Output Space Entropy Search Framework for Multi-Objective Bayesian Optimization

Syrine Belakaria   SYRINE.BELAKARIA@WSU.EDU
Aryan Deshwal   ARYAN.DESHWAL@WSU.EDU
Janardhan Rao Doppa   JANA.DOPPA@WSU.EDU
School of Electrical Engineering and Computer Science
Washington State University
Pullman, Washington 99163, USA

Abstract

We consider the problem of black-box multi-objective optimization (MOO) using expensive function evaluations (also referred to as experiments), where the goal is to approximate the true Pareto set of solutions by minimizing the total resource cost of experiments. For example, in hardware design optimization, we need to find the designs that trade-off performance, energy, and area overhead using expensive computational simulations. The key challenge is to select the sequence of experiments to uncover high-quality solutions using minimal resources. In this paper, we propose a general framework for solving MOO problems based on the principle of output space entropy (OSE) search: select the experiment that maximizes the information gained per unit resource cost about the true Pareto front. We appropriately instantiate the principle of OSE search to derive efficient algorithms for the following four MOO problem settings: 1) The most basic *single-fidelity setting*, where experiments are expensive and accurate; 2) Handling *black-box constraints* which cannot be evaluated without performing experiments; 3) The *discrete multi-fidelity setting*, where experiments can vary in the amount of resources consumed and their evaluation accuracy; and 4) The *continuous-fidelity setting*, where continuous function approximations result in a huge space of experiments. Experiments on diverse synthetic and real-world benchmarks show that our OSE search based algorithms improve over state-of-the-art methods in terms of both computational-efficiency and accuracy of MOO solutions.

1. Introduction

Many engineering and scientific applications involve making design choices to optimize multiple objectives. Some examples include tuning the knobs of a compiler to optimize performance and efficiency of a set of software programs; designing new materials to optimize strength, elasticity, and durability; and designing hardware to optimize performance, power, and area. There are a few common challenges in solving these kind of multi-objective optimization (MOO) problems: **1)** The objective functions are unknown and we need to perform expensive experiments to evaluate each candidate design choice, where expense is measured in terms of the consumed resources (physical or computational). For example, performing computational simulations and physical lab experiments for hardware optimization and material design applications respectively. **2)** The objectives are conflicting in nature and all of them cannot be optimized simultaneously. Therefore, we need to find the *Pareto optimal* set of solutions. A solution is called Pareto optimal if it cannot be improved in any of the objectives without compromising some other objective. **3)** The solutions may need to

satisfy black-box constraints, which cannot be evaluated without performing experiments. For example, in aviation power system design applications, we need to find the designs that trade-off total energy and mass while satisfying specific thresholds for motor temperature and voltage of cells. **4)** We have the ability to perform multi-fidelity experiments (discrete or continuous) to evaluate objective functions via cheaper approximations, which vary in the amount of resources consumed and their accuracy. For example, in hardware design optimization, we can use multi-fidelity simulators for design evaluations. We want to leverage this additional freedom to reduce the overall cost for optimization. Real-world MOO problems come with two or more of the above challenges and the overall goal is to approximate the optimal Pareto set while minimizing the total resource cost of conducted experiments.

Bayesian Optimization (BO) (Shahriari, Swersky, Wang, Adams, & De Freitas, 2016) is an effective framework to solve black-box optimization problems with expensive function evaluations. The key idea behind BO is to build a cheap surrogate statistical model, e.g., Gaussian Process (Williams & Rasmussen, 2006), using the real experimental data; and employ it to intelligently select the sequence of experiments or function evaluations using an acquisition function, e.g., expected improvement (EI) and upper-confidence bound (UCB). There is a large body of literature on single-objective BO algorithms (Shahriari et al., 2016) and their applications including hyper-parameter tuning of machine learning methods (Snoek, Larochelle, & Adams, 2012; Kotthoff, Thornton, Hoos, Hutter, & Leyton-Brown, 2017). However, there is relatively less work on the more challenging problem of BO for multiple objective functions (first and second challenges) (Hernández-Lobato, Hernandez-Lobato, Shah, & Adams, 2016), very limited work on the constrained multi-objective optimization problem (third challenge), and no prior work on multi-objective optimization in the multi-fidelity setting (fourth challenge). *To the best of our knowledge, this is the first work on discrete and continuous-fidelity settings for multi-objective BO within the ML literature* as discussed in the related work section.

Prior work on multi-objective BO is lacking in the following ways. Many algorithms reduce the problem to single-objective optimization by designing appropriate acquisition functions, e.g., expected improvement in Pareto hypervolume (Knowles, 2006; Emmerich & Klinkenberg, 2008). This can potentially lead to aggressive exploitation behavior. Additionally, algorithms to optimize Pareto Hypervolume (PHV) based acquisition functions scale poorly as the number of objectives and the dimensionality of input space grows. There are also methods that rely on *input space entropy* based acquisition function (Hernández-Lobato et al., 2016) to select the candidate inputs for evaluation. However, it is computationally expensive to approximate and optimize this acquisition function.

In this paper, we study a general framework for solving a large-class of black-box MOO problems based on the principle of *output space entropy (OSE)* search (Wang & Jegelka, 2017; Hoffman & Ghahramani, 2015). Our work is inspired by the prior success of the OSE principle for solving single-objective BO problems and is an extension of Wang and Jegelka (2017) to several multi-objective optimization settings. The key idea is to select the input and fidelity vector (if applicable) that maximizes the information gain per unit resource cost about the optimal Pareto front in each iteration. Output space entropy search has many advantages over algorithms based on input space entropy search (Belakaria, Deshwal, & Doppa, 2019): a) it allows much tighter approximation; b) it is cheaper to compute; and c) it naturally lends itself to robust optimization with respect to the number of samples

used for acquisition function computation. We appropriately instantiate the OSE principle to derive efficient algorithms for solving four qualitatively different MOO problems: the most basic single-fidelity setting (Belakaria et al., 2019), MOO with black-box constraints, discrete multi-fidelity setting (Belakaria, Deshwal, & Doppa, 2020a), and continuous-fidelity setting. Comprehensive experiments on diverse synthetic and real-world benchmarks show that our OSE search based algorithms are computationally-efficient and perform better than the state-of-the-art algorithms.

Contributions. The main contribution of this paper is the development and evaluation of multi-objective BO algorithms based on the principle of output space entropy search for four different MOO problem settings. Specific contributions include the following:

- Development of an approach referred to as MESMO to solve the most basic MOO problem in the single-fidelity setting, where experiments are expensive and accurate (Belakaria et al., 2019).

- Development of an approach referred to as MESMOC to handle MOO problems with black-box constraints, which cannot be evaluated without performing experiments.

- Development of an approach referred to as MF-OSEMO to solve MOO problems in the discrete multi-fidelity setting, where experiments can vary in the amount of resources consumed and their evaluation accuracy (Belakaria et al., 2020a).

- Development of an approach referred to as iMOCA to solve MOO problems in the continuous-fidelity setting, where continuous function approximations result in a huge space of experiments with varying cost. We provide two qualitatively different approximations for iMOCA.

- Experimental evaluation on diverse synthetic and real-world benchmark problems to demonstrate the effectiveness of the proposed algorithms over existing MOO algorithms and a naive continuous-fidelity baseline.

- Open-source code for all methods: MESMO[1], MESMOC[2], MF-OSEMO[3], and iMOCA[4]

2. Background and Problem Setup

In this section, we first provide an overview of the generic Bayesian optimization framework. Next, we formally define the different MOO problem settings considered in this work.

2.1 Bayesian Optimization Framework

Bayesian Optimization (BO) is a very efficient framework to solve global optimization problems using *black-box evaluations of expensive objective functions*. Let $\mathfrak{X} \subseteq \Re^d$ be an input space. In the single-objective BO formulation, we are given an unknown real-valued objective function $f : \mathfrak{X} \mapsto \Re$, which can evaluate each input $\mathbf{x} \in \mathfrak{X}$ to produce an evaluation $y = f(\mathbf{x})$. Each evaluation $f(\mathbf{x})$ is expensive in terms of the consumed resources. The main

---

1. github.com/belakaria/MESMO
2. github.com/belakaria/MESMOC
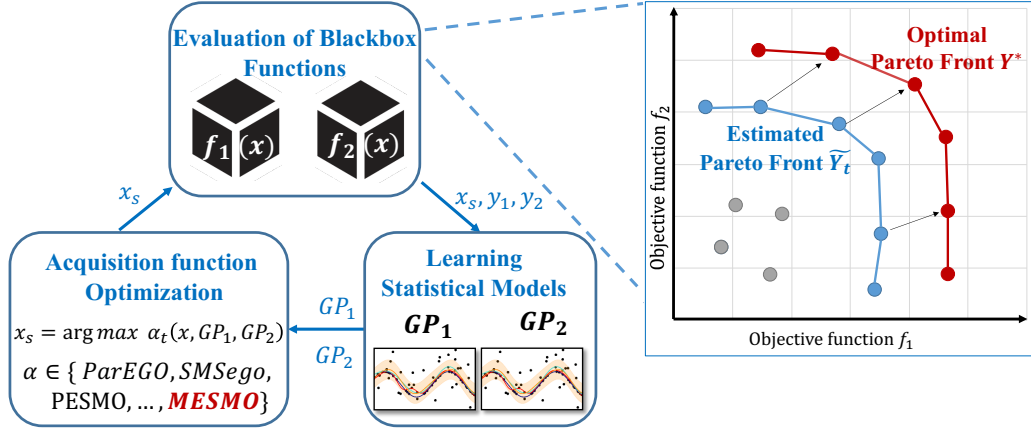3. github.com/belakaria/MF-OSEMO
4. github.com/belakaria/iMOCA

Figure 1: Overview of the Bayesian optimization process for two objective functions (k=2).

goal is to find an input $\mathbf{x}^* \in \mathfrak{X}$ that approximately optimizes f by performing a limited number of function evaluations. BO algorithms learn a cheap surrogate model from training data obtained from past function evaluations. They intelligently select the next input for evaluation by trading-off exploration and exploitation to quickly direct the search towards optimal inputs. The three key elements of BO framework are:

1) Statistical Model of the true function $f(x)$. *Gaussian Process (GP)* (Williams & Rasmussen, 2006) is the most commonly used model. A GP over a space $\mathfrak{X}$ is a random process from $\mathfrak{X}$ to $\mathfrak{R}$. It is characterized by a mean function $\mu : \mathfrak{X} \mapsto \mathfrak{R}$ and a covariance or kernel function $\kappa : \mathfrak{X} \times \mathfrak{X} \mapsto \mathfrak{R}$. If a function f is sampled from $\mathcal{GP}(\mu, \kappa)$, then $f(x)$ is distributed normally $\mathcal{N}(\mu(x), \kappa(x, x))$ for a set of inputs from $x \in \mathcal{X}$.

2) Acquisition Function (α) to score the utility of evaluating a candidate input $\mathbf{x} \in \mathfrak{X}$ based on the statistical model. Some popular acquisition functions in the single-objective literature include expected improvement (EI), upper confidence bound (UCB), predictive entropy search (PES) (Hernández-Lobato, Hoffman, & Ghahramani, 2014), and max-value entropy search (MES) (Wang & Jegelka, 2017).

3) Optimization Procedure to select the best scoring candidate input according to α depending on statistical model. DIRECT (Jones, Perttunen, & Stuckman, 1993) is a very popular approach for acquisition function optimization.

2.2 Multi-Objective Optimization Problem Setting Overview

Multi-objective optimization (MOO) problems can be formalized in terms of the following key elements: number of objectives, necessity to satisfy black-box constraints, and availability of cheaper approximations or fidelities (discrete/continuous) for function evaluations. Below we provide a brief overview of the four different MOO problem settings that are addressed in this paper noting that a more detailed problem setup is specified under each technical section.

Notation	Definition
$\mathbf{x}, \mathbf{y}, \mathbf{f}, \mathbf{m}$	bold notation represents vectors
$\mathbf{x}$	input vector of d dimensions
$[n]$	set of first n natural numbers $\{1, 2, \cdots, n\}$
$f_1, f_2, \cdots, f_K$	true objective functions
$C_1, C_2, \cdots, C_L$	Constraints functions
$\tilde{f}_j$	function sampled from the highest fidelity of the jth Gaussian process model
$\mathcal{X}$	Input space
I	Information gain
$\mathcal{Y}^*$	true pareto front of the objective functions $[f_1, f_2, \cdots, f_K]$
$\mathcal{Y}^*_s$	Pareto front of the sampled functions $[\tilde{f}_1, \tilde{f}_2, \cdots, \tilde{f}_K]$

Table 1: Table describing the general mathematical notations.

Basic Multi-objective Optimization Problem. The goal is to maximize real-valued objective functions $f_1(\mathbf{x}), f_2(\mathbf{x}), \cdots, f_K(\mathbf{x})$, with $K \geq 2$, over continuous space $\mathfrak{X} \subseteq \Re^d$. Each evaluation (also called an experiment) of an input $\mathbf{x} \in \mathfrak{X}$ produces a vector of objective values $\mathbf{y} = (y_1, y_2, \cdots, y_K)$ where $y_i = f_i(x)$ for all $i \in \{1, 2, \cdots, K\}$.

MOO Problem with Constraints. This is a generalization of the basic MOO problem, where we need to satisfy some black-box constraints. Our goal is to maximize real-valued objective functions $f_1(\mathbf{x}), f_2(\mathbf{x}), \cdots, f_K(\mathbf{x})$, with $K \geq 2$, while satisfying L black-box constraints of the form $C_1(x) \geq 0, C_2(x) \geq 0, \cdots, C_L(x) \geq 0$ over continuous space $\mathfrak{X} \subseteq \Re^d$. Each evaluation of an input $\mathbf{x} \in \mathfrak{X}$ produces a vector of objective values and constraint values $\mathbf{y} = (y_{f_1}, y_{f_2}, \cdots, y_{f_K}, y_{c_1} \cdots y_{c_L})$ where $y_{f_j} = f_j(x)$ for all $j \in \{1, 2, \cdots, K\}$ and $y_{c_i} = C_i(x)$ for all $i \in \{1, 2, \cdots, L\}$.

MOO Problem with Discrete Multi-fidelity Experiments. This is a general version of the MOO problem, where we have access to M_j fidelities for each function f_j that vary in the amount of resources consumed and the accuracy of evaluation. The evaluation of an input $\mathbf{x} \in \mathfrak{X}$ with fidelity vector $\mathbf{m} = [m_1, m_2, \cdots, m_K]$ produces an evaluation vector of K values denoted by $\mathbf{y}^{\mathbf{m}} \equiv [y_1^{(m_1)}, \cdots, y_K^{(m_K)}]$, where $y_j^{(m_j)} = f_j^{(m_j)}(x)$ for all $j \in \{1, 2, \cdots, K\}$.

MOO Problem with Continuous-fidelity Experiments. In this general version of the multi-fidelity setting, we have access to $g_i(\mathbf{x}, z_i)$ where g_i is an alternative function through which we can evaluate cheaper approximations of f_i by varying the fidelity variable $z_i \in \mathcal{Z}$ (continuous function approximations). The evaluation of an input $\mathbf{x} \in \mathcal{X}$ with fidelity vector $\mathbf{z} = [z_1, z_2, \cdots, z_K]$ produces an evaluation vector of K values denoted by $\mathbf{y} \equiv [y_1, y_2, \cdots, y_K]$, where $y_i = g_i(\mathbf{x}, z_i)$ for all $i \in \{1, 2, \cdots, K\}$.

3. Related Work

In this section, we discuss prior work from the BO literature that is related to the four MOO problem settings considered in this paper.

Single-fidelity Multi-Objective Optimization. There is a family of model based multi-objective BO algorithms that reduce the problem to single-objective optimization. The ParEGO method (Knowles, 2006) employs random scalarization for this purpose: scalar

weights of K objective functions are sampled from a uniform distribution to construct a single-objective function and expected improvement is employed as the acquisition function to select the next input for evaluation. ParEGO is simple and fast, but more advanced approaches often outperform it. Many methods optimize the Pareto hypervolume (PHV) metric (Emmerich & Klinkenberg, 2008) that captures the quality of a candidate Pareto set. This is done by extending the standard acquisition functions to PHV objective, e.g., expected improvement in PHV (EHI) (Emmerich & Klinkenberg, 2008) and probability of improvement in PHV (SUR) (Picheny, 2015). Unfortunately, algorithms to optimize PHV based acquisition functions scale very poorly and are not feasible for more than two objectives. SMSego is a relatively faster method (Ponweiser, Wagner, Biermann, & Vincze, 2008). To improve scalability, the gain in hypervolume is computed over a limited set of points: SMSego finds those set of points by optimizing the posterior means of the GPs. A common drawback of this family of algorithms is that reduction to single-objective optimization can potentially lead to more exploitation behavior resulting in sub-optimal solutions.

PAL (Zuluaga, Sergent, Krause, & Püschel, 2013), PESMO (Hernández-Lobato et al., 2016), and the concurrent works USeMO (Belakaria, Deshwal, Jayakodi, & Doppa, 2020b) and MESMO (Belakaria et al., 2019) are principled algorithms based on information theory. PAL tries to classify the input points based on the learned models into three categories: Pareto optimal, non-Pareto optimal, and uncertain. In each iteration, it selects the candidate input for evaluation towards the goal of minimizing the size of uncertain set. PAL provides theoretical guarantees, but it is only applicable for input space $\mathfrak{X}$ with finite set of discrete points. USeMO is a general framework that iteratively generates a cheap Pareto front using the surrogate models and then selects the input with highest uncertainty for evaluation. PESMO (Hernández-Lobato et al., 2016) relies on input space entropy based acquisition function and iteratively selects the input that maximizes the information gained about the optimal Pareto set $\mathcal{X}^*$. Unfortunately, optimizing this acquisition function poses significant challenges: a) it requires a series of approximations, which can be potentially sub-optimal; b) the optimization, even after approximations, is expensive; and c) the performance is strongly dependent on the number of Monte-Carlo samples. In comparison, our proposed output space entropy based acquisition function partially overcomes the above challenges, and allows efficient and robust optimization with respect to the number of samples used for acquisition function computation. More specifically, the time complexities of acquisition function computation in PESMO and MESMO ignoring the time to solve the cheap MO problem that is common for both algorithms are $\mathcal{O}(SKm^3)$ and $\mathcal{O}(SK)$ respectively, where S is the number of Monte-Carlo samples, K is the number of objectives, and m is the size of the sample Pareto set in PESMO. In fact, PESMO formulation relies on an expensive and high-dimensional ($l \cdot d$ dimensions) distribution over the input space, where l is size of the optimal Pareto set $\mathcal{X}^*$ while MESMO relies on a computationally cheap and low-dimensional distribution over the output space ($l \cdot K$ dimensions, which is considerably less than $l \cdot$d as $K \ll d$ in practice). Additionally, Belakaria et al. (2019) demonstrated that MESMO is very robust and performs very well even with one sample.

Constrained Multi-Objective Optimization. There exists very limited prior work to address constrained MO problems (Garrido-Merchán & Hernández-Lobato, 2019; Feliot, Bect, & Vazquez, 2017). PESMOC (Garrido-Merchán & Hernández-Lobato, 2019) is the

current state-of-the-art method for this problem setting. PESMOC extends the information-theoretic approach PESMO that relies on the principle of input space entropy search to the constrained setting. As a consequence, it inherits the drawbacks of PESMO. Our proposed MESMOC algorithm based on OSE search is intended to improve over PESMOC. MES-MOC+ (Fernández-Sánchez, Garrido-Merchán, & Hernández-Lobato, 2020) is a concurrent work that also employs the principle of output space entropy search to solve constrained multi-objective optimization problems. However, this paper uses a completely different approximation of the information gain leading to a different expression of the acquisition function. This method employs a series of complex mathematical approximations based on Assumed Density Filtering (ADF). Our proposed MESMOC algorithm uses the truncated Gaussian distribution approximation that results in a closed-form expression, fast, and easy to implement acquisition function. Additionally, the ADF based method (Fernández-Sánchez et al., 2020) considers blackbox constraints only in the acquisition function definition while MESMOC addresses the constraints both in the acquisition function expression and in the acquisition function optimization to ensure the selection of valid inputs.

Multi-fidelity Single-Objective Optimization. Acquisition functions (AFs) for single-fidelity and single-objective BO are extensively studied (Shahriari et al., 2016). AFs can be broadly classified into two categories. First, *myopic* AFs rely on improving a "local" measure of utility (e.g., expected improvement). Second, *non-myopic* AFs measure the "global" utility of evaluating a candidate input for solving the black-box optimization problem (e.g., predictive entropy search). Canonical examples of myopic acquisition function include expected improvement (EI) and upper-confidence bound (UCB). EI was extended to multi-fidelity setting (Huang, Allen, Notz, & Miller, 2006; Picheny, Ginsbourger, & et al., 2013a; Lam, Allaire, & et al, 2015). The popular GP-UCB method (Srinivas, Krause, Kakade, & Seeger, 2009) was also extended to multi-fidelity setting with discrete fidelities (Kandasamy, Dasarathy, Oliva, & et al, 2016) and continuous fidelities (Kandasamy, Dasarathy, Schneider, & Poczos, 2017). Entropy based methods fall under the category of non-myopic AFs. Some examples include entropy search (ES) (Hennig & Schuler, 2012) and predictive entropy search (PES) (Hernández-Lobato et al., 2014). Their multi-fidelity extensions include MT-ES (Swersky, Snoek, & Adams, 2013; Klein, Falkner, Bartels, Hennig, & Hutter, 2017) and MF-PES (Zhang, Hoang, & et al, 2017; McLeod, Osborne, & Roberts, 2017). Unfortunately, they inherit the computational difficulties of the original ES and PES. Max-value entropy search (MES) (Wang & Jegelka, 2017) and output space predictive entropy search (Hoffman & Ghahramani, 2015) are recent approaches that rely on the principle of output space entropy (OSE) search. Prior work (Wang & Jegelka, 2017) has shown advantages of OSE search in terms of compute-time, robustness, and accuracy over input space entropy search methods. Recent work (Song, Chen, & Yue, 2019) proposed a general approach based on mutual information. Takeno, Fukuoka, Tsukada, Koyama, Shiga, Takeuchi, and Karasuyama (2019) extended MES to multi-fidelity setting and showed its effectiveness over MF-PES. MUMBO (Moss, Leslie, & Rayson, 2020) extended MES to the continuous-fidelity and multi-task setting.

