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

ARCO-BO: Adaptive Resource-Aware COllaborative Bayesian Optimization for Heterogeneous Multi-Agent Design Optimization

Zihan Wang
Department of Mechanical Engineering,
Northwestern University,
Evanston, IL 60208
e-mail: zihan.wang5@northwestern.edu

Yi-Ping Chen
Department of Mechanical Engineering,
Northwestern University,
Evanston, IL 60208
e-mail: chenyp@u.northwestern.edu

Tuba Dolar
Department of Mechanical Engineering,
Northwestern University,
Evanston, IL 60208
e-mail: tubadolar@u.northwestern.edu

Wei Chen[1]
Department of Mechanical Engineering,
Northwestern University,
Evanston, IL 60208
e-mail: weichen@northwestern.edu

Many real-world optimization problems in scientific discovery and engineering design optimization involve multiple design evaluation sources, such as simulators, experiments, or manufacturing sites, operate independently, and evaluate different functions of the same underlying objective (quantity of interest). In this work, we refer to these design evaluation sources as agents. Such agents often differ in their objective function mappings, evaluation budgets, and accessible optimization variables, which complicate coordination and information sharing. Bayesian optimization (BO) is a widely used framework for expensive blackbox optimization, yet its standard single-agent formulation assumes centralized control and full data sharing. Recent collaborative BO methods relax these assumptions but still rely on uniform resources, fully shared input spaces, and closely aligned tasks, and these requirements are seldom met in real applications. To address these limitations, we introduce adaptive resource-aware collaborative Bayesian optimization (ARCO-BO), a framework that explicitly accounts for heterogeneity in multi-agent optimization. ARCO-BO integrates three key components: a similarity- and optimal-location-aware consensus mechanism for adaptive information sharing, a budget-aware asynchronous sampling strategy for resource coordination, and a partial input-space sharing scheme for heterogeneous optimization variables. Experiments on synthetic benchmarks and high-dimensional engineering optimization problems demonstrate that ARCO-BO consistently outperforms independent BO and the existing consensus-based collaborative BO, achieving robust and efficient performance in complex heterogeneous multi-agent optimization settings.
[DOI: 10.1115/1.4071073]

Keywords: Bayesian optimization, collaborative optimization, resource-aware optimization, multi-agent systems, task similarity

1 Introduction

In decentralized design optimization, multiple design evaluation sources operate independently and hold different views of the same underlying system. We refer to each of these heterogeneous evaluation sources as an agent. Many real-world optimization problems in scientific discovery and engineering design optimization involve such agents working toward a shared underlying objective (quantity of interest), while differing in their evaluation processes, resource availability, and accessible design spaces. In multifidelity engineering design optimization [1–3], for example, low- and high-fidelity simulators target the same physical phenomenon, yet differences in model resolution, numerical approximation, and cost result in distinct objective function landscapes corresponding to a shared underlying quantity of interest. Simulators can be queried many times when computationally feasible, whereas experiments often permit only a limited number of evaluations due to time, expense, or material constraints. The accessible design inputs may also differ, since some variables cannot be measured or controlled experimentally, while simulations may include idealized or unphysical parameters. Similar heterogeneity arises in experiment-driven materials discovery [4–8], where laboratories often use different instruments, environmental conditions, or measurement protocols. As a result, they probe the same high-level structure–property relationship, yet induce distinct objective function mappings of the same underlying quantity of interest, and their accessible regions of the design space may only partially overlap. A comparable situation occurs in decentralized manufacturing [9–11], where facilities implement nominally similar process recipes but differ in equipment settings, operating environments, or calibration factors, leading to variations in local performance metrics even when sharing as many controllable parameters as possible. This inherent variability creates a unifying challenge: *agents aim to improve performance on a shared underlying system while contending with differences arising from their distinct data sources, operational environments, and evaluation processes.*

[1]Corresponding author.

Contributed by the Design Automation Committee of ASME for publication in the Journal of Mechanical Design. Manuscript received October 7, 2025; final manuscript received January 26, 2026; published online March 9, 2026. Assoc. Editor: Jitesh H. Panchal.

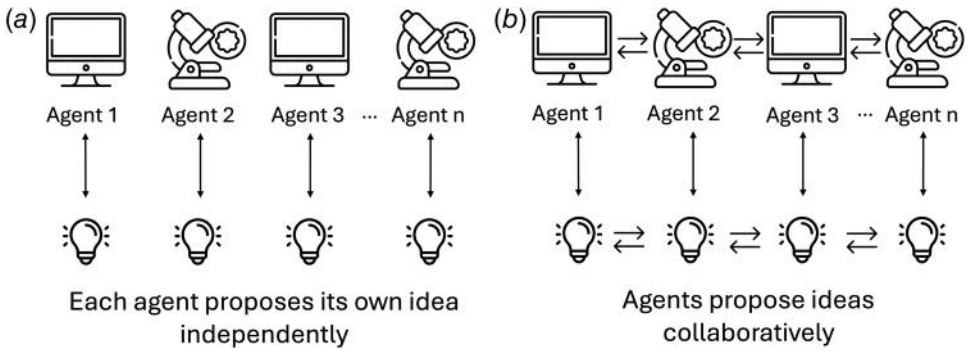

Fig. 1 Conceptual illustration of a multi-agent design optimization: (*a*) without collaboration, agents rely solely on their own information and (*b*) with collaboration, agents share information to accelerate optimization

These application settings motivate a multi-agent design optimization perspective [12,13], in which agents optimize different objective function mappings of a shared underlying objective while leveraging useful information from one another. Building on the forms of heterogeneity described above, we distinguish three categories that commonly arise in practice: *function heterogeneity*, where agents target a common underlying objective but are associated with a distinct objective function [14]; *budget heterogeneity*, where evaluation costs differ and lead to unequal query capacities; and *input-space heterogeneity*, where agents access overlapping but nonidentical regions of the design domain due to measurement limits or model-specific constraints. In this work, we emphasize cooperation over competition, enabling agents to coordinate and accelerate learning despite heterogeneity. Figure 1 illustrates this perspective by contrasting independent optimization, where agents operate in isolation, with collaborative optimization, where agents selectively share information to accelerate learning despite heterogeneity.

A broad range of methods has been developed for coordinating multiple entities in optimization and decision-making systems [15–25]. Classical distributed optimization methods [15–17] assume that all participating nodes collectively solve a single global objective function, typically through decomposition and coordinated local updates such as distributed gradient descent or the alternating direction method of multipliers. Consensus-based approaches [18–21] similarly align shared variables so that all nodes converge to a common system-level solution. Game-theoretic and agent-based frameworks [22–25] also operate in shared environments where interactions among decision-makers influence a unified objective function or equilibrium outcome. While powerful, these approaches generally assume a common optimization goal or explicit interaction models and rely on gradient information, making them unsuitable for costly blackbox evaluations or for scenarios where evaluation sources do not share the same objective function. Recent developments in multi-actor reinforcement learning [26–28] likewise emphasize coordinated behavior under a shared environment, but their focus on reward maximization differs from the sample-efficient exploration required for expensive blackbox optimization with limited evaluation budgets. In contrast, our setting involves autonomous evaluation sources, which we refer to as agents, that do not jointly minimize a single global objective function. Each agent optimizes its own objective function mapping of a shared underlying objective, shaped by differences in fidelity, experimental conditions, or operating environments, while sharing only a subset of design variables with others. Because naive search strategies and ad hoc heuristics are not sample-efficient in such heterogeneous and resource-constrained settings, an adaptive and uncertainty-aware approach is needed to guide evaluations effectively.

Bayesian optimization (BO) has been a popular effective adaptive learning framework for data-efficient decision-making in scenarios where evaluating objective functions is expensive [29–31]. BO leverages probabilistic surrogate models such as Gaussian processes (GPs) [32] to estimate uncertainty and guide exploration through acquisition functions that balance exploration and exploitation to efficiently locate global optima with minimal samples. However, standard BO frameworks inherently assume a centralized, single-agent setup in which all evaluations and optimization decisions are controlled by a unified model operating over the entire design space. This assumption limits the applicability of traditional BO in decentralized and collaborative environments where optimization tasks are performed by multiple agents or in a parallelized setup.

This gap motivates the shift to collaborative Bayesian optimization, which extends BO from a single-agent setting to a distributed multi-agent scenario in which multiple agents optimize concurrently and selectively share information. Collaborative BO [11,33,34] introduces a new paradigm for multi-agent design optimization by enabling agents to share surrogate information and/or coordinate sampling decisions *without requiring centralized control or full data sharing*. The collaborative BO strategy proposed by Chen et al. [33] introduces constrained Gaussian process surrogates that enable agents to leverage informative evaluations discovered by high-performing collaborators. Rather than treating each agent's optimization process in isolation, this method facilitates cross-agent knowledge sharing by selectively incorporating promising designs observed by others, subject to compatibility constraints. Another collaboration strategy proposed by Zhan et al. [34] adopts a federated optimization paradigm, where each agent trains its own model using its local data and then shares only certain parameters (like gradients or parts of the model) with others. By doing this, the agents work together to build a shared global model without directly sharing their raw data. More recently, consensus-based collaborative BO methods have been proposed [11] in which agents agree on their next-to-sample designs by sharing their local designs, but not design performances.

In parallel, emerging multisource data fusion methods [2,35–37] offer an alternative paradigm for knowledge sharing through unified surrogate models. A representative example is the latent variable Gaussian process (LVGP) [36], which embeds discrete or high-dimensional variables into a continuous latent space to capture shared structures across heterogeneous tasks. In multifidelity optimization, LVGP has demonstrated strong potential for capturing shared structures and correlations across tasks or fidelity levels by implicitly learning task similarities through latent embeddings [2]. This enables effective information transfer across heterogeneous sources [37], thereby improving sample efficiency. Nevertheless, unlike collaborative BO frameworks that safeguard agent-level privacy by avoiding the exchange of true evaluation outcomes, LVGP requires access to both the input variables and the observed outputs from all agents in order to train a unified surrogate model. Additionally, the joint optimization of latent embeddings and hyperparameters may become computationally intensive

as the number of agents increases, limiting LVGP's scalability and applicability in many-agent systems.

Although these advances mark important progress in collaborative optimization, extending them to accommodate the heterogeneity observed in real-world applications remains a substantial challenge. Prior work has generally considered only limited aspects of heterogeneity, without providing a unified framework that addresses all simultaneously. In this article, we propose adaptive resource-aware collaborative Bayesian optimization (ARCO-BO), a framework designed for multi-agent optimization in heterogeneous settings that overcomes the limitations of existing approaches. We use the term *collaboration* to denote a cooperative multi-agent scenario in which agents seek to improve their own objective function mappings of a shared underlying objective while benefiting from one another's information. As existing collaborative BO methods typically assume shared objective functions, uniform resource budgets, or fully overlapping input spaces, ARCO-BO explicitly tackles three critical forms of heterogeneity: differing objective functions, asymmetric evaluation budgets, and partially shared input domains. ARCO-BO introduces three key capabilities. First, it employs an adaptive, similarity- and an optimal-location-aware consensus mechanism that modulates information sharing according to surrogate model alignment, enabling selective and dynamic collaboration among agents. Second, it implements a resource-aware sampling strategy that schedules agent participation based on their available evaluation budgets, promoting efficient and balanced exploration. Third, it enables collaboration over partially shared input spaces, allowing agents to coordinate on common variables while independently optimizing their private inputs. We present proof-of-concept numerical studies rather than domain-specific engineering deployments. We evaluate ARCO-BO on a suite of low-dimensional illustrative functions and high-dimensional synthetic design optimization problems, which serve as controlled abstractions of real engineering settings. These benchmarks enable reproducible analysis of algorithmic behavior while capturing key characteristics of heterogeneous multi-agent systems. We compare the proposed ARCO-BO's performance to two baselines: (1) standard Bayesian optimization with independent agents and (2) conventional consensus-based collaborative BO with static update rules [11]. Our results show that ARCO-BO consistently achieves superior design quality and faster convergence across all tested scenarios.

The contributions of this work are summarized as follows:

- We introduce ARCO-BO, a novel framework for multi-agent Bayesian optimization under heterogeneous settings, including function heterogeneity, asymmetric evaluation budgets, and partially shared input spaces.
- We develop three key algorithmic components: a similarity- and optimal-location-aware consensus mechanism for dynamic information sharing, a budget-aware sampling strategy to coordinate evaluations based on remaining agent resources, and a partially shared input model to handle heterogeneous design spaces across agents.
- We demonstrate that ARCO-BO consistently outperforms both independent optimization and static-consensus baselines on synthetic benchmarks and high-dimensional engineering design optimization problems.

The remainder of this article is organized as follows. Section 2 reviews the technical background, including the existing Bayesian optimization and collaborative Bayesian optimization via consensus. Section 3 introduces the proposed ARCO-BO framework, detailing its adaptive consensus mechanism, resource-aware sampling strategy, and support for partially shared input spaces. Section 4 outlines the setup of numerical studies, including evaluation metrics and computational configurations. Section 5 presents and analyzes the results, comparing ARCO-BO to baseline methods across diverse multi-agent optimization scenarios. Section 6 concludes the article and discusses potential directions for future research.

