rigorous mathematical foundation, serves as a novel benchmark. It can be used in empirical studies to quantitatively evaluate how humans actually make design decisions within a team setting.

Our findings are subject to the following limitations. First, we used the fixed number of agents in the MAS, which means that we did not test the impact of the scalability of the MAS on the cost-aware strategy. In future work, we plan to dive into studying how the size of an MAS affects cost strategies, with the objective of identifying how different team sizes can change the efficiency and financial planning of design projects. Second, in this article, we adopted the global evaluator in BO for information exchange, which means that the agents in the MAS are fully connected by the global evaluator. In our future work, we will explore the topology of communication and team structures within distributed MAS, trying to understand how various structures might influence system efficiency and reliability. A key goal will be to create a model for MAS that reflects the varied preferences of individual agents, such as different attitudes toward risk, to model heterogeneity within human design teams. Third, this article initiates by partitioning the design space, ensuring that agents operate in distinct regions and maintaining their independence. In our future work, another strategy could be adopted to set overlapping boundaries between agents, using probability to choose who can sample this overlapping region or comparing the improvement of two agents in the same overlapping region. Fourth, we conclude that incentive structures to find solutions should be customized according to the specifics of the design problem to be solved. However, we did not examine how incentive structures influence the performance of the MAS. In future work, we intend to examine how different initial funds or budgets impact the performance of our MAS, especially in systems that represent varied agent preferences and decisions. Finally, to ensure that the insights about effective team coordination obtained from this study are grounded in reality beyond theoretical simulations, more validation is needed. This involves extensive testing in real-world DSE scenarios, demonstrating the practical applicability and effectiveness of our approach in cost management in design teams. With the aim of examining the prescriptive feature of the proposed model, we plan to conduct human-subject experiments to collect human behavior data in collaborative DSE and compare their actual behaviors (i.e., when to stop and what the next point sampled) against those predicted by the MABO model. One potential value of such a comparison is a measure of design irrationality quantified by the distance between the empirical data and the simulation results.

Funding Data

- The National Science Foundation (Grant Nos. CMMI-2321463 and CMMI-2419423).

Conflict of Interest

There are no conflicts of interest.

Data Availability Statement

The datasets generated and supporting the findings of this article are obtainable from the corresponding author upon reasonable request.

References

[1] Kang, E., Jackson, E., and Schulte, W., 2011, "An Approach for Effective Design Space Exploration," Foundations of Computer Software. Modeling, Development, and Verification of Adaptive Systems: 16th Monterey Workshop 2010, Redmond, WA, Mar. 31–Apr. 2, Revised Selected Papers 16, Springer, pp. 33–54.

[2] Marcin, A., Denil, M., Gomez, S., Hoffman, M. W., Pfau, D., Schaul, T., Shillingford, B., and De Freitas, N., 2016, "Learning to Learn by Gradient Descent by Gradient Descent," Adv. Neural Infor. Process. Syst. , **29**.

[3] Bottou, L., 2010, "Large-Scale Machine Learning With Stochastic Gradient Descent"," Proceedings of COMPSTAT'2010: 19th International Conference on Computational Statistics, Paris France, Aug. 22–27, Keynote, Invited and Contributed Papers, Springer, pp. 177–186.

[4] Zijun, J., 2018, "Improved Adam Optimizer for Deep Neural Networks," 2018 IEEE/ACM 26th international Symposium on Quality of Service (IWQoS), pp. 1–2. IEEE.

[5] Bajaj, I., Arora, A., and Hasan, M. F., 2021, "Black-Box Optimization: Methods and Applications," *Black Box Optimization, Machine Learning, and No-Free Lunch Theorems*, P. M. Pardalos, V. Rasskazova, and M. N. Vrahatis, eds., Springer, Cham, Switzerland, pp. 35–65.

[6] Shahriari, B., Swersky, K., Wang, Z., Adams, R. P., and De Freitas, N., 2015, "Taking the Human Out of the Loop: A Review of Bayesian Optimization," Proc. IEEE, **104**(1), pp. 148–175.

[7] Gelbart, M. A., Snoek, J., and Adams, R. P., 2014, "Bayesian Optimization With Unknown Constraints," 30th Conference on Uncertainty in Artificial Intelligence, UAI 2014, AUAI Press, pp. 250–259.