Multi-fidelity Multi-Objective Optimization. Prior work outside ML literature has considered domain-specific methods that employ single-fidelity multi-objective approaches

in the context of multi-fidelity setting by using the lower fidelities *only as an initialization* (Kontogiannis, Demange, Kipouros, & et al., 2018; Ariyarit & et al., 2017). Specifically, Ariyarit and et al. (2017) employs the single-fidelity algorithm based on expected hypervolume improvement acquisition function and Kontogiannis et al. (2018) employs an algorithm that is very similar to SMSego. Also, both these methods model all fidelities with the same GP and assume that higher fidelity evaluation is a sum of lower-fidelity evaluation and offset error. These are strong assumptions and may not hold in general multi-fidelity settings including the problems from our experimental evaluation. Our proposed MF-OSEMO (Belakaria et al., 2020a) and iMOCA algorithms (generalized versions of MESMO (Belakaria et al., 2019) solve MOO problem in discrete and continuous-fidelity settings respectively using the principle of output space entropy search and leverage some technical ideas from the prior work on single-objective optimization. We are not aware of any prior work on generic discrete/continuous-fidelity algorithms for MOO problems in the BO literature.

4. MESMO Algorithm for the Basic MOO Problem

In this section, we address the most basic MOO problem in the single-fidelity setting, where the goal is to optimize multiple black-box objective functions. To solve this problem, we propose an algorithm referred to as **M**ax-value **E**ntropy **S**earch for **M**ulti-objective **O**ptimization (MESMO). In what follows, we first describe the problem setup and surrogate models. Next, we mathematically describe the output space entropy based acquisition function and provide an algorithmic approach to efficiently compute it.

Problem Setup (Basic Multi-Objective Optimization Problem). The goal is to maximize real-valued objective functions $f_1(\mathbf{x}), f_2(\mathbf{x}), \cdots, f_K(\mathbf{x})$, with $K \geq 2$, over continuous space $\mathfrak{X} \subseteq \Re^d$. Each evaluation (also called an experiment) of an input $\mathbf{x} \in \mathfrak{X}$ produces a vector of objective values $\mathbf{y} = (y_1, y_2, \cdots, y_K)$ where $y_i = f_i(x)$ for all $i \in \{1, 2, \cdots, K\}$. We say that an point $\mathbf{x}$ *Pareto-dominates* another point $\mathbf{x}'$ if $f_i(\mathbf{x}) \geq f_i(\mathbf{x}') \ \forall i$ and there exists some $j \in \{1, 2, \cdots, K\}$ such that $f_j(\mathbf{x}) > f_j(\mathbf{x}')$. The optimal solution of MOO problem is a set of points $\mathcal{X}^* \subset \mathfrak{X}$ such that no point $\mathbf{x}' \in \mathfrak{X} \setminus \mathcal{X}^*$ Pareto-dominates a point $\mathbf{x} \in \mathcal{X}^*$. The solution set $\mathcal{X}^*$ is called the optimal *Pareto set* and the corresponding set of function values $\mathcal{Y}^*$ is called the optimal *Pareto front*. The goal of multi-objective BO is to approximate $\mathcal{X}^*$ while minimizing the number of expensive function evaluations. In the application of hardware design optimization, $\mathbf{x} \in \mathfrak{X}$ is a candidate hardware design; evaluation of of design $\mathbf{x}$ to get output objectives such as power, performance, and area involve performing computationally-expensive simulation to mimic the real hardware; and our goal is to find the optimal Pareto set of hardware designs to trade-off power, performance, and area. Table 1 contains all the mathematical notations used in this section.

Surrogate Models. Gaussian processes (GPs) are shown to be effective surrogate models in prior work on single and multi-objective BO (Hernández-Lobato et al., 2014; Wang, Zhou, & Jegelka, 2016; Wang & Jegelka, 2017; Srinivas et al., 2009; Hernández-Lobato et al., 2016). Similar to prior work (Hernández-Lobato et al., 2016), we model the objective functions $f_1, f_2, \cdots, f_K$ using K independent GP models $\mathcal{GP}_1, \mathcal{GP}_2, \cdots, \mathcal{GP}_K$ with zero mean and i.i.d. observation noise. Let $\mathcal{D} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{t-1}$ be the training data from past $t-1$ function evaluations, where $\mathbf{x}_i \in \mathfrak{X}$ is an input and $\mathbf{y}_i = \{y_1^i, y_2^i, \cdots, y_K^i\}$ is the output

vector resulting from evaluating functions $f_1, f_2, \cdots, f_K$ at $\mathbf{x}_i$. We learn surrogate models $\mathcal{GP}_1, \mathcal{GP}_2, \cdots, \mathcal{GP}_K$ from $\mathcal{D}$.

4.1 MESMO Algorithm

Output Space Entropy Based Acquisition Function. Input space entropy based methods like PESMO (Hernández-Lobato et al., 2016) selects the next candidate input $\mathbf{x}_t$ (for ease of notation, we drop the subscript in below discussion) by maximizing the information gain about the optimal Pareto set $\mathcal{X}^*$. The acquisition function based on input space entropy is given as follows:

$$\alpha(\mathbf{x}) = I(\{\mathbf{x}, \mathbf{y}\}, \mathcal{X}^* \mid D) \tag{4.1}$$

$$= H(\mathcal{X}^* \mid D) - \mathbb{E}_y[H(\mathcal{X}^* \mid D \cup \{\mathbf{x}, \mathbf{y}\})] \tag{4.2}$$

$$= H(\mathbf{y} \mid D, \mathbf{x}) - \mathbb{E}_{\mathcal{X}^*}[H(\mathbf{y} \mid D, \mathbf{x}, \mathcal{X}^*)] \tag{4.3}$$

Information gain is defined as the expected reduction in entropy $H(.)$[5] of the posterior distribution $P(\mathcal{X}^* \mid D)$ over the optimal Pareto set $\mathcal{X}^*$ as given in equations (4.2) and (4.3) (resulting from symmetric property of information gain). This mathematical formulation relies on an expensive and high-dimensional ($l \cdot d$ dimensions) distribution $P(\mathcal{X}^* \mid D)$, where l is size of the optimal Pareto set $\mathcal{X}^*$. Furthermore, optimizing the second term in r.h.s poses significant challenges: a) it requires a series of approximations (Hernández-Lobato et al., 2016) which can be potentially sub-optimal; and b) the optimization, even after approximations, is expensive c) the performance is strongly dependent on the number of Monte-Carlo samples.

To overcome the above challenges of computing input space entropy based acquisition function, we take an alternative route and propose to maximize the information gain about the optimal **Pareto front** $\mathcal{Y}^*$. This is equivalent to expected reduction in entropy over the Pareto front $\mathcal{Y}^*$, which relies on a computationally cheap and low-dimensional ($l \cdot K$ dimensions, which is considerably less than $l \cdot d$ as $K \ll d$ in practice) distribution $P(\mathcal{Y}^* \mid D)$. Our acquisition function that maximizes the information gain between the next candidate input for evaluation $\mathbf{x}$ and Pareto front $\mathcal{Y}^*$ is given as:

$$\alpha(\mathbf{x}) = I(\{\mathbf{x}, \mathbf{y}\}, \mathcal{Y}^* \mid D) \tag{4.4}$$

$$= H(\mathcal{Y}^* \mid D) - \mathbb{E}_y[H(\mathcal{Y}^* \mid D \cup \{\mathbf{x}, \mathbf{y}\})] \tag{4.5}$$

$$= H(\mathbf{y} \mid D, \mathbf{x}) - \mathbb{E}_{\mathcal{Y}^*}[H(\mathbf{y} \mid D, \mathbf{x}, \mathcal{Y}^*)] \tag{4.6}$$

The first term in the r.h.s of equation (4.6) (entropy of a factorizable K-dimensional Gaussian distribution $P(\mathbf{y} \mid D, \mathbf{x})$) can be computed in closed form as shown below:

$$H(\mathbf{y} \mid D, \mathbf{x}) = \frac{K(1 + \ln(2\pi))}{2} + \sum_{j=1}^{K} \ln(\sigma_j(\mathbf{x})) \tag{4.7}$$

where $\sigma_i^2(\mathbf{x})$ is the predictive variance of ith GP at input $\mathbf{x}$. The second term in the r.h.s of equation (4.6) is an expectation over the Pareto front $\mathcal{Y}^*$. We can approximately compute

---

5. The conditioning on D and $\mathbf{x}$ in $H(\mathbf{y} \mid D, \mathbf{x})$ is on fixed values and not random variables

this term via Monte-Carlo sampling as shown below:

$$\mathbb{E}_{\mathcal{Y}^*}[H(\mathbf{y} \mid D, \mathbf{x}, \mathcal{Y}^*)] \simeq \frac{1}{S} \sum_{s=1}^{S} [H(\mathbf{y} \mid D, \mathbf{x}, \mathcal{Y}_s^*)] \tag{4.8}$$

where S is the number of samples and $\mathcal{Y}_s^*$ denote a sample Pareto front. The main advantages of our acquisition function are: computational efficiency and robustness to the number of samples. Our experiments demonstrate these advantages over input space entropy based acquisition function.

There are two key algorithmic steps to compute equation (4.8). We want to know: 1) how to compute Pareto front samples $\mathcal{Y}_s^*$?; and 2) and how to compute the entropy with respect to a given Pareto front sample $\mathcal{Y}_s^*$? We provide solutions for these two questions.

1) Computing Pareto Front Samples via Cheap Multi-Objective optimization. To compute a Pareto front sample $\mathcal{Y}_s^*$, we first sample functions from the posterior GP models via random Fourier features (Hernández-Lobato et al., 2014; Rahimi & Recht, 2008) and then solve a cheap multi-objective optimization over the K sampled functions.

Sampling functions from posterior GP. Similar to prior work (Hernández-Lobato et al., 2014, 2016; Wang & Jegelka, 2017), we employ random Fourier features based sampling procedure. We approximate each GP prior as $\tilde{f} = \phi(\mathbf{x})^T \theta$, where $\theta \sim N(0, \mathbf{I})$. The key idea behind random Fourier features is to construct each function sample $\tilde{f}(\mathbf{x})$ as a finitely parametrized approximation: $\phi(\mathbf{x})^T \theta$, where θ is sampled from its corresponding posterior distribution conditioned on the data $\mathcal{D}$ obtained from past function evaluations: $\theta | \mathcal{D} \sim N(\mathbf{A}^{-1}\mathbf{\Phi}^{\mathbf{T}}\mathbf{y}_n, \sigma^2 \mathbf{A}^{-1})$, where $\mathbf{A} = \mathbf{\Phi}^{\mathbf{T}}\mathbf{\Phi} + \sigma^2 \mathbf{I}$ and $\Phi^T = [\phi(\mathbf{x}_1), \cdots, \phi(\mathbf{x}_{t-1})]$.

Cheap MO solver. We sample $\tilde{f}_i$ from GP model $\mathcal{GP}_i$ for each of the K functions as described above. A *cheap* multi-objective optimization problem over the K sampled functions $\tilde{f}_1, \tilde{f}_2, \cdots, \tilde{f}_k$ is solved to compute sample Pareto front $\mathcal{Y}_s^*$. This cheap multi-objective optimization also allows us to capture the interactions between different objectives. We employ the popular NSGA-II algorithm (Deb, Pratap, Agarwal, Meyarivan, & Fast, 2002a) to solve the MO problem with cheap objective functions noting that any other algorithm can be used to similar effect.

2) Entropy Computation with a Sample Pareto Front. Let $\mathcal{Y}_s^* = \{\mathbf{v}^1, \cdots, \mathbf{v}^l\}$ be the sample Pareto front, where l is the size of the Pareto front and each $\mathbf{v}^i = \{v_1^i, \cdots, v_K^i\}$ is a K-vector evaluated at the K sampled functions. The following inequality holds for each component y_j of the K-vector $\mathbf{y} = \{y_1, \cdots, y_K\}$ in the entropy term $H(\mathbf{y} \mid D, \mathbf{x}, \mathcal{Y}_s^*)$:

$$y_j \leq y_{j_s}^* \quad \forall j \in \{1, \cdots, K\} \tag{4.9}$$

where $y_{j_s}^* = \max\{v_j^1, \cdots v_j^l\}$. The inequality essentially says that the jth component of $\mathbf{y}$ (i.e., y_j) is upper-bounded by a value obtained by taking the maximum of jth components of all l K-vectors in the Pareto front $\mathcal{Y}_s^*$. This inequality can be proven by a contradiction argument. Suppose there exists some component y_j of $\mathbf{y}$ such that $y_j > y_{j_s}^*$. However, by definition, $\mathbf{y}$ is a non-dominated point because no point dominates it in the jth dimension. This results in $\mathbf{y} \in \mathcal{Y}_s^*$, which is a contradiction. Therefore, our hypothesis that $y_j > y_{j_s}^*$ is incorrect and inequality (4.9) holds.

By combining the inequality (4.9) and the fact that each function is modeled as a GP, we can approximate each component y_j as a truncated Gaussian distribution since the

676

distribution of y_j needs to satisfy $y_j \leq y_{j_s}^*$. Furthermore, a common property of entropy measure allows us to decompose the entropy of a set of independent variables into a sum over entropies of individual variables (Cover & Thomas, 2012):

$$H(\mathbf{y} \mid D, \mathbf{x}, \mathcal{Y}_s^*) = \sum_{j=1}^{K} H(y_j | D, \mathbf{x}, y_{j_s}^*) \qquad (4.10)$$

The r.h.s is a summation over entropies of K variables $\{y_1, \cdots, y_K\}$. The probability distribution of each variable y_j is a truncated Gaussian with upper bound $y_{j_s}^*$ (Michalowicz, Nichols, & Bucholtz, 2013). The differential entropy for each y_j is given as:

$$H(y_j \mid D, \mathbf{x}, \mathcal{Y}_s^*) \simeq \left[\frac{(1 + \ln(2\pi))}{2} + \ln(\sigma_j(\mathbf{x})) + \ln \Phi(\gamma_s^j(\mathbf{x})) - \frac{\gamma_s^j(\mathbf{x})\phi(\gamma_s^j(\mathbf{x}))}{2\Phi(\gamma_s^j(\mathbf{x}))} \right] \qquad (4.11)$$

equation (4.10) and equation (4.11) give the followong expression of $H(\mathbf{y} \mid D, \mathbf{x}, \mathcal{Y}_s^*)$.

$$H(\mathbf{y} \mid D, \mathbf{x}, \mathcal{Y}_s^*) \simeq \sum_{j=1}^{K} \left[\frac{(1 + \ln(2\pi))}{2} + \ln(\sigma_j(\mathbf{x})) + \ln \Phi(\gamma_s^j(\mathbf{x})) - \frac{\gamma_s^j(\mathbf{x})\phi(\gamma_s^j(\mathbf{x}))}{2\Phi(\gamma_s^j(\mathbf{x}))} \right] \qquad (4.12)$$

where $\gamma_s^j(x) = \frac{y_{j_s}^* - \mu_j(\mathbf{x})}{\sigma_j(\mathbf{x})}$, and ϕ and Φ are the p.d.f and c.d.f of a standard normal distribution respectively. By combining equations (4.7) and (4.12) with equation (4.6), we get the final form of our acquisition function as shown below:

$$\alpha(\mathbf{x}) \simeq \frac{1}{S} \sum_{s=1}^{S} \sum_{j=1}^{K} \left[\frac{\gamma_s^j(\mathbf{x})\phi(\gamma_s^j(\mathbf{x}))}{2\Phi(\gamma_s^j(\mathbf{x}))} - \ln \Phi(\gamma_s^j(\mathbf{x})) \right] \qquad (4.13)$$

A complete description of the MESMO algorithm is given in Algorithm 1. The blue colored steps correspond to computation of our output space entropy based acquisition function.

5. MESMOC Algorithm for MOO Problem with Constraints

In this section, we address the MOO problem with constraints, where the goal is to optimize multiple real-valued objective functions while satisfying several black-box constraints over continuous space. To solve this problem, we propose an algorithm referred to as **M**ax-value **E**ntropy **S**earch for **M**ulti-objective **O**ptimization with **C**onstraints (MESMOC). In what follows, we explain the technical details and acquisition function derivation.

Problem Setup (MOO Problem with Constraints). This is a generalization of the basic MOO problem, where we need to satisfy some black-box constraints. Our goal is to maximize real-valued objective functions $f_1(\mathbf{x}), f_2(\mathbf{x}), \cdots, f_K(\mathbf{x})$, with $K \geq 2$, while satisfying L black-box constraints of the form $C_1(x) \geq 0, C_2(x) \geq 0, \cdots, C_L(x) \geq 0$ over continuous space $\mathfrak{X} \subseteq \Re^d$. Each evaluation of an input $\mathbf{x} \in \mathfrak{X}$ produces a vector of objective values and constraint values $\mathbf{y} = (y_{f_1}, y_{f_2}, \cdots, y_{f_K}, y_{c_1} \cdots y_{c_L})$ where $y_{f_j} = f_j(x)$ for all $j \in \{1, 2, \cdots, K\}$ and $y_{c_i} = C_i(x)$ for all $i \in \{1, 2, \cdots, L\}$. We say that a valid input $\mathbf{x}$ (satisfies all constraints) *Pareto-dominates* another input $\mathbf{x}'$ if $f_j(\mathbf{x}) \geq f_j(\mathbf{x}') \ \forall j$ and there

---

Algorithm 1 MESMO Algorithm

Input: input space $\mathfrak{X}$; K blackbox objective functions $f_1(x), f_2(x), \cdots, f_K(x)$; and maximum no. of iterations T_{max}

1: Initialize Gaussian process models $\mathcal{GP}_1, \cdots, \mathcal{GP}_K$ by evaluating at N_0 initial points
2: **for** each iteration $t = N_0 + 1$ to T_{max} **do**
3:     Select $\mathbf{x}_t \leftarrow \arg max_{\mathbf{x} \in \mathfrak{X}} \; \alpha_t(\mathbf{x})$, where $\alpha_t(.)$ is computed as:
4:       for each sample $s \in 1, \cdots, S$:
5:         Sample $\tilde{f}_j \sim \mathcal{GP}_j, \quad \forall j \in \{1, \cdots, K\}$
6:         $\mathcal{Y}_s^* \leftarrow$ Pareto front of *cheap* multi-objective optimization over $(\tilde{f}_1, \cdots, \tilde{f}_K)$
7:       Compute $\alpha_t(.)$ based on the S samples of $\mathcal{Y}_s^*$ as given in equation (4.13)
8:     Evaluate $\mathbf{x}_t$: $\mathbf{y}_t \leftarrow (f_1(\mathbf{x}_t), \cdots, f_K(\mathbf{x}_t))$
9:     Aggregate data: $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\mathbf{x}_t, \mathbf{y}_t)\}$
10:    Update models $\mathcal{GP}_1, \mathcal{GP}_2, \cdots, \mathcal{GP}_K$
11:    $t \leftarrow t + 1$
12: **end for**
13: **return** Pareto front of $f_1(x), f_2(x), \cdots, f_K(x)$ based on $\mathcal{D}$

---

exists some $j \in \{1, 2, \cdots, K\}$ such that $f_j(\mathbf{x}) > f_j(\mathbf{x}')$. The goal of multi-objective BO with constraints is to approximate the Pareto set over valid inputs $\mathcal{X}^*$ while minimizing the number of expensive function evaluations. For example, in electric aviation power system design applications, we need to find the designs that trade-off total energy and the mass while satisfying specific thresholds for motor temperature and voltage of cells. Table 1 contains all the mathematical notations used in this section.

Surrogate Models. Similar to section 4, we model the objective functions and black-box constraints by independent GP models $\mathcal{GP}_{f_1}, \mathcal{GP}_{f_2}, \cdots, \mathcal{GP}_{f_K}$ and $\mathcal{GP}_{c_1}, \mathcal{GP}_{c_2}, \cdots, \mathcal{GP}_{f_K}$ with zero mean and i.i.d. observation noise. Let $\mathcal{D} = \{(\mathbf{x}_i, \mathbf{y}_i)\}_{i=1}^{t-1}$ be the training data from past $t-1$ function evaluations, where $\mathbf{x}_i \in \mathfrak{X}$ is an input and $\mathbf{y}_i = \{y_{f_1}^i, \cdots, y_{f_K}^i, y_{c_1}^i, \cdots y_{c_L}^i\}$ is the output vector resulting from evaluating the objective functions and constraints at $\mathbf{x}_i$. We learn surrogate models from $\mathcal{D}$.

5.1 MESMOC Algorithm

Output Space Entropy Based Acquisition Function. To overcome the challenges of computing input space entropy based acquisition function, MESMO (Belakaria et al., 2019) proposed to maximize the information gain about the optimal **Pareto front**. However, MESMO did not address the challenge of constrained Pareto front. We propose an extension of MESMO's acquisition function to maximize the information gain between the next candidate input for evaluation $\mathbf{x}$ and constrained Pareto front $\mathcal{Y}^*$ given as:

$$\alpha(\mathbf{x}) = I(\{\mathbf{x}, \mathbf{y}\}, \mathcal{Y}^* \mid D) = H(\mathbf{y} \mid D, \mathbf{x}) - \mathbb{E}_{\mathcal{Y}^*}[H(\mathbf{y} \mid D, \mathbf{x}, \mathcal{Y}^*)] \tag{5.1}$$

In this case, the output vector $\mathbf{y}$ is $K + L$ dimensional: $\mathbf{y} = (y_{f_1}, y_{f_2}, \cdots, y_{f_K}, y_{c_1} \cdots y_{c_L})$ where $y_{f_j} = f_j(x)$ for all $j \in \{1, 2, \cdots, K\}$ and $y_{c_i} = C_i(x)$ for all $i \in \{1, 2, \cdots, L\}$. Consequently, the first term in the r.h.s of equation (5.1), entropy of a factorizable $(K + L)$-dimensional Gaussian distribution $P(\mathbf{y} \mid D, \mathbf{x}$, can be computed in closed form as shown
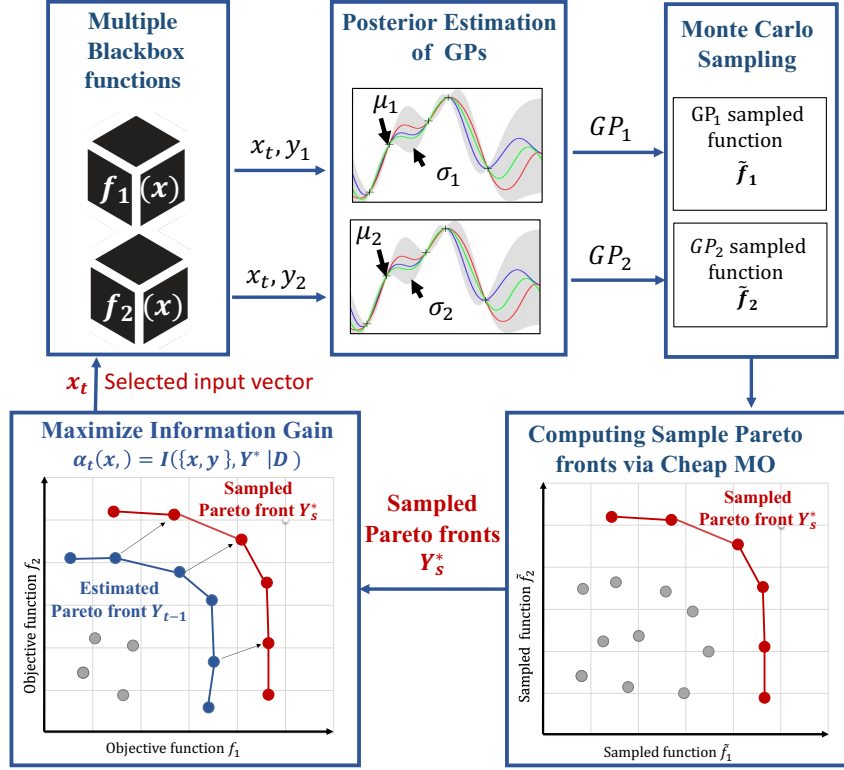
Figure 2: Overview of the MESMO algorithm for two objective functions ($K{=}2$). We build statistical models $\mathcal{GP}_1$, $\mathcal{GP}_2$ for the two objective functions $f_1(x)$ and $f_2(x)$ respectively. First, we sample functions from the statistical models. We compute sample pareto fronts by solving a cheap MO problem over the sampled functions. Second, we select the best candidate input x_t that maximizes the information gain. Finally, we evaluate the functions for x_t to get (y_1, y_2) and update the statistical models using the new training example.

below:

$$H(\mathbf{y} \mid D, \mathbf{x}) = \frac{(K + C)(1 + \ln(2\pi))}{2} + \sum_{j=1}^{K} \ln(\sigma_{f_j}(\mathbf{x})) + \sum_{i=1}^{L} \ln(\sigma_{c_i}(\mathbf{x})) \qquad (5.2)$$

where $\sigma_{f_j}^2(\mathbf{x})$ and $\sigma_{c_i}^2(\mathbf{x})$ are the predictive variances of jth function and i^{th} constraint GPs respectively at input $\mathbf{x}$.