2 Technical Background

This section introduces the foundational concepts that motivate and inform our proposed ARCO-BO. We begin with a review of BO, followed by its extension to collaborative multi-agent settings via consensus-based collaborative BO. These components highlight the heterogeneity and coordination challenges that ARCO-BO is designed to address.

2.1 Bayesian Optimization. BO [29,38–40] has emerged as a powerful framework for the global optimization of expensive, blackbox functions. In these settings, the objective function f is not analytically accessible and can only be evaluated through potentially noisy observations of the form $y = f(\mathbf{x}) + \varepsilon$, where $\varepsilon \sim \mathcal{N}(0, \sigma^2)$ denotes independent Gaussian noise. The goal is to identify the global optimal solution of the function:

$$\mathbf{x}^* \in \arg\min_{\mathbf{x}} f(\mathbf{x}) \tag{1}$$

with $\mathbf{x}$ residing in a d-dimensional design space. A central challenge in BO lies in the high cost of function evaluations, which severely limits the number of allowable queries. As a result, exhaustive search strategies become impractical, and conventional optimization methods often prove inefficient. BO addresses this challenge by leveraging a probabilistic surrogate model to approximate the unknown objective function. This model guides the selection of future query points in a way that strategically balances exploration of uncertain regions with exploitation of areas likely to yield high performance. BO relies on two fundamental components: a surrogate model (typically a Gaussian process) and an acquisition function that quantifies the utility of sampling each candidate point.

The GP defines a distribution over functions and is characterized by a mean function $\mu(\cdot)$ and a covariance (kernel) function $k(\cdot, \cdot)$:

$$f(\mathbf{x}) \sim \mathcal{GP}\big(\mu(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')\big) \tag{2}$$

Given an observed dataset $\mathcal{D} = \{(\mathbf{x}^{(n)}, y^{(n)})\}_{n=1}^{N}$, the posterior distribution of f at a set of new input locations $\mathbf{X}_* = [\mathbf{x}_*^{(1)}, \ldots, \mathbf{x}_*^{(m)}]$ is

$$f(\mathbf{X}_*) \mid \mathcal{D} \sim \mathcal{N}\big(\mu_n(\mathbf{X}_*), \Sigma_n(\mathbf{X}_*, \mathbf{X}_*)\big) \tag{3}$$

where μ_n and Σ_n denote the posterior mean and covariance functions, respectively.

An acquisition function is a heuristic used to select the next sampling point. It takes the GP posterior as input and balances exploration and exploitation. Common acquisition functions [29,31,41] include probability of improvement, expected improvement (EI), lower confidence bound, and Thompson sampling. We adopt EI as a well-established baseline acquisition function with well-understood behavior. The proposed framework is acquisition-agnostic: it operates by reconciling the candidate input locations proposed by individual agents through consensus on shared variables, rather than by introducing a new acquisition strategy. Consequently, it can be combined with alternative acquisition functions without modification.

Assuming we have a minimization problem, the improvement utility function is

$$u(\mathbf{x}) = \max(0, f^* - f(\mathbf{x})) \tag{4}$$

where f^* is the incumbent, i.e., the best (minimum) observed value so far. EI is formulated as

$$\begin{aligned} \mathcal{A}_{\mathrm{EI}}(\mathbf{x}) &= \mathbb{E}[u(\mathbf{x}) \mid \mathcal{D}] \\ &= (f^* - \mu(\mathbf{x}))\Phi\left(\frac{f^* - \mu(\mathbf{x})}{\sigma(\mathbf{x})}\right) + \sigma(\mathbf{x})\phi\left(\frac{f^* - \mu(\mathbf{x})}{\sigma(\mathbf{x})}\right) \end{aligned} \tag{5}$$

where $\Phi(\cdot)$ and $\phi(\cdot)$ are the cumulative distribution function (CDF) and probability density function (PDF) of the standard normal

distribution, and $\mu(\mathbf{x})$ and $\sigma^2(\mathbf{x})$ are the GP posterior mean and variance. The next sampling point is chosen by maximizing $\mathcal{A}_{\mathrm{EI}}(\mathbf{x})$.

2.2 Collaborative Bayesian Optimization Via Consensus. As conventional BO is typically formulated for a single-agent setting, collaborative BO [11,33,34,42] generalizes the BO framework to accommodate these multi-agent scenarios by allowing agents to share information on the query samples to improve convergence efficiency.

In the consensus-based Collabroative Bayesian Optimization (CBO) framework proposed by Yue et al. [11], each agent $i \in \{1, \ldots, K\}$ maintains its own surrogate model and acquisition function $\mathcal{A}_i$, selecting its candidate design point at iteration t as

$$\mathbf{x}_i^{(t)} \in \arg\max_{\mathbf{x}} \mathbb{E}_{\hat{f}_i|\mathcal{D}_i^{(t)}}\left[\mathcal{A}_i(\hat{f}_i(\mathbf{x}))\right] \tag{6}$$

where $\hat{f}_i$ is the surrogate model fitted on agent i's dataset $\mathcal{D}_i^{(t)}$.

However, instead of immediately evaluating these proposals, the algorithm introduces a consensus step to enable collaboration. At each iteration, agents exchange only their proposed values for the shared variables and update them through a weighted averaging scheme defined by a row-stochastic consensus matrix $\mathbf{W}^{(t)} \in \mathbb{R}^{K\times K}$. Specifically, the shared variables for each agent are updated as

$$\mathbf{x}_i^{(t+1)} = \sum_{j=1}^{K} \mathbf{W}_{ij}^{(t)} \mathbf{x}_j^{(t)}, \quad i = 1, \ldots, K \tag{7}$$

while the agent-specific private variables remain unchanged. The resulting consensus-informed design vector for agent i is then formed by combining the updated shared variables with its original private variables.

Figure 2 illustrates this operation for a two-agent system, in which each agent proposes a candidate design independently and then updates its shared variables via consensus, blending local proposals with information from the other agent to coordinate over the shared design subspace.

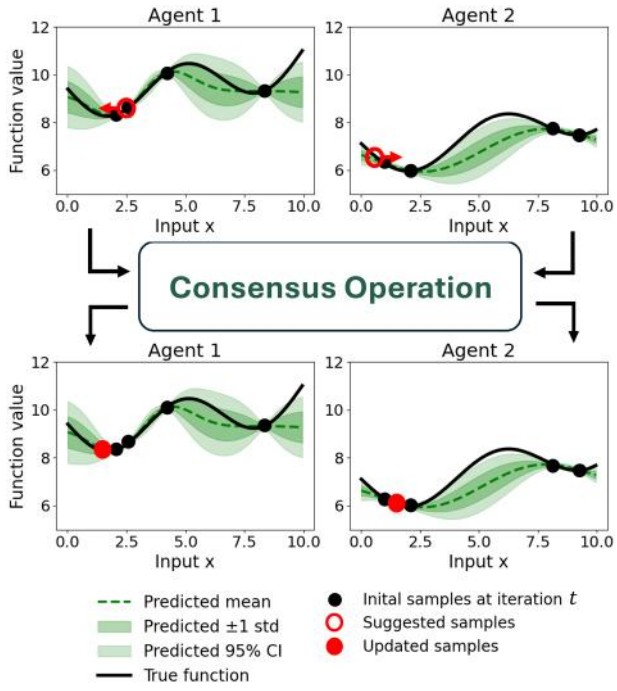

Fig. 2 Consensus operation in collaborative Bayesian optimization. Agents first propose samples independently, then update their decisions via a weighted consensus based on the matrix $\mathbf{W}^{(0)}$.

Collaboration starts with full uniform averaging and gradually transitions toward independent optimization. Specifically, the consensus matrix is initialized as

$$\mathbf{W}^{(0)} = \frac{1}{K}\mathbf{1}_{K\times K} \tag{8}$$

where $\mathbf{1}_{K\times K}$ is a matrix of ones, representing equal weighting of all agents' proposals when initialized. As optimization progresses, the consensus matrix evolves according to

$$\mathbf{W}^{(t+1)} = \mathbf{W}^{(t)} + \frac{1}{TK}[(K-1)\mathbf{I}_K - \mathbf{1}_{K\times K}] \tag{9}$$

where T denotes the total number of iterations. This update increases the diagonal entries $\mathbf{W}_{ii}^{(t)}$ while decreasing the off-diagonal entries $\mathbf{W}_{ij}^{(t)}$ for $i \neq j$, thereby progressively reducing the influence of other agents' shared-variable proposals. In the limit, $\mathbf{W}^{(t)}$ converges to the identity matrix $\mathbf{I}_K$, enabling agents to focus solely on their own objective functions in later stages. The intuition behind this transitional design optimization is that in early optimization stages, each agent's surrogate model is trained on limited data and benefits from the information pooled from other agents. As each agent accumulates more observations and constructs higher-quality surrogate models, the optimization naturally shifts toward agent-specific refinement.

This consensus-based collaborative BO framework improves sample efficiency by coordinating exploration across agents while preserving task-specific goals. However, it relies on restrictive assumptions: it presumes that agents' objective function mappings of a shared underlying objective are well aligned, with optima located in similar regions of the design space, thereby overlooking broader forms of functional heterogeneity. When these mappings are weakly correlated or poorly aligned, uniform information sharing can lead to negative transfer and slow convergence. As shown in Fig. 3, which depicts a three-agent system, agents 2 and 3 are highly correlated ($r = 0.77$), whereas agents 1 and 3 are nearly uncorrelated ($r = -0.14$) with distinct optima. The framework further assumes equal evaluation budgets, ignoring resource asymmetries that lead to inefficient sampling and unbalanced contributions. It also neglects partially shared input spaces, where coordination is possible only over subsets of design variables. These limitations highlight the need for a more flexible, resource-aware, and heterogeneity-adaptive framework for decentralized multi-agent design optimization.

3 Adaptive Resource-Aware COllaborative Bayesian Optimization (ARCO-BO)

We propose an ARCO-BO framework to enhance collaborative BO in the presence of task heterogeneity, resource disparity, and partially shared input spaces. ARCO-BO extends the consensus-based collaborative BO by introducing three key mechanisms: (1) similarity and optimal-location-aware dynamic consensus weighting, which promotes targeted knowledge exchange by adaptively weighting contributions from agents whose objective function mappings of the shared underlying objective are well aligned. (2) Budget-aware asynchronous sampling, which adjusts each agent's sampling frequency based on its evaluation budget, ensuring efficient resource utilization. (3) Partial input-space sharing, which allows agents to collaborate even when their design spaces only partially overlap. The overall workflow of ARCO-BO is summarized in Algorithm.

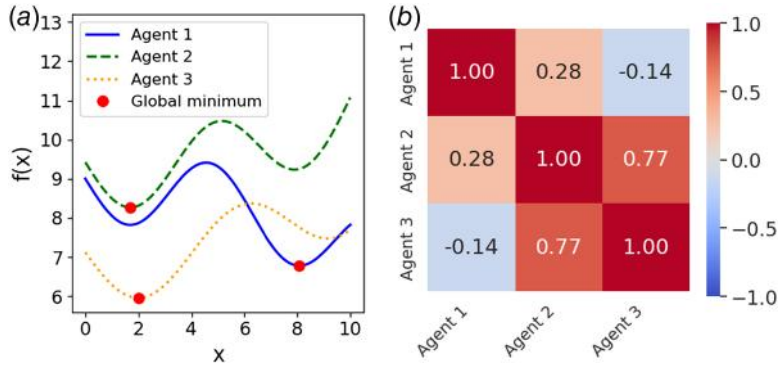

Fig. 3 Illustration of function heterogeneity in a three-agent system. (*a*) Agent-specific objective functions $f_i(\mathbf{x})$, whose global minima lie in different regions of the domain. (*b*) Pairwise similarity shown as a correlation matrix.

Algorithm 1. ARCO-BO

Given (definitions):

- K: number of agents.
- For each agent $i \in \{1, \ldots, K\}$:
  - $\mathcal{D}_i = \{(\mathbf{x}_{ij}, y_{ij})\}_{j=1}^{n_i}$: initial dataset.
  - f_i: objective function; B_i: evaluation budget.
  - GP_i: Gaussian process surrogate model fit to $\mathcal{D}_i$.
  - Agent decision vector

$$\mathbf{x}_i = \begin{bmatrix} \mathbf{x}_{i,\text{shared}} \\ \mathbf{x}_{i,\text{private}} \end{bmatrix} \in \mathbb{R}^d, \quad d_i = d_s + d_p,$$

    where $\mathbf{x}_{i,\text{shared}} \in \mathbb{R}^{d_s}$ denotes shared variables and $\mathbf{x}_{i,\text{private}} \in \mathbb{R}^{d_p}$ denotes agent-specific private variables.
  - $\mathcal{A}_i(\cdot\,; \mathrm{GP}_i)$: acquisition function for agent i.
- T: maximum number of algorithm iterations.
- $\mathbf{S} \in \mathbb{R}^{K\times K}$: surrogate model similarity matrix.
- $\boldsymbol{\Omega} \in \mathbb{R}^{K\times K}$: row-stochastic consensus weight matrix derived from $\mathbf{S}$.

