Robust Bayesian Optimization for Flexibility Analysis of Expensive Simulation-based Models with Rigorous Uncertainty Bounds

Akshay Kudva, Wei-Ting Tang, Joel A. Paulson*

Department of Chemical and Biomolecular Engineering, The Ohio State University, Columbus OH, USA

Abstract

The performance of emerging biochemical systems rests on their potential to adapt to uncertainties quickly and accurately. Flexibility analysis is a quantitative framework for determining if a system can maintain safe and feasible operation despite uncertainty. Most available methods assume access to equation-oriented models, which can be difficult to obtain in practice. In this paper, we propose a sequential black-box flexibility analysis method, BoFlex, that overcomes this challenge by constructing probabilistic surrogate models over the joint space of uncertain and recourse variables. BoFlex is based on a special alternating confidence bound procedure, which we show finitely converges to a correct solution under mild assumptions on the unknown functions. We also establish a rigorous upper bound on the convergence rate in terms of the maximum information gain of the surrogate model. The advantages of BoFlex are demonstrated on several case studies including a heat exchanger network and a bubble column reactor.

Key words: Flexibility analysis, Robust Bayesian optimization, Black-box optimization, Gaussian process regression

1. Introduction

Uncertainty and other sources of variability are inevitable in real-world systems and can arise from many different factors such as noisy and incomplete data, unknown model parameters, external disturbances, and implementation errors, to name a few. Flexibility analysis provides a quantitative framework for systematically identifying if a particular design of interest is feasible over a range of uncertainty values given the potential for recourse (or feedback) from control inputs that represent degrees of freedom in the system. The mathematical formalism of flexibility analysis was originally introduced in the process systems engineering (PSE) community by [1]; a historical perspective describing the evolution of the concepts and models used for flexibility analysis is given in [2]. Since this early work, there have been several additional works focused on developing more efficient strategies for solving the underlying mathematical programming problem defining the flexibility conditions [3, 4] as well as extensions to new problem formulations such as stochastic [5, 6] and dynamic problems [7].

*Corresponding author

Email addresses: kudva.7@osu.edu (Akshay Kudva), tang.1856@osu.edu (Wei-Ting Tang), paulson.82@osu.edu (Joel A. Paulson)

As discussed in detail in [8], the concept of flexibility is closely related to ideas from robust optimization [9, 10] that have become more popular in recent years due to an increased number of theoretical developments and applications [11, 12, 13]. A major difference between flexibility analysis and standard robust optimization approaches, however, is how one treats the recourse (control input) decisions. Here, recourse refers to the ability to take corrective action by manipulating a control variable (such as flow rates or heat inputs) in response to the realization of a random event whose outcome is unknown at the design stage but accurately known at a later stage of operation. Specifically, flexibility analysis allows the control inputs to be optimally adjusted to the realization of the uncertain parameters, while traditional robust optimization (RO) neglects recourse, leading to overly conservative solutions. The concept of "adjustable robust optimization" (ARO) [14] partially addresses this gap by incorporating recourse in the form of parametrized "decision rules" that map the realization of the uncertainties to a control input. However, depending on the choice of the structure of the decision rule, one often introduces some degree of suboptimality compared to comprehensive form of the flexibility analysis problem. As such, since we are interested in the general flexibility analysis problem in this work, we cannot directly rely on advances in RO or ARO to tackle this problem.

Although there has been a significant amount of work on flexibility analysis, an important assumption in the vast majority of methods is that the mathematical structure of the model is known and can be efficiently exploited by global optimization techniques. For example, in the case that all functions defining feasibility are linearly related to the uncertain parameters and control inputs, one can exploit knowledge of the Karush–Kuhn–Tucker (KKT) optimality conditions to specify equations for the worst-case active constraints that depend on the choice of these variables. The KKT conditions can then be enforced as constraints when solving flexibility analysis problems for which a variety of reformulation techniques exist for linear and other simple convex functions [3]. However, such approaches critically rely on the assumptions that: (i) the KKT conditions are equivalent to the global optimality conditions and (ii) the terms appearing in the KKT conditions are known. It turns out that, in many engineering design problems, neither of these assumptions are valid since the model is inherently non-convex and the feasibility conditions are defined in terms of an expensive computer simulation whose internal structure is inaccessible to the modeler. One example is when we want to evaluate the flexibility of a system with significant spatial variability that is best characterized by a computational fluid dynamics (CFD) simulation [15]. Even though many mature CFD codes exist, it may be very difficult (or even impossible) to convert them to form that is compatible with existing flexibility analysis methods. These cases are often referred to as "simulation-based," "derivative-free," or "black-box" models for which the main assumption is that the model can be queried at desired input values within the range of interest.

Due to their generality, there has been a significant amount of interest in utilizing black-box models within optimization. In fact, several previous works have tackled a special case of the black-box flexibility analysis problem in which there are no recourse variables [16, 17, 18, 19, 20]. In the absence of recourse variables, flexibility analysis reduces to a "feasibility test" problem that requires one to identify the worst-case uncertain parameter and constraint combination, which can be formulated as a single-level optimization problem. This single-level problem can then be tackled using standard or custom derivative-free optimization (DFO) methods [21]. For example, [16], [17], and [19] use high-dimensional model representations, Kriging models, and cubic radial basis function models, respectively, to build a surrogate model of the expensive feasibility function that can then be used to actively design new input points. An alternative approach, pursued in [20], is to convert the feasibility problem into a sequence of univariate DFO models that are

then tackled by the established BOBYQA [22] method. Since these methods neglect recourse variables, they result in a conservative view of flexibility in real-world problems. Furthermore, all of the aforementioned methods are heuristic in nature in the general non-convex case, meaning they do not provide any form of guarantee of convergence or bounds on the rate of convergence.

In this paper, we propose a novel black-box flexibility analysis method for expensive models that directly accounts for the challenging tri-level "max-min-max" optimization problem structure of the flexibility test. The proposed method, BoFlex, effectively extends the Bayesian optimization (BO) framework [23, 24, 25] (which is an efficient surrogate-based DFO method) to such three-level max-min-max problems. Similar to standard BO, BoFlex uses Gaussian process (GP) regression [26] to construct non-parametric probabilistic surrogate models of the black-box functions directly from data. By combining the probabilistic model with an acquisition (or expected utility) function, BoFlex simultaneously selects uncertain parameters and recourse variables that are likely to improve the information we have about the solution to the flexibility test problem. A key source of novelty in BoFlex is the design of the acquisition function, which is defined in terms of alternating confidence bounds on the flexibility test parameter. We develop a new strategy for computing these confidence bounds through transformations of a multi-output GP model that avoid challenges with alternative approaches such as constraint aggregation. We show that, because of this structure, BoFlex enjoys strong theoretical guarantees regarding performance and convergence. In particular, we derive worst-case bounds on the number of iterations required for convergence to a correct answer that hold in the general non-convex case. We also introduce an extension of the BoFlex that directly quantifies the flexibility index, which represents the largest scaling factor for the uncertainties (around a nominal value) that ensures the system remains flexible [4]. Finally, we show that BoFlex works well in practice on a variety of test problems including heat exchanger networks (HENs) and a bubble column fermentation problem defined in terms of a genome-scale dynamic flux balance analysis (DFBA) simulator. Not only does BoFlex consistently provide accurate solutions in a limited number of expensive simulations, we perform ablation studies (that remove/modify certain components of the algorithm) to demonstrate that the unique components of BoFlex all contribute toward its success in practice.

The remainder of this paper is organized as follows. In Section 2, we provide a formal introduction to the flexibility test problem of interest in this work. We provide relevant background material on GPs and BO in Section 3. In Section 4, we introduce our BoFlex algorithm, analyze its theoretical properties, and discuss important practical implementation details. We then derive an extension of BoFlex that works for flexibility index problems in Section 5. We evaluate the performance of Boflex on several case studies in Section 6, and provide some concluding remarks in Section 7. Lastly, for readability purposes, the proofs of all theoretical results are provided in Section 8.