The l.h.s of equation (5.1) can be decomposed in a similar way to equation (4.8). There are two key algorithmic steps to compute this part of the equation: 1) The first is how to compute Pareto front samples $\mathcal{Y}_s^*$?; and 2) The second is how to compute the entropy with respect to a given Pareto front sample $\mathcal{Y}_s^*$? We provide solutions for these two questions below.

1) Computing Pareto Front Samples via Cheap Multi-Objective Optimization. To compute a Pareto front sample $\mathcal{Y}_s^*$, we first sample functions and constraints from the posterior GP models via random Fourier features (Hernández-Lobato et al., 2014; Rahimi & Recht, 2008) and then solve a cheap constrained multi-objective optimization over the K sampled functions and L sampled constraints.

Cheap MO solver. We sample $\tilde{f}_i$ from GP model $\mathcal{GP}_{f_j}$ for each of the K functions and $\tilde{C}_i$ from GP model $\mathcal{GP}_{c_i}$ for each of the L constraints. A *cheap* constrained multi-objective optimization problem over the K sampled functions $\tilde{f}_1, \tilde{f}_2, \cdots, \tilde{f}_k$ and the L sampled constraints $\tilde{C}_1, \tilde{C}_2, \cdots, \tilde{C}_L$ is solved to compute the sample Pareto front $\mathcal{Y}_s^*$. We employ the popular constrained NSGA-II algorithm (Deb et al., 2002a; Deb, Pratap, Agarwal, & Meyarivan, 2002b) to solve the constrained MO problem with cheap sampled objective functions and constrained.

2) Entropy Computation with a Sample Pareto Front. Let $\mathcal{Y}_s^* = \{\mathbf{v}^1, \cdots, \mathbf{v}^l\}$ be the sample Pareto front, where l is the size of the Pareto front and each $\mathbf{v}^i$ is a $(K + L)$-vector evaluated at the K sampled functions and L sampled constraints $\mathbf{v}^i = \{v_{f_1}^i, \cdots, v_{f_K}^i, v_{c_1}^i, \cdots, v_{c_L}^i\}$. The following inequality holds for each component y_j of the $(K+L)$-vector $\mathbf{y} = \{y_{f_1}, \cdots, y_{f_K}, y_{c_1}, \cdots y_{c_L}\}$ in the entropy term $H(\mathbf{y} \mid D, \mathbf{x}, \mathcal{Y}_s^*)$:

$$y_j \leq \max\{v_j^1, \cdots v_j^l\} \quad \forall j \in \{f_1, \cdots, f_K, c_1, \cdots, c_L\} \tag{5.3}$$

The inequality essentially says that the jth component of $\mathbf{y}$ (i.e., y_j) is upper-bounded by a value obtained by taking the maximum of jth components of all l $(K + L)$-vectors in the Pareto front $\mathcal{Y}_s^*$. This inequality had been proven by a contradiction for MESMO (Belakaria et al., 2019) for $j \in \{f_1, \cdots, f_K\}$. We assume the same for $j \in \{c_1, \cdots, c_L\}$.

By combining the inequality (5.3) and the fact that each function is modeled as an independent GP, we can approximate each component y_j as a truncated Gaussian distribution since the distribution of y_j needs to satisfy $y_j \leq \max\{v_j^1, \cdots v_j^l\}$. Let $y_s^{c_i*} = \max\{v_{c_i}^1, \cdots v_{c_i}^l\}$ and $y_s^{f_j*} = \max\{v_{f_j}^1, \cdots v_{f_j}^l\}$. Furthermore, a common property of entropy measure allows us to decompose the entropy of a set of independent variables into a sum over entropies of individual variables (Cover & Thomas, 2012):

$$H(\mathbf{y} \mid D, \mathbf{x}, \mathcal{Y}_s^*) = \sum_{j=1}^{K} H(y_{f_j}|D, \mathbf{x}, y_s^{f_j*}) + \sum_{i=1}^{C} H(y_{c_i}|D, \mathbf{x}, y_s^{c_i*}) \tag{5.4}$$

The r.h.s is a summation over entropies of $(K+L)$-variables $\mathbf{y} = \{y_{f_1}, \cdots, y_{f_K}, y_{c_1}, \cdots y_{c_L}\}$. The differential entropy for each y_j is the entropy of a truncated Gaussian distribution (Michalowicz et al., 2013) and given by the following equations:

$$H(y_{f_j}|D, \mathbf{x}, y_s^{f_j*}) \simeq \left[\frac{(1 + \ln(2\pi))}{2} + \ln(\sigma_{f_j}(\mathbf{x})) + \ln \Phi(\gamma_s^{f_j}(\mathbf{x})) - \frac{\gamma_s^{f_j}(\mathbf{x})\phi(\gamma_s^{f_j}(\mathbf{x}))}{2\Phi(\gamma_s^{f_j}(\mathbf{x}))}\right] \tag{5.5}$$

$$H(y_{c_i}|D, \mathbf{x}, y_s^{c_i*}) \simeq \left[\frac{(1 + \ln(2\pi))}{2} + \ln(\sigma_{c_i}(\mathbf{x})) + \ln \Phi(\gamma_s^{c_i}(\mathbf{x})) - \frac{\gamma_s^{c_i}(\mathbf{x})\phi(\gamma_s^{c_i}(\mathbf{x}))}{2\Phi(\gamma_s^{c_i}(\mathbf{x}))}\right] \tag{5.6}$$

Consequently we have:

$$H(\mathbf{y} \mid D, \mathbf{x}, \mathcal{Y}_s^*) \simeq \sum_{j=1}^{K} \left[\frac{(1 + \ln(2\pi))}{2} + \ln(\sigma_{f_j}(\mathbf{x})) + \ln \Phi(\gamma_s^{f_j}(\mathbf{x})) - \frac{\gamma_s^{f_j}(\mathbf{x})\phi(\gamma_s^{f_j}(\mathbf{x}))}{2\Phi(\gamma_s^{f_j}(\mathbf{x}))} \right]$$
$$+ \sum_{i=1}^{L} \left[\frac{(1 + \ln(2\pi))}{2} + \ln(\sigma_{c_i}(\mathbf{x})) + \ln \Phi(\gamma_s^{c_i}(\mathbf{x})) - \frac{\gamma_s^{c_i}(\mathbf{x})\phi(\gamma_s^{c_i}(\mathbf{x}))}{2\Phi(\gamma_s^{c_i}(\mathbf{x}))} \right] \quad (5.7)$$

where $\gamma_s^{c_i}(x) = \frac{y_s^{c_i*} - \mu_{c_i}(\mathbf{x})}{\sigma_{c_i}(\mathbf{x})}$, $\gamma_s^{f_j}(x) = \frac{y_s^{f_j*} - \mu_{f_j}(\mathbf{x})}{\sigma_{f_j}(\mathbf{x})}$, ϕ and Φ are the p.d.f and c.d.f of a standard normal distribution respectively. By combining equations (5.2) and (5.7) with equation (5.1), we get the final form of our acquisition function as shown below:

$$\alpha(\mathbf{x}) \simeq \frac{1}{S} \sum_{s=1}^{S} \left[\sum_{j=1}^{K} \frac{\gamma_s^{f_j}(\mathbf{x})\phi(\gamma_s^{f_j}(\mathbf{x}))}{2\Phi(\gamma_s^{f_j}(\mathbf{x}))} - \ln \Phi(\gamma_s^{f_j}(\mathbf{x})) + \sum_{i=1}^{L} \frac{\gamma_s^{c_i}(\mathbf{x})\phi(\gamma_s^{c_i}(\mathbf{x}))}{2\Phi(\gamma_s^{c_i}(\mathbf{x}))} - \ln \Phi(\gamma_s^{c_i}(\mathbf{x})) \right]$$
$$(5.8)$$

A complete description of the MESMOC algorithm is given in Algorithm 2.

Algorithm 2 MESMOC Algorithm

Input: input space $\mathfrak{X}$; K blackbox functions $f_1(x), f_2(x), \cdots, f_K(x)$; L blackbox constraints $C_1(x), C_2(x), \cdots, C_L(x)$; and maximum no. of iterations T_{max}

1: Initialize Gaussian process models $\mathcal{GP}_{f_1}, \mathcal{GP}_{f_2}, \cdots, \mathcal{GP}_{f_K}$ and $\mathcal{GP}_{c_1}, \mathcal{GP}_{c_2}, \cdots, \mathcal{GP}_{c_L}$ by evaluating at N_0 initial points
2: **for** each iteration $t = N_0 + 1$ to T_{max} **do**
3:    Select $\mathbf{x}_t \leftarrow \arg max_{\mathbf{x} \in \mathfrak{X}} \; \alpha_t(\mathbf{x})$
       s.t $(\mu_{c_1} \geq 0, \cdots, \mu_{c_L} \geq 0)$
4:    $\alpha_t(.)$ is computed as:
5:     for each sample $s \in 1, \cdots, S$:
6:      Sample $\tilde{f}_j \sim \mathcal{GP}_{f_j}, \quad \forall j \in \{1, \cdots, K\}$
7:      Sample $\tilde{C}_i \sim \mathcal{GP}_{c_i}, \quad \forall i \in \{1, \cdots, L\}$
8:      // Solve *cheap* MOO over $(\tilde{f}_1, \cdots, \tilde{f}_K)$ constrained by $(\tilde{C}_1, \cdots, \tilde{C}_L)$
9:      $\mathcal{Y}_s^* \leftarrow \arg max_{x \in \mathcal{X}}(\tilde{f}_1, \cdots, \tilde{f}_K)$
        s.t $(\tilde{C}_1 \geq 0, \cdots, \tilde{C}_L \geq 0)$
10:    Compute $\alpha_t(.)$ based on the S samples of $\mathcal{Y}_s^*$ as given in equation (5.8)
11:   Evaluate $\mathbf{x}_t$; $\mathbf{y}_t \leftarrow (f_1(\mathbf{x}_t), \cdots, f_K(\mathbf{x}_t), C_1(\mathbf{x}_t), \cdots, C_L(\mathbf{x}_t))$
12:   Aggregate data: $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\mathbf{x}_t, \mathbf{y}_t)\}$
13:   Update models $\mathcal{GP}_{f_1}, \mathcal{GP}_{f_2}, \cdots, \mathcal{GP}_{f_K}$ and $\mathcal{GP}_{c_1}, \mathcal{GP}_{c_2}, \cdots, \mathcal{GP}_{c_L}$
14:   $t \leftarrow t + 1$
15: **end for**
16: **return** Pareto front of $f_1(x), f_2(x), \cdots, f_K(x)$ based on $\mathcal{D}$

---

6. MF-OSEMO Algorithm for Discrete Multi-Fidelity MOO Problem

In this section, we address the multi-fidelity version of MOO problem, where we have access to multiple fidelities for each function that vary in the amount of resources consumed and the accuracy of evaluation. To solve this problem, we propose an algorithm referred to as **M**ulti-**F**idelity **O**utput **S**pace *Entropy Search for* **M**ulti-objective **O**ptimization (MF-OSEMO). We first describe the complete details related to the multi-fidelity MOO problem. Subsequently, we explain our proposed MF-OSEMO algorithm with two mathematically different approximations of the output space entropy based acquisition function.

Problem Setup (Discrete Multi-Fidelity MOO Problem). This is a general version of the MOO problem, where we have access to M_j fidelities for each function f_j that vary in the amount of resources consumed and the accuracy of evaluation. The evaluation of an input $\mathbf{x} \in \mathfrak{X}$ with fidelity vector $\mathbf{m} = [m_1, m_2, \cdots, m_K]$ produces an evaluation vector of K values denoted by $\mathbf{y^m} \equiv [y_1^{(m_1)}, \cdots, y_K^{(m_K)}]$, where $y_j^{(m_j)} = f_j^{(m_j)}(x)$ for all $j \in \{1, 2, \cdots, K\}$. Let $\lambda_j^{(m_j)}$ be the cost of evaluating i^{th} function f_j at $m_j \in [M_j]$ fidelity, where $m_j = M_j$ corresponds to the highest fidelity for f_j. Our goal is to approximate the optimal Pareto set $\mathcal{X}^*$ over the highest fidelities functions while minimizing the overall cost of function evaluations (experiments). For example, in power system design optimization, we need to find designs that trade-off cost, size, efficiency, and thermal tolerance using multi-fidelity simulators for design evaluations. Table 2 contains all the mathematical notations used in this section (MF-OSEMO).

Cost of Function Evaluations. The total normalized evaluation cost is $\lambda^{(\mathbf{m})} \equiv \sum_{j=1}^{K} \left(\lambda_j^{(m_j)} / \lambda_j^{(M_j)} \right)$. We normalize the total cost since the cost units can be different for different objectives (e.g. cost unit for f_1 is computation time while cost unit for f_2 could be memory space size). If the cost is known, it can be directly injected in the latter expression. However, in some real world settings, the cost of a function evaluation can be only known after the function evaluation. For example, in hyper-parameter tuning of a neural network, the cost of the experiment is defined by the training and inference time. However, we cannot know the exact needed time until after the experiment is finalised. In this case, the cost can be modeled by an independent Gaussian process. The predictive mean can be used during the optimization. Our goal is to approximate $\mathcal{X}^*$ by minimizing the overall cost of function evaluations.

Multi-Fidelity Gaussian Process Model. Let $D = \{(\mathbf{x}_i, \mathbf{y}_i^{(\mathbf{m})})\}_{i=1}^{t-1}$ be the training data from past $t-1$ function evaluations, where $\mathbf{x}_i \in \mathfrak{X}$ is an input and $\mathbf{y}_i^{(\mathbf{m})} = [y_1^{(m_1)}, \cdots, y_K^{(m_K)}]$ is the output vector resulting from evaluating functions $f_1^{(m_1)}, f_2^{(m_2)}, \cdots, f_K^{(m_k)}$ at $\mathbf{x}_i$. Gaussian processes (GPs) are known to be effective surrogate models in prior work on single and multi-objective BO (Srinivas et al., 2009; Hernández-Lobato et al., 2016). We learn K surrogate models $\mathcal{GP}_1, \mathcal{GP}_2, \cdots, \mathcal{GP}_K$ from $\mathcal{D}$, where each $\mathcal{GP}_j$ corresponds to the jth function f_j. In our setting, each function has multiple fidelities. So one ideal property desired for the surrogate model of a single function is to take into account all the fidelities in a single model. Multi-fidelity GPs (MF-GP) are capable of modeling functions with multiple fidelities in a single model. Hence, each of our surrogate model $\mathcal{GP}_j$ is a multi-fidelity GP.

Notation	Definition
$f_1^{(m_1)}, f_2^{(m_1)}, \cdots, f_K^{(m_K)}$	functions the m_j fidelity of the true objective functions
$\tilde{f}_j^{(m_j)}$	function sampled from jth Gaussian process model at m_jth fidelity
$M_1, M_2, \cdots, M_K$	no. of fidelities for each function
$\mathbf{m} = [m_1, m_2, \cdots, m_K]$	fidelity vector where each fidelity $m_j \in [M_j]$
$y_j^{m_j}$	jth function f_j evaluated at m_jth fidelity where $m_j \in [M_j]$
$\mathbf{y^m}$	output vector equivalent to $[y_1^{(m_1)}, \cdots, y_K^{(m_K)}]$
$\lambda_j^{(m_j)}$	cost of evaluating jth function f_j at m_jth fidelity
$\lambda^{(\mathbf{m})}$	total normalized cost $\lambda^{(\mathbf{m})} \equiv \sum_{j=1}^{K} \left(\lambda_j^{(m_j)} / \lambda_j^{(M_j)} \right)$
$\mathcal{Y}^*$	true pareto front of the objective functions $[f_1, f_2, \cdots, f_K]$ (the highest fidelities)
$\mathcal{Y}_s^*$	Pareto front of the sampled highest fidelities $[\tilde{f}_1, \tilde{f}_2, \cdots, \tilde{f}_K]$

Table 2: Table describing additional mathematical notations used in this section (MF-OSEMO).

Specifically, we use the MF-GP model as proposed in Kennedy and O'Hagan (2000), Takeno et al. (2019). We describe the complete details of the MF-GP model below for the sake of completeness. One key thing to note about MF-GP model is that the kernel function $(k((\mathbf{x_i}, m_i), (\mathbf{x_j}, m_j)))$ is dependent on both the input and the fidelity. For a given input $\mathbf{x}$, the MF-GP model returns a *vector* (one for each fidelity) of predictive mean, a *vector* of predictive variance, and a matrix of predictive covariance. The MF-GP model has two advantages. The first is that all fidelities are integrated into one single GP. The second is that difference among fidelities are adaptively estimated without any additional feature representation for fidelities. It should be noted that we employ an independent multi-fidelity GP for each function.

We describe full details of a MF-GP model for one objective function f_j (without loss of generality) below:

Let $y_j^{(1)}(\mathbf{x}), \ldots, y_j^{(M_j)}(\mathbf{x})$ represent the values obtained by evaluating the function f_j at its 1st, 2nd, ..., M_jth fidelity respectively. In a MF-GP model, each fidelity is represented by a Gaussian process and the observation is modeled as

$$y_j^{(m_j)}(\mathbf{x}) = f_j^{(m_j)}(\mathbf{x}) + \epsilon, \quad \epsilon \sim \mathcal{N}(0, \sigma_{\text{noise}}^2).$$

Let $f_j^{(1)} \sim GP(0, k_1(\mathbf{x}, \mathbf{x}'))$ be a Gaussian process for the 1st fidelity i.e. $m_j = 1$, where $k_1 : \mathcal{R}^d \times \mathcal{R}^d \to \mathcal{R}$ is a suitable kernel. The output for successively fidelities $m_j = 2, \ldots, M_j$ is recursively defined as

$$f_j^{(m_j)}(\mathbf{x}) = f_j^{(m_j-1)}(\mathbf{x}) + f_{j_e}^{(m_j-1)}(\mathbf{x}), \quad (6.1)$$

where, $f_{j_e}^{(m_j-1)} \sim GP(0, k_e(\mathbf{x}, \mathbf{x}'))$ with $k_e : \mathcal{R}^d \times \mathcal{R}^d \to \mathcal{R}$. It is assumed that $f_{j_e}^{(m_j-1)}$ is conditionally independent from all fidelities lower than m_j. As a result, *the kernel for a pair of points evaluated at the same fidelity* becomes:

$$k_{m_j}(\mathbf{x}, \mathbf{x}') \equiv k_1(\mathbf{x}, \mathbf{x}') + (m_j - 1)k_e(\mathbf{x}, \mathbf{x}') \quad (6.2)$$

683

and as a result, the output for m_jth fidelity is also modeled as a Gaussian process:

$$f_j^{(m_j)} \sim GP(0, k_{m_j}(\mathbf{x}, \mathbf{x}')).$$

The kernel function for a pair of inputs evaluated at different fidelities m_j and m_j' is:

$$k((\mathbf{x}, m_j), (\mathbf{x}', m_j')) = \text{cov}\left(f_j^{(m_j)}(\mathbf{x}), f_j^{(m_j')}(\mathbf{x}')\right) = k_{m_j}(\mathbf{x}, \mathbf{x}')$$

where $m_j \leq m_j'$ and cov represents covariance. Using a kernel matrix $K \in \mathcal{R}^{n \times n}$ in which the p, q element is defined by $k((\mathbf{x}, m_j^p), (\mathbf{x}', m_j^q))$, all fidelities $f_j^{(1)}, \ldots, f_j^{(M_j)}$ can be integrated into one common Gaussian process by which predictive mean and variance are obtained as

$$\mu^{(m_j)}(\mathbf{x}) = K + \sigma_{\text{noise}}^2 I^{-1} \mathbf{Y}, \tag{6.3}$$

$$\sigma^{2(m_j)}(\mathbf{x}) = k((\mathbf{x}, m_j), (\mathbf{x}, m_j)) - k_n^{(m_j)}(\mathbf{x})^\top K + \sigma_{\text{noise}}^2 I^{-1} k_n^{(m_j)}(\mathbf{x}), \tag{6.4}$$

where $k_n^{(m_j)}(\mathbf{x}) \equiv (k((\mathbf{x}, m_j), (\mathbf{x}_1, m_{j_1})), \ldots, k((\mathbf{x}, m_j), (\mathbf{x}_n, m_{j_n})))^\top$ and $\mathbf{Y} = (y_1^{(m_{j_1})}(\mathbf{x}_1), \ldots, y_n^{(m_{j_n})}(\mathbf{x}_n))^\top$. We also define $\sigma^{2(m_j m_j')}(\mathbf{x})$ as the predictive covariance between $(\mathbf{x}, m_j)$ and $(\mathbf{x}, m_j')$, i.e., covariance for identical $\mathbf{x}$ at different fidelities:

$$\sigma^{2(m_j m_j')}(\mathbf{x}) = k((\mathbf{x}, m_j), (\mathbf{x}, m_j')) - k_n^{(m_j)}(\mathbf{x})^\top K + \sigma_{\text{noise}}^2 I^{-1} k_n^{(m_j')}(\mathbf{x}). \tag{6.5}$$

6.1 MF-OSEMO Algorithm with Two Approximations

We describe our proposed acquisition function for the multi-fidelity MOO problem setting. We leverage the information-theoretic principle of output space information gain to develop an efficient and robust acquisition function. This method is applicable for the general case, where at each iteration, the objective functions can be evaluated at different fidelities.