Goal: Minimize each f_i under budget constraints.

Initialize: Fit GP_i using $\mathcal{D}_i$ for each agent. Let $B_{\max} = \max_i B_i$ and define $\tau_i = \lfloor B_{\max}/B_i \rfloor$ with $\tau_i \geq 1$. Set iteration counter $t = 0$.

While any $B_i > 0$ and $t < T$ **do**:

(1) **Acquisition**: For each agent i, if $t \bmod \tau_i = 0$ and $B_i > 0$, select a proposal

$$\mathbf{x}_i^{(t)} \in \arg\max_{\mathbf{x}} \mathcal{A}_i(\mathbf{x}; \mathrm{GP}_i), \qquad \mathbf{x}_i^{(t)} = \begin{bmatrix} \mathbf{x}_{i,\text{shared}}^{(t)} \\ \mathbf{x}_{i,\text{private}}^{(t)} \end{bmatrix}.$$

(2) **Consensus on shared variables:** Update only the shared component via weighted averaging:

$$\mathbf{x}_{i,\text{shared}}^{(t),\text{new}} \leftarrow \sum_{j=1}^{K} \Omega_{ij}^{(t)}\, \mathbf{x}_{j,\text{shared}}^{(t)}, \qquad i = 1, \ldots, K.$$

(3) **Evaluation and model update:** For each agent i with $B_i > 0$:
- Form the full input using the consensus-updated shared part and the private part from the proposal:

$$\mathbf{x}_i^{(t),\text{new}} = \begin{bmatrix} \mathbf{x}_{i,\text{shared}}^{(t),\text{new}} \\ \mathbf{x}_{i,\text{private}}^{(t)} \end{bmatrix} \in \mathbb{R}^d.$$

- Evaluate $y_i^{(t),\text{new}} = f_i(\mathbf{x}_i^{(t),\text{new}})$.
- Update dataset $\mathcal{D}_i \leftarrow \mathcal{D}_i \cup \{(\mathbf{x}_i^{(t),\text{new}}, y_i^{(t),\text{new}})\}$.
- Retrain GP_i; update $B_i \leftarrow B_i - 1$.

(4) **Refresh similarity/weights:** Recompute $\mathbf{S}^{(t)}$ and update the row-stochastic $\boldsymbol{\Omega}^{(t)}$.

(5) **Increment:** $t \leftarrow t + 1$.

end while

Output: For each agent i, return final dataset $\mathcal{D}_i$ and solution $\mathbf{x}_i^*$ (best observed point).

3.1 Similarity- and Optimal-Location-Aware Dynamic Consensus Weighting. Previous collaborative BO via consensus strategy [11] predominantly assumes uniform consensus weights, treating all agents as equally informative regardless of differences in their underlying objective functions. While this assumption simplifies implementation, it is inherently limiting in heterogeneous settings where agents are associated with distinct objective function mappings of a shared underlying objective. Blindly aggregating information across dissimilar agents can degrade surrogate fidelity, introduce misleading guidance, and ultimately hinder optimization convergence. To address this limitation, ARCO-BO introduces a similarity- and optimal-location-aware consensus mechanism that dynamically adjusts the influence of each agent based on both global functional similarity and local alignment of the predicted optimal solution. This approach ensures that information transfer is targeted and beneficial, promoting collaboration only where it enhances optimization performance.

Let K denote the number of agents, each maintaining a local GP surrogate model GP_i. To quantify interagent similarity, ARCO-BO computes behavior-based embeddings derived from each model's predictive mean over a shared input grid. Specifically, let $X_{\text{test}} \in \mathbb{R}^{N\times d}$ be a common test set sampled via Latin hypercube sampling within the design bounds. Following established heuristics in surrogate modeling and design of computer experiments, we set $N = 50d$, which provides sufficient resolution for model comparison in moderate dimensions (commonly interpreted as d up to about 10) [43,44]. For each agent i, the predictive mean at each test input is computed to form a vector:

$$\boldsymbol{\mu}_i = \mathbb{E}[\mathrm{GP}_i(X_{\text{test}})] \tag{10}$$

and its predicted minima location is determined by

$$\mathbf{x}_i^* \in \arg\min_{x\in X_{\text{test}}} \boldsymbol{\mu}_i \tag{11}$$

If there happens to be more than one point with the same minimum value, we choose the one that appears first in the test set.

Stacking $\boldsymbol{\mu}_i$ across all agents yields the behavior matrix $\mathbf{M} \in \mathbb{R}^{K\times N}$, while stacking $\mathbf{x}_i^*$ yields the minima matrix $\mathbf{X}^* \in \mathbb{R}^{K\times d}$. These matrices encode both global predictive behavior and local optimal solutions for all agents.

The optimal-location-aware similarity between agents i and j is then defined as

$$s_{ij} = s_{ij}^{\text{Pearson}} \times s_{ij}^{\text{Proximity}} \tag{12}$$

This multiplicative formulation ensures that high similarity is assigned only when both global trends and the local optimal solution are aligned. If either component is low, their product is suppressed, acting as a gating mechanism that avoids misleading collaboration. By enforcing joint agreement, the multiplicative

form conservatively identifies agent pairs for which information sharing is both reliable and beneficial.

s_{ij}^{Pearson} captures global similarity in functional behavior using the Pearson correlation coefficient, which measures the linear alignment between the predictive means of agents i and j. Let $\mu_{i,n}$ denote the nth element of agent i's predictive mean vector $\boldsymbol{\mu}_i$, and let

$$\bar{\mu}_i = \frac{1}{N}\sum_{n=1}^{N}\mu_{i,n}$$

be the average predictive value over the test set. The Pearson correlation coefficient is then computed as

$$\rho_{ij} = \frac{\sum_{n=1}^{N}(\mu_{i,n}-\bar{\mu}_i)(\mu_{j,n}-\bar{\mu}_j)}{\sqrt{\sum_{n=1}^{N}(\mu_{i,n}-\bar{\mu}_i)^2}\sqrt{\sum_{n=1}^{N}(\mu_{j,n}-\bar{\mu}_j)^2}}, \quad s_{ij}^{\text{Pearson}} = \frac{\rho_{ij}+1}{2} \tag{13}$$

This normalization maps the Pearson correlation from its original range of $[-1, 1]$ to $[0, 1]$, ensuring non-negative similarity scores that are compatible with weighting in the consensus framework.

The second term, $s_{ij}^{\text{Proximity}}$, encodes minima proximity similarity, quantifying how close the predicted optimal solutions are

$$s_{ij}^{\text{Proximity}} = \exp\left(-\lambda_p \|\mathbf{x}_i^* - \mathbf{x}_j^*\|^2\right) \tag{14}$$

where λ_p is a tunable hyperparameter that controls the sensitivity of the similarity score to the distance between predicted minima. This exponential formulation ensures that agents with nearby optimal solutions are weighted more strongly in the consensus.

In our implementation, we set λ_p such that $s_{ij}^{\text{Proximity}} = 0.1$ when the distance between the predicted optimal solution is 10% of the input-domain range $\Delta = \max(\mathbf{x}) - \min(\mathbf{x})$:

$$\lambda_p = \frac{-\ln(0.1)}{(0.1\Delta)^2} \tag{15}$$

The 10% threshold is an empirically motivated choice based on initial experiments and reflects a moderate tolerance for optimal solution misalignment. However, this threshold can be adjusted depending on the problem context and desired sensitivity to proximity. In general, λ_p should be set to reflect the scale at which optimal solution misalignment becomes detrimental to collaborative inference.

Together, these yield a symmetric similarity matrix $\mathbf{S}^{(t)} \in \mathbb{R}^{K\times K}$ that captures both global predictive consistency and alignment in locally optimal regions. To illustrate the motivation behind our metric, Fig. 4 shows how different function transformations impact shape similarity and optimal solution proximity. Each subplot compares a base function $f(x)$ (solid black) with a transformed variant (dashed gray). Transformations like vertical shift and amplitude scaling (Figs. 4(*a*) and 4(*b*)) preserve both shape and minimizer location, resulting in high similarity. In contrast, sign flips (Fig. 4(*c*)) or input shifts (Figs. 4(*d*)–4(*f*)) degrade similarity by altering the function shape, the optimum location, or both. These examples highlight the need for a similarity metric that considers both global trends and local optimal solutions. Our multiplicative formulation conservatively gates information sharing, emphasizing collaboration only when both aspects are well aligned. In this work, we define similarity in a manner consistent with Bayesian optimization: functions are considered similar when they share comparable trends and optimal solution locations, even if their numerical values differ. This optimization-oriented definition explains why horizontal shifts, such as in case (d), decrease similarity despite only modest changes in function values.

To enable dynamic, similarity-aware collaboration, ARCO-BO constructs a time-varying consensus matrix:

$$\boldsymbol{\Omega}^{(t)} = \gamma(t)\cdot\mathbf{S}^{(t)} + \big(1-\gamma(t)\big)\cdot\mathbf{I} \tag{16}$$

where $\mathbf{S}^{(t)} \in \mathbb{R}^{K\times K}$ is a non-negative similarity matrix, and $\mathbf{I}$ is the identity matrix. The scalar weighting function

$$\gamma(t) = \exp\left(-\frac{\alpha t}{T}\right) \tag{17}$$

controls the transition from early-stage collaboration ($\gamma(0) = 1$) to late-stage independent optimization ($\gamma(T) \approx 0$). Here, T denotes the total number of BO iterations, and α is a tunable hyperparameter that governs the decay rate. Larger values of α lead to faster reduction in collaboration, while smaller values preserve stronger consensus for a longer duration.

To ensure that $\boldsymbol{\Omega}^{(t)}$ defines a valid consensus mechanism, it is normalized using Sinkhorn normalization [45], which iteratively rescales the rows and columns of a non-negative matrix to make it doubly stochastic. The normalization is applied as

$$\boldsymbol{\Omega}^{(t)} \leftarrow \text{diag}(\mathbf{r})^{-1}\boldsymbol{\Omega}^{(t)}\,\text{diag}(\mathbf{c})^{-1} \tag{18}$$

where $\mathbf{r}$ and $\mathbf{c}$ are row and column scaling vectors, updated alternately until convergence. In ARCO-BO, each agent updates its consensus variables only at the beginning of its own sampling step, using the most recently available information. Updates are not applied immediately upon receiving messages from other agents. Because each agent maintains and updates its own surrogate model, communication latency affects only how timely the consensus information is, not the correctness of the GP update.

This similarity- and optimal-location-aware consensus strategy enables ARCO-BO to integrate information across agents only when it is mutually beneficial, thereby enhancing the sample efficiency and improving convergence robustness in heterogeneous multi-agent optimization settings.

3.2 Budget-Aware Asynchronous Sampling. To realize the agent-specific sampling intervals τ_i introduced in Algorithm, we incorporate a budget-aware asynchronous sampling mechanism.

In real-world collaborative optimization, agents often operate under heterogeneous resource constraints. For instance, one agent might perform high-fidelity physical experiments that are expensive and time-consuming, while another relies on low-fidelity simulations that are faster and cheaper to evaluate. Similarly, some agents may be limited by equipment availability or operational cost, while others can afford frequent queries due to lower evaluation overhead. Despite this variability, most existing collaborative BO methods assume a uniform sampling rate, where all agents carry out the same number of iterations or participate in collaborative updates at the same frequency. This assumption can lead to inefficient use of resources, as fast agents are forced to wait for slower ones, and costly evaluations are scheduled just as frequently as cheap ones. ARCO-BO addresses this by introducing a budget-aware asynchronous sampling mechanism. Specifically, each agent i is assigned a sampling interval τ_i based on its relative budget:

$$\tau_i = \left\lfloor \frac{B_{\max}}{B_i} \right\rfloor \tag{19}$$

where B_i is the total budget for agent i and $B_{\max}$ is the maximum budget among all agents. This means agents with larger budgets sample more frequently, while those with smaller budgets sample less often, helping them conserve their limited evaluations for the most important points.