2. Problem Formulation

In this work, we are interested in the following flexibility test problem that can be formulated as a tri-level "max-min-max" optimization problem [1]

$$\chi = \max_{\boldsymbol{\theta}\in\Theta} \min_{\mathbf{z}\in\mathcal{Z}} \max_{j\in\mathcal{J}} f_j(\boldsymbol{\theta}, \mathbf{z}), \tag{1}$$

where $\boldsymbol{\theta} \in \Theta$ denotes the set of uncertain parameters restricted to some specified domain $\Theta \subset \mathbb{R}^{d_1}$, $\boldsymbol{z} \in \mathcal{Z}$ denotes the set of control (or recourse) variables that can be adjusted during operation restricted to some specified domain $\mathcal{Z} \subset \mathbb{R}^{d_2}$, and

$\{f_j\}_{j\in\mathcal{J}}$ are a set $q = |\mathcal{J}|$ functions that define inequality constraints of the form $f_j(\boldsymbol{\theta}, \mathbf{z}) \leq 0$, $f_j : \Theta \times \mathcal{Z} \to \mathbb{R}$, $j = 1, \ldots, q$ that must be satisfied for the system to achieve feasible operation. The scalar quantity χ thus represents a feasibility measure for a given system under recourse. A value of χ satisfying $\chi \leq 0$ implies that feasible operation can be attained over the full set of uncertain parameters, i.e., the flexibility test is passed. On the other hand, if $\chi > 0$, there is at least one $\theta \in \Theta$ value for which feasible operation cannot be achieved for any feasible control action $\mathbf{z} \in \mathcal{Z}$, i.e., the flexibility test is failed.

In the absence of control variables ($d_2 = 0$), the flexibility test (1) reduces to a standard feasibility problem, e.g., [17] for which the goal is to identify if constraints are satisfied for all uncertainty realizations. However, the presence of $\mathbf{z}$ fundamentally changes the behavior of (1) since we must identify the best control action $\mathbf{z}$ to compensate for any specific realization of $\boldsymbol{\theta}$. In fact, we can interpret (1) as a sequential three-player game with $\boldsymbol{\theta}$ representing Player 1, $\mathbf{z}$ representing Player 2 that can adapt to the decisions made by Player 1, and the constraint index j representing Player 3 that can adapt to the decisions made by Players 1 and 2.

There has been a substantial amount of previous work on solving flexibility problems of the form (1) when the functional form of the inequalities $\{f_j(\boldsymbol{\theta}, \mathbf{z})\}_{j\in\mathcal{J}}$ are known, e.g., [7, 27, 4, 2, 8].[1] Typically, these inequalities are defined implicitly in terms of equality and inequality constraints of the form

$$\mathbf{h}(\mathbf{x}, \boldsymbol{\theta}, \mathbf{z}) = \mathbf{0}, \quad \mathbf{g}(\mathbf{x}, \boldsymbol{\theta}, \mathbf{z}) \leq \mathbf{0}, \tag{2}$$

where $\mathbf{x}$ denote internal states variables of the system (e.g., temperatures, concentrations), $\mathbf{h}(\cdot)$ is a function representing equality constraints that uniquely define the state for fixed uncertainty and control input values (e.g., steady-state versions of the material and energy balances), and $\mathbf{g}(\cdot)$ is a function defining the important constraints directly in terms of the state (e.g., physical, safety, and quality constraints). Letting $\mathbf{x}(\boldsymbol{\theta}, \mathbf{z})$ denote the values of $\mathbf{x}$ that satisfy $\mathbf{h}(\mathbf{x}, \boldsymbol{\theta}, \mathbf{z}) = \mathbf{0}$, we can recover the definition of the inequality constraints in (1) by

$$[\mathbf{g}(\mathbf{x}(\boldsymbol{\theta}, \mathbf{z}), \boldsymbol{\theta}, \mathbf{z})]_j = f_j(\boldsymbol{\theta}, \mathbf{z}) \leq 0, \quad \forall j \in \mathcal{J}, \tag{3}$$

where $[\mathbf{g}(\cdot)]_j$ denotes the j^{th} output of $\mathbf{g}(\cdot)$. In many practical applications, however, it is difficult or impossible to obtain cheap equation-oriented "white-box" models whose structure can be exploited by existing flexibility test methods. Even in cases where part of the model is known, many engineering design problems involve complex (integrated and multi-scale) computer simulations, implying $\{f_j(\boldsymbol{\theta}, \mathbf{z})\}_{j\in\mathcal{J}}$ are time-consuming-to-evaluate "black-box" functions (i.e., we can only query zeroth-order information at specific input values).

The main goal of this paper is to develop a black-box flexibility analysis method for functions in which limited data can be collected. Specifically, we consider the *bandit feedback* setting in which the unknown functions $\{f_j\}_{j\in\mathcal{J}}$ can be sequentially queried at specific $(\boldsymbol{\theta}, \mathbf{z}) \in \Theta \times \mathcal{Z}$ values. Since we are interested in integrated simulators that take in $d = d_1 + d_2$ inputs and return q outputs, we require that the same query point is used for all functions at each iteration (though this assumption can be relaxed without major modifications). Since the querying process is assumed to be expensive, we would like to design an algorithm that can determine if $\chi \leq 0$ (test passed) or $\chi > 0$ (test failed) in as few iterations as possible. Furthermore,

[1] By "known," we mean that there is a representation of the function as a composition of known elementary functions, which can be equivalently represented as a directed acyclic graph [28]. This structure is crucial for computing derivatives and/or convex relaxations that are assumed available by traditional "white-box" optimization algorithms.

we would like to have some guarantees that the method converges and has minimal chance of returning an inaccurate result for the flexibility test. Before introducing our proposed method in Section 4, we first provide relevant background on Bayesian optimization, which is the core framework that we look to extend in this work.

Lastly, since the simulator of interest may involve random variables, we further assume that, whenever we evaluate the inequality constraints, we only receive a noisy estimate of these functions. That is, for each input $(\boldsymbol{\theta}, \mathbf{z})$, we obtain measurements $\hat{f}_j(\boldsymbol{\theta}, \mathbf{z}) = f_j(\boldsymbol{\theta}, \mathbf{z}) + \varepsilon_j$ where ε_j is a zero-mean, R-sub-Gaussian noise for all $j = 1, \ldots, q$. Although $\hat{f}_j(\boldsymbol{\theta}, \mathbf{z})$ is a random variable, we will use $\hat{f}_j(\boldsymbol{\theta}_t, \mathbf{z}_t)$ to denote the measurement (realization) obtained at a given iteration $t \in \mathbb{N}$. In general, the noise variables could be correlated but we do not consider this case in our theoretical analysis in Section 4.3.

3. Background

In this section, we provide a brief review of Gaussian processes (GPs) and robust Bayesian optimization, which is the foundation of our flexibility test Bayesian optimization algorithm in Section 4. The introduction to GPs and nominal Bayesian optimization are standard and follow from [26] and [25], respectively. Robust Bayesian optimization, on the other hand, is relatively new and so we will briefly summarize the method introduced in [29, 30].

3.1. Gaussian Processes (GPs)

All inequality constraint functions $\{f_j(\boldsymbol{\theta}, \mathbf{z})\}_{j \in \mathcal{J}}$ in Section 2 are unknown *a priori*, meaning they must be learned from observations. For notational simplicity, let $\mathbf{x} = (\boldsymbol{\theta}, \mathbf{z}) \in \mathbb{R}^d$ denote the concatenated vector of the uncertain parameters and control inputs with total input dimension $d = d_1 + d_2$. Also note that we will interchangeably denote $f(\boldsymbol{\theta}, \mathbf{z})$ as $f(\mathbf{x})$ for any function $f : \Theta \times \mathcal{Z} \to \mathbb{R}$ or $f : \mathcal{X} \to \mathbb{R}$ where $\mathcal{X} = \Theta \times \mathcal{Z}$. We will use GPs as a non-parametric model to approximate the unknown inequality functions over the common domain $\mathcal{X}$. Since traditional Bayesian optimization methods focus on a single function, we will initially present results for a single performance function $\phi(\mathbf{x})$ – we will keep this function general for now but one can imagine selecting $\phi(\mathbf{x}) = \max_{j \in \mathcal{J}} f_j(\mathbf{x})$ as the largest constraint value to connect the following exposition to (1). We extend this model to multiple functions in order to represent all inequalities simultaneously in Section 4.1.

GP surrogate models are one of the most popular choices for non-parametric regression in machine learning, where the goal is to find an approximation to the nonlinear map $\phi : \mathcal{X} \to \mathbb{R}$ from some input vector $\mathbf{x} \in \mathcal{X}$ to the function value $\phi(\mathbf{x})$. This learning is achieved by assuming the function values $\phi(\mathbf{x})$ at different inputs $\mathbf{x}$ are random variables and, further, that any finite subset of these random variables has a joint Gaussian distribution. In other words, GPs model the correlation between any set of evaluation points and thus generalizes the concept of distributions over vector spaces to distributions over function spaces [26].