The key idea behind the proposed acquisition function is to find the pair $\{\mathbf{x}, \mathbf{m}\}$ that maximizes the information gain about the **Pareto front of the highest fidelities (denoted by $\mathcal{Y}^*$)** per unit cost, where $\{\mathbf{x}, \mathbf{m}\}$ represents a candidate input $\mathbf{x}$ evaluated at a vector of fidelities $\mathbf{m} = [m_1, m_2, \cdots, m_K]$. This idea can be expressed mathematically as given below:

$$\alpha(\mathbf{x}, \mathbf{m}) = I(\{\mathbf{x}, \mathbf{y}^{(\mathbf{m})}\}, \mathcal{Y}^* \mid D) / \lambda^{(\mathbf{m})} \tag{6.6}$$

where $\lambda^{(\mathbf{m})}$ is the total *normalized* cost of evaluating the objective functions at $\mathbf{m}$ and D is the data collected so far. Figure 3 provides an overview of the MF-OSEMO algorithm. The information gain in equation (6.6) is defined as the expected reduction in entropy $H(.)$ of the posterior distribution $P(\mathcal{Y}^* \mid D)$ as a result of evaluating $\mathbf{x}$ at fidelity vector $\mathbf{m}$:

$$I(\{\mathbf{x}, \mathbf{y}^{(\mathbf{m})}\}, \mathcal{Y}^* \mid D) = H(\mathcal{Y}^* \mid D) - \mathbb{E}_{y^{(\mathbf{m})}}[H(\mathcal{Y}^* \mid D \cup \{\mathbf{x}, \mathbf{y}^{(\mathbf{m})}\})] \tag{6.7}$$

$$= H(\mathbf{y}^{(\mathbf{m})} \mid D, \mathbf{x}) - \mathbb{E}_{\mathcal{Y}^*}[H(\mathbf{y}^{(\mathbf{m})} \mid D, \mathbf{x}, \mathcal{Y}^*)] \tag{6.8}$$
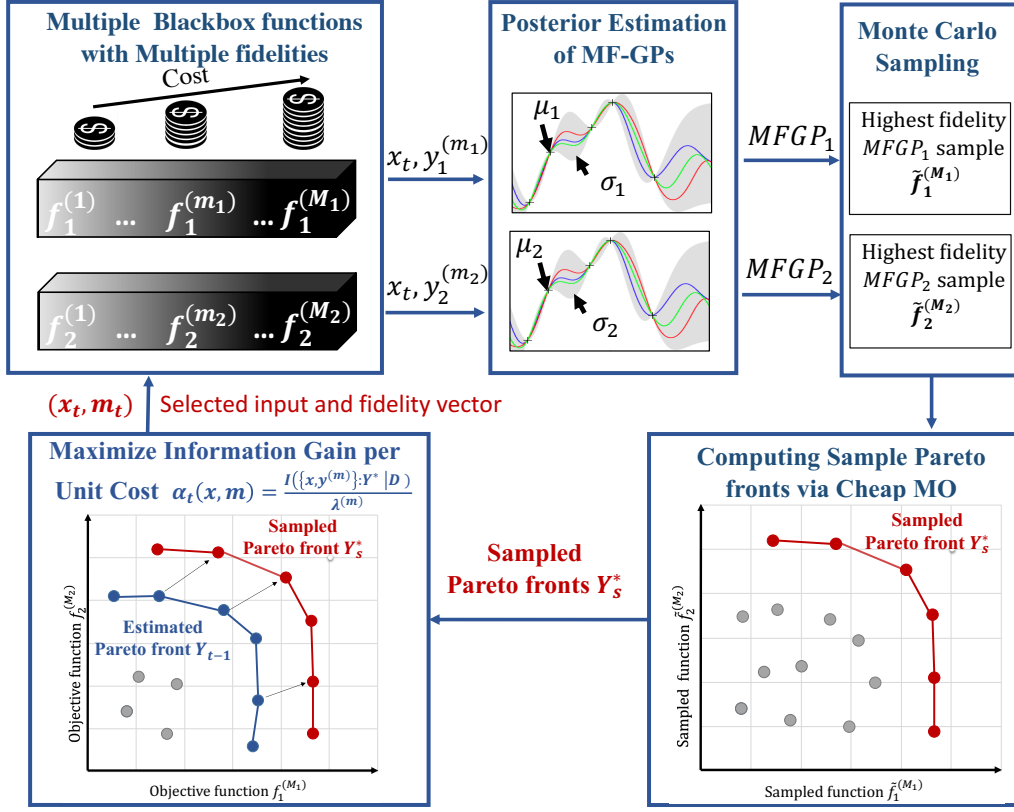
Figure 3: Overview of the MF-OSEMO algorithm for two objective functions (K=2). We build multi-fidelity statistical models $\mathcal{MFGP}_1$, $\mathcal{MFGP}_2$ for the two objective functions $f_1(x)$ and $f_2(x)$ with M_1 and M_2 fildelities respectively. First, we sample highest fidelity functions from the statistical models. We compute sample pareto fronts by solving a cheap MO problem over the sampled functions. Second, we select the best candidate input x_t and fidelity vector $m_t = (m_1, m_2)$ that maximizes the information gain per unit cost . Finally, we evaluate the functions for x_t at fidelities m_t to get $(y_1^{(m_1)}, y_2^{(m_2)})$ and update the statistical models using the new training example.

equation (6.8) follows from equation (6.7) as a result of the symmetric property of information gain. The first term in the r.h.s of equation (6.8) is the entropy of a factorizable K-dimensional Gaussian distribution $P(\mathbf{y}^{(\mathbf{m})} \mid D, \mathbf{x}))$ which can be computed in closed form as shown below:

$$H(\mathbf{y}^{(\mathbf{m})} \mid D, \mathbf{x}) = \frac{K(1 + \ln(2\pi))}{2} + \sum_{j=1}^{K} \ln(\sigma_j^{(m_j)}(\mathbf{x})) \qquad (6.9)$$

where $\sigma_j^{(m_j)}(\mathbf{x})$ is the predictive variance of jth surrogate model GP_j at input $\mathbf{x}$ and fidelity m_j. The second term in the r.h.s of equation (6.8) is an expectation over the Pareto front

of the highest fidelities $\mathcal{Y}^*$. We can approximately compute this term via Monte-Carlo sampling as shown below:

$$\mathbb{E}_{\mathcal{Y}^*}[H(\mathbf{y}^{(\mathbf{m})} \mid D, \mathbf{x}, \mathcal{Y}^*)] \simeq \frac{1}{S} \sum_{s=1}^{S} [H(\mathbf{y}^{(\mathbf{m})} \mid D, \mathbf{x}, \mathcal{Y}_s^*)] \tag{6.10}$$

where S is the number of samples and $\mathcal{Y}_s^*$ denote a sample Pareto front obtained over the highest fidelity functions sample from K surrogate models. The main advantages of our acquisition function are: cost efficiency, computational-efficiency, and robustness to the number of samples. Our experiments demonstrate these advantages over state-of-the-art single fidelity AFs for multi-objective optimization.

There are two key algorithmic steps to compute equation (6.10). The first is computing Pareto front samples $\mathcal{Y}_s^*$; and the second is computing the entropy with respect to a given Pareto front sample $\mathcal{Y}_s^*$. We provide solutions for these two steps below.

1) Computing Pareto Front Samples via Cheap Multi-Objective optimization. To compute a Pareto front sample $\mathcal{Y}_s^*$, we first sample highest fidelity functions from the posterior MF-GP models via random Fourier features (Hernández-Lobato et al., 2014; Rahimi & Recht, 2008) and then solve a cheap multi-objective optimization over the K sampled high fidelity functions. It is important to note that we are sampling only the **highest fidelity function** from each MF-GP surrogate model.

Sampling functions from the posterior of MF-GP model. Similar to prior work (Hernández-Lobato et al., 2014, 2016; Wang & Jegelka, 2017), we employ random Fourier features based sampling procedure. We approximate each GP prior of the highest fidelity as $\tilde{f}^{(M)} = \phi(\mathbf{x})^T \theta$, where $\theta \sim N(0, \mathbf{I})$. The key idea behind random Fourier features is to construct each function sample $\tilde{f}^{(M)}(\mathbf{x})$ as a finitely parametrized approximation: $\phi(\mathbf{x})^T \theta$, where θ is sampled from its corresponding posterior distribution conditioned on the data D obtained from past function evaluations: $\theta \mid D \sim N(\mathbf{A}^{-1}\mathbf{\Phi}^{\mathbf{T}}\mathbf{y}_n, \sigma^2 \mathbf{A}^{-1})$, where $\mathbf{A} = \mathbf{\Phi}^{\mathbf{T}}\mathbf{\Phi} + \sigma^2 \mathbf{I}$ and $\Phi^T = [\phi(\mathbf{x}_1), \cdots, \phi(\mathbf{x}_{t-1})]$.

Cheap MO solver. We sample $\tilde{f}_i^{(M_i)}$ from each surrogate model $\mathcal{MF} - \mathcal{GP}_i$ as described above. A *cheap* multi-objective optimization problem over the K sampled functions $\tilde{f}_1^{(M_1)}, \tilde{f}_2^{(M_2)}, \cdots, \tilde{f}_K^{(M_K)}$ is solved to compute the sample Pareto front $\mathcal{Y}_s^*$. This cheap multi-objective optimization also allows us to capture the interactions between different objectives. We employ the popular NSGA-II algorithm (Deb et al., 2002a) to solve the MO problem with cheap objective functions noting that any other algorithm can be used.

2) Entropy Computation with a Sample Pareto Front. Let $\mathcal{Y}_s^* = \{\mathbf{v}^1, \cdots, \mathbf{v}^l\}$ be the sample Pareto front, where l is the size of the Pareto front and each $\mathbf{v}^i = \{v_1^i, \cdots, v_K^i\}$ is a K-vector evaluated at the K sampled high fidelity functions. The following inequality holds for each component $y_j^{(m_j)}$ of the K-vector $\mathbf{y}^{(\mathbf{m})} = \{y_1^{(m_1)}, \cdots, y_K^{(m_k)}\}$ in the entropy term $H(\mathbf{y}^{(\mathbf{m})} \mid D, \mathbf{x}, \mathcal{Y}_s^*)$:

$$y_j^{(m_j)} \leq y_{j_s}^* \quad \forall j \in \{1, \cdots, K\} \tag{6.11}$$

where $y_{j_s}^* = \max\{v_j^1, \cdots v_j^l\}$. The inequality essentially says that the jth component of $\mathbf{y}^{\mathbf{m}}$ (i.e., $y_j^{m_j}$) is upper-bounded by a value obtained by taking the maximum of jth components

of all l vectors $\{\mathbf{v}^1, \cdots, \mathbf{v}^l\}$ in the Pareto front $\mathcal{Y}_s^*$. The proof of 6.11 can be divided into two cases:

Case I. If y_j is evaluated at its highest fidelity (i.e $m_j = M_j$), inequality (6.11) can be proven by a contradiction argument. Suppose there exists some component $y_j^{(M_j)}$ of $\mathbf{y}^{(\mathbf{M})}$ such that $y_j^{(M_j)} > y_{j_s}^*$. However, by definition, $\mathbf{y}^{(\mathbf{M})}$ is a non-dominated point because no point dominates it in the jth dimension. This results in $\mathbf{y}^{(\mathbf{M})} \in \mathcal{Y}_s^*$ which is a contradiction. Therefore, our hypothesis that $y_j^{(M_j)} > y_{j_s}^*$ is incorrect and inequality (6.11) holds.

Case II. If y_j is evaluated at one of its lower fidelities (i.e, $m_j \neq M_j$), the proof follows from the assumption that the value of lower fidelity of a objective is usually smaller than the corresponding higher fidelity, i.e., $y_j^{(m_j)} \leq y_j^{(M_j)} \leq y_{j_s}^*$. This is especially true for most real-world experiments. For example, in optimizing a neural network's accuracy with respect to its hyperparameters, a commonly employed fidelity is the number of data samples used for training. It is reasonable to believe that the accuracy is always higher for the higher fidelity (more data samples to train on) when compared to a lower fidelity (less data samples). By combining the inequality (6.11) and the fact that each function is modeled as an independent MF-GP, a common property of entropy measure allows us to decompose the entropy of a set of independent variables into a sum over entropies of individual variables (Cover & Thomas, 2012):

$$H(\mathbf{y}^{(\mathbf{m})} \mid D, \mathbf{x}, \mathcal{Y}_s^*) = \sum_{j=1}^{K} H(y_j^{(m_j)} | D, \mathbf{x}, y_{j_s}^*) \tag{6.12}$$

The computation of equation (6.12) requires the computation of the entropy of $p(y_j^{(m_j)} | D, \mathbf{x}, y_{j_s}^*)$. This is a conditional distribution that depends on the value of m_j and can be expressed as $H(y_j^{(m_j)} | D, \mathbf{x}, y_j^{(m_j)} \leq y_{j_s}^*)$. This entropy is dealt with in two cases.

First, for $\mathbf{m_j} = \mathbf{M_j}$, the density function of this probability is approximated by truncated Gaussian distribution and its entropy can be expressed as (Michalowicz et al., 2013):

$$H(y_j^{(M_j)} | D, \mathbf{x}, y_j^{(M_j)} \leq y_{j_s}^*) \simeq \frac{(1 + \ln(2\pi))}{2} + \ln(\sigma_j^{(M_j)}(\mathbf{x})) + \ln \Phi(\gamma_s^{(M_j)}(\mathbf{x}))$$
$$- \frac{\gamma_s^{(M_j)}(\mathbf{x})\phi(\gamma_s^{(M_j)}(\mathbf{x}))}{2\Phi(\gamma_s^{(M_j)}(\mathbf{x}))} \tag{6.13}$$

where $\gamma_s^{(M_j)}(\mathbf{x}) = \frac{y_{j_s}^* - \mu_j^{(M_j)}(\mathbf{x})}{\sigma_j^{(M_j)}(\mathbf{x})}$, and ϕ and Φ are the p.d.f and c.d.f of a standard normal distribution respectively.

Second, for $\mathbf{m_j} \neq \mathbf{M_j}$, the density function of $p(y_j^{(m_j)} | D, \mathbf{x}, y_{j_s}^*)$ can be computed using two different approximations as described below.

Approximation 1 (MF-OSEMO-TG): As a consequence of **Case II**, which states that $y_j^{(m_j)} \leq y_{j_s}^*$ also holds for all lower fidelities, the entropy of $p(y_j^{(m_j)} | D, \mathbf{x}, y_{j_s}^*)$ can also be

approximated by the entropy of a truncated Gaussian distribution and expressed as follow:

$$H(y_j^{(m_j)}|D, \mathbf{x}, y_j^{(m_j)} \leq y_{j_s}^*) \simeq \frac{(1 + \ln(2\pi))}{2} + \ln(\sigma_j^{(m_j)}(\mathbf{x})) + \ln \Phi(\gamma_s^{(m_j)}(\mathbf{x}))$$
$$- \frac{\gamma_s^{(m_j)}(\mathbf{x})\phi(\gamma_s^{(m_j)}(\mathbf{x}))}{2\Phi(\gamma_s^{(m_j)}(\mathbf{x}))} \tag{6.14}$$

where $\gamma_s^{(m_j)}(\mathbf{x}) = \frac{y_{j_s}^* - \mu_j^{(m_j)}(\mathbf{x})}{\sigma_j^{(m_j)}(\mathbf{x})}$.

Approximation 2 (MF-OSEMO-NI): Although equation (6.14) is sufficient for computing the entropy for $m_j \neq M_j$, it can be improved by conditioning on a tighter inequality $y_j^{(M_j)} \leq y_{j_s}^*$ as compared to the general one, i.e., $y_j^{(m_j)} \leq y_{j_s}^*$. As we show below, this improvement comes at the expense of not obtaining a final closed-form expression, but it can be efficiently computed via numerical integration. We apply the derivation of the entropy based on numerical integration for single-objective problem, proposed in (Takeno et al., 2019), for the multi-objective setting.

Now, for calculating $H(y_j^{(m_j)}|D, \mathbf{x}, y_j^{(m_j)} \leq y_{j_s}^*)$ by replacing $p(y_j^{(m_j)}|D, \mathbf{x}, y_j^{(m_j)} \leq y_{j_s}^*)$ with $p(y_j^{(m_j)}|D, \mathbf{x}, y_j^{(M_j)} \leq y_{j_s}^*)$ and using Bayes' theorem, we have:

$$p(y_j^{(m_j)}|D, \mathbf{x}, y_j^{(M_j)} \leq y_{j_s}^*) = \frac{p(y_j^{(M_j)} \leq y_{j_s}^*|y_j^{(m_j)}, D, \mathbf{x})p(y_j^{(m_j)}, D, \mathbf{x})}{p(y_j^{(M_j)} \leq y_{j_s}^*|D, \mathbf{x})} \tag{6.15}$$

Both the densities, $p(y_j^{(M_j)} \leq y_{j_s}^*|D, \mathbf{x})$ and $p(y_j^{(m_j)}, D, \mathbf{x})$ can be obtained from the predictive distribution of MF-GP model and is given as follows:

$$p(y_j^{(m_j)}, D, \mathbf{x}) = \frac{\phi(\gamma_j^{(m_j)}(\mathbf{x}))}{\sigma_j^{(m_j)}} \tag{6.16}$$

$$p(y_j^{(M_j)} \leq y_{j_s}^*|D, \mathbf{x}) = \Phi(\gamma_s^{(M_j)}(\mathbf{x}))) \tag{6.17}$$

where $\gamma_j^{(m_j)}(\mathbf{x}) = \frac{y_j^{(m_j)} - \mu_j^{(m_j)}(\mathbf{x})}{\sigma_j^{(m_j)}(\mathbf{x})}$. Since MF-GP represents all fidelities as one unified Gaussian process, the joint marginal distribution $p(y_j^{(M_j)}, y_j^{(m_j)}|D, \mathbf{x})$ can be immediately obtained from the posterior distribution of the corresponding model $\mathcal{GP}_j$ as given below:

$$p(y_j^{(M_j)}|y_j^{(m_j)}, \mathbf{x}, D) \sim \mathcal{N}(\mu_j(\mathbf{x}), s_j^2(\mathbf{x})) \tag{6.18}$$

where $\mu_j(\mathbf{x}) = \frac{\sigma_j^{2(m_j M_j)}(\mathbf{x})(y_j^{(m_j)} - \mu_j^{m_j}(\mathbf{x}))}{\sigma_j^{2(m_j)}(\mathbf{x})}$ and $s_j^2(\mathbf{x}) = \sigma_j^{2(M_j)}(\mathbf{x}) - \frac{(\sigma_j^{2(m_j M_j)}(\mathbf{x}))^2}{\sigma_j^{2(m_j)}(\mathbf{x})}$. As a result, $p(y_j^{(M_j)} \leq y_{j_s}^*|y_j^{(m_j)}, D, \mathbf{x})$ is expressed as the cumulative distribution of the Gaussian in (6.18):

$$p(y_j^{(M_j)} \leq y_{j_s}^*|y_j^{(m_j)}, D, \mathbf{x}) = \Phi(\frac{y_{j_s}^* - \mu_j(\mathbf{x})}{s_j(\mathbf{x})}) \tag{6.19}$$

Algorithm 3 MF-OSEMO Algorithm

Input: input space $\mathfrak{X}$; K blackbox objective functions where each function f_j has multiple fidelities M_j $\left(\{f_1^{(1)}(\mathbf{x}), \cdots, f_1^{(M_1)}(\mathbf{x})\}, \cdots, \{f_K^{(1)}(\mathbf{x}), \cdots, f_K^{(M_K)}(\mathbf{x})\}\right)$; and total budget λ_{Total}

1: Initialize multi-fidelity Gaussian process models $\mathcal{GP}_1, \cdots, \mathcal{GP}_K$ by evaluating at initial points D
2: **While** $\lambda_t \leq \lambda_{total}$ **do**
3:    for each sample $s \in 1, \cdots, S$:
4:      Sample highest-fidelity functions $\tilde{f}_i^{(M_i)} \sim \mathcal{GP}_i, \quad \forall i \in \{1, \cdots, K\}$
5:      $\mathcal{Y}_s^* \leftarrow$ Pareto front of *cheap* multi-objective optimization over $(\tilde{f}_1^{(M_1)}, \cdots, \tilde{f}_K^{(M_K)})$
6:    Find the next point to evaluate: select $(\mathbf{x}_t, \mathbf{m}_t) \leftarrow \arg max_{\mathbf{x} \in \mathfrak{X}, \mathbf{m}} \; \alpha_t(\mathbf{x}, \mathbf{m}, \mathcal{Y}^*)$
7:    Update the total cost consumed: $\lambda_t \leftarrow \lambda_t + \lambda^{(\mathbf{m_t})}$
8:    Aggregate data: $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\mathbf{x}_t, \mathbf{y}_t^{\mathbf{m}})\}$
9:    Update models $\mathcal{GP}_1, \cdots, \mathcal{GP}_K$
10:    $t \leftarrow t + 1$
11: **end while**
12: **return** Pareto front and Pareto set of $f_1(x), \cdots, f_K(x)$ based on $\mathcal{D}$
13: **Procedure** $\alpha_t(\mathbf{x}, \mathbf{m}, \mathcal{Y}_s^*)$
14: // Computes information gain (I) about the posterior of true Pareto front ($\mathcal{Y}^*$) per unit cost as a result of evaluating $\mathbf{x}$
15: // I = H_1 - H_2;   where H_1 = Entropy of $\mathbf{y}^{(\mathbf{m})}$ conditioned on D and $\mathbf{x}$
   //   and H_2 = Expected entropy of $\mathbf{y}^{(\mathbf{m})}$ conditioned on D, $\mathbf{x}$ and ($\mathcal{Y}^*$)
16: Set $H_1 = H(\mathbf{y}^{(\mathbf{m})} \mid D, \mathbf{x}) = K(1 + \ln(2\pi))/2 + \sum_{j=1}^{K} \ln(\sigma_j^{(m_j)}(\mathbf{x}))$ (entropy of K-factorizable Gaussian)
17: To compute $H_2 \simeq \frac{1}{S} \sum_{s=1}^{S} \sum_{j=1}^{K} H(y_j^{(m_j)} | D, \mathbf{x}, y_{j_s}^*)$, initialize $H_2 = 0$
18: **for** each sample $\mathcal{Y}_s^*$ **do**
19:   **for** $j \in 1 \cdots K$ **do**
20:     Set $y_{j_s}^* =$ maximum of jth component of all vectors in $\mathcal{Y}_s^*$
21:     **If** $m_j = M_j$ // if evaluating jth function at highest fidelity
22:       H_2 += $H(y_j^{(M_j)}|D, \mathbf{x}, y_j^{(M_j)} \leq y_{j_s}^*)$ (entropy of truncated Gaussian $p(y_j^{(M_j)}|D, \mathbf{x}, \underline{y_j^{(M_j)} \leq y_{j_s}^*})$)
23:      **If** $m_j \neq M_j$ // if evaluating jth function at lower fidelity
24:       // two approximations are provided
25:       **If** approximation = TG
26:         H_2 += $H(y_j^{(m_j)}|D, \mathbf{x}, y_j^{(m_j)} \leq y_{j_s}^*)$ (entropy of truncated Gaussian $p(y_j^{(M_j)}|D, \mathbf{x}, \underline{y_j^{(m_j)} \leq y_{j_s}^*})$)
27:       **If** approximation = NI
28:         H_2 += $H(y_j^{(m_j)}|D, \mathbf{x}, y_j^{(M_j)} \leq y_{j_s}^*)$ (entropy computed via numerical integration)
29:   **end for**
30: **end for**
31: Divide by number of samples: $H_2 = H_2/S$
32: **return** $(H_1 - H_2)/\lambda^{(\mathbf{m})}$

By substituting (6.16), (6.17), and 6.19 into (6.15) we get:

$$H(y_j^{(m_j)}|D, \mathbf{x}, y_j^{(M_j)} \leq y_{j_s}^*) = -\int \Psi(y_j^{(m_j)}) \log(\Psi(y_j^{(m_j)})) dy_j^{(m_j)} \tag{6.20}$$

With $\Psi(y_j^{(m_j)}) = \Phi\left(\frac{y_{j_s}^* - \mu_j(\mathbf{x})}{s_j(\mathbf{x})}\right) \frac{\Phi(\gamma_s^{(M_j)}(\mathbf{x})))\phi(\gamma_j^{(m_j)}(\mathbf{x}))}{\sigma_j^{(m_j)}}$. Since this integral is over one-dimension

variable $y_j^{(m_j)}$, numerical integration can result in a tight approximation.