Agent i proposes a new point every τ_i iterations when it still has remaining budget. Otherwise, it skips that iteration. This asynchronous sampling strategy allows each agent to operate at a frequency

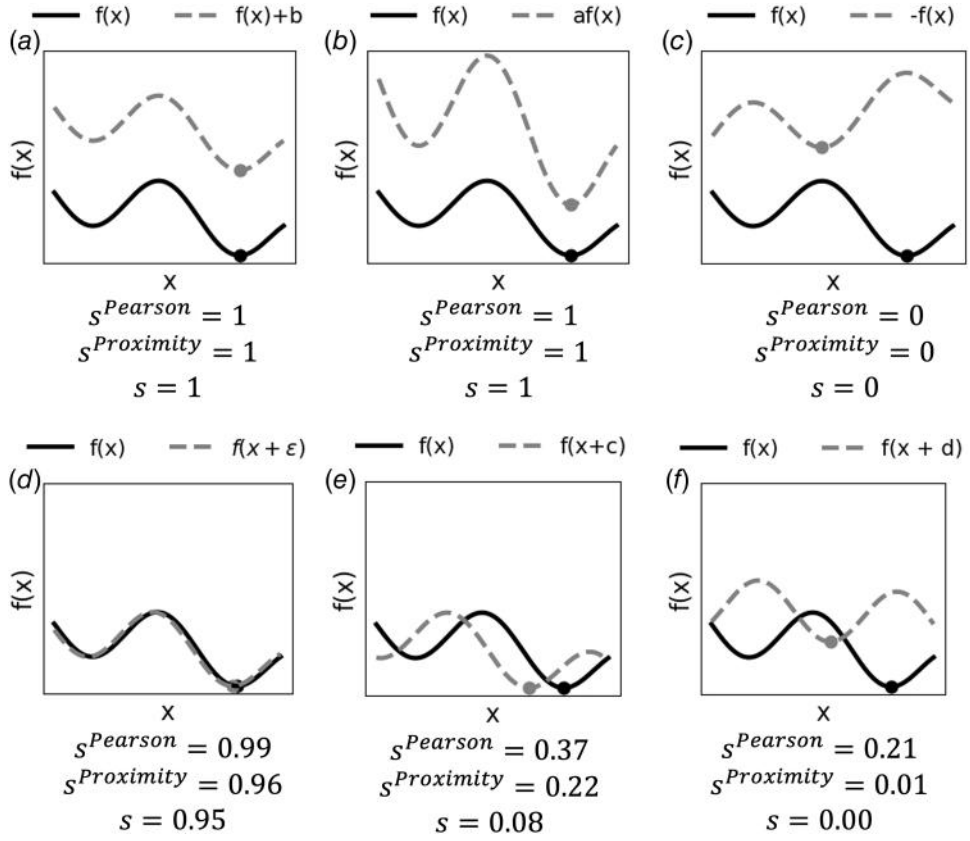

Fig. 4 Illustration of function transformations and their impact on similarity. Each panel compares a base function $f(x)$ (solid black) with a transformed variant (dashed gray). (*a*) Vertical shift: $f(x) + b$ preserves both shape and minimizer location. (*b*) Amplitude scaling: $af(x)$ changes magnitude but keeps the same shape and minimizer. (c) Sign flip: $-f(x)$ inverts the shape and shifts the minimizer. (*d*) Small input shift: $f(x+\varepsilon)$ retains high similarity in both shape and minimizer location. (*e*) Moderate input shift: $f(x + c)$ preserves shape but misaligns the minimizer. (*f*) Large input shift: $f(x + d)$ leads to low shape and proximity similarity. Pearson correlation measures shape similarity, while proximity reflects alignment between minimizers.

proportional to its allocated budget, enabling more efficient resource utilization and improved overall performance in collaborative optimization.

A key implementation detail is when agents exchange and use shared information during asynchronous execution. In ARCO-BO, each agent updates its shared information and consensus weights only when it begins its own sampling step, rather than immediately upon receiving updates from others. This update rule aligns consensus updates with each agent's local BO cycle, prevents inconsistent or partial updates during acquisition optimization, and preserves the pacing induced by the budget-aware intervals τ_i. In practice, each agent uses the most recently available peer information when starting its next sampling iteration, while delayed or out-of-order messages are incorporated during the following local update without interrupting ongoing computation. As a result, agents operate fully asynchronously but maintain locally synchronized consensus updates tied to their individual sampling schedules. Because each agent maintains and updates its own surrogate model, communication latency affects only how timely the consensus information is, not the correctness of the GP update.

Currently, ARCO-BO assumes that the evaluation budgets B_i are fixed and known in advance. However, the framework can be extended to accommodate dynamic budget adjustment by periodically updating τ_i based on remaining budgets or runtime constraints. Similarly, the model could be adapted to account for varying evaluation costs by defining an effective budget in terms of cumulative cost rather than the number of evaluations. We leave these extensions for future work.

3.3 Partial Input-Space Sharing. Building on the dynamic consensus mechanism in Algorithm, we extend the framework to settings in which agents share only a subset of the design variables.

In many collaborative optimization problems, agents do not operate over identical input spaces. Differences may arise due to unmeasurable parameters in physical experiments, agent-specific design freedoms, or privacy and ownership constraints. As a result, only a subset of variables may be commonly accessible and meaningfully comparable across all agents, while the remaining variables are agent-specific. Conventional BO frameworks typically assume a fully shared input space, which limits their applicability in such heterogeneous scenarios.

ARCO-BO addresses this challenge by explicitly identifying the subset of design variables that are commonly shared across all agents and restricting information exchange to this shared subspace. Each agent's decision vector is therefore decomposed as

$$\mathbf{x}_i = \begin{bmatrix} \mathbf{x}_{i,\text{shared}} \\ \mathbf{x}_{i,\text{private}} \end{bmatrix} \tag{20}$$

where $\mathbf{x}_{i,\text{shared}} \in \mathbb{R}^{d_s}$ denotes the variables that are shared among all agents, and $\mathbf{x}_{i,\text{private}} \in \mathbb{R}^{d_p}$ denotes agent-specific private variables,

with $d_s + d_p = d$.

During the consensus step, information exchange occurs *only* in the shared subspace. Specifically, the shared variables are updated via a weighted averaging scheme,

$$\mathbf{x}_{i,\text{shared}}^{(t),\text{new}} = \sum_{j=1}^{K} \Omega_{ij}^{(t)} \mathbf{x}_{j,\text{shared}}^{(t)}, \quad i = 1, \ldots, K \tag{21}$$

while the private variables remain unchanged.

The resulting consensus-informed design for agent i is then given by

$$\mathbf{x}_i^{(t),\text{new}} = \begin{bmatrix} \mathbf{x}_{i,\text{shared}}^{(t),\text{new}} \\ \mathbf{x}_{i,\text{private}}^{(t)} \end{bmatrix} \tag{22}$$

By limiting collaboration to variables that are commonly shared across agents, this partial-sharing scheme facilitates effective coordination while preserving agent-specific privacy. At the same time, it enables ARCO-BO to operate in heterogeneous settings where fully aligned input spaces are neither feasible nor desirable.

3.4 Privacy Considerations and Information Sharing Trade-Offs. ARCO-BO enables collaboration without requiring agents to share raw input–output data. Instead, agents exchange surrogate-level summaries, such as predictive means and estimated optimal locations, that convey useful structural information about their objective functions while concealing exact measurements or experimental conditions. This surrogate-based communication strikes a balance between cooperative information exchange and data confidentiality.

Information sharing plays a key role in collaborative optimization, and existing approaches span a wide range of privacy-utility trade-offs. At one end of the spectrum, data-fusion methods [2,35–37] pool full datasets across agents, enabling maximum information sharing but offering no protection for proprietary or sensitive data. Federated learning [46,47] takes a more conservative approach by sharing surrogate model parameters or gradient information; however, this can still reveal the underlying surrogate and provides only limited privacy guarantees. At the other end, consensus-based collaborative BO methods,such as [11], restrict communication to decision vectors, which offers stronger privacy but removes surrogate-level context for assessing similarity across agents and can lead to negative transfer in heterogeneous settings. ARCO-BO sits between these extremes. It enables similarity-aware coordination through limited surrogate-level information exchange while keeping all raw data local to each agent. Although this avoids direct data sharing, the exchanged summaries may still convey coarse functional trends or regions of interest, reflecting an inherent trade-off between privacy and utility.

In this framework, the design variables x are assumed to lie in a shared, nonproprietary domain, as their proposed values must be communicated during consensus updates. All private variables and observed objective values remain strictly local to each agent. Only surrogate-level statistics are exchanged, providing the necessary signals for collaboration without disclosing raw data or gradients. Although these shared quantities may reveal limited functional characteristics, they maintain a practical level of privacy suitable for decentralized optimization. Future extensions could integrate formal privacy-preserving mechanisms, such as differential privacy or encrypted surrogate aggregation, for stricter guarantees when required.

4 Evaluation Metrics

In this section, we introduce the quantitative metrics used to evaluate the performance of multi-agent Bayesian optimization methods. In ARCO-BO, each of the K agents performs one function evaluation per iteration, so one global iteration corresponds to up to K parallel function calls. This execution model reflects a parallel deployment setting, where agents operate independently and evaluate simultaneously without coordination.

Given that the objective is to identify the global optimum for each individual agent, the evaluation metrics are designed to assess performance across all agents collectively.

(1) *Normalized Final Regret.* To account for differences in objective function scales across agents, we evaluate performance using the normalized final regret [48]:

$$\bar{r}^{\,\text{final}} = \frac{1}{K} \sum_{i=1}^{K} \frac{f_i(\mathbf{x}_i^*) - \min_{t=1,\ldots,T} f_i(\mathbf{x}_i^{(t)})}{f_i^{\max} - f_i^{\min}} \tag{23}$$

where $f_i(\mathbf{x}_i^*)$ is the known global minimum for agent i, and $f_i^{\max}$, $f_i^{\min}$ denote the maximum and minimum of the true function. A smaller value of $\bar{r}^{\,\text{final}}$ indicates closer proximity to the optimum.

(2) *Normalized Area Under the Curve (AUC).* To evaluate early-stage convergence behavior, we compute the normalized area under the averaged convergence curve [49], which tracks the best objective value found so far over time. Specifically,

$$\overline{\text{AUC}} = \frac{1}{K} \sum_{i=1}^{K} \frac{1}{N} \sum_{t=1}^{N} \frac{\bar{f}_i^{(t)} - f_i(\mathbf{x}_i^*)}{f_i^{\max} - f_i^{\min}} \tag{24}$$

where $\bar{f}_i^{(t)}$ is the mean of the best objective value found by agent i up to iteration t, averaged across replicates, and N is the number of early-stage iterations. A smaller $\overline{\text{AUC}}$ indicates faster convergence to the optimum during the initial phase of optimization. In our test cases, we use $N = 0.1T$ to represent the early-stage phase of optimization. Convergence curves are indexed by each agent's evaluation count, so the AUC metric is not affected by asynchronous sampling.

In addition to the evaluation metrics, we also report the computational overhead of ARCO-BO relative to separate BO. All experiments were performed on a workstation running Ubuntu 20.04, equipped with an AMD EPYC 7413 processor (24 cores) and 64 GB of RAM. The code was implemented in PYTHON 3.9 and executed on the CPU.

5 Case Studies

In this section, we first present a 1D illustrative example demonstrating how our method works differently in contrast to conventional collaborative BO via consensus when optimizing heterogeneous functions. Next, we demonstrate the effectiveness of the proposed ARCO-BO method on a 2D example exhibiting three types of heterogeneity: task heterogeneity, resource disparity, and partially shared input spaces. This example highlights the advantages of ARCO-BO over applying separate BO independently to each agent. Finally, we evaluate ARCO-BO on two high-dimensional engineering benchmarks: the borehole function (8D) and the wing weight function (10D). These engineering problems are widely used in surrogate-assisted optimization and testing ARCO-BO's ability to scale in terms of both consensus overhead and decentralized sampling efficiency in higher dimensions.

It is important to note that although these benchmark functions are originally used as test cases for multifidelity design optimizations [2], we treat each fidelity level as a separate agent task in this study, each with its own objective function and ground truth, rather than treating the highest fidelity function as a ground truth, and the objective is to identify the optimal solution for each agent simultaneously.

5.1 One-Dimensional Illustrative Example. To evaluate the robustness and collaborative dynamics of ARCO-BO, we consider a multi-agent optimization problem based on variants of the Sasena function [50], where three agents each optimize agent-specific objective function variants of a shared underlying function, with different global optimal locations. These variations emulate realistic heterogeneity arising from differences in modeling fidelity, operating conditions, or task formulations, which is common in engineering design and simulation. Details of the agent-specific functions, lower and upper bounds, budgets, and true optimal are provided in Table 4. Each agent begins with three randomly sampled initial evaluations, which are kept consistent across all compared methods to ensure a fair comparison.

To isolate the effect of collaborative mechanisms, all agents in our numerical studies use an identical GP kernel and shared hyperparameters. The framework itself, however, does not impose this restriction: agents may employ different kernels or hyperparameter configurations to reflect task-specific structures or data variability. In this example, all Gaussian process surrogate models use a squared exponential kernel, with fixed lengthscale 0.5, signal variance 1.0, and Gaussian observation noise with variance 10^{-6}. Figure 5 illustrates the GP updates at the last iteration under three optimization strategies. Figure 5(*a*) shows a separate BO, where each agent optimizes independently without collaboration. While the predicted means generally align with the true functions, the models exhibit high uncertainty in unexplored regions. Figure 5(*b*) illustrates the behavior of an existing collaborative BO method based on uniform consensus, where agents share information equally regardless of task alignment. This uniform sharing often leads to incorrect collaboration, as conflicting information from agents with different objective functions is combined without regard for compatibility. Consequently, agents are misled in their sampling decisions, resulting in suboptimal exploration and poor convergence. This issue is visually highlighted by the red boxes and arrows, which indicate significant deviations from the true optimal solution. In contrast, Fig. 5(*c*) shows the results of ARCO-BO, which adaptively reweights interagent influence based on task similarity and differences in optimal solution locations. The four heatmaps (Fig. 6) depict the learned interagent weight matrix W, where each cell (i, j) is the weight agent i assigns to agent j. Because agent 1's optimum is far from those of agents 2 and 3, their pairwise similarity is near zero; consequently, ARCO-BO suppresses the cross-weights W_{12} and W_{13} (and symmetrically W_{21}, W_{31}) toward zero. In contrast, agents 2 and 3 are similar, so ARCO-BO initially assignscross-weights W_{23} and W_{32} to facilitate sharing; as both agents approach their optimal solution, these weights taper and W sharpens toward a near-diagonal structure, evidencing reduced cross-agent influence and increasing self-reliance. This targeted collaboration enables agents to focus sampling near their true optimal solution while avoiding misleading guidance from incompatible tasks.