[8] Panchal, J. H., Sha, Z., and Kannan, K. N., 2017, "Understanding Design Decisions Under Competition Using Games With Information Acquisition and a Behavioral Experiment," ASME J. Mech. Des., **139**(9), p. 091402.

[9] Cao, F., Zhu, M. M., and Ding, D., 2013, "Distributed Workflow Scheduling Under Throughput and Budget Constraints in Grid Environments," Workshop on Job Scheduling Strategies for Parallel Processing, Boston, MA, May 24, Springer, pp. 62–80.

[10] Yu, J., and Buyya, R., 2006, "A Budget Constrained Scheduling of Workflow Applications on Utility Grids Using Genetic Algorithms," 2006 Workshop on Workflows in Support of Large-Scale Science, Paris, France, June 19, IEEE, pp. 1–10.

[11] Zhu, H., 2017, "Maximizing Group Performance While Minimizing Budget," IEEE. Trans. Syst. Man. Cybernet.: Syst., **50**(2), pp. 633–645.

[12] Chen, S., Bayrak, A. E., and Sha, Z., 2023, "Multi-agent Bayesian Optimization for Unknown Design Space Exploration," International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Boston, MA, Aug. 20–23, American Society of Mechanical Engineers, Vol. 87318, p. V03BT03A047.

[13] Lowe, R., Wu, Y. I., Tamar, A., Harb, J., Pieter Abbeel, O., and Mordatch, I., 2017, "Multi-agent Actor-Critic for Mixed Cooperative-Competitive Environments," Advances in Neural Information Processing Systems, Long Beach, CA, Dec. 4–9, Vol. 30, pp. 6379–6390.

[14] Snoek, J., Larochelle, H., and Adams, R. P., 2012, "Practical Bayesian Optimization of Machine Learning Algorithms," Advances in Neural Information Processing Systems, Lake Tahoe, NV, Dec. 3–6, Vol. 25, pp. 2951–2959.

[15] Kontar, R., Shi, N., Yue, X., Chung, S., Byon, E., Chowdhury, M., Jin, J., Kontar, W., Masoud, N., Nouiehed, M., and Okwudire, C. E., 2021, "The Internet of Federated Things (IoFT)," IEEE Access, **9**(1), p. 156071.

[16] Peralta, F., Reina, D. G., and Toral, S., 2023, "Water Quality Online Modeling Using Multi-objective and Multi-agent Bayesian Optimization With Region Partitioning," Mechatronics, **91**, p. 102953.

[17] Gramacy, R. B., Gray, G. A., Le Digabel, S., Lee, H. K., Ranjan, P., Wells, G., and Wild, S. M., 2014, *Modeling an Augmented Lagrangian for Improved Blackbox Constrained Optimization*, GERAD HEC Montréal, Canada.

[18] Gramacy, R. B., Gray, G. A., Le Digabel, S., Lee, H. K., Ranjan, P., Wells, G., and Wild, S. M., 2016, "Modeling an Augmented Lagrangian for Blackbox Constrained Optimization," Technometrics, **58**(1), pp. 1–11.

[19] Tran, A., Eldred, M., McCann, S., and Wang, Y., 2020, "srMO-BO-3GP: A Sequential Regularized Multi-objective Constrained Bayesian Optimization for Design Applications," International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Vol. 83983, Virtual Online, Aug. 17–20, American Society of Mechanical Engineers, p. V009T09A015.

[20] Letham, B., Karrer, B., Ottoni, G., and Bakshy, E., 2019, "Constrained Bayesian Optimization With Noisy Experiments." Bayesian Analysis, **14**(2), pp. 495–519.

[21] Bernardo, J., Bayarri, M., Berger, J., Dawid, A., Heckerman, D., Smith, A., and West, M., 2011, "Optimization Under Unknown Constraints," Bayesian Statistics, **9**(9), p. 229. .

[22] Freriks, H., Heemels, W., Muller, G., and Sandee, J., 2006, "5.3. 2 on the Systematic Use of Budget-Based Design: Sixteenth Annual International Symposium of the International Council on Systems Engineering (INCOSE)," INCOSE International Symposium, Orlando, FL, July 8–14, Vol. 16, Wiley Online Library, pp. 788–803.