    A complete description of the MF-OSEMO algorithm is given in Algorithm 3. The blue colored steps correspond to computation of our acquisition function via sampling.

7. iMOCA Algorithm for Continuous-Fidelity MOO Problem

In this section, we address the continuous-fidelity MOO problem, we have access to alternative functions through which we can evaluate cheaper approximations of objective functions by varying a continuous fidelity variable. To solve this problem, we propose an algorithm referred to as **i**nformation-*Theoretic* **M**ulti-*Objective Bayesian* **O**ptimization *with* **C**ontinuous **A**pproximations (iMOCA). We first describe the complete details related to the continuous-fidelity MOO problem. Subsequently, we explain our proposed iMOCA algorithm with two mathematically different approximations of the output space entropy based acquisition function.

Problem Setup (Continuous-Fidelity MOO Problem). The continuous-fidelity MOO problem is the general version of the discrete multi-fidelity setting where we have access to $g_i(\mathbf{x}, z_i)$ where g_i is an alternative function through which we can evaluate cheaper approximations of f_i by varying the fidelity variable $z_i \in \mathcal{Z}$ (continuous function approximations). Without loss of generality, let $\mathcal{Z}=[0,1]$ be the fidelity space. Fidelities for each function f_i vary in the amount of resources consumed and the accuracy of evaluation, where $z_i=0$ and $z_i^*=1$ refer to the lowest and highest fidelity respectively. At the highest fidelity z_i^*, $g_i(\mathbf{x}, z_i^*) = f_i(\mathbf{x})$. The evaluation of an input $\mathbf{x} \in \mathcal{X}$ with fidelity vector $\mathbf{z} = [z_1, z_2, \cdots, z_K]$ produces an evaluation vector of K values denoted by $\mathbf{y} \equiv [y_1, y_2, \cdots, y_K]$, where $y_i = g_i(\mathbf{x}, z_i)$ for all $i \in \{1, 2, \cdots, K\}$. Let $\mathcal{C}_i(\mathbf{x}, z_i)$ be the cost of evaluating $g_i(\mathbf{x}, z_i)$. Our goal is to approximate the optimal Pareto set $\mathcal{X}^*$ over the highest fidelities functions while minimizing the overall cost of function evaluations (experiments). For example, in rocket launching research, we need to find designs that trade-off return-time and angular distance using continuous-fidelity simulators (e.g., varying tolerance parameter to trade-off simulation time and accuracy) for design evaluations. Table 3 contains all the mathematical notations used in this section (iMOCA).

Cost of Function Evaluations. The total normalized cost of function evaluation is $\mathcal{C}(\mathbf{x}, \mathbf{z}) = \sum_{i=1}^K \left(\mathcal{C}_i(\mathbf{x}, z_i)/\mathcal{C}_i(\mathbf{x}, z_i^*) \right)$. We normalize the cost of each function by the cost of its highest fidelity because the cost units of different objectives can be different. If the cost is known, it can be directly injected in the latter expression. However, in some real-world settings, the cost of a function evaluation can be only known after the function evaluation. In this case, the cost can be modeled by an independent Gaussian process. The predictive mean can be used during the optimization. The final goal is to recover $\mathcal{X}^*$ while minimizing the total cost of function evaluations.

Continuous-Fidelity GPs as Surrogate Models. Let $D = \{(\mathbf{x}_i, \mathbf{y}_i, \mathbf{z}_i)\}_{i=1}^{t-1}$ be the training data from past t-1 function evaluations, where $\mathbf{x}_i \in \mathcal{X}$ is an input and $\mathbf{y}_i = [y_1, y_2, \cdots, y_K]$ is the output vector resulting from evaluating functions $g_1, g_2, \cdots, g_K$ for $\mathbf{x}_i$ at fidelities $z_1, z_2, \cdots, z_K$ respectively. We learn from $\mathcal{D}$, K surrogate statistical models $\mathcal{GP}_1, \cdots, \mathcal{GP}_K$, where each model $\mathcal{GP}_j$ corresponds to the jth function g_j. Continuous fidelity GPs (CF-GPs) are capable of modeling functions with continuous fidelities within a single model. Hence, we employ CF-GPs to build surrogate statistical models for each function. Specifically, we use the CF-GP model proposed in (Kandasamy et al., 2017). W.l.o.g, we assume that our functions g_j are defined over the spaces $\mathcal{X} = [0,1]^d$ and $\mathcal{Z} = [0,1]$. Let $g_j \sim \mathcal{GP}_j(0, \kappa_j)$ such that $y_j = g_j(z_j, \mathbf{x}) + \epsilon$, where $\epsilon \sim \mathcal{N}(0, \eta^2)$ and κ :

$(\mathcal{Z} \times \mathcal{X})^2 \to \mathbb{R}$ is the prior covariance matrix defined on the product of input and fidelity spaces.

$$\kappa_j([z_j, \mathbf{x}], [z'_j, \mathbf{x}']) = \kappa_{j\mathcal{X}}(\mathbf{x}, \mathbf{x}') \cdot \kappa_{j\mathcal{Z}}(z_j, z'_j)$$

where $\kappa_{j\mathcal{X}}, \kappa_{j\mathcal{Z}}$ are radial kernels over $\mathcal{X}$ and $\mathcal{Z}$ spaces respectively. $\mathcal{Z}$ controls the smoothness of g_j over the fidelity space to be able to share information across different fidelities. A key advantage of this model is that it integrates all fidelities into one single GP for information sharing. We denote the posterior mean and standard deviation of g_j conditioned on D by $\mu_{g_j}(\mathbf{x}, z_j)$ and $\sigma_{g_j}(\mathbf{x}, z_j)$. We denote the posterior mean and standard deviation of the highest fidelity functions $f_j(\mathbf{x}) = g_j(\mathbf{x}, z_j^*)$ by $\mu_{f_j}(x) = \mu_{g_j}(\mathbf{x}, z_j^*)$ and $\sigma_{f_j}(\mathbf{x}) = \sigma_{g_j}(\mathbf{x}, z_j^*)$ respectively. We define $\sigma^2_{g_j, f_j}(x)$ as the predictive co-variance between a lower fidelity z_j and the highest fidelity z_j^* at the same $\mathbf{x}$.

Notation	Definition
$g_1, g_2, \cdots, g_K$	General objective functions with low and high fidelities
$\tilde{g}_j$	Function sampled from the jth Gaussian process model at fidelity z_j
$z_1, z_2, \cdots, z_K$	The fidelity variables for each function
$\mathbf{z}$	Fidelities vector
$\mathbf{z}^* = [z_1^*, z_2^*, \cdots, z_K^*]$	Fidelities vector with all fidelities at their highest value
y_j	jth function g_j evaluated at fidelity z_j
$\mathbf{y} = [y_1, y_2, \cdots, y_K]$	Output vector resulting from evaluating $g_1, g_2, \cdots, g_K$ for $\mathbf{x}_i$ at fidelities $z_1, z_2, \cdots, z_K$ respectively
$\mathbf{f} = [f_1, f_2, \cdots, f_K]$	Output vector resulting from evaluating functions $f_1, f_2, \cdots, f_K$ or equivalently $g_1, g_2, \cdots, g_K$ for $\mathbf{x}_i$ at fidelities $z_1^*, z_2^*, \cdots, z_K^*$ respectively
$\mathcal{C}_j(\mathbf{x}, z_j)$	Cost of evaluating jth function g_j at fidelity z_j
$\mathcal{C}(\mathbf{x}, \mathbf{z})$	Total normalized cost $\mathcal{C}(\mathbf{x}, \mathbf{z}) = \sum_{i=1}^{K} (\mathcal{C}_i(\mathbf{x}, z_i) / \mathcal{C}_i(\mathbf{x}, z_i^*))$
$\mathcal{Z}$	Fidelity space
$\mathcal{Z}_t^{(j)}$	Reduced fidelity space for function g_j at iteration t
$\mathcal{Z}_r$	Reduced fidelity space
ξ	Information gap
$\beta_t^{(j)}$	Exploration/exploitation parameter for function g_j at iteration t

Table 3: Mathematical notations and their associated definition used in this section (iMOCA)

7.1 iMOCA Algorithm with Two Approximations

We first describe the key idea behind our proposed iMOCA algorithm including the main challenges. Next, we present our algorithmic solution to address those challenges.

Key Idea of iMOCA: The acquisition function behind iMOCA employs principle of output space entropy search to select the sequence of input and fidelity-vector (one for each objective) pairs. iMOCA is applicable for solving MO problems in both continuous and discrete fidelity settings. *The key idea is to find the pair $\{\mathbf{x}_t, \mathbf{z}_t\}$ that maximizes the information gain I per unit cost about the Pareto front of the highest fidelities* (denoted by

$\mathcal{Y}^*$), where $\{\mathbf{x}_t, \mathbf{z}_t\}$ represents a candidate input $\mathbf{x}_t$ evaluated at a vector of fidelities $\mathbf{z}_t = [z_1, z_2, \cdots, z_K]$ at iteration t. Importantly, iMOCA performs joint search over input space $\mathcal{X}$ and reduced fidelity space $\mathcal{Z}_r$ over fidelity vectors for this selection.

$$(\mathbf{x}_t, \mathbf{z}_t) \leftarrow \arg max_{\mathbf{x} \in \mathcal{X}, \mathbf{z} \in \mathcal{Z}_r} \; \alpha_t(\mathbf{x}, \mathbf{z}) \text{ , where } \quad \alpha_t(\mathbf{x}, \mathbf{z}) = I(\{\mathbf{x}, \mathbf{y}, \mathbf{z}\}, \mathcal{Y}^*|D)/\mathcal{C}(\mathbf{x}, \mathbf{z}) \quad (7.1)$$

In the following sections, we describe the details and steps of our proposed algorithm iMOCA. We start by explaining the bottlenecks of continuous fidelity optimization due to the infinite size of the fidelity space followed by describing a principled approach to reduce the fidelity space. Subsequently, we present the computational steps of our proposed acquisition function: Information gain per unit cost for each candidate input and fidelity-vector pair.

7.1.1 APPROACH TO REDUCE FIDELITY SEARCH SPACE

In this work, we focus primarily on MO problems with continuous fidelity space. The continuity of this space results in infinite number of fidelity choices. Thus, selecting an informative and meaningful fidelity becomes a major bottleneck. Therefore, we reduce the search space over fidelity-vector variables in a principled manner guided by the learned statistical models (Kandasamy et al., 2017). Our fidelity space reduction method is inspired from BOCA for single-objective optimization (Kandasamy et al., 2017). We apply the method in BOCA to each of the objective functions to be optimized in MO setting.

A favourable setting for continuous-fidelity methods would be for the lower fidelities g_j to be informative about the highest fidelity f_j. Let h_j be the bandwith parameter of the fidelity kernel $\kappa_{j\mathcal{Z}}$ and let $\xi : \mathcal{Z} \rightarrow [0, 1]$ be a measure of the gap in information about $g_j(., z_j^*)$ when queried at $z_j \neq z_j^*$ with $\xi(z_j) \approx \frac{\|z_j - z_j^*\|}{h_j}$ for the squared exponential kernels (Kandasamy et al., 2017). A larger h_j will result in g_j being smoother across $\mathcal{Z}$. Consequently, lower fidelities will be more informative about f_j and the information gap $\xi(z_j)$ will be smaller.

To determine an informative fidelity for each function in iteration t, we reduce the space $\mathcal{Z}$ and select z_j from the subset $\mathcal{Z}_t^{(j)}$ defined as follows:

$$\mathcal{Z}_t^{(j)}(\mathbf{x}) = \{\{z_j \in \mathcal{Z}_{\setminus\{z_j^*\}}, \sigma_{g_j}(\mathbf{x}, z_j) > \gamma(z_j), \xi(z_j) > \beta_t^{(j)}\|\xi\|_\infty\} \cup \{z_j^*\}\} \quad (7.2)$$

where $\gamma(z_j) = \xi(z_j)(\frac{\mathcal{C}_j(\mathbf{x}, z_j)}{\mathcal{C}_j(\mathbf{x}, z_j^*)})^q$ and $q = \frac{1}{p_j + d + 2}$ with p_j, d the dimensions of $\mathcal{Z}$ and $\mathcal{X}$ respectively. Without loss of generality, we assume that $p_j = 1$. $\beta_t^{(j)} = \sqrt{0.5d \cdot log \,(2tl + 1)}$ is the exploration/exploitation parameter (Kandasamy et al., 2017). where, l is the effective L_1 diameter of $\mathcal{X}$ and is computed by scaling each dimension by the inverse of the bandwidth of the SE kernel for that dimension. We denote by $\mathcal{Z}_r = \{\mathcal{Z}_t^{(j)}, j \in \{1 \ldots K\}\}$, the reduced fidelity space for all K functions.

We filter out the fidelities for each objective function at BO iteration t using the above-mentioned two conditions. We provide intuitive explanation of these conditions below.
The first condition $\sigma_{g_j}(\mathbf{x}, z_j) > \gamma(z_j)$: A reasonable multi-fidelity strategy would query the cheaper fidelities in the beginning to learn about the function g_j by consuming the least possible cost budget and later query from higher fidelities in order to gain more accurate

information. Since the final goal is to optimize f_j, the algorithm should also query from the highest-fidelity. However, the algorithm might never query from higher fidelities due to their high cost. This condition will make sure that lower fidelities are likely to be queried, but not excessively and the algorithm will move toward querying higher fidelities as iterations progress. Since $\gamma(z_j)$ is monotonically increasing in $\mathcal{C}_j$, this condition can be easily satisfied by cheap fidelities. However, if a fidelity is very far from z_j^*, then the information gap ξ will increase and hence, uninformative fidelities would be discarded. Therefore, $\gamma(z_j)$ will guarantee achieving a good trade-off between resource cost and information.

The second condition $\xi(z_j) > \beta_t^{(j)}\|\xi\|_\infty$: We recall that if the first subset of $\mathcal{Z}_t^{(j)}$ is empty, the algorithm will automatically evaluate the highest-fidelity z_j^*. However, if it is not empty, and since the fidelity space is continuous (infinite number of choices for z_j), the algorithm might query fidelities that are very close to z_j^* and would cost nearly the same as z_j^* without being as informative as z_j^*. The goal of this condition is to prevent such situations by excluding fidelities in the small neighborhood of z_j^* and querying z_j^* instead. Since $\beta_t^{(j)}$ increases with t and ξ is increasing as we move away from z_j^*, this neighborhood is shrinking and the algorithm will eventually query z_j^*.

7.1.2 NAIVE-CFMO: A SIMPLE CONTINUOUS-FIDELITY MO BASELINE

In this section, we first describe a simple baseline approach referred to as *Naive-CFMO* to solve continuous-fidelity MO problems by combining the above-mentioned fidelity space reduction approach with existing multi-objective BO methods. Next, we explain the key drawbacks of Naive-CFMO and how our proposed iMOCA algorithm overcomes them.

A straightforward way to construct a continuous-fidelity MO method is to perform a two step selection process similar to the continuous-fidelity single-objective BO algorithm proposed in (Kandasamy et al., 2017):

Step 1) Select the input $\mathbf{x}$ that maximizes the acquisition function at the *highest fidelity*. This can be done using any existing multi-objective BO algorithm.

Step 2) Evaluate $\mathbf{x}$ at the *cheapest valid fidelity for each function* in the reduced fidelity space $\mathcal{Z}_t^{(j)}(\mathbf{x})$ computed using the reduction approach mentioned in the previous section. Since we are studying information gain based methods in this work, we instantiate Naive-CFMO using the state-of-the-art information-theoretic MESMO algorithm (Belakaria et al., 2019) for Step 1. Algorithm 5 shows the complete pseudo-code of Naive-CFMO.

Drawbacks of Naive-CFMO: Unfortunately, Naive-CFMO has two major drawbacks.

- The acquisition function solely relies on the highest-fidelity f_j. Therefore, it does not capture and leverage the statistical relation between different fidelities and full-information provided by the global function g_j.
- Generally, there is a dependency between the fidelity space and the input space in continuous-fidelity problems. Therefore, selecting an input that maximizes the highest-fidelity and then evaluating it at a different fidelity can result in a mismatch in the evaluation process leading to poor performance and slower convergence.

iMOCA vs. Naive-CFMO: Our proposed iMOCA algorithm overcomes the drawbacks of Naive-CFMO as follows.

- iMOCA's acquisition function maximizes the information gain per unit cost across all fidelities by capturing the relation between fidelities and the impact of resource cost on information gain.

- iMOCA performs joint search over input and fidelity space to select the input variable $\mathbf{x} \in \mathcal{X}$ and fidelity variables $\mathbf{z} \in \mathcal{Z}_r$ while maximizing the proposed acquisition function. Indeed, our experimental results demonstrate the advantages of iMOCA over Naive-CFMO.

7.1.3 INFORMATION-THEORETIC CONTINUOUS-FIDELITY ACQUISITION FUNCTION

In this section, we explain the technical details of the acquisition function behind iMOCA. We propose two approximations for the computation of information gain per unit cost.

The information gain in equation (7.1) is defined as the expected reduction in entropy $H(.)$ of the posterior distribution $P(\mathcal{Y}^*|D)$ due to evaluating $\mathbf{x}$ at fidelity vector $\mathbf{z}$. Based on the symmetric property of information gain, the latter can be rewritten as follows:

$$I(\{\mathbf{x}, \mathbf{y}, \mathbf{z}\}, \mathcal{Y}^*|D) = H(\mathbf{y}|D, \mathbf{x}, \mathbf{z}) - \mathbb{E}_{\mathcal{Y}^*}[H(\mathbf{y}|D, \mathbf{x}, \mathbf{z}, \mathcal{Y}^*)] \qquad (7.3)$$

In equation (7.3), the first term is the entropy of a K-dimensional Gaussian distribution that can be computed in closed form as follows:

$$H(\mathbf{y}|D, \mathbf{x}, \mathbf{z}) = \sum_{j=1}^{K} \ln(\sqrt{2\pi e} \; \sigma_{g_j}(\mathbf{x}, z_j)) \qquad (7.4)$$

In equation (7.3), the second term is an expectation over the Pareto front of the highest fidelities $\mathcal{Y}^*$. This term can be approximated using Monte-Carlo sampling:

$$\mathbb{E}_{\mathcal{Y}^*}[H(\mathbf{y}|D, \mathbf{x}, \mathbf{z}, \mathcal{Y}^*)] \simeq \frac{1}{S} \sum_{s=1}^{S} [H(\mathbf{y}|D, \mathbf{x}, \mathbf{z}, \mathcal{Y}_s^*)] \qquad (7.5)$$

where S is the number of samples and $\mathcal{Y}_s^*$ denote a sample Pareto front obtained over the highest fidelity functions sampled from K surrogate models. To compute equation (7.5), we provide algorithmic solutions to construct Pareto front samples $\mathcal{Y}_s^*$ and to compute the entropy with respect to a given Pareto front sample $\mathcal{Y}_s^*$.

1) Computing Pareto Front Samples: We first sample highest fidelity functions $\tilde{f}_1, \cdots, \tilde{f}_K$ from the posterior CF-GP models via random Fourier features (Hernández-Lobato et al., 2014; Rahimi & Recht, 2008). This is done similar to prior work (Hernández-Lobato et al., 2016; Wang & Jegelka, 2017). We solve a cheap MO optimization problem over the K sampled functions $\tilde{f}_1, \cdots, \tilde{f}_K$ using the popular NSGA-II algorithm (Deb et al., 2002a) to compute the sample Pareto front $\mathcal{Y}_s^*$.