Figure 7 further quantifies the benefits of ARCO-BO based on 50 independent replicates, each generated using different initial samples drawn for the agents. In the top row, each plot shows the mean best function value (y^*) over iterations, comparing ARCO-BO with conventional collaborative BO via consensus and separate BO. The results demonstrate that ARCO-BO enables faster convergence to each agent's optimum. In contrast, the baseline collaborative BO fails to outperform separate BO and often does not reach the true optimal solution, due to overly

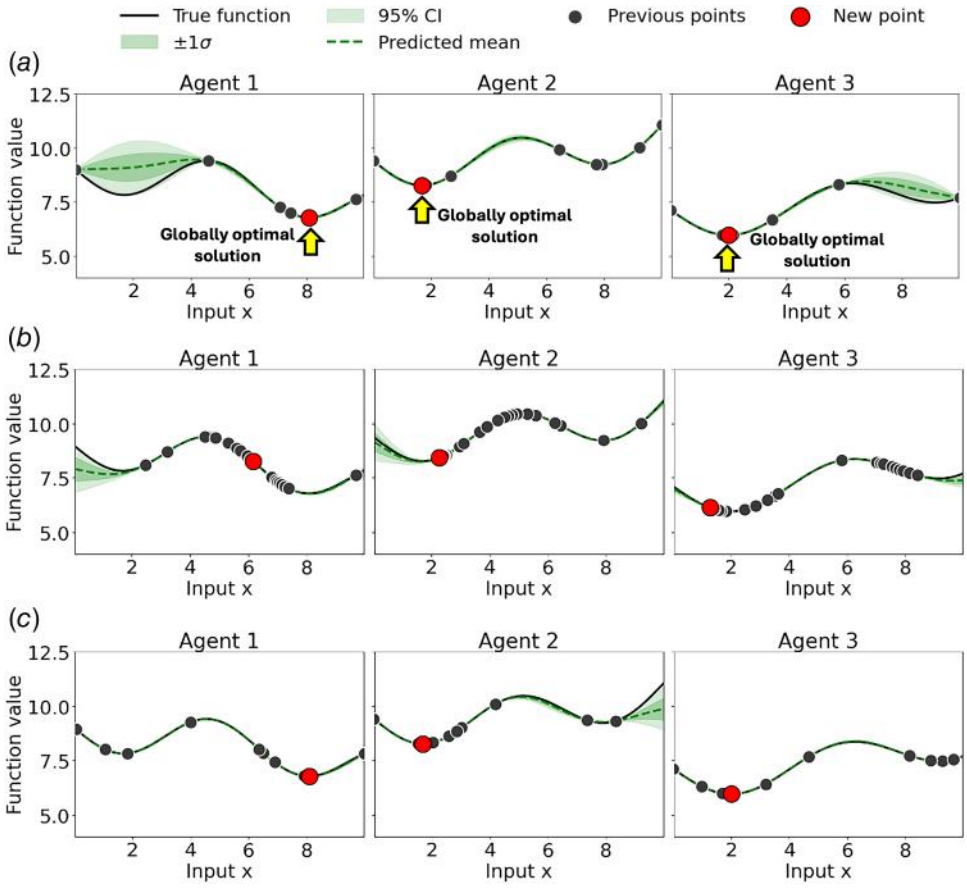

Fig. 5 Comparison of surrogate model predictions for three agents optimizing distinct variants of the Sasena function at iteration 20. (*a*) Separate BO, (*b*) benchmark collaborative BO, and (c) the proposed ARCO-BO method. Previous solutions are included not to list every sampled point, but to indicate the span of the regions explored by each agent. The globally optimal solution is marked for reference.

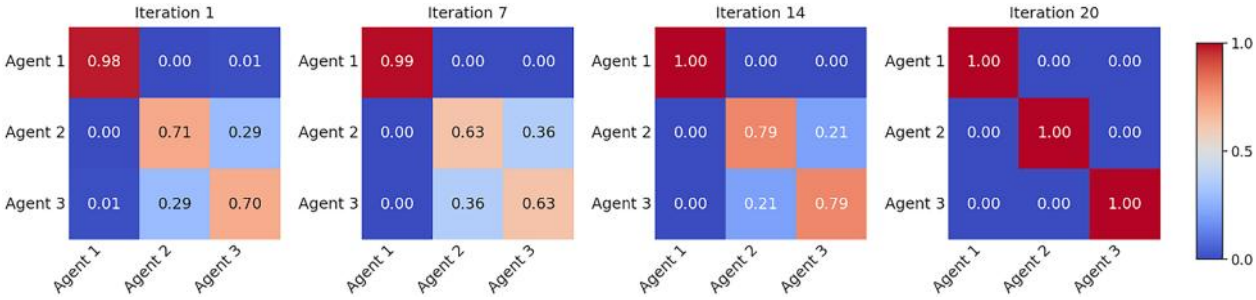

Fig. 6 Progression of the learned interagent weights *W* under ARCO-BO at iterations 1, 7, 14, and 20

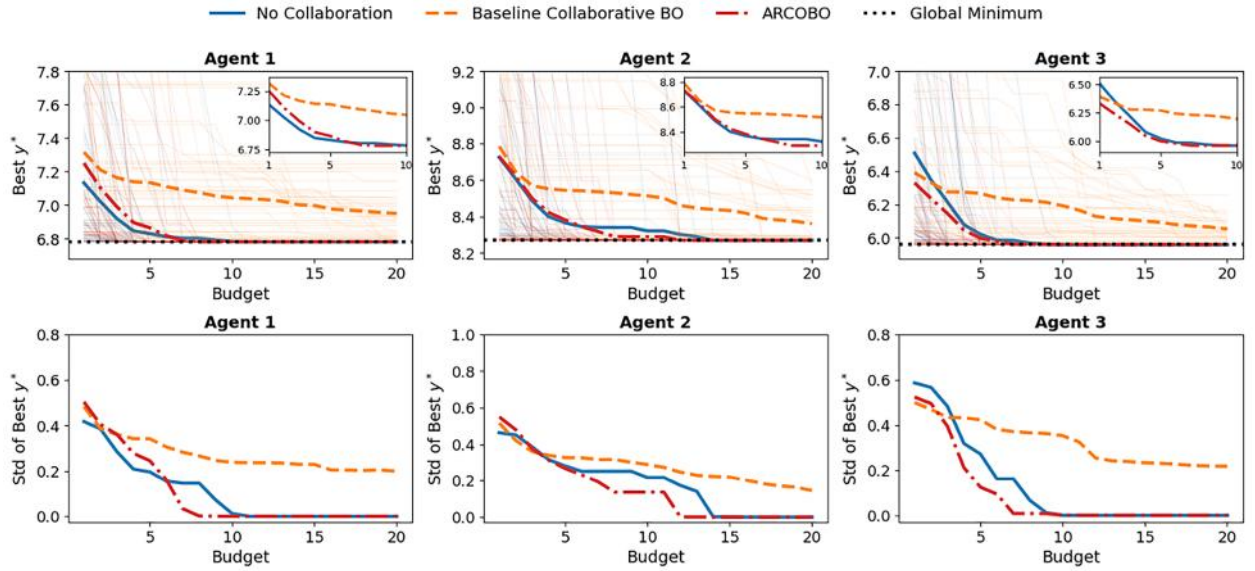

Fig. 7 Convergence performance of ARCO-BO compared to baseline methods on the multi-agent Sasena problem, evaluated over 50 independent replicates

strong information sharing that misguides agents with divergent objective functions. The bottom row shows the standard deviation of the best function values, indicating that ARCO-BO also achieves greater robustness in optimization performance. To further assess performance, we compute the normalized AUC and the normalized final regret, as summarized in Table 1. Both separate BO and ARCO-BO are able to reach the global optimal solution for all agents, while the baseline collaborative BO fails to do so. Among all methods, ARCO-BO achieves the fastest early convergence, highlighting the effectiveness of its adaptive and similarity-aware collaboration strategy. The additional computational overhead of ARCO-BO relative to the separate BO is reported in Table 8.

5.2 Two-Dimensional Illustrative Example. We consider a multi-agent problem based on variations of the 2D Ackley function [51], a widely used benchmark known for its numerous local minima and challenging nonconvex landscape. We design three scenarios to assess ARCO-BO's capabilities: (1) agents have the same evaluation budgets with both input variables fully shareable; (2) agents have different evaluation budgets with both input variables shareable, simulating heterogeneous resource availability; and (3) agents have the same budgets but only one input variable is shareable, reflecting partial input sharing constraints. In all cases, ARCO-BO is compared against separate BO and, where applicable, a benchmark collaborative BO via consensus.

Detailed information about the six agents is listed in Table 5. In the second scenario, agents 2, 3, and 6 are assigned only half of the original evaluation budgets to reflect resource differences. Each

Table 1 Normalized AUC and normalized final regrets for the 1D Sasena problem, reported as mean ± standard deviation across 50 replicates

Method	AUC	Final regret
Separate BO	0.1623 ± 0.2048	0.0000 ± 0.0000
Benchmark CBO	0.1840 ± 0.1872	0.0483 ± 0.0790
ARCO-BO	0.1562 ± 0.2017	0.0000 ± 0.0000

Table 2 Normalized AUC and normalized final regrets for the 2D Ackley problem in three different scenarios, reported as mean ± standard deviation across 50 replicates

Scen.	Method	AUC	Final regret
1	Separate BO	0.2784 ± 0.1483	0.0169 ± 0.0159
	Benchmark CBO	0.2050 ± 0.0870	0.0929 ± 0.0669
	ARCO-BO	0.2008 ± 0.0932	0.0145 ± 0.0158
2	Separate BO	0.2770 ± 0.1486	0.0143 ± 0.0159
	ARCO-BO	0.1992 ± 0.0933	0.0125 ± 0.0158
3	Separate BO	0.2784 ± 0.1483	0.0186 ± 0.0186
	ARCO-BO	0.1861 ± 0.0975	0.0145 ± 0.0158

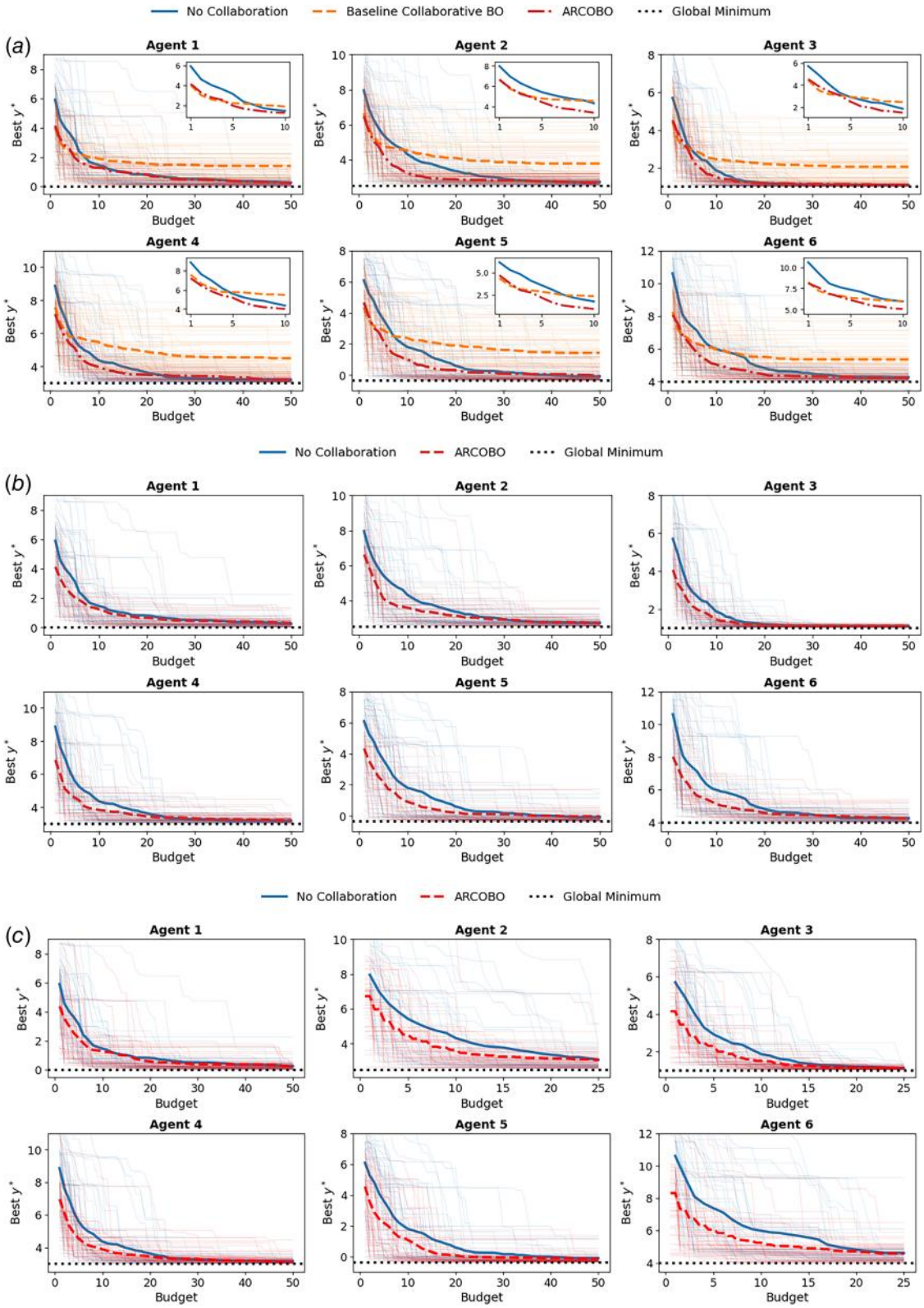

Fig. 8 Convergence performance across 50 replicates for ARCO-BO and separate BO on the multi-agent Ackley problem with (*a*) the same budgets and fully shareable input, (*b*) different budgets for each agent, and (*c*) different shareable inputs

scenario is evaluated across 50 independent replicates, and we report both the normalized AUC and the normalized final regret for all agents (Table 2). We use an radial basis function (RBF) kernel with fixed lengthscale 0.5, signal variance 1.0, and observation noise variance 10^{-6}.