A GP is fully specified by a prior mean function $\mu(\mathbf{x})$ and a prior covariance function $k(\mathbf{x}, \mathbf{x}')$ that defines the covariance of any two function values $\phi(\mathbf{x})$ and $\phi(\mathbf{x}')$ for any $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$. The covariance function is also commonly referred to as "the kernel" due to its relationship to kernel methods in machine learning. The prior mean function is mainly used to capture some expected trend in the data; we will assume $\mu(\mathbf{x}) = 0$ without loss of generality in this work. The zero mean assumption can be met in practice by normalizing the output data when defining the target function. The choice of kernel is problem-dependent, as it encodes information about the smoothness and rate of change of the unknown function. A review of commonly used kernels can be found in [26, Chapter 4]. The specific kernel used

in this paper is discussed in Section 4.4. It is worth noting that the presented algorithm can be applied for *any choice of kernel*, though some of the theoretical results hold under certain assumptions on the kernel.

Using the GP framework, we can make a prediction of the function value $\phi(\mathbf{x}_\star)$ for an arbitrary test input $\mathbf{x}_\star \in \mathcal{X}$ given a set of t past observations, $\{\hat{\phi}(\mathbf{x}_i)\}_{i=1}^t$, at specifically selected input values $\{\mathbf{x}_i\}_{i=1}^t$. The GP model assumes noisy observations of the true function, i.e., $\hat{\phi}(\mathbf{x}) = \phi(\mathbf{x}) + \varepsilon$ where $\varepsilon \sim \mathcal{N}(0, \lambda)$ with variance λ. It is important to note that this Gaussian noise model is used to simplify the design of the algorithm and does not have to match the true noise distribution for which mild assumptions are made in Section 4.3. In fact, we will consider the *agnostic* setting introduced in [31] that consists of a misspecified prior and noise model. Conditioned on these t observations, the posterior remains a GP with the following predictive posterior mean and variance

$$\mu_t(\mathbf{x}_\star) = \mathbf{k}_t^\top(\mathbf{x}_\star)(\mathbf{K}_t + \lambda \mathbf{I}_t)^{-1}\hat{\boldsymbol{\phi}}_t, \tag{4a}$$

$$\sigma_t^2(\mathbf{x}_\star) = k(\mathbf{x}_\star, \mathbf{x}_\star) - \mathbf{k}_t^\top(\mathbf{x}_\star)(\mathbf{K}_t + \lambda \mathbf{I}_t)^{-1}\mathbf{k}_t(\mathbf{x}_\star), \tag{4b}$$

where $\hat{\boldsymbol{\phi}}_t = [\hat{\phi}(\mathbf{x}_1), \ldots, \hat{\phi}(\mathbf{x}_t)]^\top$ is the vector of observed noisy function values, $\mathbf{K}_t \in \mathbb{R}^{n \times n}$ is the covariance matrix between the observation input data with entries $[\mathbf{K}_t]_{(i,j)} = k(\mathbf{x}_i, \mathbf{x}_j)$ for all $i, j \in \{1, \ldots, n\}$, $\mathbf{I}_t \in \mathbb{R}^{t \times t}$ denotes the identity matrix, and $\mathbf{k}_t(\mathbf{x}_\star) = [k(\mathbf{x}_\star, \mathbf{x}_1), \ldots, k(\mathbf{x}_\star, \mathbf{x}_t)]^\top$ is a vector of covariance values between the new input $\mathbf{x}_\star$ and the observed inputs $\{\mathbf{x}_i\}_{i=1}^t$.

3.2. Bayesian Optimization

Bayesian optimization (BO) aims to find the global maximum of an unknown function by sequentially querying it at particular inputs [25, 32, 24]. The underlying assumption is that the evaluation of the function is expensive (and thus the bottleneck) while computational resources are relatively cheap for designing the next query point. This fits our problem described in Section 2 where each evaluation of the inequality constraints corresponds to running a process simulator that takes a significant amount more CPU time than the BO design process (potentially by multiple orders of magnitude). In general, BO constructs a probabilistic surrogate model for the objective function that is used to identify informative sample locations. GPs are the most popular surrogate model due to their non-parametric nature and analytic expressions for the posterior mean and variance in (4). The main drawback of GPs is their $\mathcal{O}(t^3)$ scaling with respect to the number of data points (due to the matrix inversion in these expressions); however, this is typically not an issue in most BO applications wherein we are operating in a low data regime.

For example, the upper confidence bound (UCB) algorithm in [31] selects the next sample location as follows

$$\mathbf{x}_{t+1} \in \underset{\mathbf{x} \in \mathcal{X}}{\operatorname{argmax}} \; \mu_t(\mathbf{x}) + \beta_t^{1/2}\sigma_t(\mathbf{x}), \tag{5}$$

where β_t is an iteration-dependent positive scalar that reflects the size of the confidence interval of the GP model. We see that (5) suggests to evaluate the objective function at the location where the confidence interval is largest. It has been shown that by repeatedly evaluating the system at $\{\mathbf{x}_1, \mathbf{x}_2, \ldots\}$ suggested by (5) improves the estimate (decreases uncertainty) of the global maximum and thus provably converges under some assumptions [31]. In our setting, however, we cannot apply such a sampling scheme because it treats all input variables as "design variables" whereas in our case these variables are partitioned into adversarial players $(\boldsymbol{\theta}, \mathbf{z})$ that are "fighting against" each other.

3.3. Adversarially Robust Bayesian Optimization

Adversarially robust Bayesian optimization (ARBO) is a conceptually straightforward extension of BO to handle the adversarial nature of two player games of the form $\max_{\boldsymbol{\theta}\in\Theta} \min_{\mathbf{z}\in\mathcal{Z}} \phi(\boldsymbol{\theta}, \mathbf{z})$ [29, 30]. ARBO enables one to solve so-called robust optimization problems for which we have a set of "here-and-now" variables $\boldsymbol{\theta}$ and "wait-and-see" variables $\mathbf{z}$ that can be used to model a variety of different real-world problems involving design under uncertainty. The key difference in ARBO compared to UCB (5) is that we now need to sequentially solve two optimization problems defined in terms of the confidence bounds

$$\boldsymbol{\theta}_{t+1} \in \operatorname*{argmax}_{\boldsymbol{\theta}\in\Theta} \min_{\mathbf{z}\in\mathcal{Z}} \ \mu_t(\boldsymbol{\theta}, \mathbf{z}) + \beta_t^{1/2}\sigma_t(\boldsymbol{\theta}, \mathbf{z}), \tag{6a}$$

$$\mathbf{z}_{t+1} \in \operatorname*{argmin}_{\mathbf{z}\in\mathcal{Z}} \ \mu_t(\boldsymbol{\theta}_t, \mathbf{z}) - \beta_t^{1/2}\sigma_t(\boldsymbol{\theta}_t, \mathbf{z}). \tag{6b}$$

The core idea underpinning (6) is that we must alternate between upper confidence bounds for selecting the here-and-now variables and lower confidence bounds for selecting the wait-and-see variables. Intuitively, this structure follows the same reasoning as UCB (5) in that we need a relaxed view of the problem being solved. Since we have a minimization problem in the second stage, we need to use a lower confidence bound to ensure we have a valid lower bound on the minimum.

For sufficiently smooth functions, it has been shown in [29] that ARBO enjoys similar convergence properties to the original UCB algorithm described in Section 3.2. However, the flexibility test problem (1) is a tri-level optimization problem and so does not directly fit the setting of ARBO. In principle, we could apply ARBO by defining a new worst-case constraint function $\phi(\boldsymbol{\theta}, \mathbf{z}) = \max_{j\in\mathcal{J}} f_j(\boldsymbol{\theta}, \mathbf{z})$; however, the $\max_{j\in\mathcal{J}}$ operator is non-smooth, which makes ϕ difficult to accurately model with a GP. In fact, useful confidence bound results that have been established (that allow us to rigorously select the β_t parameter and will be discussed further in Section 4.3) do not hold in this case. This prevents us from implementing any sort of guaranteed stopping criteria during the sampling process, meaning such an algorithm would be heuristic at best. As proposed in [33], one way to overcome this challenge is to replace $\max_{j\in\mathcal{J}}$ with a smooth approximation such as the Kreisselmeier–Steinhsauser (KS) function that is defined as follows [34]

$$\mathrm{KS}(\boldsymbol{\theta}, \mathbf{z}; \rho) = M + \frac{1}{\rho} \ln \left[\sum_{j=1}^{q} \exp\left(\rho(f_j(\boldsymbol{\theta}, \mathbf{z}) - M)\right) \right], \tag{7}$$

where $\rho > 0$ is an "aggregation" parameter whose value controls the accuracy of the approximation and $M \approx \max_{j\in\mathcal{J}} f_j(\boldsymbol{\theta}, \mathbf{z})$ is a constant used to reduce overflow/underflow errors in the exponential function. Using established properties of the KS function [27], it can be directly used to bound the max operator from above and below as follows

$$\mathrm{KS}(\boldsymbol{\theta}, \mathbf{z}; \rho) - \ln(q)/\rho \leq \max_{j\in\mathcal{J}} f_j(\boldsymbol{\theta}, \mathbf{z}) \leq \mathrm{KS}(\boldsymbol{\theta}, \mathbf{z}; \rho), \tag{8}$$

Furthermore, $\mathrm{KS}(\boldsymbol{\theta}, \mathbf{z}; \rho) \to \max_{j\in\mathcal{J}} f_j(\boldsymbol{\theta}, \mathbf{z})$ as $\rho \to \infty$, such that in principle we could select ρ to be large enough to yield nearly 0 error in practice. However, the downside of this approach is that large ρ significantly increases the maximum rate of change in the function – therefore, even though we can technically model it with a GP similarly to (4), we need to develop specialized non-stationary kernels to construct reasonable confidence bounds. Another disadvantage of this approach is that it aggregates the information we have collected on the individual inequality

constraint functions such that we are not exploiting the full set of available information at every iteration. Therefore, in this paper, we develop a new method that avoids these complications by developing a multi-output GP model for the inequality constraints that is combined with a new acquisition function to design the next sample location $(\boldsymbol{\theta}_{t+1}, \mathbf{z}_{t+1})$, as discussed in the next section.