2) Entropy Computation for a Given Pareto Front Sample: Let $\mathcal{Y}_s^* = \{\mathbf{v}^1, \cdots, \mathbf{v}^l\}$ be the sample Pareto front, where l is the size of the Pareto front and each $\mathbf{v}^i = \{v_1^i, \cdots, v_K^i\}$

is a K-vector evaluated at the K sampled highest-fidelity functions. The following inequality holds for each component y_j of $\mathbf{y} = (y_1, \cdots, y_K)$ in the entropy term $H(\mathbf{y}|D, \mathbf{x}, \mathbf{z}, \mathcal{Y}_s^*)$:

$$y_j \leq f_s^{j*} \quad \forall j \in \{1, \cdots, K\} \tag{7.6}$$

where $f_s^{j*} = \max\{v_j^1, \cdots v_j^l\}$. Essentially, this inequality says that the jth component of $\mathbf{y}$ (i.e., y_j) is upper-bounded by a value, which is the maximum of jth components of all l vectors $\{\mathbf{v}^1, \cdots, \mathbf{v}^l\}$ in the Pareto front $\mathcal{Y}_s^*$. Inequality (7.6) holds by the same proof of (6.11). *For the ease of notation, we drop the dependency on $\mathbf{x}$ and $\mathbf{z}$. We use f_j to denote $f_j(x) = g_j(x, z_j^*)$, the evaluation of the highest fidelity at x and y_j to denote $g_j(x, z_j)$ the evaluation of g_j at a lower fidelity $z_j \neq z_j^*$.*

By combining the inequality (7.6) and the fact that each function is modeled as an independent CF-GP, a common property of entropy measure allows us to decompose the entropy of a set of independent variables into a sum over entropies of individual variables (Cover & Thomas, 2012):

$$H(\mathbf{y}|D, \mathbf{x}, \mathbf{z}, \mathcal{Y}_s^*) = \sum_{j=1}^{K} H(y_j|D, \mathbf{x}, z_j, f_s^{j*}) \tag{7.7}$$

The computation of (7.7) requires the computation of the entropy of $p(y_j|D, \mathbf{x}, z_j, f_s^{j*})$. This is a conditional distribution that depends on the value of z_j and can be expressed as $H(y_j|D, \mathbf{x}, z_j, y_j \leq f_s^{j*})$. This entropy can be computed using two different approximations as described below.

Truncated Gaussian Approximation (iMOCA-T): As a consequence of (7.6), which states that $y_j \leq f_s^{j*}$ also holds for all fidelities, the entropy of $p(y_j|D, \mathbf{x}, z_j, f_s^{j*})$ can also be approximated by the entropy of a truncated Gaussian distribution and expressed as follows:

$$H(y_j|D, \mathbf{x}, z_j, y_j \leq f_s^{j*}) \simeq \ln(\sqrt{2\pi e}\, \sigma_{g_j}) + \ln \Phi(\gamma_s^{(g_j)}) - \frac{\gamma_s^{(g_j)}\phi(\gamma_s^{(g_j)})}{2\Phi(\gamma_s^{(g_j)})} \text{ where } \gamma_s^{(g_j)} = \frac{f_s^{j*} - \mu_{g_j}}{\sigma_{g_j}} \tag{7.8}$$

From equations (7.5), (7.4), and (7.8), we get the final expression of iMOCA-T as shown below:

$$\alpha_t(\mathbf{x}, \mathbf{z}, \mathcal{Y}^*) \simeq \frac{1}{\mathcal{C}(\mathbf{x}, \mathbf{z})S} \sum_{j=1}^{K} \sum_{s=1}^{S} \left[\frac{\gamma_s^{(g_j)}\phi(\gamma_s^{(g_j)})}{2\Phi(\gamma_s^{(g_j)})} - \ln(\Phi(\gamma_s^{(g_j)})) \right] \tag{7.9}$$

Extended-skew Gaussian Approximation (iMOCA-E): Although equation (7.9) is sufficient for computing the entropy, this entropy can be mathematically interpreted and computed with a different approximation. The condition $y_j \leq f_s^{j*}$, is originally expressed as $f_j \leq f_s^{j*}$. Substituting this condition with it's original equivalent, the entropy becomes $H(y_j|D, \mathbf{x}, z_j, f_j \leq f_s^{j*})$. Since y_j is an evaluation of the function g_j while f_j is an evaluation of the function f_j, we observe that $y_j|f_j \leq f_s^{j*}$ can be approximated by an extended-skew

Gaussian (ESG) distribution (Moss et al., 2020; Azzalini, 1985). It has been shown that the differential entropy of an ESG does not have a closed form expression (Arellano-Valle, CONTRERAS-REYES, & Genton, 2013). Therefore, we derive a simplified expression where most of the terms are analytical by manipulating the components of the entropy. We apply the derivation of the entropy based on ESG formulation, proposed by Moss et al. (2020), to the multi-objective setting.

In order to simplify the calculation $H(y_j | D, \mathbf{x}, z_j, f_j \leq f_s^{j*})$, let us define the normalized variable $\Gamma_{f_s^{j*}}$ as $\Gamma_{f_s^{j*}} \sim \frac{y_j - \mu_{g_j}}{\gamma_{g_j}} | f_j \leq f_s^{j*}$. $\Gamma_{f_s^{j*}}$ is distributed as an ESG with p.d.f whose mean $\mu_{\Gamma_{f_s^{j*}}}$ and variance $\sigma_{\Gamma_{f_s^{j*}}}$ are defined in Appendix A. We define the predictive correlation between y_j and f_j as $\tau = \frac{\sigma^2_{g_j, f_j}}{\sigma_{g_j} \sigma_{f_j}}$. The entropy can be computed using the following expression. Due to lack of space, we only provide the final expression. Complete derivation for equations (7.10) and (7.11) are provided in Appendix A.

$$H(y_j | D, \mathbf{x}, z_j, f_j \leq f_s^{j*}) \simeq \ln(\sqrt{2\pi e} \, \sigma_{g_j}) + \ln(\Phi(\gamma_s^{(f_j)})) - \tau^2 \frac{\phi(\gamma_s^{(f_j)}) \gamma_s^{(f_j)}}{2\Phi(\gamma_s^{(f_j)})}$$
$$- \mathbb{E}_{u \sim \Gamma_{f_s^{j*}}} \left[\ln(\Phi(\frac{\gamma_s^{(f_j)} - \tau u}{\sqrt{1 - \tau^2}})) \right] \qquad (7.10)$$

From equations (7.5), (7.4) and (7.10), the final expression of iMOCA-E can be expressed as follow:

$$\alpha_t(\mathbf{x}, \mathbf{z}, \mathcal{Y}^*) \simeq \frac{1}{\mathcal{C}(\mathbf{x}, \mathbf{z}) S} \sum_{j=1}^{K} \sum_{s=1}^{S} \left[\tau^2 \frac{\gamma_s^{(f_j)} \phi(\gamma_s^{(f_j)})}{2\Phi(\gamma_s^{(f_j)})} - \ln(\Phi(\gamma_s^{(f_j)})) + \mathbb{E}_{u \sim \Gamma_{f_s^{j*}}} \left[\ln(\Phi(\frac{\gamma_s^{(f_j)} - \tau u}{\sqrt{1 - \tau^2}})) \right] \right]$$
$$(7.11)$$

The expression given by equation (7.11) is mostly analytical except for the last term. We perform numerical integration via Simpson's rule using $\mu_{\Gamma_{f_s^{j*}}} \mp \gamma \sqrt{\sigma(\Gamma_{f_s^{j*}})}$ as the integral limits. Since this integral is over one-dimension variable, numerical integration can result in a tight approximation with low computational cost. Complete pseudo-code of iMOCA is shown in Algorithm 4.

Generality of the Two Approximations: We observe that for any fixed value of $\mathbf{x}$, when we choose the highest-fidelity for each function $\mathbf{z} = \mathbf{z}^*$, **a)** For iMOCA-T, we will have $g_i = f_j$; and **b)** For iMOCA-E, we will have $\tau = 1$. Consequently, both equation (7.9) and equation (7.11) will degenerate to the acquisition function of MESMO optimizing only highest-fidelity functions given in equation (4.13) in section 4.

The main advantages of our proposed acquisition function are: cost-efficiency, computational-efficiency, and robustness to the number of Monte-Carlo samples. Indeed, our experiments demonstrate these advantages over state-of-the-art single-fidelity MO algorithms.

Algorithm 4 iMOCA Algorithm

Input: input space $\mathcal{X}$; K blackbox functions f_j and their continuous approximations g_j; total budget $\mathcal{C}_{total}$

1: Initialize continuous fidelity Gaussian process $\mathcal{GP}_1, \cdots, \mathcal{GP}_K$ by initial points D
2: **While** $\mathcal{C}_t \leq \mathcal{C}_{total}$ **do**
3:     for each sample $s \in 1, \cdots, S$:
4:        Sample highest-fidelity functions $\tilde{f}_j \sim \mathcal{GP}_j(., z_j^*)$

5:        $\mathcal{Y}_s^* \leftarrow$ Solve *cheap* MOO over $(\tilde{f}_1, \cdots, \tilde{f}_K)$
6:     Find the query based on $\mathcal{Y}^* = \{\mathcal{Y}_s^*, s \in \{1 \ldots S\}\}$
7:     // Choose one of the two approximations
8:     **If** approx = T // Use eq (7.9) for α_t (iMOCA-T)
9:        select $(\mathbf{x}_t, \mathbf{z}_t) \leftarrow \arg max_{\mathbf{x} \in \mathcal{X}, \mathbf{z} \in \mathcal{Z}_r}\ \alpha_t(\mathbf{x}, \mathbf{z}, \mathcal{Y}^*)$
10:    **If** approx = E // Use eq (7.11) for α_t (iMOCA-E)
11:       select $(\mathbf{x}_t, \mathbf{z}_t) \leftarrow \arg max_{\mathbf{x} \in \mathcal{X}, \mathbf{z} \in \mathcal{Z}_r}\ \alpha_t(\mathbf{x}, \mathbf{z}, \mathcal{Y}^*)$

12:    Update the total cost: $\mathcal{C}_t \leftarrow \mathcal{C}_t + \mathcal{C}(\mathbf{x}_t, \mathbf{z}_t)$
13:    Aggregate data: $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\mathbf{x}_t, \mathbf{y}_t, \mathbf{z}_t)\}$
14:    Update models $\mathcal{GP}_1, \cdots, \mathcal{GP}_K$
15:    $t \leftarrow t + 1$
16: **end while**
17: **return** Pareto front and Pareto set of black-box functions $f_1(x), \cdots, f_K(x)$

Algorithm 5 Naive-CFMO Algorithm

Input: input space $\mathcal{X}$; K blackbox functions f_j and their continuous approximations g_j; total budget $\mathcal{C}_{total}$

1: Initialize continuous fidelity Gaussian process $\mathcal{GP}_1, \cdots, \mathcal{GP}_K$ by evaluating at initial points D
2: **While** $\mathcal{C}_t \leq \mathcal{C}_{total}$ **do**
3:     for each sample $s \in 1, \cdots, S$:
4:        Sample highest-fidelity functions $\tilde{f}_j \sim \mathcal{GP}_j(., z_j^*)$

5:        $\mathcal{Y}_s^* \leftarrow$ Solve *cheap* MOO over $(\tilde{f}_1, \cdots, \tilde{f}_K)$
6:     Find the query based on $\mathcal{Y}^* = \{\mathcal{Y}_s^*, s \in \{1 \ldots S\}\}$:
7:     // Use eq (4.13) for α_t (MESMO)
8:     select $\mathbf{x}_t \leftarrow \arg max_{\mathbf{x} \in \mathcal{X}}\ \alpha_t(\mathbf{x}, \mathcal{Y}^*)$
9:     **for** $j \in 1 \cdots K$ **do**
10:       select $z_j \leftarrow \arg min_{\mathbf{z_j} \in \mathcal{Z}_t^{(j)}(\mathbf{x_t}) \cup \{z_j^*\}}\ \mathcal{C}_i(x_t, z_j)$
11:    Fidelity vector $\mathbf{z}_t \leftarrow [z_1 \ldots z_K]$
12:    Update the total cost: $\mathcal{C}_t \leftarrow \mathcal{C}_t + \mathcal{C}(\mathbf{x}_t, \mathbf{z}_t)$
13:    Aggregate data: $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\mathbf{x}_t, \mathbf{y}_t, \mathbf{z}_t)\}$
14:    Update models $\mathcal{GP}_1, \cdots, \mathcal{GP}_K$
15:    $t \leftarrow t + 1$
16: **end while**
17: **return** Pareto front and Pareto set of black-box functions $f_1(x), \cdots, f_K(x)$

---

8. Experiments and Results

In this section, we first describe the experimental evaluation of MESMO (single-fidelity algorithm), MF-OSEMO (discrete multi-fidelity algorithm) and iMOCA (continuous-fidelity algorithm) on synthetic and real-world engineering problems. Subsequently, we present experimental results of MESMOC (constrained MO algorithm) on two real-world engineering problems, namely, electrified aviation power system design and analog circuit design.

8.1 Experimental Evaluation of iMOCA, MF-OSEMO, and MESMO

We mainly present the results for iMOCA with MESMO and MF-OSEMO as baselines for the following reasons: First, iMOCA is the generalisation of both MESMO and MF-OSEMO to the most general setting (continuous-fidelity); and second, the performance, robustness, and effectiveness of MESMO and MF-OSEMO have been shown in (Belakaria et al., 2019) and (Belakaria et al., 2020a) respectively.

Experimental Setup. In our experiments, we employed CF-GP models as described in section 7 with squared exponential kernels. We initialize the surrogate models of all functions with the same number of points selected randomly from both lower and higher fidelities. We compare iMOCA with several baselines: six state-of-the-art single-fidelity MO algorithms (ParEGO, SMSego, EHI, SUR, PESMO, and MESMO) and one naive continuous-fidelity baseline that we proposed in Section 7.1.2. We employ the code for ParEGO, PESMO, SMSego, EHI, and SUR from the BO library Spearmint[6]. The code for all four of our algorithms are available in public Github repositories. We provide more details about the algorithms parameters, libraries, and computational resources in the Ap-

---

6. github.com/HIPS/Spearmint/tree/PESM

pendix B.2. For experiments in discrete fidelity setting, the number of fidelities is very limited. Thus, the fidelity space reduction method deem meaningless in this case. Therefore, we employ iMOCA without fidelity space reduction for those scenarios. Additionally, we compare to the state-of-the-art discrete fidelity method MF-OSEMO. MF-OSEMO has two variants: MF-OSEMO-TG and MF-OSEMO-NI. Since MF-OSEMO-TG has the same formulation as iMOCA-T and provide similar results, we compare only to MF-OSEMO-NI.

8.1.1 Synthetic Benchmarks

We evaluate our most general algorithm iMOCA and baselines on four different synthetic benchmarks. We construct two problems using a combination of benchmark functions for continuous-fidelity and single-objective optimization (Surjanovic & Bingham, 2020): *Branin,Currin* (with $K{=}2$, $d{=}2$) and *Ackley, Rosen, Sphere* (with $K{=}3$, $d{=}5$). To show the effectiveness of iMOCA on settings with discrete fidelities, we employ two of the known general MO benchmarks: *QV* (with $K{=}2$, $d{=}8$) and *DTLZ1* (with $K{=}6$, $d{=}5$) (Habib, Singh, & et al., 2019; Shu, Jiang, Zhou, Shao, Hu, & Meng, 2018). We provide their complete details in Appendix B.1. The titles of plots in Fig. 4, Fig. 8, and Fig. 5 denote the corresponding experiments.

8.1.2 Real-world Engineering Design Optimization Problems

We evaluate iMOCA and baselines on four real-world design optimization problems from diverse engineering domains. We provide the details of these problems below.

1) Analog Circuit Design Optimization. Design of a voltage regulator via Cadence circuit simulator that imitate the real hardware (Belakaria, Zhou, Deshwal, Doppa, Pande, & Heo, 2020d; Hong & et al, 2019). The simulation time can be adjusted to vary the simulation from fast and inaccurate to long and accurate. Each candidate circuit design is defined by 33 input variables ($d{=}33$). We optimize nine objectives: efficiency, four output voltages, and four output ripples. This problem has a continuous-fidelity space with cost varying from 10 mins to 120 mins.

2) Panel Structure Design for Large Vessels. The deck structure in large vessels commonly require the design of panels resisting uni-axial compression in the direction of the stiffeners (Zhu, Wang, & Collette, 2014). We consider optimizing the trade-off between two objective functions: weight and strength of the panel. These functions depend on six input variables ($d{=}6$): one of them is the number of stiffeners used and five others relating to the plate thickness and stiffener dimensions. This problem has a discrete fidelity setting: two fidelities with computational costs 1 min and 21 mins respectively.

3) Rocket Launching Simulation. Rocket launching studies (Hasbun, 2012) require several long and computationally-expensive simulations to reach an optimal design. In this problem, we have three input variables ($d = 3$): mass of fuel, launch height, and launch angle. The three objective functions are return time, angular distance, and difference between the launch angle and the radius at the point of launch. The simulator has a parameter that can be adjusted to perform continuous fidelity simulations. We employ the parameter range to vary the cost from 0.05 to 30 mins.

4) Network-On-Chip Design. Communication infrastructure is critical for efficient data movement in hardware chips (Joardar et al., 2018; Deshwal et al., 2019; Choi et al., 2018; Das et al., 2017) and they are designed using cycle-accurate simulators. We consider a dataset of 1024 configurations of a network-on-chip with ten input variables (d=10) (Che, Boyer, Meng, Tarjan, Sheaffer, Lee, & et al., 2009). We optimize two objectives: latency and energy. This problem has two discrete fidelities with costs 3 mins and 45 mins respectively.

8.1.3 RESULTS AND DISCUSSION

We compare iMOCA with both approximations (iMOCA-T and iMOCA-E) to all baselines. We employ two known metrics for evaluating the quality of a given Pareto front: *Pareto hypervolume (PHV)* metric and R_2 indicator. *PHV* (Zitzler, 1999) is defined as the volume between a reference point and the given Pareto front; and R_2 (Picheny, Wagner, & Ginsbourger, 2013b) is a distance-based metric defined as the average distance between two Pareto-fronts. We report both the difference in the hyper-volume, and the average distance between an optimal Pareto front ($\mathcal{F}^*$) and the best recovered Pareto front estimated by optimizing the posterior mean of the models at the highest fidelities (Hernández-Lobato et al., 2016). The mean and variance of *PHV* and R_2 metrics across 10 different runs are reported as a function of the total cost.

Fig. 4 shows the *PHV* results of all the baselines and iMOCA for synthetic and real-world experiments (Fig. 5 shows the corresponding R_2 results). We observe that iMOCA consistently outperforms all baselines. Both iMOCA-T and iMOCA-E have lower converge cost. Additionally, iMOCA-E shows a better convergence rate than iMOCA-T. This result can be explained by its tighter approximation. Nevertheless, iMOCA-T displays very close or sometimes better results than iMOCA-E. This demonstrates that even with loose approximation, using the iMOCA-T approximation can provide consistently competitive results using less computation time. For experiments with the discrete fidelity setting, iMOCA most of the times outperformed MF-OSEMO or produced very close results. It is important to note that MF-OSEMO is an algorithm designed specifically for the discrete-fidelity setting. Therefore, the competitive performance of iMOCA shows its effectiveness and generalisability.

Figure 8 in appendix B.3 shows the results of evaluating iMOCA and PESMO with varying number of Monte-Carlo samples $S \in \{1, 10, 100\}$. For ease of comparison and readability, we present these results in two different figures side by side. We observe that the convergence rate of PESMO is dramatically affected by the number of MC samples S. However, iMOCA-T and iMOCA-E maintain a better performance consistently even with a single sample. These results strongly demonstrate that our method *iMOCA is much more robust to the number of Monte-Carlo samples*.