[23] Wertz, J. R., Larson, W. J., Kirkpatrick, D., and Klungle, D., 1999, *Space Mission Analysis and Design*, Vol. 8, Springer, Torrance, CA.

[24] Jones, D. R., Schonlau, M., and Welch, W. J., 1998, "Efficient Global Optimization of Expensive Black-Box Functions," J. Global Optim., **13**(4), pp. 455–492.

[25] Pandita, P., Bilionis, I., and Panchal, J., 2016, "Extending Expected Improvement for High-Dimensional Stochastic Optimization of Expensive Black-Box Functions," ASME J. Mech. Des., **138**(11), p. 111412.

[26] Moore, R. A., Romero, D. A., and Paredis, C. J., 2014, "Value-Based Global Optimization," ASME J. Mech. Des., **136**(4), p. 041003.

[27] Lorenz, R., Monti, R. P., Violante, I. R., Faisal, A. A., Anagnostopoulos, C., Leech, R., and Montana, G., 2015, "Stopping Criteria for Boosting Automatic Experimental Design Using Real-Time FMRI With Bayesian Optimization," arXiv preprint arXiv:1511.07827.

[28] McLeod, M., Roberts, S., and Osborne, M. A., 2018, "Optimization, Fast and Slow: Optimally Switching Between Local and Bayesian Optimization," International Conference on Machine Learning, Stockholm, Sweden, July 10–15, PMLR, pp. 3443–3452.

[29] Chaudhari, A. M., Bilionis, I., and Panchal, J. H., 2018, "How Do Designers Choose Among Multiple Noisy Information Sources in Engineering Design Optimization? An Experimental Study," International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, Vol. 51753, Quebec, Canada, Aug. 26–29, American Society of Mechanical Engineers, p. V02AT03A021.

[30] Frazier, P. I., 2018, "Bayesian Optimization," *Recent Advances in Optimization and Modeling of Contemporary Problems*, Informs, pp. 255–278.

[31] Rasmussen, C. E., 2003, "Gaussian Processes in Machine Learning," *Summer School on Machine Learning*, B. Schölkopf and M. K. Warmuth, eds., Springer, Berlin, Germany, pp. 63–71.

[32] Chaudhari, A. M., Bilionis, I., and Panchal, J. H., 2020, "Descriptive Models of Sequential Decisions in Engineering Design: An Experimental Study," ASME J. Mech. Des., **142**(8), p. 081704.

[33] Srinivas, N., Krause, A., Kakade, S. M., and Seeger, M., 2009, "Gaussian Process Optimization in the Bandit Setting: No Regret and Experimental Design," Proceedings of the 27th International Conference on Machine Learning, Omnipress, pp. 1015–1022.

[34] Silver, D., Lever, G., Heess, N., Degris, T., Wierstra, D., and Riedmiller, M., 2014, "Deterministic Policy Gradient Algorithms," International Conference on Machine Learning, Beijing, China, June 21–26, PMLR, pp. 387–395.

[35] Agrawal, A., and McComb, C., 2023, "Reinforcement Learning for Efficient Design Space Exploration With Variable Fidelity Analysis Models," ASME J. Comput. Inf. Sci. Eng., **23**(4), p. 041004.

[36] González, J., Dai, Z., Hennig, P., and Lawrence, N., 2016, "Batch Bayesian Optimization via Local Penalization," Artificial Intelligence and Statistics, Cadiz, Spain, May 9–11, PMLR, pp. 648–657.

[37] Kaipa, K. N., and Ghose, D., 2017, "Multimodal Function Optimization," *Glowworm Swarm Optimization. Studies in Computational Intelligence*, vol 698. Springer, Cham.

[38] Floudas, C. A., Pardalos, P. M., Adjiman, C., Esposito, W. R., Gümüs, Z. H., Harding, S. T., Klepeis, J. L., Meyer, C. A., and Schweiger, C. A., 2013, "Certified Global Minima for a Benchmark of Difficult Optimization Problems," *Handbook of Test Problems in Local and Global Optimization*. Vol. 33. Springer Science & Business Media.