4. Bayesian Optimization for Flexibility Analysis (BoFlex)

In this section, we introduce the BoFlex algorithm for solving black-box flexibility test problems under a limited budget of function queries. We first describe a multi-output extension of GPs that is critical for overcoming the aforementioned challenges of ARBO applied to (1). We then summarize the BoFlex algorithm, which we show is capable of guaranteeing a correct answer is provided to the flexibility test problem within a finite number of steps with probability greater than some user specified level. As we show in this section, the theoretical guarantees will rely on the continuity of the underlying functions and so one can roughly expect the smoother the behavior of $f_j(\boldsymbol{\theta}, \mathbf{z})$ for all $j \in \mathcal{J}$ the quicker BoFlex will converge. We finally discuss some important implementation issues that can be helpful when applying BoFlex in practice.

4.1. GPs with Multiple Outputs

In general, we have some finite number of q simulator outputs that define the key inequality constraints that must be satisfied to consider the system "safe" (feasible). This implies that we need to consider multiple, possibly correlated functions when designing our algorithm, which differs from traditional BO. Here, we choose to use the representation from [35] that suggests the use of an equivalent surrogate function

$$h(\boldsymbol{\theta}, \mathbf{z}, j) = \begin{cases} f_1(\boldsymbol{\theta}, \mathbf{z}) & \text{if } j = 1 \\ \quad \vdots & \\ f_q(\boldsymbol{\theta}, \mathbf{z}) & \text{if } j = q \end{cases} \tag{9}$$

which simply returns the corresponding constraint function depending on the additional input $j \in \mathcal{J} = \{1, \ldots, q\}$. The function $h(\cdot, \cdot, \cdot)$ remains a single-output function such that it can still be modeled by a scalar GP over the extended parameter space $\Theta \times \mathcal{Z} \times \mathcal{J}$. A key advantage of this representation is that we have a significant amount of flexibility when it comes to specifying the kernel. For example, consider the case of $q = 2$, we may select the kernel to behave as follows

$$k((\mathbf{x}, j), (\mathbf{x}', j')) = \begin{cases} \delta_{jj'} k_1(\mathbf{x}, \mathbf{x}') + k_{12}(\mathbf{x}, \mathbf{x}') & \text{if } j = 0 \\ \delta_{jj'} k_2(\mathbf{x}, \mathbf{x}') + k_{12}(\mathbf{x}, \mathbf{x}') & \text{if } j = 1 \end{cases}$$

where $\delta_{jj'}$ is the Kronecker delta, k_1 and k_2 are kernels for functions $f_1(\mathbf{x})$ and $f_2(\mathbf{x})$, respectively, and k_{12} is an additional covariance function that models the similarities between the two function outputs. If we wanted to model the two functions independently, we could directly set $k_{12}(\mathbf{x}, \mathbf{x}') = 0$, which would simplify training and prediction (aka inference). Note that $k((\mathbf{x}, j), (\mathbf{x}', j'))$ is a valid kernel function as long as it satisfies symmetry and positive semi-definiteness [36]. These two properties are satisfied by the kernel choice shown above as long as k_1, k_2, and k_{12} are valid kernels since the set of kernel functions is closed under addition, multiplication, and the application of linear operators.

Therefore, by merely augmenting the training data with an additional input j, we can use the standard GP framework to predict the posterior mean and variance at any test point using (4) – we now just add the function index j to the input

Algorithm 1 BoFlex: Bayesian Optimization for Black-Box Flexibility Tests

Input: Uncertain parameter domain Θ;
Recourse variable domain $\mathcal{Z}$;
Kernel for GP prior $k((\boldsymbol{\theta}, \mathbf{z}, j), (\boldsymbol{\theta}', \mathbf{z}', j'))$;
Confidence interval parameters $\{\beta_t^{1/2}\}_{t \geq 0}$.

1: **for** $t = 0, 1, 2, \ldots$ **do**
2: $\quad \hat{\chi}_t^U \leftarrow \max_{\boldsymbol{\theta} \in \Theta} \min_{\mathbf{z} \in \mathcal{Z}} \max_{j \in \mathcal{J}} u_t(\boldsymbol{\theta}, \mathbf{z}, j)$
3: $\quad \hat{\chi}_t^L \leftarrow \max_{\boldsymbol{\theta} \in \Theta} \min_{\mathbf{z} \in \mathcal{Z}} \max_{j \in \mathcal{J}} l_t(\boldsymbol{\theta}, \mathbf{z}, j)$
4: $\quad$ **if** $\hat{\chi}_t^U \leq 0$ **then**
5: $\quad\quad$ Declare test passed (system is *flexible*) and stop.
6: $\quad$ **end if**
7: $\quad$ **if** $\hat{\chi}_t^L > 0$ **then**
8: $\quad\quad$ Declare test failed (system is *not flexible*) and stop.
9: $\quad$ **end if**
10: $\quad \boldsymbol{\theta}_{t+1} \leftarrow \operatorname{argmax}_{\boldsymbol{\theta} \in \Theta} \min_{\mathbf{z} \in \mathcal{Z}} \max_{j \in \mathcal{J}} u_t(\boldsymbol{\theta}, \mathbf{z}, j)$
11: $\quad \mathbf{z}_{t+1} \leftarrow \operatorname{argmin}_{\mathbf{z} \in \mathcal{Z}} \max_{j \in \mathcal{J}} l_t(\boldsymbol{\theta}_{t+1}, \mathbf{z}, j)$
12: $\quad$ Query noisy simulator: $\hat{f}_j(\boldsymbol{\theta}_{t+1}, \mathbf{z}_{t+1})$, $\forall j = 1, \ldots, q$
13: $\quad$ Update GP model with new data collected in previous step
14: **end for**

parameter set $\mathbf{x}$ for each observation. In this setting, we obtain q measurements at every iteration t; however, for simplicity of notation, we continue to write μ_t and σ_t to refer to the posterior mean and variance for $h(\boldsymbol{\theta}, \mathbf{z}, j)$ given all information at iteration t (implying there are tq measurements instead of just t as before).

4.2. The Algorithm

We are now in a position to state our proposed algorithm, which we refer to as BoFlex. To facilitate the description of BoFlex, we first define the upper and lower confidence bounds on $h(\boldsymbol{\theta}, \mathbf{z}, j)$ based on the posterior GP model as follows

$$u_t(\boldsymbol{\theta}, \mathbf{z}, j) = \mu_t(\boldsymbol{\theta}, \mathbf{z}, j) + \beta_t^{1/2} \sigma_t(\boldsymbol{\theta}, \mathbf{z}, j), \tag{10a}$$

$$l_t(\boldsymbol{\theta}, \mathbf{z}, j) = \mu_t(\boldsymbol{\theta}, \mathbf{z}, j) - \beta_t^{1/2} \sigma_t(\boldsymbol{\theta}, \mathbf{z}, j). \tag{10b}$$

The probability of the true functions $\{f_j(\boldsymbol{\theta}, \mathbf{z})\}_{j \in \mathcal{J}}$ lying in the confidence interval $[l_t(\boldsymbol{\theta}, \mathbf{z}, j), u_t(\boldsymbol{\theta}, \mathbf{z}, j)]$ clearly depends on the choice of β_t as well as our assumptions on these functions. We give a formal treatment of these choices in Lemma 1 in Section 4.3 wherein a sufficiently large β_t is derived to ensure these bounds hold for all parameter values and iterations with at least some user-specified probability. Given these confidence bounds, BoFlex operates as described in Algorithm 1. This algorithm is conceptually simple and only requires one to solve tri-level optimization problems in terms of the upper and lower confidence bounds, which are significantly cheaper to evaluate than the expensive black-box functions by assumption. Specifically, Lines 2 and 3 involve computing confidence bounds on the test parameter $\chi \in [\hat{\chi}_t^L, \hat{\chi}_t^U]$ (with probability dependent on the choice of β_t) that are used to determine if the test is passed or failed on Lines 4–9. If the confidence region has yet to cross zero, then we continue on to Lines 10 and 11 to sequentially compute the next query point $(\boldsymbol{\theta}_{t+1}, \mathbf{z}_{t+1})$ using the alternating confidence bound principle described previously. Finally, on Lines 12 and 13, we query the simulator at this designed point and update the GP with this newly collected data before repeating these steps until a definitive answer to the test is found.