Cost Reduction Factor. We also provide the *cost reduction factor* for experiments with continuous fidelities, which is the percentage of gain in the convergence cost when compared to the best performing baseline (the earliest cost for which any of the single-fidelity baselines converge). Although this metric gives advantage to baselines, the results in Table 4 show a consistently high gain ranging from 52.1% to 85%.
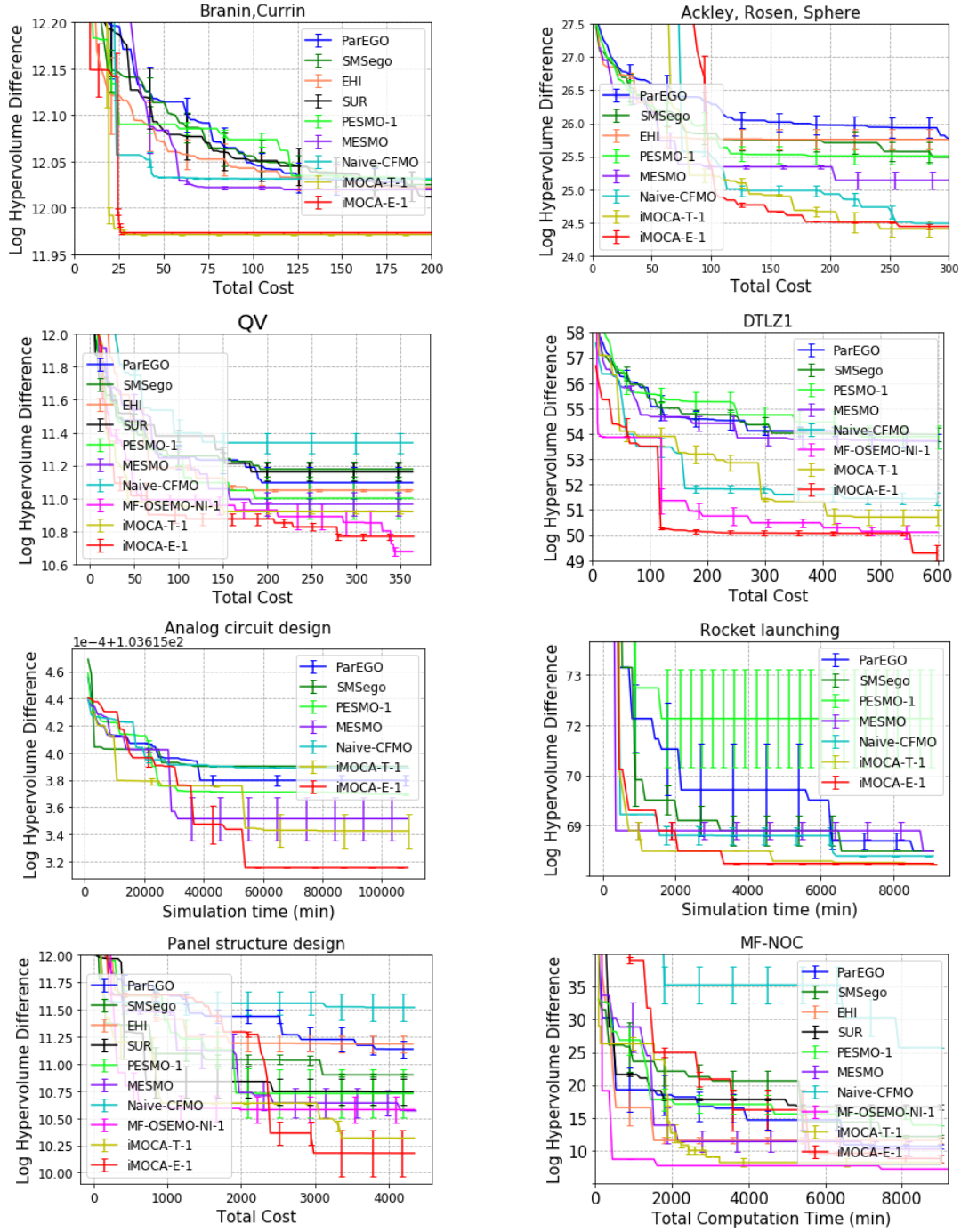
Figure 4: Results of iMOCA and the baselines algorithms on synthetic benchmarks and real-world problems. The PHV metric is presented against the total resource cost of function evaluations.
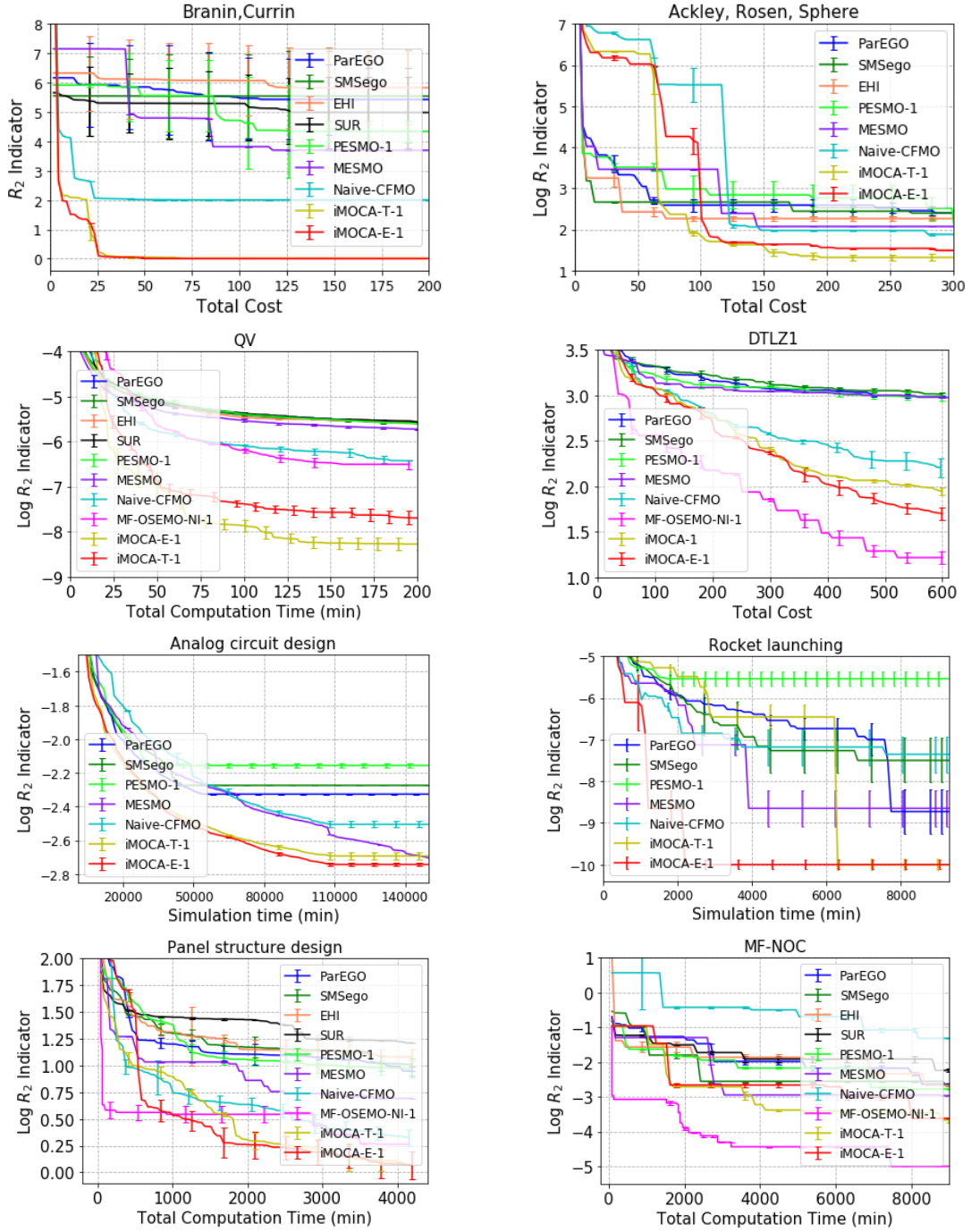
Figure 5: Results of iMOCA and the baselines algorithms on synthetic benchmarks and real-world problems. The R_2 metric is presented against the total resource cost of function evaluations.

Table 4: *Best* convergence cost from all baselines $\mathcal{C}_B$, *Worst* convergence cost for iMOCA $\mathcal{C}$, and cost reduction factor $\mathcal{G}$.

Name	BC	ARS	Circuit	Rocket
$\mathcal{C}_B$	200	300	115000	9500
$\mathcal{C}$	30	100	55000	2000
$\mathcal{G}$	85%	66.6%	52.1%	78.9%

8.2 Experimental Evaluation of MESMOC

Experimental Setup: In this section, we compare MESMOC with PESMOC (Garrido-Merchán & Hernández-Lobato, 2019), the state-of-the-art BO algorithm for solving constrained MO problems and MESMOC+ (Fernández-Sánchez et al., 2020), the concurrent approach which also relies on the same principle of output space entropy search. Due to lack of BO approaches for constrained MO setting, we also compare to known genetic algorithms: NSGA-II (Deb et al., 2002a) and MOEAD (Zhang & Li, 2007). However, they require large number of function evaluations to converge which is not practical for the optimization of expensive functions. We employ a GP based statistical model with squared exponential (SE) kernel in all our experiments. The hyper-parameters are estimated after every five function evaluations (iterations). We initialize the GP models for all functions by sampling the initial points at random. We employ the code for PESMOC and MESMOC+ from the BO library Spearmint[7]. We employ NSGA-II and MOEAD from the Platypus library[8]. Our code for MESMOC is available at the following Github repository [9]. We provide additional details about the algorithms parameters, libraries, and computational resources in the Appendix B.2.

8.2.1 REAL-WORLD ENGINEERING DESIGN PROBLEMS

Below we provide the details of the two real-world problems and associated optimization task that are employed for our experimental evaluation.

1) Electrified Aviation Power System Design. We consider optimizing the design of electrified aviation power system of unmanned aerial vehicle (UAV) via a time-based static simulation. The UAV system architecture consists of a central Li-ion battery pack, hex-bridge DC-AC inverters, PMSM motors, and necessary wiring (Belakaria, Jackson, Cao, Doppa, & Lu, 2020c). Each candidate input consists of a set of 5 (d=5) variable design parameters such as the battery pack configuration (battery cells in series, battery cells in parallel) and motor size (number of motors, motor stator winding length, motor stator winding turns). We minimize two objective functions: mass and total energy. This problem

---

7. github.com/EduardoGarrido90/Spearmint
8. platypus.readthedocs.io/en/latest/getting-started.html#installing-platypus
9. github.com/belakaria/MESMOC

has 5 black-box constraints:

$$C_0 : \text{Maximum final depth of discharge} \leq 75\%$$
$$C_1 : \text{Minimum cell voltage} \geq 3V$$
$$C_2 : \text{Maximum motor temperature} \leq 125°C$$
$$C_3 : \text{Maximum inverter temperature} \leq 120°C$$
$$C_5 : \text{ Maximum modulation index} \leq 1.3$$

2) Analog Circuit Optimization Domain. We consider optimizing the design of a multi-output switched-capacitor voltage regulator via Cadence circuit simulator that imitates the real hardware (Belakaria et al., 2020d). This circuit relies on a dynamic frequency switching clock. Each candidate circuit design is defined by 33 input variables (d=33). The first 24 variables are the width, length, and unit of the eight capacitors of the circuit $W_i, L_i, M_i \ \forall i \in 1 \cdots 8$. The remaining input variables are four output voltage references $V_{ref_i} \ \forall i \in 1 \cdots 4$ and four resistances $R_i \ \forall i \in 1 \cdots 4$ and a switching frequency f. We optimize nine objectives: maximize efficiency Eff, maximize four output voltages $V_{o_1} \cdots V_{o_4}$, and minimize four output ripples $OR_1 \cdots OR_4$. Our problem has a total of nine constraints. Since some of the constraints have upper bounds and lower bounds, they are defined in the problem by 15 different constraints:

$$C_0 : Cp_{total} \simeq 20nF \ with \ Cp_{total} = \sum_{i=1}^{8}(1.955W_iL_i + 0.54(W_i + L_i))M_i$$

$$C_1 \ to \ C_4 : V_{o_i} \geq V_{ref_i} \ \forall \in 1 \cdots 4$$
$$C_5 \ to \ C_8 \ : \ OR_{lb} \leq OR_i \leq OR_{ub} \ \forall i \ \in 1 \cdots 4$$
$$C_9 : Eff \leq 100\%$$

where OR_{lb} and OR_{ub} are the predefined lower-bound and upper-bound of OR_i respectively. Cp_{total} is the total capacitance of the circuit.

8.2.2 RESULTS AND DISCUSSION

We evaluate the performance of our algorithm and the baselines using the Pareto hypervolume (PHV) metric. PHV is a commonly employed metric to measure the quality of a given Pareto front (Zitzler, 1999). Figure 6 shows that MESMOC outperforms existing baselines. It recovers a better Pareto front with a significant gain in the number of function evaluations. Both of these experiments are motivated by real-wold engineering applications where further analysis of the designs in the Pareto front is crucial.

Electrified Aviation Power System Design. In this setting, the input space is discrete with 250,000 combinations of design parameters. Out of the entire design space, only 9% of design combinations passed all the constraints and only five points are in the optimal Pareto front. From a domain expert perspective, satisfying all the constraints is critical. Hence, the results reported for the hypervolume include only points that satisfy all the constraints. Despite the hardness of the problem, 90% (180 out of 200 inputs) of the designs selected by MESMOC satisfy all the constraints while for MESMOC+, PESMOC, MOEAD, and

NSGA-II, this was 49% (98 out of 200), 1.5% (3 out of 200 inputs), 9.5% (19 out of 200 inputs), and 7.5% (15 out of 200 inputs) respectively. MESMOC was not able to recover all the five points from the optimal Pareto front. However, it was able to closely approximate the optimal Pareto front and recover better designs than the baselines.

Analog Circuit Design Optimization. In this setting, the input space is continuous, consequently there is an infinite number of candidate designs. From a domain expert perspective, satisfying all the constraints is not critical and is impossible to achieve. The main goal is to satisfy most of the constraints (and getting close to satisfying the threshold for violated constraints) while reaching the best possible objective values. Therefore, the results reported for the hypervolume include all the evaluated points. In this experiment, the efficiency of circuit is the most important objective function. The table in Figure 2 shows the optimized circuit parameters from different algorithms.
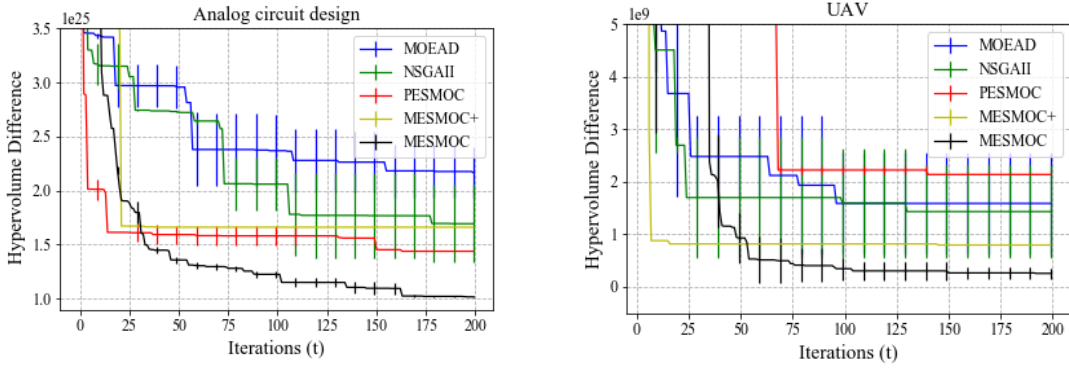

Figure 6: Results of different constrained multi-objective algorithms including MESMOC. The PHV metric is shown as a function of the number function evaluations.

SPECS	NSGA-II		PESMOC		MESMOC	
V_{ref1}(V)	0.6	0.5	0.52	0.53	0.63	0.52
V_{ref2}(V)	0.55	0.62	0.55	0.61	0.51	0.53
V_{ref3}(V)	1.06	1.06	1.07	1.12	1.05	1.13
V_{ref4}(V)	1.07	1.09	1.09	1.06	1.05	1.06
V_{o1}(mV)	699.6	713.1	677.10	760.60	**678.40**	**551.62**
V_{o2} (mV)	700.4	712.2	690.70	725.70	**520.61**	**632.80**
V_{o3} (V)	1.10	1.06	*1.08*	1.15	**1.12**	**1.16**
V_{o4} (V)	1.09	1.09	*1.08*	0.99	**1.14**	**1.08**
Eff (%)	73.26	71.85	**76.20**	**74.82**	*88.81*	*88.53*

Figure 7: Comparison table of optimized circuit parameters obtained from different algorithms (designs are selected from the Pareto set prioritized by efficiency)

All algorithms can generate design parameters for the circuit that meets the voltage reference requirements. The optimized circuit using MESMOC can achieve the highest conversion efficiency of 88.81% (12.61% improvement when compared with PESMOC with fixed frequency optimization and 17.86% improvement when compared with NSGA-II) with similar output ripples. The circuit with optimized parameters can generate the target output voltages within the range of 0.52V to 0.76V (1/3x ratio) and 0.99V to 1.17V (2/3x ratio) under the loads varying from 14 Ohms to 1697 Ohms.

9. Summary and Future Work

We introduced a novel and general framework for solving multi-objective (MO) Bayesian optimization problems based on the principle of output space entropy (OSE) search. The key idea is to select the sequence of experiments that maximize the information gained per unit cost about the optimal Pareto front. We instantiated this principle appropriately to solve a variety of MO problems from the most basic setting and its constrained version to the multi-fidelity and continuous-fidelity settings. Our comprehensive experimental results on both synthetic and real-world benchmarks showed that all our OSE based algorithms yield consistently better results than state-of-the-art methods, and are more efficient and robust than methods based on input space entropy search.

Future work includes extending this framework to handle high-dimensional BO problems (Oh, Gavves, & Welling, 2018) and combinatorial spaces, e.g., sets, sequences, and graphs (Doppa, 2021; Deshwal, Belakaria, Doppa, & Fern, 2020; Oh, Tomczak, Gavves, & Welling, 2019; Deshwal, Belakaria, & Doppa, 2021b, 2021a); and investigating important scientific applications including biological sequence design (Yang, Wu, & Arnold, 2019) and molecule design (Deshwal, Simon, & Doppa, 2021).

Appendix A. Full Derivation of iMOCA's Acquisition Function

Our goal is to derive a full approximation for iMOCA algorithm. In this appendix, we provide the technical details of the extended-skew Gaussian approximation (iMOCA-E) for the computation of the information gain per unit cost.

The information gain in equation (7.1) is defined as the expected reduction in entropy $H(.)$ of the posterior distribution $P(\mathcal{Y}^*|D)$ due to evaluating $\mathbf{x}$ at fidelity vector $\mathbf{z}$. Based on the symmetric property of information gain, we can rewrite it as shown below:

$$I(\{\mathbf{x}, \mathbf{y}, \mathbf{z}\}, \mathcal{Y}^*|D) = H(\mathbf{y}|D, \mathbf{x}, \mathbf{z}) - \mathbb{E}_{\mathcal{Y}^*}[H(\mathbf{y}|D, \mathbf{x}, \mathbf{z}, \mathcal{Y}^*)] \tag{A.1}$$

In equation (A.1), the first term is the entropy of a K-dimensional Gaussian distribution that can be computed in closed form as follows:

$$H(\mathbf{y}|D,\mathbf{x},\mathbf{z}) = \sum_{j=1}^{K} \ln(\sqrt{2\pi e}\; \sigma_{g_j}(\mathbf{x},z_j)) \tag{A.2}$$

The second term of equation (A.1) is an expectation over the Pareto front of the highest fidelities $\mathcal{Y}^*$. This term can be approximated using Monte-Carlo sampling:

$$\mathbb{E}_{\mathcal{Y}^*}[H(\mathbf{y}|D,\mathbf{x},\mathbf{z},\mathcal{Y}^*)] \simeq \frac{1}{S}\sum_{s=1}^{S}[H(\mathbf{y}|D,\mathbf{x},\mathbf{z},\mathcal{Y}_s^*)] \tag{A.3}$$

In the main paper, we showed that :

$$y_j \le f_s^{j*} \quad \forall j \in \{1,\cdots,K\} \tag{A.4}$$

By combining the inequality (A.4) and the fact that each function is modeled as an independent CF-GP, a common property of entropy measure allows us to decompose the entropy of a set of independent variables into a sum over entropies of individual variables (Cover & Thomas, 2012):

$$H(\mathbf{y}|D,\mathbf{x},\mathbf{z},\mathcal{Y}_s^*) \simeq \sum_{j=1}^{K} H(y_j|D,\mathbf{x},z_j,f_s^{j*}) \tag{A.5}$$

In what follows, we provide details of iMOCA-E approximation to compute $H(y_j|D,\mathbf{x},z_j,f_s^{j*})$.

The condition $y_j \le f_s^{j*}$, is originally expressed as $f_j \le f_s^{j*}$. Substituting this condition with it's original equivalent, the entropy becomes $H(y_j|D,\mathbf{x},z_j,f_j \le f_s^{j*})$. Since y_j is an evaluation of the function g_j and f_j is an evaluation of the function f_j, we make the observation that $y_j|f_j \le f_s^{j*}$ can be approximated by an extended-skew Gaussian (ESG) distribution (Azzalini, 1985). It had been shown that the differential entropy of an ESG does not have a closed-form expression (Arellano-Valle et al., 2013). Therefore, we derive a simplified expression where most of the terms are analytical by manipulating the components of the entropy as shown below.

In order to simplify the calculation $H(y_j|D,\mathbf{x},z_j,f_j \le f_s^{j*})$, we start by deriving an expression for its probability distribution. Based on the definition of the conditional distribution of a bi-variate normal, $f_j|y_j$ is normally distributed with mean $\mu_{f_j} + \frac{\sigma_{f_j}}{\sigma_{g_j}}\tau(y_j - \mu_{g_j})$ and variance $\sigma_{f_j}^2(1-\tau)^2$, where $\tau = \frac{\sigma_{g_j,f_j}^2}{\sigma_{g_j}\sigma_{f_j}}$ is the predictive correlation between y_j and f_j. We can now write the cumulative distribution function for $y_j|f_j \le f_s^{j*}$ as shown below:

$$P(y_j \le u|f_j \le f_s^{j*}) = \frac{P(y_j \le u, f_j \le f_s^{j*})}{P(f_j \le f_s^{j*})} = \frac{\int_{-\infty}^{u} \phi\left(\frac{\theta-\mu_{g_j}}{\sigma_{g_j}}\right)\Phi\left(\frac{f_s^{j*}-\mu_{f_j}-\frac{\sigma_{f_j}}{\sigma_{g_j}}\tau(\theta-\mu_{g_j})}{\sqrt{\sigma_{f_j}^2(1-\tau)^2}}\right)d\theta}{\sigma_{g_j}\Phi\left(\frac{f_s^{j*}-\mu_{f_j}}{\sigma_{f_j}}\right)}$$

Let us define the normalized variable $\Gamma_{f_s^{j*}}$ as $\Gamma_{f_s^{j*}} \sim \frac{y_j - \mu_{g_j}}{\gamma_{g_j}} | f_j \leq f_s^{j*}$. After differentiating with respect to u, we can express the probability density function for $\Gamma_{f_s^{j*}}$ as:

$$P(u) = \frac{\phi(u)}{\Phi(\gamma_s^{(f_j)})} \Phi(\frac{\gamma_s^{(f_j)} - \tau u}{\sqrt{1 - \tau^2}})$$

which is the density of an ESG with mean and variance defined as follows:

$$\mu_{\Gamma_{f_s^{j*}}} = \tau \frac{\phi(\gamma_s^{(f_j)})}{\Phi(\gamma_s^{(f_j)})}, \sigma_{\Gamma_{f_s^{j*}}} = 1 - \tau^2 \frac{\phi(\gamma_s^{(f_j)})}{\Phi(\gamma_s^{(f_j)})} \left[\gamma_s^{(f_j)} + \frac{\phi(\gamma_s^{(f_j)})}{\Phi(\gamma_s^{(f_j)})} \right] \tag{A.6}$$

Therefore, we can express the entropy of the ESG as shown below:

$$H(\Gamma_{f_s^{j*}}) = - \int P(u) \ln(P(u)) du \tag{A.7}$$

We also derive a more simplified expression of the iMOCA-E acquisition function based on ESG. For a fixed sample f_s^j*, $H(\Gamma_{f_s^{j*}})$ can be decomposed as follows:

$$H(\Gamma_{f_s^{j*}}) = \mathbb{E}_{u \sim \Gamma_{f_s^{j*}}} \left[- \ln(\phi(u)) + \ln(\Phi(\gamma_s^{(f_j)})) - \ln(\Phi(\frac{\gamma_s^{(f_j)} - \tau u}{\sqrt{1 - \tau^2}})) \right] \tag{A.8}$$

We expand the first term as shown below:

$$\mathbb{E}_{u \sim \Gamma_{f_s^{j*}}} [- \ln(\phi(u))] = \frac{1}{2} \ln(2\pi) + \frac{1}{2} \mathbb{E}_{u \sim \Gamma_{f_s^{j*}}} [u^2] \tag{A.9}$$

From the mean and variance of $\Gamma_{f_s^{j*}}$ in equation (A.6), we get:

$$\mathbb{E}_{u \sim \Gamma_{f_s^{j*}}} [u^2] = \mu_{\Gamma_{f_s^{j*}}}^2 + \sigma_{\Gamma_{f_s^{j*}}} = 1 - \tau^2 \frac{\phi(\gamma_s^{(f_j)}) \gamma_s^{(f_j)}}{\Phi(\gamma_s^{(f_j)})} \tag{A.10}$$

We note that the final entropy can be computed using the following expression.

$$H(y_j | D, \mathbf{x}, z_j, y_j \leq f_s^{j*}) = H(\Gamma_{f_s^{j*}}) + \ln(\sigma_{g_j}) \tag{A.11}$$

By combining equations (A.8) and (A.11), we get:

$$H(y_j | D, \mathbf{x}, z_j, f_j \leq f_s^{j*}) \simeq \ln(\sqrt{2\pi e}\, \sigma_{g_j}) + \ln(\Phi(\gamma_s^{(f_j)})) - \tau^2 \frac{\phi(\gamma_s^{(f_j)}) \gamma_s^{(f_j)}}{2\Phi(\gamma_s^{(f_j)})}$$

$$- \mathbb{E}_{u \sim \Gamma_{f_s^{j*}}} \left[\ln(\Phi(\frac{\gamma_s^{(f_j)} - \tau u}{\sqrt{1 - \tau^2}})) \right] \tag{A.12}$$

From equations (A.3), (A.2), and (A.12), the final expression of iMOCA-E can be expressed as follows:

$$\alpha_t(\mathbf{x}, \mathbf{z}, \mathcal{Y}^*) \simeq \frac{1}{\mathcal{C}(\mathbf{x}, \mathbf{z}) S} \sum_{j=1}^{K} \sum_{s=1}^{S} \tau^2 \frac{\gamma_s^{(f_j)} \phi(\gamma_s^{(f_j)})}{2\Phi(\gamma_s^{(f_j)})} - \ln(\Phi(\gamma_s^{(f_j)})) + \mathbb{E}_{u \sim \Gamma_{f_s^{j*}}} [\ln(\Phi(\frac{\gamma_s^{(f_j)} - \tau u}{\sqrt{1 - \tau^2}}))]$$

Since the differential entropy of an ESG cannot be computed analytically, we perform numerical integration via Simpson's rule using $\mu_{\Gamma_{f_s^{j*}}} \mp \gamma \sqrt{\sigma_{\Gamma_{f_s^{j*}}}}$ as the integral limits. In practice, we set γ to 5. Since this integral is over one-dimension variable, numerical integration can result in a tight approximation with small amount of computation.