In scenario 1 (Fig. 8(*a*)), where agents have equal evaluation budgets and both input variables are fully shareable, ARCO-BO achieves rapid convergence. Compared to the benchmark collaborative BO, which often fails to converge to each agent's local optimum due to detrimental information transfer, ARCO-BO

Table 3 Normalized AUC and final regrets for borehole and wing weight problems (mean ± std over 20 replicates)

Benchmark	Method	AUC	Final regret
Borehole	Separate BO	0.0251 ± 0.0178	0.0018 ± 0.0038
	ARCO-BO	0.0174 ± 0.0101	0.0008 ± 0.0016
Wing weight	Separate BO	0.0807 ± 0.0757	0.0274 ± 0.0576
	ARCO-BO	0.0471 ± 0.0188	0.0026 ± 0.0050

demonstrates clear advantages by enabling agents to collaborate selectively only when it is mutually beneficial.

In scenario 2 (Fig. 8(*b*)), where agents have different evaluation budgets but both input variables remain shareable, ARCO-BO maintains robust optimization performance despite the heterogeneous resource distribution. Notably, separate BO struggles under this setting because agents with low budgets cannot adequately explore their objective landscapes, which leads to the convergence on suboptimal solutions (agent 2, 3, and 6 in Fig. 8(*b*)). In contrast, ARCO-BO utilizes knowledge transferred

Table 4 One-dimensional illustrative example: Sasena function

Agent	Formulation	Upper/lower bound	Init. sample	Budget	True optimal
1	$y_1(x) = -\sin x - \exp\left(\frac{x}{10}\right) + 10$	$0 \le x \le 10$	3	20	6.782
2	$y_2(x) = -\sin(0.95x) - \exp\left(\frac{x}{50}\right) + 0.03(x-2)^2 + 10.3$		3	20	8.269
3	$y_3(x) = -\sin(0.8x) - \exp\left(\frac{x}{50}\right) + 0.03(x-2)^2 + 8$		3	20	5.959

Formulation, upper/lower bounds, initial sample size, budget, and true optimal value of each agent.

Table 5 Two-dimensional illustrative example: multifidelity Ackley function

Agent	Formulation	Upper/lower bound	Init. sample	Budget	True optimal
1	$y_0(x) = -20\exp\left(-0.2\sqrt{0.5\sum x_i^2}\right) - \exp\left(0.5\sum\cos(\pi x_i)\right) + 20 + e$	$-5 \le x_i \le 5$	5	50	0.000
2	$y_1(x) = -20\exp\left(-0.2\sqrt{0.5\sum (x_i+0.2)^2}\right) - \exp\left(0.5\sum\cos(1.1\pi(x_i+0.2))\right) + 20 + e + 2.5$		5	50	2.500
3	$y_2(x) = -20\exp\left(-0.2\sqrt{0.5\sum (0.8(x_i-0.3))^2}\right) - \exp\left(0.5\sum\cos(0.9\pi\cdot 0.8(x_i-0.3))\right) + 20 + e + 1.0$		5	50/25	1.000
4	$y_3(x) = -20\exp\left(-0.2\sqrt{(x_1+0.4)^2}\right) - \exp(\cos(\pi(x_1+0.4))) + 20 + e + 3.0$		5	50/25	3.000
5	$y_4(x) = -20\exp\left(-0.2\sqrt{0.5\sum (x_i-0.5)^2}\right) - 1.5\exp\left(0.5\sum\cos(\pi(x_i-0.5))\right) + 20 + e + 1.0$		5	50	−0.359
6	$y_5(x) = 1.1[-20\exp\left(-0.2\sqrt{0.5\sum (x_i-0.1)^2}\right) - \exp\left(0.5\sum\cos(\pi(x_i-0.1))\right) + 20 + e] + 4.0,$		5	50/25	3.978

Note: Formulation, bounds, initial sample size, budget, and true optimal value (illustrative) for each fidelity level.

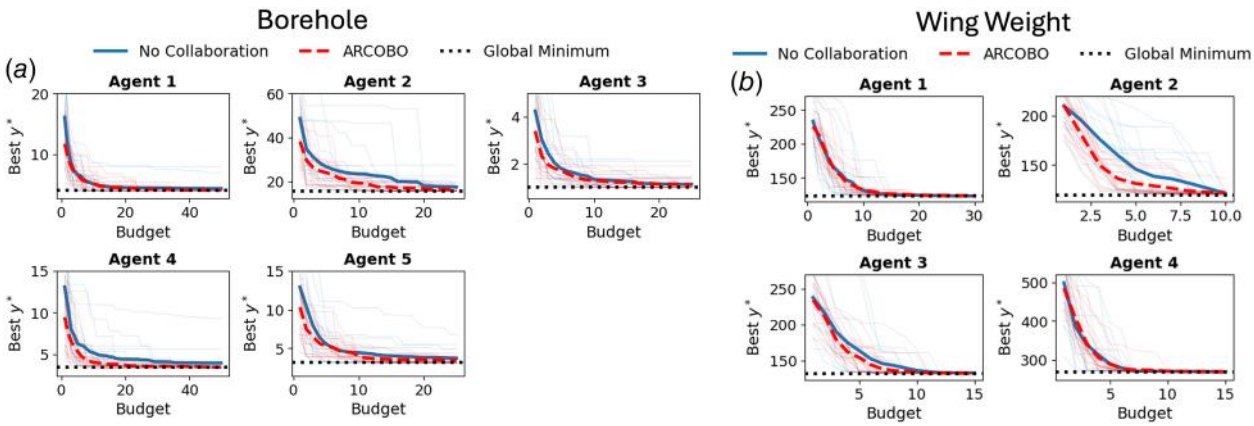

Fig. 9 Convergence performance across 20 replicates for ARCO-BO and separate BO on the (*a*) borehole problem and (*b*) wing weight problem.

Table 6 Borehole function with five agents

Agent	Formulation	Variable bounds	Shareable inputs	Init. sample	Budget	True optimal
1	$y_1(\mathbf{x}) = \frac{2\pi T_u(H_u - H_l)}{\ln(r/r_w)\left(1 + \frac{2LT_u}{r_w^2 K_w \ln(r/r_w)} + \frac{T_u}{T_l}\right)}$	$0.05 \le r_w \le 0.15$ $100 \le r \le 10000$ $100 \le T_u \le 1000$ $990 \le H_u \le 1110$ $10 \le T_l \le 500$ $700 \le H_l \le 820$ $1000 \le L \le 2000$ $6000 \le K_w \le 12000$	r_w T_u H_u T_l H_l	8	50	3.985
2	$y_2(\mathbf{x}) = \frac{2\pi T_u(H_u - 0.8H_l)}{\ln(r/r_w)\left(1 + \frac{LT_u}{r_w^2 K_w \ln(r/r_w)} + \frac{T_u}{T_l}\right)}$			8	25	15.582
3	$y_3(\mathbf{x}) = \frac{2\pi T_u(H_u - H_l)}{\ln(r/r_w)\left(1 + \frac{8LT_u}{r_w^2 K_w \ln(r/r_w)} + 0.75\frac{T_u}{T_l}\right)}$			8	25	1.000
4	$y_4(\mathbf{x}) = \frac{2\pi T_u(1.09H_u - H_l)}{\ln(4r/r_w)\left(1 + \frac{3LT_u}{r_w^2 K_w \ln(r/r_w)} + \frac{T_u}{T_l}\right)}$			8	50	3.434
5	$y_5(\mathbf{x}) = \frac{2\pi T_u(1.05H_u - H_l)}{\ln(2r/r_w)\left(1 + \frac{3LT_u}{r_w^2 K_w \ln(r/r_w)} + \frac{T_u}{T_l}\right)}$			8	25	3.153

Note: Each agent is assigned a modified variant of the borehole function. Variable bounds, shareable inputs, initial sample size, budget, and true optimal value are listed.

from higher-budget agents to guide low-budget agents toward better solutions, leading to improvements in convergence speed and final regret compared to baselines.

In scenario 3 (Fig. 8(*c*)), where agents have the same budgets but only one input variable is shareable, ARCO-BO continues to demonstrate its advantages over separate BO. While partial input sharing limits the extent of collaboration, ARCO-BO's adaptive consensus mechanism ensures that agents only share useful and consistent information along the shared dimensions. As a result, it still outperforms independent BO, which lacks collaborative benefits.

Overall, these results demonstrate that ARCO-BO's adaptive collaboration strategy works with varying input sharing constraints and resource heterogeneity. It provides an efficient optimization framework for scenarios where traditional BO frameworks struggle due to limited budgets or incomplete information sharing. The additional computational overhead of ARCO-BO relative to separate BO, averaged across all test cases and replicates, is reported in Table 4.

5.3 Example on High-Dimensional Engineering Benchmarks. We performed 20 independent replicates for both the borehole and wing weight problems under a multi-agent setting. These two functions are both modified from Ref. [2]. The results are summarized in Table 3 and visualized in Fig. 9. In each benchmark, agents operate under different evaluation budgets, and only a subset of the input space is shared across agents. Full specifications of the agent-specific functions, budget allocations, and shared input dimensions are provided in Tables 6 and 7.

Table 7 Wing weight function with four agents

Agent	Formulation	Variable bounds	Shareable inputs	Init. sample	Budget	True optimal
1	$y_1(\mathbf{x}) = 0.036\, s_w^{0.758} w_{fw}^{0.0035} \left(\frac{A}{\cos^2(\Lambda)}\right)^{0.6} q^{0.006} \lambda^{0.04} \left(\frac{100 t_c}{\cos(\Lambda)}\right)^{-0.3} \left((N_z W_{dg})^{0.49} + s_w w_p\right)$	$150 \le s_w \le 200$ $220 \le w_{fw} \le 300$ $6 \le A \le 10$ $-10 \le \Lambda \le 10$ $16 \le q \le 45$ $0.5 \le \lambda \le 1$ $0.08 \le t_c \le 0.18$ $2.5 \le N_z \le 6$ $1700 \le W_{dg} \le 2500$ $0.025 \le w_p \le 0.08$	s_w w_{fw} A q W_{dg}	5	30	123.25
2	$y_2(\mathbf{x}) = 0.036\, s_w^{0.758} w_{fw}^{0.0035} \left(\frac{A}{\cos^2(\Lambda)}\right)^{0.6} q^{0.006} \lambda^{0.04} \left(\frac{100 t_c}{\cos(\Lambda)}\right)^{-0.3} \left((N_z W_{dg})^{0.49} + w_p\right)$			5	10	119.53
3	$y_3(\mathbf{x}) = 0.036\, s_w^{0.758} w_{fw}^{0.0035} \left(\frac{A}{\cos^2(\Lambda)}\right)^{0.6} q^{0.005} \lambda^{0.04} \left(\frac{100 t_c}{\cos(\Lambda)}\right)^{-0.3} \left((N_z W_{dg})^{0.49} + w_p\right)$			5	20	131.65
4	$y_4(\mathbf{x}) = 0.036\, s_w^{0.9} w_{fw}^{0.0035} \left(\frac{A}{\cos^2(\Lambda)}\right)^{0.6} q^{0.005} \lambda^{0.04} \left(\frac{100 t_c}{\cos(\Lambda)}\right)^{-0.3} (N_z W_{dg})^{0.49}$			5	20	268.13

Note: Each agent is assigned a modified variant of the wing weight function. Variable bounds, shareable inputs, initial sample size, budget, and true optimal value are listed.