A detailed illustration of how the BoFlex algorithm works on a simple test problem is given in Section 6.1 (Figure 2).

4.3. Theoretical Results

Before we can analyze the performance of BoFlex, we must make some assumptions regarding the inequality constraint functions and the noise sequence.

Assumption 1. *The function $h(\boldsymbol{\theta}, \mathbf{z}, j)$ has bounded norm in a reproducing kernel Hilbert space (RKHS), which includes functions of the form*

$$h(\boldsymbol{\theta}, \mathbf{z}, j) = \textstyle\sum_k \alpha_k k((\boldsymbol{\theta}, \mathbf{z}, j), (\boldsymbol{\theta}_k, \mathbf{z}_k, j_k)),$$

with $\alpha_k \in \mathbb{R}$ and representer points $(\boldsymbol{\theta}_k, \mathbf{z}_k, j_k) \in \Theta \times \mathcal{Z} \times \mathcal{J}$ [37].

The bounded norm property in Assumption 1 implies that the coefficients α_k decay sufficiently fast as k increases. In other words, these functions are well-behaved in the sense that they are regular with respect to the choice of kernel. Let $\mathcal{H}_k$ denote the RKHS for which the kernel k determines the roughness and size of the function space and the induced norm $\|h\|_k = \sqrt{\langle h, h \rangle}$ measures the complexity of a given function $h \in \mathcal{H}_k$ with respect to the kernel.

Assumption 2. *The kernel function satisfies $k((\boldsymbol{\theta}, \mathbf{z}, j), (\boldsymbol{\theta}, \mathbf{z}, j)) \leq 1$ for all inputs $(\boldsymbol{\theta}, \mathbf{z}, j) \in \Theta \times \mathcal{Z} \times \mathcal{J}$.*

Assumption 2 implies that we have bounded variance for all possible inputs and can be made without loss of generality, as any multiplicative scaling can be absorbed into the norm bound B.

Assumption 3. *Let $\{\mathbf{x}_t\}_{t=0}^{\infty}$ be some $\mathcal{X}$-valued discrete-time stochastic process that is predictable with respect to the filtration $\mathbb{F} = \{\mathcal{F}_t\}_{t=0}^{\infty}$, i.e., $\mathbf{x}_t$ is $\mathcal{F}_{t-1}$-measurable for all $t \geq 1$. The noise sequence $\{\varepsilon_t\}_{t=1}^{\infty}$ is an $\mathbb{R}$-valued stochastic process adapted to $\mathbb{F}$ such that ε_t conditioned on $\mathcal{F}_{t-1}$ is R-sub-Gaussian for some $R \geq 0$, i.e.,*

$$\mathbb{E}\left[e^{\gamma \varepsilon_t} \mid \mathcal{F}_{t-1}\right] \leq \exp\left(\frac{\gamma^2 R^2}{2}\right), \quad \forall \gamma \in \mathbb{R}, \quad \forall t \geq 0.$$

Assumption 3 is a relatively mild assumption on the noise that is common in the bandit optimization literature (e.g., [38, 39, 40, 41]) and includes, for example, noise models bounded in $[-R, R]$ as well as Gaussian noise.

The GP framework uses a statistical model that makes different assumptions about h and ε than those shown in Assumptions 1–3. Specifically, samples from the GP prior are rougher than RKHS functions and are not contained in $\mathcal{H}_k$ an the noise model used to derive (4) is different. However, it is known that GPs and RKHS functions are closely related (see, e.g., [42]) and it turns out that one can use the GP mean and variance to infer reliable confidence bounds on $h(\boldsymbol{\theta}, \mathbf{z}, j)$. The proofs of all results in this section are given in Section 8.

Lemma 1 (Based on [43])**.** *Assume that h has bounded RKHS norm $\|h\|_k \leq B$ and that the measurements are corrupted by R-sub-Gaussian noise (i.e., Assumptions 1, 2, and 3 hold). Then, for any $\delta \in (0, 1)$ and*

$$\beta_t^{1/2} = B + \frac{R}{\lambda^{1/2}} \sqrt{2I(\mathbf{y}_t; h) + 2\ln(1/\delta)}, \tag{11}$$

one has

$$\mathbb{P}\left[|\mu_t(\mathbf{x}, j) - h(\mathbf{x}, j)| \leq \beta_t^{1/2} \sigma_t(\mathbf{x}, j), \ \forall \mathbf{x} \in \mathcal{X}, \forall j \in \mathcal{J}, \forall t \geq 0\right] \geq 1 - \delta, \tag{12}$$

where $I(\mathbf{y}_t; h)$ denotes the mutual information between the GP prior for h and the tq measurements $\mathbf{y}_t$ of h for a specific noise realization sequence, i.e.,

$$\mathbf{y}_t = [\hat{h}(\mathbf{x}_1, \mathcal{J})^\top, \ldots, \hat{h}(\mathbf{x}_t, \mathcal{J})^\top]^\top \in \mathbb{R}^{tq},$$

where $\hat{h}(\mathbf{x}_i, \mathcal{J}) = [\hat{h}(\mathbf{x}_i, 1), \ldots, \hat{h}(\mathbf{x}_i, q)]^\top \in \mathbb{R}^q$ *is a shorthand notation for all the measurements collected at iteration* i*.*

It is interesting to note that, for GP models, the mutual information $I(\mathbf{y}_t; h)$ depends only on the inputs and not the corresponding measurement of the function. In particular, for a given set of measurements $\mathbf{y}_\mathcal{D}$ at inputs $\mathbf{a} = (\mathbf{x}, j) \in \mathcal{D} \subset \mathcal{X} \times \mathcal{J}$, the mutual information is given by

$$I(\mathbf{y}_\mathcal{D}; h) = \tfrac{1}{2} \ln \left(\det \left(\mathbf{I} + \lambda^{-1} \mathbf{K}_\mathcal{D} \right) \right), \tag{13}$$

where $\mathbf{K}_\mathcal{D}$ is the kernel matrix $[k(\mathbf{a}, \mathbf{a}')]_{\mathbf{a}, \mathbf{a}' \in \mathcal{D}}$. Intuitively, the mutual information measures how informative the set of $|\mathcal{D}|$ samples $\mathbf{y}_\mathcal{D}$ are for the function h. If the function values are independent of each other under the GP prior, they will provide large amounts of new information. On the other hand, if the measurements are taken close to each other as measured by the kernel, they are correlated under the GP prior and thus provide less information. Furthermore, the more prior information that we encode in the GP prior, the less information we are able to gain given the same number of samples. Notice that Lemma 1 makes no probabilistic assumption for h, meaning h could be "optimally bad" (in terms of inducing the worst-case prediction error for the mean function) as long as it has bounded RKHS norm.

The scaling factor β_t in Lemma 1 depends on the specific measurements of the function so it leads to instance-specific values. It is common to replace this with a worst-case upper bound by defining the worst-case mutual information that could be obtained by any algorithm with at most n measurements

$$\gamma_n = \max_{\mathcal{D} \subseteq \mathcal{X} \times \mathcal{J}, |\mathcal{D}| \leq n} I(\mathbf{y}_\mathcal{D}; h). \tag{14}$$

Intuitively, γ_n quantifies a best case scenario where we select the measurements in the most informative manner possible and can be directly interpreted as a measure of the complexity of the function class associated with the GP prior. Since $I(\mathbf{y}_t; h) \leq \gamma_{tq}$ by definition, we can replace $I(\mathbf{y}_t; h)$ with γ_{tq} in (11) and the results still hold. We will present the remaining theoretical results in terms of γ_{tq} to show worst-case bounds; however, note that the instance-specific bounds also hold and will be useful when it comes to practical implementation details discussed in Section 4.4.

Next, we derive a bound on the gap between the upper and lower bounds on the flexibility test parameter χ derived by BoFlex as a function of the number iterations.