Appendix B. Additional Experiments and Results

B.1 Description of Synthetic Benchmarks

In what follows, we provide complete details of the synthetic benchmarks employed in this paper. Since our algorithm is designed for maximization settings, we provide the benchmarks in their maximization form.

1) BRANIN, CURRIN EXPERIMENT

In this experiment, we construct a multi-objective problem using a combination of existing single-objective optimization benchmarks (Kandasamy et al., 2017). It has two functions with two dimensions ($K=2$ and $d=2$).

Branin Function: We use the following function where $\mathcal{C}(z) = 0.05 + z^{6.5}$

$$g(\mathbf{x}, z) = -\left(a(x_2 - b(z)x_1^2 + c(z)x_1 - r)^2 + s(1 - t(z))cos(x_1) + s\right)$$

where $a = 1$, $b(z) = 5.1/(4\pi^2) - 0.01(1-z)$, $c(z) = 5/\pi - 0.1(1-z)$, $r = 6$, $s = 10$ and $t(z) = 1/(8\pi) + 0.05(1-z)$.

Currin Exponential Function: We use $\mathcal{C}(z) = 0.1 + z^2$

$$g(\mathbf{x}, z) = -\left(1 - 0.1(1-z)\exp\left(\frac{-1}{2x_2}\right)\right)\left(\frac{2300x_1^3 + 1900x_1^2 + 2092x_1 + 60}{100x_1^3 + 500x_1^2 + 4x_1 + 20}\right).$$

2) ACKLEY, ROSEN, SPHERE EXPERIMENT

In this experiment, we construct a multi-objective problem using a combination of existing single-objective optimization benchmarks (Wu & Frazier, 2018). It has three functions with five dimensions ($K=3$ and $d=5$). For all functions, we employed $\mathcal{C}(z) = 0.05 + z^{6.5}$

Ackley Function

$$g(\mathbf{x}, z) = -\left(-20\exp\left[-0.2\sqrt{\frac{1}{d}\sum_{i=1}^{d}x_i^2}\right] - \exp\left[\frac{1}{d}\sum_{i=1}^{d}\cos(2\pi x_i)\right] + e + 20\right) - 0.01(1-z)$$

Rosenbrock Function:

$$g(\mathbf{x}, z) = -\sum_{i=1}^{d-1}\left[100\left(x_{i+1} - x_i^2 + 0.01(1-z)\right)^2 + (1-x_i)^2\right]$$

Sphere Function:

$$g(\mathbf{x}, z) = -\sum_{i=1}^{d}x_i^2 - 0.01(1-z)$$

3) DTLZ1 EXPERIMENT

In this experiment, we solve a problem from the general multi-objective optimization benchmarks (Habib et al., 2019). We have six functions with five dimensions ($K=6$ and

d=5) with a discrete fidelity setting. Each function has three fidelities in which z takes three values from $\{0.2, 0.6, 1\}$ with z^*=1. The cost of evaluating each fidelity function is $\mathcal{C}(z)$=$\{0.01, 0.1, 1\}$

$$g_j(\mathbf{x}, z) = f_j(\mathbf{x}) - e(\mathbf{x}, z)$$

$f_1(\mathbf{x}) = -(1 + r)0.5\Pi_{i=1}^5 x_i$
$f_j(\mathbf{x}) = -(1 + r)0.5(1 - x_{6-j+1})\Pi_{i=1}^{6-j} x_i$ with $j = 2 \ldots 5$
$f_6(\mathbf{x}) = -(1 + r)0.5(1 - x_1)$
$r = 100[d + \sum_{i=1}^d ((x_i - 0.5)^2) - cos(10\pi(x_i - 0.5))]$
$e(\mathbf{x}, z) = \sum_{i=1}^d \alpha(z)cos(10\pi\alpha(z)x_i + 0.5\pi\alpha(z) + \pi)$ with $\alpha(z) = 1 - z$

4) QV EXPERIMENT

In this experiment, we solve a problem from the general multi-objective optimization benchmarks (Shu et al., 2018). We have two functions with eight dimensions (K=2 and d=8) with a discrete fidelity setting.

Function 1 has only one fidelity which is the highest fidelity

$$f_1(\mathbf{x}) = -(\frac{1}{d}\sum_{i=1}^d (x_i^2 - 20\pi x_i + 10))^{\frac{1}{4}}$$

Function 2 has two fidelities with cost $\{0.1, 1\}$ respectively and the following expressions:

High fidelity: $f_2(\mathbf{x}, High) = -(\frac{1}{d}\sum_{i=1}^d ((x_i - 1.5)^2 - 20\pi(x_i - 1.5) + 10))^{\frac{1}{4}}$
Low fidelity: $f_2(\mathbf{x}, Low) = -(\frac{1}{d}((\sum_{i=1}^d (\alpha[\mathbf{i}](x_i - 1.5)^2 - 20\pi(x_i - 1.5) + 10))^{\frac{1}{4}}$
with α=$[0.9, 1.1, 0.9, 1.1, 0.9, 1.1, 0.9, 1.1]$

B.2 Additional Information About Experimental Setup

Experimental Setup For Our Proposed Algorithms:

- The hyper-parameters are estimated after every five function evaluations (BO iterations) for MESMO and MESMOC. For iMOCA and MF-OSEMO, the number of evaluations would be higher due to the low cost of lower fidelities. Therefore, the hyper-parameters are estimated every twenty iterations.

- During the computation of Pareto front samples, we solve a cheap MO optimization problem over sampled functions using NSGA-II. We use Platypus[10] library for the implementation. For NSGA-II, the most important parameter is the number of function calls. We experimented with several values. We noticed that increasing this number does not result in any performance improvement for our algorithms. Therefore, we fixed it to 1500 for all our experiments.

Parameters Used for NSAG-II and MOEAD as Constrained Baselines:

---

10. platypus.readthedocs.io/en/latest/getting-started.html#installing-platypus

- Since we allow only 200 evaluations for MESMOC and PESMOC, we also set the number of functions evaluations for NSGA-II and MOEAD to 200. We leave any other parameter to the default value provided by the Platypus library.

Computational Resources

- We performed all experiments on a machine with the following configuration: Intel i7-7700K CPU @ 4.20GHz with 8 cores and 32 GB memory.
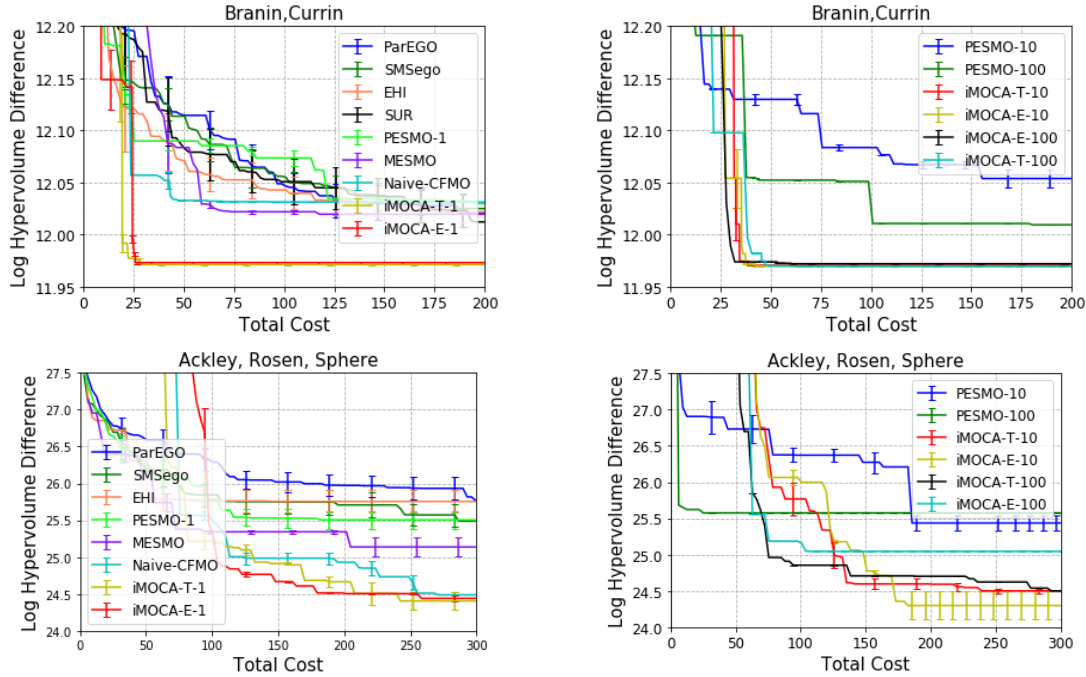
B.3 Additional Results

Figure 8: Results of synthetic benchmarks showing the effect of varying the number of Monte-Carlo samples for iMOCA, MESMO, and PESMO. The hypervolume difference is shown against the total resource cost of function evaluations.

References

Arellano-Valle, R. B., CONTRERAS-REYES, J. E., & Genton, M. G. (2013). Shannon entropy and mutual information for multivariate skew-elliptical distributions. *Scandinavian Journal of Statistics*, *40*(1).

Ariyarit, A., & et al. (2017). Multi-fidelity multi-objective efficient global optimization applied to airfoil design problems. *Applied Sciences*, *7*(12).

Azzalini, A. (1985). A class of distributions which includes the normal ones. *Scandinavian Journal of Statistics*.

Belakaria, S., Deshwal, A., & Doppa, J. R. (2019). Max-value entropy search for multi-objective Bayesian optimization. In *Conference on Neural Information Processing Systems*, pp. 7823–7833.

Belakaria, S., Deshwal, A., & Doppa, J. R. (2020a). Multi-fidelity multi-objective Bayesian optimization: An output space entropy search approach.. In *AAAI*, pp. 10035–10043.

Belakaria, S., Deshwal, A., Jayakodi, N. K., & Doppa, J. R. (2020b). Uncertainty-aware search framework for multi-objective Bayesian optimization. In *AAAI conference on artificial intelligence*.

Belakaria, S., Jackson, D., Cao, Y., Doppa, J. R., & Lu, X. (2020c). Machine learning enabled fast multi-objective optimization for electrified aviation power system design. In *IEEE Energy Conversion Congress and Exposition (ECCE)*.

Belakaria, S., Zhou, Z., Deshwal, A., Doppa, J. R., Pande, P., & Heo, D. (2020d). Design of multi-output switched-capacitor voltage regulator via machine learning. In *Proceedings of the Twenty-Third IEEE/ACM Design Automation and Test in Europe Conference (DATE)*.

Che, S., Boyer, M., Meng, J., Tarjan, D., Sheaffer, J. W., Lee, S., & et al. (2009). Rodinia: A benchmark suite for heterogeneous computing. In *2009 IEEE international symposium on workload characterization (IISWC)*.

Choi et al. (2018). On-chip communication network for efficient training of deep convolutional networks on heterogeneous manycore systems. *IEEE Transactions on Computers (TC)*, *67*(5), 672–686.

Cover, T. M., & Thomas, J. A. (2012). *Elements of information theory.* John Wiley and Sons.

Das et al. (2017). Design-space exploration and optimization of an energy-efficient and reliable 3D small-world network-on-chip. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, *36*(5), 719–732.

Deb, K., Pratap, A., Agarwal, S., Meyarivan, T., & Fast, A. (2002a). Nsga-ii. *IEEE Transactions on Evolutionary Computation*, *6*(2), 182–197.

Deb, K., Pratap, A., Agarwal, S., & Meyarivan, T. (2002b). A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE transactions on evolutionary computation*, *6*(2), 182–197.

Deshwal, A., Belakaria, S., & Doppa, J. R. (2021a). Bayesian optimization over hybrid spaces. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*, Vol. 139 of *Proceedings of Machine Learning Research*, pp. 2632–2643. PMLR.

Deshwal, A., Belakaria, S., & Doppa, J. R. (2021b). Mercer features for efficient combinatorial bayesian optimization. In *Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI)*, pp. 7210–7218. AAAI Press.

Deshwal, A., Belakaria, S., Doppa, J. R., & Fern, A. (2020). Optimizing discrete spaces via expensive evaluations: A learning to search framework. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence (AAAI)*, pp. 3773–3780. AAAI Press.

Deshwal, A., Simon, C., & Doppa, J. R. (2021). Bayesian optimization of nanoporous materials. *ChemRxiv*.

Deshwal et al. (2019). MOOS: A multi-objective design space exploration and optimization framework for NoC enabled manycore systems. *ACM Transactions on Embedded Computing Systems (TECS)*, *18*(5s), 77:1–77:23.

Doppa, J. R. (2021). Adaptive experimental design for optimizing combinatorial structures. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 4940–4945.

Emmerich, M., & Klinkenberg, J.-w. (2008). The computation of the expected improvement in dominated hypervolume of pareto front approximations. *Technical Report, Leiden University, 34*.

Feliot, P., Bect, J., & Vazquez, E. (2017). A Bayesian approach to constrained single-and multi-objective optimization. *Journal of Global Optimization*, *67*(1-2), 97–133.

Fernández-Sánchez, D., Garrido-Merchán, E. C., & Hernández-Lobato, D. (2020). Max-value entropy search for multi-objective bayesian optimization with constraints. *arXiv preprint arXiv:2011.01150v1*.

Garrido-Merchán, E. C., & Hernández-Lobato, D. (2019). Predictive entropy search for multi-objective Bayesian optimization with constraints. *Neurocomputing, 361*, 50–68.

Habib, A., Singh, H. K., & et al. (2019). A multiple surrogate assisted multi/many-objective multi-fidelity evolutionary algorithm. *Information Sciences*.

Hasbun, J. E. (2012). *Classical mechanics with MATLAB applications*. Jones & Bartlett Publishers.

Hennig, P., & Schuler, C. J. (2012). Entropy search for information-efficient global optimization. *Journal of Machine Learning Research (JMLR)*, *13*(Jun), 1809–1837.

Hernández-Lobato, D., Hernandez-Lobato, J., Shah, A., & Adams, R. (2016). Predictive entropy search for multi-objective Bayesian optimization. In *Proceedings of International Conference on Machine Learning (ICML)*, pp. 1492–1501.

Hernández-Lobato, J. M., Hoffman, M. W., & Ghahramani, Z. (2014). Predictive entropy search for efficient global optimization of black-box functions. In *Advances in Neural Information Processing Systems*, pp. 918–926.

Hoffman, M. W., & Ghahramani, Z. (2015). Output-space predictive entropy search for flexible global optimization. In *NIPS workshop on Bayesian Optimization*.

Hong, W., & et al (2019). A dual-output step-down switched-capacitor voltage regulator with a flying capacitor crossing technique for enhanced power efficiency. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, *27*(12).

Huang, D., Allen, T. T., Notz, W. I., & Miller, R. A. (2006). Sequential kriging optimization using multiple-fidelity evaluations. *Structural and Multidisciplinary Optimization*.

Joardar et al. (2018). Learning-based application-agnostic 3D NoC design for heterogeneous manycore systems. *IEEE Transactions on Computers*, *68*(6), 852–866.

Jones, D. R., Perttunen, C. D., & Stuckman, B. E. (1993). Lipschitzian optimization without the lipschitz constant. *Journal of Optimization Theory and Applications*, *79*(1), 157–181.

Kandasamy, K., Dasarathy, G., Oliva, J. B., & et al (2016). Gaussian process bandit optimisation with multi-fidelity evaluations. In *Conference on Neural Information Processing Systems*.

Kandasamy, K., Dasarathy, G., Schneider, J., & Poczos, B. (2017). Multi-fidelity Bayesian optimisation with continuous approximations. *ICML*.

Kennedy, M. C., & O'Hagan, A. (2000). Predicting the output from a complex computer code when fast approximations are available. *Biometrika*.

Klein, A., Falkner, S., Bartels, S., Hennig, P., & Hutter, F. (2017). Fast Bayesian optimization of machine learning hyperparameters on large datasets. In *International Conference on Artificial Intelligence and Statistics*.

Knowles, J. (2006). Parego: a hybrid algorithm with on-line landscape approximation for expensive multiobjective optimization problems. *IEEE Transactions on Evolutionary Computation*, *10*(1), 50–66.

Kontogiannis, S. G., Demange, J., Kipouros, T., & et al. (2018). A comparison study of two multifidelity methods for aerodynamic optimization. In *56th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*.

Kotthoff, L., Thornton, C., Hoos, H. H., Hutter, F., & Leyton-Brown, K. (2017). Auto-weka 2.0: Automatic model selection and hyperparameter optimization in weka. *Journal of Machine Learning Research (JMLR)*, *18*(1), 826–830.

Lam, R., Allaire, D. L., & et al (2015). Multifidelity optimization using statistical surrogate modeling for non-hierarchical information sources. In *56th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*.

McLeod, M., Osborne, M. A., & Roberts, S. J. (2017). Practical Bayesian optimization for variable cost objectives. *arXiv preprint arXiv:1703.04335*.

Michalowicz, J. V., Nichols, J. M., & Bucholtz, F. (2013). *Handbook of differential entropy*. Chapman and Hall/CRC.

Moss, H. B., Leslie, D. S., & Rayson, P. (2020). Mumbo: Multi-task max-value Bayesian optimization. *The European Conference on Machine Learnin*.

Oh, C., Gavves, E., & Welling, M. (2018). BOCK : Bayesian optimization with cylindrical kernels. In Dy, J. G., & Krause, A. (Eds.), *Proceedings of the 35th International Conference on Machine Learning (ICML)*, Vol. 80 of *Proceedings of Machine Learning Research*, pp. 3865–3874. PMLR.

Oh, C., Tomczak, J., Gavves, E., & Welling, M. (2019). Combinatorial Bayesian Optimization using the Graph Cartesian Product. In *NeurIPS*.

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

Picheny, V., Ginsbourger, D., & et al. (2013a). Quantile-based optimization of noisy computer experiments with tunable precision. *Technometrics*.

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

Ponweiser, W., Wagner, T., Biermann, D., & Vincze, M. (2008). Multiobjective optimization on a limited budget of evaluations using model-assisted s-metric selection. In *International Conference on Parallel Problem Solving from Nature*, pp. 784–794. Springer.

Rahimi, A., & Recht, B. (2008). Random features for large-scale kernel machines. In *Advances in Neural Information Processing Systems*, pp. 1177–1184.

Shahriari, B., Swersky, K., Wang, Z., Adams, R. P., & De Freitas, N. (2016). Taking the human out of the loop: A review of Bayesian optimization. *Proceedings of the IEEE*, *104*(1), 148–175.

Shu, L., Jiang, P., Zhou, Q., Shao, X., Hu, J., & Meng, X. (2018). An on-line variable fidelity metamodel assisted multi-objective genetic algorithm for engineering design optimization. *Applied Soft Computing*, *66*.

Snoek, J., Larochelle, H., & Adams, R. P. (2012). Practical Bayesian optimization of machine learning algorithms. In *Advances in Neural Information Processing Systems*, pp. 2951–2959.

Song, J., Chen, Y., & Yue, Y. (2019). A general framework for multi-fidelity Bayesian optimization with Gaussian processes. *International Conference on Artificial Intelligence and Statistics*.

Srinivas, N., Krause, A., Kakade, S. M., & Seeger, M. (2009). Gaussian process optimization in the bandit setting: No regret and experimental design. *arXiv preprint arXiv:0912.3995*.

Surjanovic, S., & Bingham, D. (2020). Virtual library of simulation experiments: Test functions and datasets. Retrieved January 21, 2020, from `http://www.sfu.ca/~ssurjano`.

Swersky, K., Snoek, J., & Adams, R. P. (2013). Multi-task Bayesian optimization. In *Conference on Neural Information Processing Systems*.

Takeno, S., Fukuoka, H., Tsukada, Y., Koyama, T., Shiga, M., Takeuchi, I., & Karasuyama, M. (2019). Multi-fidelity Bayesian optimization with max-value entropy search. *arXiv:1901.08275*.

Wang, Z., & Jegelka, S. (2017). Max-value entropy search for efficient Bayesian optimization. In *Proceedings of International Conference on Machine Learning (ICML)*.

Wang, Z., Zhou, B., & Jegelka, S. (2016). Optimization as estimation with Gaussian processes in bandit settings. In *Proceedings of International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 1022–1031.

Williams, C. K., & Rasmussen, C. E. (2006). *Gaussian processes for machine learning*, Vol. 2. MIT Press.

Wu, J., & Frazier, P. I. (2018). Continuous-fidelity Bayesian optimization with knowledge gradient. *NIPS Workshop on Bayesian Optimization*.

Yang, K. K., Wu, Z., & Arnold, F. H. (2019). Machine-learning-guided directed evolution for protein engineering. *Nature methods*, *16*(8), 687–694.

Zhang, Q., & Li, H. (2007). Moea/d: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, *11*(6), 712–731.

Zhang, Y., Hoang, T. N., & et al (2017). Information-based multi-fidelity Bayesian optimization. In *Conference on Neural Information Processing Systems Workshop on Bayesian Optimization*.

Zhu, J., Wang, Y.-J., & Collette, M. (2014). A multi-objective variable-fidelity optimization method for genetic algorithms. *Engineering Optimization*, *46*(4), 521–542.

Zitzler, E. (1999). *Evolutionary algorithms for multiobjective optimization: Methods and applications*, Vol. 63. Ithaca: Shaker.

Zuluaga, M., Sergent, G., Krause, A., & Püschel, M. (2013). Active learning for multi-objective optimization. In *Proceedings of International Conference on Machine Learning (ICML)*, pp. 462–470.