Table 8 Average per-iteration wall-clock time (s) for separate BO and ARCO-BO across the four case studies

Case study	Separate BO (s/iter)	ARCO-BO (s/iter)	Overhead
1D Sasena	0.4547	0.4999	+9.9%
2D Ackley	0.7724	0.8548	+10.7%
Borehole	0.8521	0.9565	+12.3%
Wing weight	0.8576	0.9624	+12.2%

Across both benchmark problems, ARCO-BO consistently outperforms separate BO in terms of both early convergence and final solution quality, as shown in Table 3. Figure 9(*a*) illustrates these gains on the Borehole benchmark. For example, agents 2, 3, and 5 operate under reduced evaluation budgets, which would typically limit their ability to reach the global optimum. However, with ARCO-BO, these agents are able to benefit from collaborative information sharing with higher-budget peers, allowing them to converge effectively despite fewer evaluations. A similar trend is observed in Fig. 9(*b*) for the wing weight benchmark, particularly for agent 2 where ARCO-BO shows significant improvement over separate BO. Without collaboration, the agent lacks sufficient information to escape poor regions of the design space. ARCO-BO, by contrast, enables faster convergence by leveraging interagent knowledge to guide optimization across heterogeneous settings.

The superior performance of ARCO-BO across these engineering benchmarks stems from its ability to adaptively coordinate learning across heterogeneous agents. Using similarity-aware consensus, it avoids negative transfer from dissimilar tasks, which improves convergence. Its budget-aware asynchronous sampling helps agents with limited evaluations allocate resources more effectively. Partial input-space sharing enables collaboration even when agents operate on different subsets of the design space. Together, these mechanisms let ARCO-BO exploit complementary information, increase sample efficiency, and deliver consistently better solutions than independent BO in complex multi-agent settings. Finally, the additional computational overhead introduced by ARCO-BO is modest compared to the cost of function evaluations and is averaged across all agents and replicates. Quantitative timing results are reported in Table 8.

6 Conclusion

This work introduces ARCO-BO, a framework for multi-agent BO in heterogeneous settings. ARCO-BO addresses key challenges such as task heterogeneity, resource disparity, and partially shared input spaces, which are common in real-world collaborative scenarios such as decentralized manufacturing, material discovery, and multifidelity design optimization. The proposed framework integrates three core innovations to enable effective collaboration:

- Similarity and optimal-location-aware consensus mechanisms that adaptively regulate interagent influence based on surrogate model alignment.
- A budget-aware sampling strategy that allocates evaluations according to each agent's available resources.
- A strategy to handle partially shared inputs that enables collaboration while preserving private design variables.

Experiments on synthetic problems and engineering benchmarks show that ARCO-BO consistently outperforms independent BO and conventional collaborative BO via consensus. It converges faster and achieves lower regret under heterogeneous budgets and partially shared inputs, with low-budget agents gaining from targeted knowledge transfer by high-budget peers without being misled by irrelevant information.

These gains stem from ARCO-BO's distinctive approach to information sharing, which sets it apart from existing multi-agent design optimization strategies. Consensus-based collaborative BO methods coordinate decisions while avoiding raw data exchange, but their reliance on uniform averaging often ignores task heterogeneity and budget asymmetry. At the other extreme, multisource data fusion methods combine all agents' input–output data into a unified surrogate, but require full data sharing and degrade in performance under strong heterogeneity or privacy constraints.

Despite its advantages, ARCO-BO currently assumes fixed evaluation budgets and focuses on unconstrained problems, and it relies on user-defined hyperparameters such as the consensus decay rate. These design choices simplify the analysis but may limit adaptability in dynamic or strongly constrained environments. Future extensions could incorporate adaptive budgeting strategies, self-tuning consensus mechanisms, and dynamic agent participation, as well as support for richer forms of input heterogeneity, including mixed discrete–continuous domains and agent-specific constraints. Extending collaborative BO to constrained multi-agent optimization is a natural next step. In line with existing constrained BO approaches [52–54], constraints could be modeled using additional surrogate models and incorporated via standard feasibility-aware objectives or acquisition functions, such as penalty-based formulations or feasibility-weighted acquisition criteria [55]. More recent methods that explicitly account for uncertainty in constraint predictions [56,57] could likewise be integrated. Importantly, the collaborative layer would continue to operate by correcting candidate sampling locations based on interagent similarity, independent of the specific constraint-handling strategy. Assuming correlated constraint landscapes across agents, analogous to the objective functions, this provides a principled extension beyond unconstrained settings.

Beyond addressing these limitations, ARCO-BO also opens broader opportunities. Its emphasis on selective and resource-aware collaboration aligns naturally with settings where data sharing is constrained and evaluations are expensive, such as federated optimization, cooperative scientific design, and decentralized engineering. Reinforcement learning offers another promising direction, where consensus updates could themselves be adapted through learned policies that regulate interagent influence over time and guide sampling with a long-term view of performance.

Acknowledgment

This work was supported by the U.S. Army Combat Capabilities Development Command Army Research Laboratory under cooperative agreement number W911NF2220121, Air Force Office of Scientific Research under award number FA9550-24-1-0301, and the National Science Foundation's MADE-PUBLIC Future Manufacturing Research Grant Program under award number 80NSSC24M0176. Yi-Ping Chen also acknowledges the Taiwan-Northwestern Doctoral Scholarship.

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.

Appendix A: Details of Test Functions

This section presents detailed formulations and configurations for all benchmark functions used in our study. These include two illustrative examples—a 1D Sasena function (Table 4) and a 2D multifidelity Ackley function (Table 5), as well as two engineering-inspired high-dimensional problems: the borehole (Table 6) and wing weight (Table 7) functions. For each case, we specify the agent-specific modifications, input bounds, sample budgets, and known optima location. These functions collectively span a range of heterogeneities in input dimension, fidelity

structure, and interagent similarity, serving to rigorously evaluate the proposed optimization framework.

Appendix B: Sensitivity Analysis of Hyperparameters

The proposed ARCO-BO method introduces two tunable hyperparameters, α and λ_p, which respectively control the smoothness of the consensus update and the strength of cross-agent information sharing. To assess the sensitivity of ARCO-BO to these parameters, we perform a systematic study using the three-agent Sasena benchmark introduced in Sec. 5.1.

We evaluate α over the range $\{0.1, 0.5, 1, 5, 10, 50\}$, which spans different rates of temporal consensus decay, and perform 20 independent trials for each setting while keeping all other conditions (initialization, kernel choices, and evaluation budgets) fixed. Smaller values such as $\alpha = 0.1$ and 0.5 slow the propagation of information across agents and can modestly affect early convergence, whereas larger values accelerate the consensus update. Among the larger settings, the differences are minor, though $\alpha = 10$ consistently yields slightly faster early progress without altering the overall convergence trend. Importantly, all tested values ultimately converge to nearly identical solution quality, indicating that ARCO-BO is not sensitive to fine-grained tuning of α.

The parameter λ_p governs how strongly agents influence one another based on the proximity of their predicted optimal solutions. To select meaningful values for the sensitivity study, we calibrate λ_p using the definition of the proximity similarity term Equation (14), We specify a tolerance level by requiring the similarity to decrease to 0.1 when the predicted optima of two agents differ by a fraction p of the input-domain range. Letting $d = p\Delta$ with $\Delta = 1$ in the normalized one-dimensional Sasena benchmark, we obtain

$$\exp(-\lambda_p d^2) = 0.1 \quad \Longrightarrow \quad \lambda_p = \frac{-\ln(0.1)}{d^2} = \frac{2.302585}{p^2}$$

This yields proximity-parameter values

$$\lambda_p \in \{23025.85,\ 920.96,\ 230.26,\ 57.56,\ 9.21\}$$

for $p \in \{0.01, 0.05, 0.10, 0.20, 0.50\}$, respectively. These values span a range from highly conservative information sharing (large λ_p) to much more permissive collaboration (small λ_p). Motivated by a moderate tolerance for predicted-optima misalignment, we adopt $p = 0.1$, which corresponds to $\lambda_p \approx 230.26$ in the Sasena benchmark. Empirically, the influence of λ_p is most pronounced in early iterations: larger values strengthen cross-agent information transfer when predicted optima are well aligned, leading to faster initial improvement, whereas smaller values result in more cautious updates with slightly wider variability. These differences, however, diminish over time, and all examined settings converge to similar final objective values. Based on this calibration and the desire to enable collaboration among agents whose predicted optima are reasonably aligned, we adopt the tolerance level $p = 0.1$, which corresponds to $\lambda_p \approx 230.26$ in the Sasena benchmark. As shown in Fig. 10, the convergence trajectories for different λ_p values differ only modestly, and the curve associated with $p = 0.1$ provides a stable and representative trajectory. Therefore, we use $p = 0.1$ as our default setting.

A default configuration of $\alpha = 10$ and the proximity parameter λ_p corresponding to the tolerance level $p = 0.1$ was used throughout all experiments and consistently produced stable and competitive performance. In practice, α and λ_p primarily influence the rate at which information flows between agents, while the final optimization outcomes remain consistent across a broad range of settings.

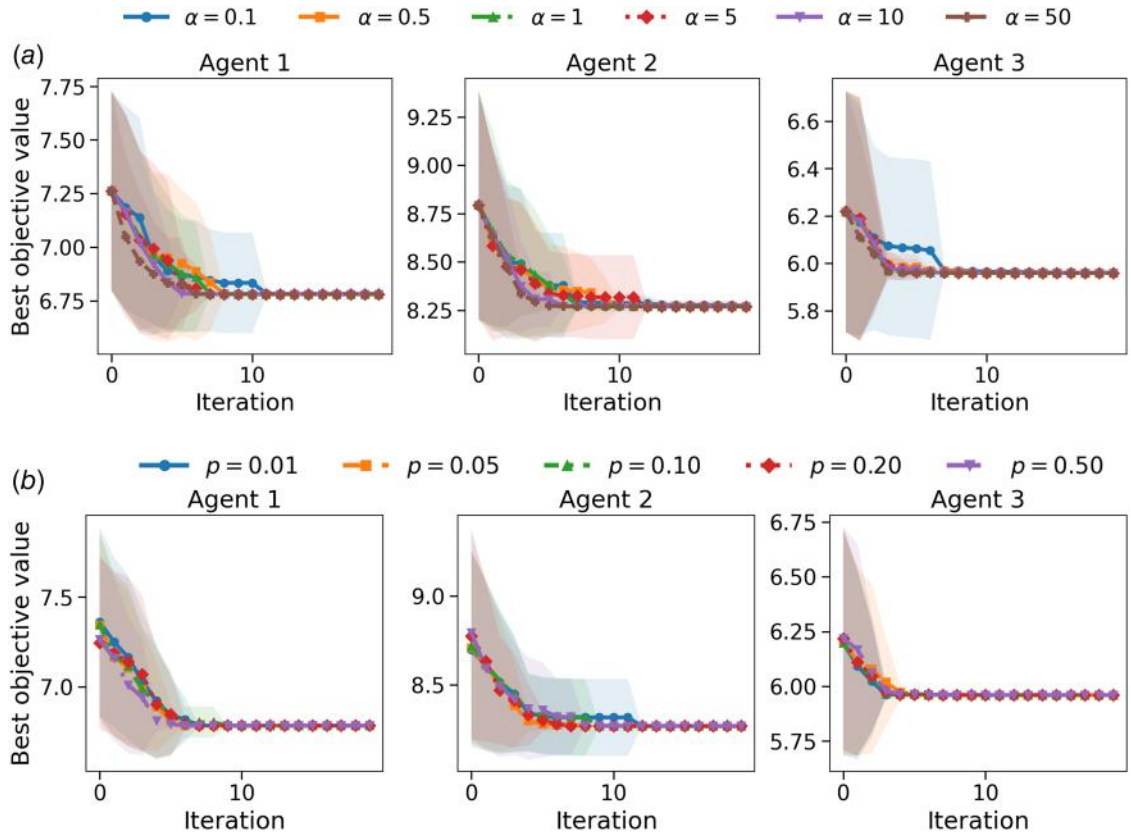

Fig. 10 Sensitivity of ARCO-BO to the hyperparameters α and λ_p, evaluated on the three-agent Sasena benchmark. (*a*) Varying $\alpha \in \{0.1, 0.5, 1, 5, 10, 50\}$ shows that smaller values slow early convergence, whereas larger values speed up information integration; all settings achieve similar final performance. (*b*) Varying the proximity tolerance $p \in \{0.01, 0.05, 0.10, 0.20, 0.50\}$ (corresponding to different λ_p values) produces only minor differences in early behavior, with nearly identical final outcomes.

Appendix C: Computational Overhead

To assess the additional cost introduced by the collaborative layer in ARCO-BO, we measured the average per-iteration wall-clock time across four representative case studies. As shown in Table 8, ARCO-BO incurs only a modest overhead of approximately 10–12% per iteration relative to running separate BO agents independently. This increase is small compared with the cost of GP model training and does not affect the practical usability of the method.