Theorem 1. *Let the assumptions of Lemma 1 hold and fix the variables* $T \in \mathbb{N}$*,* $\epsilon > 0$*, and* $\delta \in (0, 1)$*. Further, suppose that* T *satisfies*

$$\frac{T}{\beta_T \gamma_{Tq}} \geq \frac{C_1}{\epsilon^2}, \tag{15}$$

where $C_1 = 8 / \ln(1 + \lambda^{-1})$ *and define the gap in the upper and lower bound on the flexibility test parameter* χ *at every iteration as follows*

$$G_{t+1} = \hat{\chi}_t^U - \hat{\chi}_t^L. \tag{16}$$

Then, running BoFlex in Algorithm 1 with the confidence bound parameter set to $\beta_t^{1/2} = B + R\lambda^{-1/2} \sqrt{2\gamma_{tq} + 2\ln(1/\delta)}$ *for* T *iterations achieves a minimum gap value of* $G_T^\star = \min_{t \in \{1, \ldots, T\}} G_t \leq \epsilon$ *with probability at least* $1 - \delta$*.*

Theorem 1 states that, given some reasonable assumptions on the function class, BoFlex is able to find upper and lower bound estimates that are at least ϵ-close to the true flexibility test parameter value since $\hat{\chi}_t^L \leq \chi \leq \hat{\chi}_t^U$ for all $t \geq 0$ by Lemma 1. Furthermore, as long as $\beta_T \gamma_{Tq}$ scales sublinearly with respect to T, i.e.,

$\lim_{T\to\infty} \frac{T}{\beta_T \gamma_{Tq}} = \infty$, then a large enough T always exists that satisfies (15) for any $\epsilon > 0$. It turns out that γ_{Tq} satisfies this property for many commonly-used kernels, as shown in [31]. We will discuss specific examples in more detail later in this section. We also note that the minimum gap $G_t^\star$ can be recursively calculated at every iteration of BoFlex by $G_t^\star = \min\{G_{t-1}^\star, G_t\}$. An alternative termination criteria that one can use is to monitor $G_t^\star$ and stop the algorithm once $G_t^\star \leq \tau$ for some small tolerance value $\tau > 0$, though this will likely require additional iterations to achieve a more accurate estimate of χ.

Not only are we interested in the gap between the upper and lower estimates of χ, we are also interested in the termination behavior of BoFlex. Specifically, under what conditions will Line 5 or 8 of Algorithm 1 be correctly triggered. We show in the following result that BoFlex is guaranteed to converge in a finite number of iterations with high probability to a correct result.

Theorem 2. *Let the assumptions of Lemma 1 hold, $|\chi| = \eta$ for some positive constant $\eta > 0$, β_t set as in Theorem 1, and $\lim_{T\to\infty} \gamma_{Tq}/\sqrt{T} \to 0$. Then, given a confidence level $\delta \in (0, 1)$, BoFlex will terminate to a correct answer to the flexibility test in a finite number of iterations with probability at least $1 - \delta$ (i.e., Line 5 or 8 of Algorithm 1 is correctly declared for some iteration $T < \infty$).*

Not only does Theorem 2 establish that BoFlex terminates in a finite number of steps for a reasonable class of functions, it also provides guarantees that it will always converge to a correct answer whenever the bounds established in Lemma 1 hold. The probability that these bounds hold are controlled by a single tuning parameter, δ, which represents the failure probability. It can be made as close to 0 as desired, though this will increase the size of the confidence bounds that can be expected to slow convergence. The proof of Theorem 2 further establishes a link between the magnitude $|\chi| = \eta > 0$ and the number of iterations required before BoFlex terminates. In particular, larger values of η are expected to require fewer iterations since it gets easier to validate or invalidate the test $\chi \leq 0$.

We can verify the required condition that $\lim_{T\to\infty} \frac{\gamma_{Tq}}{\sqrt{T}} = 0$ in Theorem 2 by analyzing known bounds on γ_{Tq}. Specifically, it has been shown that $\gamma_T = \mathcal{O}(p \ln T)$ for the linear kernel, $\gamma_T = \mathcal{O}((\ln T)^{p+1})$ for the squared exponential kernel, and $\gamma_T = \mathcal{O}(T^{\frac{p}{2\nu+p}} \ln^{\frac{2\nu}{2\nu+p}} T)$ for the Matern-ν kernel [31, 44] where $p = d + 1$ is the total number of inputs in the GP model. Note that these same scaling laws hold for γ_{Tq} since the factor q is a constant. As such, finite convergence can be guaranteed for the linear kernel, squared exponential kernel, and any Matern-ν kernel with smoothness parameter ν satisfying $\nu > p/2$. These are sufficient conditions on the worst-case behavior so it is possible that BoFlex still converges whenever other (less smooth) kernels are used. We can also make Theorem 1 more explicit by substituting these bounds into (15). For example, in the case of the squared exponential kernel, $G_T^\star \leq \mathcal{O}(\frac{(\ln T)^{p+1}}{\sqrt{T}})$ or equivalently $G_T^\star \leq \epsilon$ for $T = \mathcal{O}^\star(\frac{1}{\epsilon^2} \ln\left(\frac{1}{\epsilon}\right)^{2p})$ where $\mathcal{O}^\star(\cdot)$ is a variant of $\mathcal{O}(\cdot)$ that hides dimension-independent log factors.

4.4. Practical Implementation

In this section, we discuss some possible changes to Algorithm 1 that make it simpler to implement in practice at the expense of loosing some of the theoretical guarantees established in the previous section.

4.4.1. Choice of exploration constant

The parameter β_t that sets the size of the GP confidence interval in Theorems 1 and 2 may be too conservative when applied in practice. As shown in Lemma 1, the GP confidence bounds still hold when γ_{Tq} is replaced by the data-dependent

empirical mutual information gained so far, $I(\mathbf{y}_t; h)$. This quantity can also be easily computed from the kernel matrix evaluated at the past measurement locations (13). Furthermore, depending on the application of interest, one may consider setting β_t to a constant value, which roughly corresponds to bounding the failure probability per iteration (as opposed to over all iterations and all possible inputs). For example, [35] used $\beta_t^{1/2} = 2$ in their experiments, which yielded good results in practice.

Since guarantees are no longer provided under such choices of β_t, Algorithm 1 may prematurely converge. To mitigate potential errors in such cases, we propose to derive *a posteriori* bounds on χ after the learning process is completed. Specifically, we can use Lemma 1 to check if we have achieved an accurate enough estimate – this can always be done regardless of the choice of β_t used during the data acquisition process. If not, then we can continue to run BoFlex until the estimated gap $G_T^\star$ is below some threshold and then repeat the *a posteriori* check. Interested readers are referred to [45] for further details on these type of *a posteriori* bounds including for cases where the kernel choice is misspecified.

4.4.2. Kernel selection

The choice of kernel k in the GP model is a critical parameter in Algorithm 1. As the theoretical results in Section 4.3 show, the slower the growth of γ_{Tq} with respect to T, the faster we can expect BoFlex to converge. As discussed previously, this growth is directly tied to the underlying degree of correlation in the function. In this section, we review the type of kernel that we use throughout our numerical experiments as well as the kind of models that they can represent. More detailed discussions on kernel choices and properties can be found in, e.g., [26, 42, 46].

We focus exclusively on the Matern-ν kernel with $\nu = 3/2$, which can be expressed as follows

$$k(\mathbf{a}, \mathbf{a}') = \zeta^2 \left(1 + \sqrt{3} r(\mathbf{a}, \mathbf{a}')\right) \exp\left(-\sqrt{3} r(\mathbf{a}, \mathbf{a}')\right), \tag{17a}$$

$$r(\mathbf{a}, \mathbf{a}') = \sqrt{(\mathbf{a} - \mathbf{a}')^\top \mathbf{L}^{-2} (\mathbf{a} - \mathbf{a}')}, \tag{17b}$$

where $r(\mathbf{a}, \mathbf{a}')$ is a scaled Euclidean distance over the combined input $\mathbf{a} = (\boldsymbol{\theta}, \mathbf{z}, j) \in \mathcal{A} = \Theta \times \mathcal{Z} \times \mathcal{J} \subset \mathbb{R}^p$, $\mathbf{L} = \mathrm{diag}(\mathbf{l})$ is a diagonal scaling matrix composed of positive lengthscale values $\mathbf{l} \in \mathbb{R}^p_{++}$, and ζ^2 is a scaling factor for the output variance. The GP model, with the Matern-ν kernel, is then parametrized by three sets of hyperparameters of total dimension $p+2$: scalar measurement noise λ in (4), scalar prior variance ζ^2, and p lengthscales $\mathbf{l}$, all of which have intuitive interpretations. In particular, λ represents the noise in the observations that includes any source of randomness in the algorithm or experiment. The prior variance ζ^2 roughly represents the expected magnitude of the function value, i.e., $|h(\mathbf{a})| \leq \zeta$ with 68% probability based on the GP prior (recall we have assumed the prior has zero mean without loss of generality). Finally, the lengthscales $\mathbf{l}$ specify how quickly the covariance changes between neighboring values along specific dimensions. Small lengthscales imply the covariance decays faster as a function of the distance between the points, meaning the function can change must faster from one point to the next. This means that we can always select a more conservative kernel representation by reducing $\mathbf{l}$. This increases the leading constant in the expression for γ_{Tq} as a function of T such that, by Theorems 1 and 2, smaller lengthscales reduce the speed of convergence of BoFlex. This fits our intuition since a large $\mathbf{l}$ means the function is not expected to change much and thus can be learned with relatively small amounts of data.