References

[1] Kvan, T., 2000, "Collaborative Design: What Is It?" Auto Construct., **9**(4), pp. 409–415.
[2] Chen, Y.-P., Wang, L., Comlek, Y., and Chen, W., 2024, "A Latent Variable Approach for Non-hierarchical Multi-Fidelity Adaptive Sampling," Comput. Methods Appl. Mech. Eng., **421**, p. 116773.
[3] Schmidt, J., Marques, M. R., Botti, S., and Marques, M. A., 2019, "Recent Advances and Applications of Machine Learning in Solid-State Materials Science," npj Comput. Mater., **5**(1), p. 83.
[4] Noack, M. M., and Sethian, J. A., 2021, "Autonomous Discovery in Science and Engineering," USDOE Office of Science (SC), Tech. Rep.
[5] Khatamsaz, D., Wagner, J., Vela, B., Arroyave, R., and Allaire, D. L., 2025, "Towards Autonomous Experimentation: Bayesian Optimization Over Problem Formulation Space for Accelerated Alloy Development," eprint arXiv:2502.05735.
[6] Coley, C. W., Eyke, N. S., and Jensen, K. F., 2020, "Autonomous Discovery in the Chemical Sciences Part I: Progress," Angew. Chem. Int. Ed., **59**(51), pp. 22858–22893.
[7] Talapatra, A., Boluki, S., Duong, T., Qian, X., Dougherty, E., and Arróyave, R., 2018, "Autonomous Efficient Experiment Design for Materials Discovery With Bayesian Model Averaging," Phys. Rev. Mater., **2**(11), p. 113803.
[8] Lookman, T., Balachandran, P. V., Xue, D., and Yuan, R., 2019, "Active Learning in Materials Science With Emphasis on Adaptive Sampling Using Uncertainties for Targeted Design," npj Comput. Mater., **5**(1), p. 21.
[9] Mourtzis, D., and Doukas, M., 2012, "Decentralized Manufacturing Systems Review: Challenges and Outlook," Robust Manufacturing Control: Proceedings of the CIRP Sponsored Conference RoMaC 2012, 18–20th June, Bremen, Springer, pp. 355–369.
[10] Kendrick, B. A., Dhokia, V., and Newman, S. T., 2017, "Strategies to Realize Decentralized Manufacture Through Hybrid Manufacturing Platforms," Rob. Comput.-Integr. Manuf., **43**, pp. 68–78.
[11] Yue, X., Liu, Y., Berahas, A. S., Johnson, B. N., and Al Kontar, R., 2025, "Collaborative and Distributed Bayesian Optimization Via Consensus," IEEE Trans. Autom. Sci. Eng., **22**, pp. 11343–11355.
[12] Juziuk, J., Weyns, D., and Holvoet, T., 2014, "Design Patterns for Multi-Agent Systems: A Systematic Literature Review," *Agent-Oriented Software Engineering*, O. Shehory and A. Sturm, eds., Springer, Berlin, Heidelberg, pp. 79–99.
[13] DeLoach, S. A., and Kumar, M., 2005, "Multi-Agent Systems Engineering: An Overview and Case Study," Agent-Orient. Methodol., pp. 317–340.
[14] Al Kontar, R., 2024, "Collaborative and Federated Black-Box Optimization: A Bayesian Optimization Perspective," 2024 IEEE International Conference on Big Data (BigData), Washington, DC, IEEE, pp. 7854–7859.
[15] Yang, T., Yi, X., Wu, J., Yuan, Y., Wu, D., Meng, Z., Hong, Y., Wang, H., Lin, Z., and Johansson, K. H., 2019, "A Survey of Distributed Optimization," Ann. Rev. Control, **47**, pp. 278–305.
[16] Chang, T.-H., Hong, M., and Wang, X., 2014, "Multi-Agent Distributed Optimization Via Inexact Consensus ADMM," IEEE Trans. Signal Process., **63**(2), pp. 482–497.
[17] Nedic, A., and Ozdaglar, A., 2009, "Distributed Subgradient Methods for Multi-Agent Optimization," IEEE Trans. Autom. Contruct., **54**(1), pp. 48–61.
[18] Amirkhani, A., and Barshooi, A. H., 2022, "Consensus in Multi-Agent Systems: A Review," Artif. Intell. Rev., **55**(5), pp. 3897–3935.
[19] Qin, J., Ma, Q., Shi, Y., and Wang, L., 2016, "Recent Advances in Consensus of Multi-Agent Systems: A Brief Survey," IEEE Trans. Ind. Electron., **64**(6), pp. 4972–4983.
[20] Li, Y., and Tan, C., 2019, "A Survey of the Consensus for Multi-Agent Systems," Syst. Sci. Control Eng., **7**(1), pp. 468–482.
[21] Olfati-Saber, R., Fax, J. A., and Murray, R. M., 2007, "Consensus and Cooperation in Networked Multi-Agent Systems," Proc. IEEE, **95**(1), pp. 215–233.
[22] Semsar-Kazerooni, E., and Khorasani, K., 2009, "Multi-Agent Team Cooperation: A Game Theory Approach," Automatica, **45**(10), pp. 2205–2213.
[23] Boella, G., and van der Torre, L., 2007, "A Game-Theoretic Approach to Normative Multi-agent Systems," Normat. Multi-Agent Syst., **7122**, pp. 1–35.
[24] Niazi, M., and Hussain, A., 2011, "Agent-Based Computing From Multi-agent Systems to Agent-Based Models: a Visual Survey," Scientometrics, **89**(2), pp. 479–499.
[25] Cardoso, R. C., and Ferrando, A., 2021, "A Review of Agent-Based Programming for Multi-Agent Systems," Computers, **10**(2), p. 16.
[26] Canese, L., Cardarilli, G. C., Di Nunzio, L., Fazzolari, R., Giardino, D., Re, M., and Spanò, S., 2021, "Multi-Agent Reinforcement Learning: A Review of Challenges and Applications," Appl. Sci., **11**(11), p. 4948.
[27] Buşoniu, L., Babuška, R., and De Schutter, B., 2010, "Multi-Agent Reinforcement Learning: An Overview," *Innovations in Multi-agent Systems and Applications-1*, A. Håkansson, N. T. Nguyen, R. Katarzyniak, and J. Kacprzyk, eds., Springer, Berlin, Heidelberg, pp. 183–221.
[28] Zhang, K., Yang, Z., and Başar, T., 2021, "Multi-Agent Reinforcement Learning: A Selective Overview of Theories and Algorithms," *Handbook of Reinforcement Learning and Control*, K. G. Vamvoudakis, Y. Wan, F. L. Lewis, and D. Cansever, eds., Springer International Publishing, Cham, Switzerland, pp. 321–384.
[29] Frazier, P. I., 2018, "A Tutorial on Bayesian Optimization," preprint arXiv:1807.02811.
[30] Pelikan, M., 2005, *Hierarchical Bayesian Optimization Algorithm*, Springer, Berlin, pp. 105–129.
[31] Frazier, P. I., 2018, "Bayesian Optimization," Recent Advances in Optimization and Modeling of Contemporary Problems, Informs, H. J. Greenberg, J. C. Smith, and A. J. Schaefer, eds., INFORMS Pubsonline, Catonsville, MD, pp. 255–278.
[32] Seeger, M., 2004, "Gaussian Processes for Machine Learning," Int. J. Neural Syst., **14**(2), pp. 69–106.
[33] Chen, Q., Jiang, L., Qin, H., and Kontar, R. A., 2025, "Multi-Agent Collaborative Bayesian Optimization Via Constrained Gaussian Processes," Technometrics, **67**(1), pp. 32–45.
[34] Zhan, D., Zhang, H., Righter, R., Zheng, Z., and Anderson, J., 2025, "Collaborative Bayesian Optimization Via Wasserstein Barycenters," 2025 IEEE 64th Conference on Decision and Control (CDC), IEEE, pp. 6284–6291.
[35] Ravi, S. K., Comlek, Y., Pathak, A., Gupta, V., Umretiya, R., Hoffman, A., Pilania, G., et al., 2025, "Interpretable Multi-Source Data Fusion Through Latent Variable Gaussian Process," Eng. Appl. Artif. Intell., **145**, p. 110033.
[36] Zhang, Y., Tao, S., Chen, W., and Apley, D. W., 2020, "A Latent Variable Approach to Gaussian Process Modeling With Qualitative and Quantitative Factors," Technometrics, **62**(3), pp. 291–302.
[37] Comlek, Y., Ravi, S. K., Pandita, P., Ghosh, S., Wang, L., and Chen, W., 2025, "Heterogeneous Multi-Source Data Fusion Through Input Mapping and Latent Variable Gaussian Process," J. Mech. Des., **147**(4), p. 041711.
[38] 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.
[39] Mockus, J., 2005, "The Bayesian Approach to Global Optimization," System Modeling and Optimization: Proceedings of the 10th IFIP Conference, New York City, Aug. 31–Sept. 4, 1981, Springer, pp. 473–481.
[40] Močkus, J., 1975, "On Bayesian Methods for Seeking the Extremum," Optimization Techniques IFIP Technical Conference, Novosibirsk, July 1–7, 1974, Springer, pp. 400–404.
[41] Snoek, J., Larochelle, H., and Adams, R. P., 2012, "Practical Bayesian Optimization of Machine Learning Algorithms," Adv. Neural Inform. Proc. Syst., **25**, pp. 2951–2959.
[42] Dolar, T., Mignerot, F., Wang, Z., Howard, H. C., Kassner, C. T., Wadley, H. N., Gianola, D. S., and Chen, W., 2026, "Accelerating Materials Discovery in Heterogeneous Ccomposition-Property Design Spaces Via Collaborative Bayesian Optimization," Materials, **261**, p. 115331.
[43] Wang, Z., Hutter, F., Zoghi, M., Matheson, D., and De Feitas, N., 2016, "Bayesian Optimization in a Billion Dimensions Via Random Embeddings," J. Artif. Intell. Res., **55**(1), pp. 361–387.
[44] Forrester, A., Sobester, A., and Keane, A., 2008, *Engineering Design Via Surrogate Modelling: A Practical Guide*, John Wiley & Sons, Hoboken, NJ.
[45] Sinkhorn, R., 1964, "A Relationship Between Arbitrary Positive Matrices and Doubly Stochastic Matrices," Anna. Math. Stat., **35**(2), pp. 876–879.
[46] Li, L., Fan, Y., Tse, M., and Lin, K.-Y., 2020, "A Review of Applications in Federated Learning," Comput. Ind. Eng., **149**, p. 106854.
[47] Mammen, P. M., 2021, "Federated Learning: Opportunities and Challenges," preprint arXiv:2101.05428.
[48] Grosnit, A., Cowen-Rivers, A. I., Tutunov, R., Griffiths, R.-R., Wang, J., and Bou-Ammar, H., 2021, "Are We Forgetting About Compositional Optimisers in Bayesian Optimisation?" J. Mach. Learn. Res., **22**(160), pp. 1–78.
[49] Xie, Z., and Chen, L., 2025, "Merge Kernel for Bayesian Optimization on Permutation Space," preprint arXiv:2507.13263.
[50] Lin, Q., Gong, L., Zhang, Y., Kou, M., and Zhou, Q., 2022, "A Probability of Improvement-Based Multi-Fidelity Robust Optimization Approach for Aerospace Products Design," Aerosp. Sci. Technol., **128**, p. 107764.
[51] Ackley, D., 2012, *A Connectionist Machine for Genetic Hillclimbing*, Vol. 28, Springer Science & Business Media, Heidelberg.
[52] Gramacy, R. B., Gray, G. A., Digabel, S. L., Lee, H. K. H., Ranjan, P., Wells, G., and Wild, S. M., 2016, "Modeling an Augmented Lagrangian for Blackbox Constrained Optimization," Technometrics, **58**(1), pp. 1–11.
[53] Gelbart, M. A., Snoek, J., and Adams, R. P., 2014, "Bayesian Optimization With Unknown Constraints," preprint arXiv:1403.5607[cs, stat].
[54] Gardner, J. R., Kusner, M. J., and Jake, G., 2014, "Bayesian Optimization With Inequality Constraints," Proceedings of the 31st International Conference on Machine Learning Series, PMLR, vol. 32, pp. 937–945.
[55] Schonlau, M., Welch, W. J., and Jones, D. R., 1998, "Global Versus Local Search in Constrained Optimization of Computer Models," *New Developments and Applications in Experimental Design*, N. Flournoy, W. F. Rosenberger, and W. K. Wong, eds., Vol. 34, Institute of Mathematical Statistics, Hayward, CA, pp. 11–26.
[56] Nguyen, Q. P., Chew, W. T. R., Song, L., Low, B. K. H., and Jaillet, P., 2023, "Optimistic Bayesian Optimization With Unknown Constraints," International Conference on Learning Representations (ICLR).
[57] Bergmann, D., and Graichen, K., 2020, "Safe Bayesian Optimization Under Unknown Constraints," 2020 59th IEEE Conference on Decision and Control (CDC), Virtual, Online, IEEE, pp. 3592–3597.