The confidence bound result established in Lemma 1 relies on the assumption that we know the true hyperparameters of the kernel (or can at least conservatively estimate them), which is not always true in practice. Therefore, a common remedy

is to apply the maximum likelihood estimation (MLE) framework to estimate the hyperparameters using already acquired data; see [26, Section 2.7] for details. It is known, however, that the MLE approach can lead to "poor results," mainly because the GP estimates may violate Lemma 1 under adaptive updates [47]. Since this issue is not the focus of this work, in our examples in Section 6, we assume that we have a reasonable set of initial data to obtain good estimates using MLE. There have been several interesting contributions on how to practically deal with this mismatch, e.g., [41, 48, 49]. Any one of these methods could be incorporated into BoFlex, though their impact on the convergence results in Section 4.3 would need to be separately analyzed, which we plan to study in future work.

4.4.3. Optimizing the acquisition function

Algorithm 1 assumes that we can find exact solutions to the acquisition optimization subproblems on Lines 2, 3, 10, and 11. The max-min-max problems, however, can be challenging to solve in practice. When Θ and $\mathcal{Z}$ are discrete sets, we can solve these problems by evaluating $u_t(\boldsymbol{\theta}, \mathbf{z}, j)$ and $l_t(\boldsymbol{\theta}, \mathbf{z}, j)$ at all $\Theta \times \mathcal{Z} \times \mathcal{J}$, which consist of only a finite number of points in this case. Since these functions can be evaluated cheaply, the cost of this exhaustive evaluation is still expected to be small relative to the cost of evaluating the expensive function as long as the cardinality $|\Theta \times \mathcal{Z} \times \mathcal{J}|$ is manageable (we found that on the order of millions can be done quite quickly). For continuous $\Theta \times \mathcal{Z}$, an alternative approach is to find an approximate solution using a local robust optimization algorithm (that assume the objective function is known) such as [50] that can be repeated from multiple initial starting points. If a guaranteed global solution is desired, then one can formulate these problems as generalized semi-infinite programs (GSIPs), as shown in [51], though the cost of these methods can be quite high in practice.

5. Extending BoFlex to Flexibility Index Problems

The flexibility test problem (1) provides a yes or no answer to the question: Can the system flexibly operate over the given set of uncertain parameters Θ? In some cases, however, one is interested in deriving a quantitative measure of the degree of flexibility. The flexibility index F is one such measure that is defined as follows

$$F = \max_{\rho \geq 0} \rho \;\text{ subject to: }\; \chi(\rho) \leq 0, \tag{18}$$

where $\chi(\rho)$ is a slightly modified definition of the flexibility test measure given by

$$\chi(\rho) = \max_{\boldsymbol{\theta} \in \Theta(\rho)} \min_{\mathbf{z} \in \mathcal{Z}} \max_{j \in \mathcal{J}} f_j(\boldsymbol{\theta}, \mathbf{z}), \tag{19}$$

with $\Theta(\rho)$ representing a *variable* parameter set

$$\Theta(\rho) = \{\boldsymbol{\theta} \in \mathbb{R}^{d_1} : \boldsymbol{\theta}_N - \rho \Delta\boldsymbol{\theta} \leq \boldsymbol{\theta} \leq \boldsymbol{\theta}_N + \rho \Delta\boldsymbol{\theta}\}. \tag{20}$$

Here, $\boldsymbol{\theta}_N \in \mathbb{R}^{d_1}$ denotes the nominal parameter values for which the system is expected to maintain feasible operation, $\Delta\boldsymbol{\theta} \in \mathbb{R}^{d_1}_+$ is the expected deviations from the nominal parameter values, and $\rho \geq 0$ is a scaling factor that controls the size of the uncertain parameter set $\Theta(\rho)$. We can then interpret the flexibility index problem (18) as finding the largest possible ρ for which there exists a feasible control input $\mathbf{z} \in \mathcal{Z}$ to keep the system feasible for all possible $\boldsymbol{\theta} \in \Theta(\rho)$. Therefore, if we compare the F values for two different designs of the same system, we can say that the design that leads to the larger F value is "more flexible" (can handle a larger range of parameter variability) than the other design.

Algorithm 2 BoFlex-Index: Extension of BoFlex to Flexibility Index Problems

Input: Starting interval $[\rho_L, \rho_U]$;
Desired tolerance ϵ_{tol};
Initial dataset $\mathscr{D}_0$ composed of noisy simulator evaluations;
Uncertain parameter domain Θ;
Recourse variable domain $\mathcal{Z}$;
Kernel for GP prior $k((\boldsymbol{\theta}, \mathbf{z}, j), (\boldsymbol{\theta}', \mathbf{z}', j'))$;
Confidence interval parameters $\{\beta_t^{1/2}\}_{t\geq 0}$.

1: **for** $i = 0, 1, 2, \ldots$ **do**
2: Calculate the midpoint $\rho_M = \frac{\rho_L + \rho_U}{2}$
3: Run BoFlex($\Theta(\rho_M), \mathcal{Z}, k, \{\beta_t^{1/2}\}_{t\geq 0}, \mathscr{D}_i$) in Algorithm 1; return result of the flexibility test $\mathcal{T}_i$, current bound gap $G_i(\rho_M)$, and complete dataset $\mathscr{D}_{i+1}$.
4: **if** $\mathcal{T}_i$ is true **then**
5: Test passed; update lower bound $\rho_L \leftarrow \rho_M$.
6: **else**
7: Test failed; update upper bound $\rho_U \leftarrow \rho_M$.
8: **end if**
9: **if** $|G_i(\rho_M)| < \epsilon_{\text{tol}}$ **then**
10: Stop and return ρ_L as the best feasible estimate for F.
11: **end if**
12: **end for**

Clearly, $\chi(\rho)$ is a monotonically non-decreasing function of ρ. In the common case that we define $\Delta\boldsymbol{\theta} = \Delta\theta[1, \ldots, 1]^\top$ for some positive scalar $\Delta\theta > 0$ (such that the uncertainty set is a hypercube), then we strengthen this result to say that $\chi(\rho)$ is a strictly increasing function of ρ such that F can be equivalently defined as the unique solution to $\chi(F) = 0$ (assuming a solution to this problem exists, which can be guaranteed as long as the system is feasible for $\boldsymbol{\theta}_N$). This is a scalar root finding problem that can be tackled with a variety of algorithms such as bisection [52]. Given a starting interval $[\rho_L, \rho_U]$ that contains the root, we define the BoFlex-Index method summarized in Algorithm 2, which is effectively a combination of BoFlex with the bisection method to tackle the flexibility index problem. Since BoFlex-Index involves solving a sequence of flexibility test problems, a key observation is that we can reuse the collected data needed to construct the multi-output GP model every time that BoFlex is called (Line 3). Thus, we can expect the quality of the bounds on χ to continually improve as the iterations increase, suggesting fewer internal BoFlex iterations will be required as the algorithm progresses.

A detailed illustration of the BoFlex-Index algorithm is shown for a test problem described in Section 6.1 (Figure 3).

6. Case Studies

In this section, we demonstrate BoFlex (Algorithm 1) on four problems using the practical implementation modifications described in Section 4.4. The first problem is a test problem that allows us to illustrate the key aspects of BoFlex and BoFlex-Index. The next two problems are based on heat exchanger network (HEN) systems from [3], which are common benchmark problems in the flexibility analysis literature. The fourth problem involves a realistic simulation-based model of a bubble column reactor that, to the best of our knowledge, has not been solved previously.

A Python implementation of BoFlex that builds upon the `BoTorch` package [53] is openly available on Github [54].

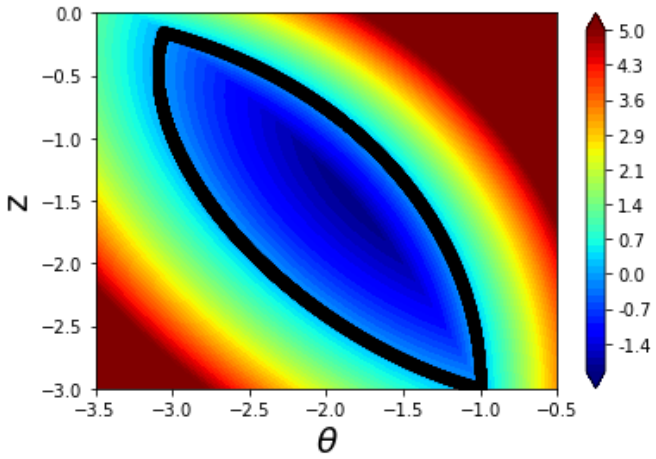

Figure 1: Contour plot of the worst-case constraint value for the illustrative test problem described in (21) and (22). The black line represents a worst-case constraint value of exactly 0.

6.1. Example 1: Illustrative Test Problem

To illustrate the behavior of the proposed BoFlex (Algorithm 1) and BoFlex-Index (Algorithm 2) methods presented in Sections 4 and 5, respectively, we first consider the following test problem consisting of two constraints $q = 2$, one uncertain parameter $d_1 = 1$, and one control input $d_2 = 1$:

$$f_1(\theta, z) = (\theta + 4)^2 + (z + 3)^2 - 9, \tag{21a}$$
$$f_2(\theta, z) = (\theta + 2)^2 + z^2 + \theta z - 5. \tag{21b}$$

The range of the uncertain parameter and control input variables are given by

$$\Theta = \{\theta \in \mathbb{R} : -3.5 \leq \theta \leq -0.5\}, \tag{22a}$$
$$\mathcal{Z} = \{z \in \mathbb{R} : -3 \leq z \leq 0\}. \tag{22b}$$

A contour plot of the value of the worst-case constraint $\phi(\theta, z) = \max_{j\in\{1,2\}} f_j(\theta, z)$ over these ranges is shown in Figure 1. From this plot, we can see that there is no feasible z value that ensures the inequality is less than zero for $\theta = -0.5$, implying the system is inflexible.

An illustration of BoFlex is shown in Figure 2. We assume BoFlex is only provided with 2 initial data points and progressively selects new points jointly in $(\boldsymbol{\theta}, \mathbf{z})$ space to improve the quality of the upper and lower confidence bounds on χ. The bottom row shows the projection of the worst-case inequality constraints onto the $\mathbf{z}$ space by calculating $\psi(\boldsymbol{\theta}) = \min_{\mathbf{z}\in\mathcal{Z}} \max_{j\in\mathcal{J}} f_j(\boldsymbol{\theta}, \mathbf{z})$, i.e., $\chi = \max_{\boldsymbol{\theta}\in\Theta} \psi(\boldsymbol{\theta})$. The confidence region calculated from the confidence bounds in (10) (shown in light purple) get progressively tighter as BoFlex progresses. Eventually, the algorithm is able to verify that the $0 < \chi$ by establishing that $0 < \hat{\chi}_t^L < \chi < \hat{\chi}_t^U$ at iteration 6.

We further illustration the BoFlex-Index algorithm in Figure 3 using a nominal value of $\theta_N = -2$ and a perturbation of $\Delta\theta = 0.5$. The true flexibility index can then be found by solving (18), which yields $F = 2$ for this problem. At every outer iteration of BoFlex-Index, we see BoFlex can efficiently certify if a specific ρ value leads to a flexible or inflexible process. By repeatedly calling BoFlex, one can hone in on the true flexibility index value F by checking if a particular candidate value is too large or too small. For example, given a starting midpoint value of $\rho = 2.75$, we see that the flexibility test fails such that we know $F < 2.75$. Running the flexibility test given the new midpoint value $\rho = 1.375$, we now see that the flexibility test is

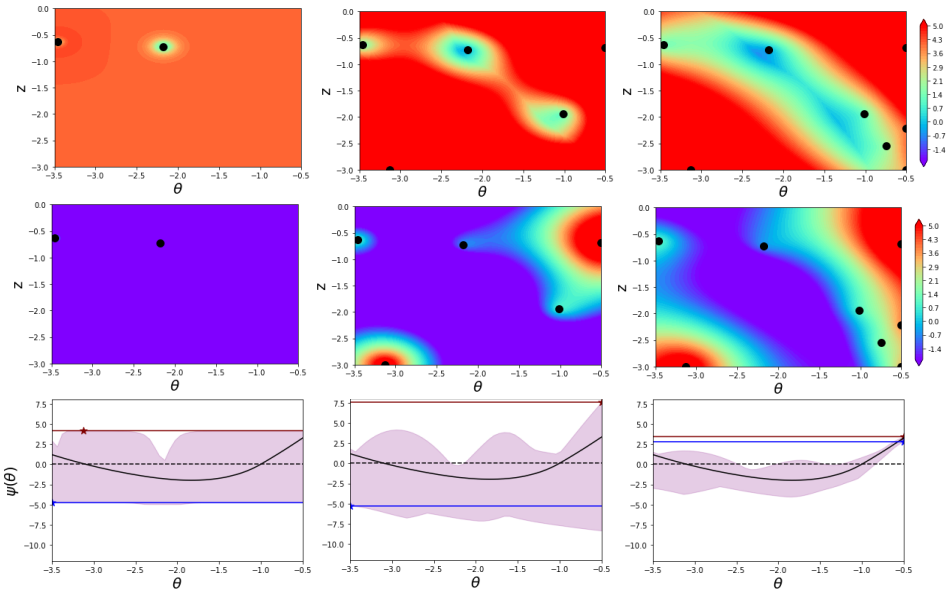

Figure 2: Optimization with BoFlex on test problem at iterations 0, 3, and 6 in the first, second, and third columns, respectively. The top row corresponds to the contours of the worst-case upper confidence bound $\max_{j\in\mathcal{J}} u_t(\boldsymbol{\theta}, \mathbf{z}, j)$ with evaluation points shown with black dots. The middle row corresponds to the contours of the worst-case lower confidence bound $\max_{j\in\mathcal{J}} l_t(\boldsymbol{\theta}, \mathbf{z}, j)$. The bottom row corresponds to the projected functions onto z, i.e., $\psi(\boldsymbol{\theta}) = \min_{\mathbf{z}\in\mathcal{Z}} \max_{j\in\mathcal{J}} f_j(\boldsymbol{\theta}, \mathbf{z})$. The shaded purple region represents the confidence region predicted by the upper and lower confidence bounds while the solid black line represents the true $\psi(\boldsymbol{\theta})$ function. The dotted black line corresponds to the 0 point, which is used to decide if the flexibility test is passed or failed. We see that BoFlex decides to jointly sample design and recourse points that provide more information about χ. After only 6 iterations, it is able to ensure that the containing interval $[\hat{\chi}_t^L, \hat{\chi}_t^U]$ is fully above 0 such that the system is correctly classified as inflexible.

passed such that we know $F \in [1.375, 2.75]$. We continue this process until a desired tolerance is met either for $\rho_U - \rho_L$ or for the gap between the upper and lower bounds for χ. After 5 iterations, we have identified $F \in [1.71, 2.06]$, which does contain the true value of $F = 2$ and can be further refined by running additional iterations.

6.2. Example 2: Small Heat Exchanger Network

We next consider [3, Example 3], which is a HEN problem illustrated in Figure 4. The goal of this system is to cool the hot stream H1 to at least 323 K despite uncertainty in the heat capacity θ of stream 1 given we can manipulate the heat load in an upstream cooler z. The range of the heat capacity is given by $\Theta = \{\theta : 0.55 \leq \theta \leq 1.05\}$ kW/K and the range of possible cooler heat duties is given by $\mathcal{Z} = \{z : 1 \leq z \leq 99\}$ kW. After eliminating intermediate state variables, we can write out the required conditions that must be satisfied by this HEN in terms of the following four inequalities

$$f_1(\theta, z) = -25 + z\left[\frac{1}{\theta} - 0.5\right] + \frac{10}{\theta} \leq 0, \tag{23a}$$

$$f_2(\theta, z) = -190 + \frac{10}{\theta} + \frac{z}{\theta} \leq 0, \tag{23b}$$

$$f_3(\theta, z) = -270 + \frac{250}{\theta} + \frac{z}{\theta} \leq 0, \tag{23c}$$

$$f_4(\theta, z) = 260 - \frac{250}{\theta} - \frac{z}{\theta} \leq 0. \tag{23d}$$

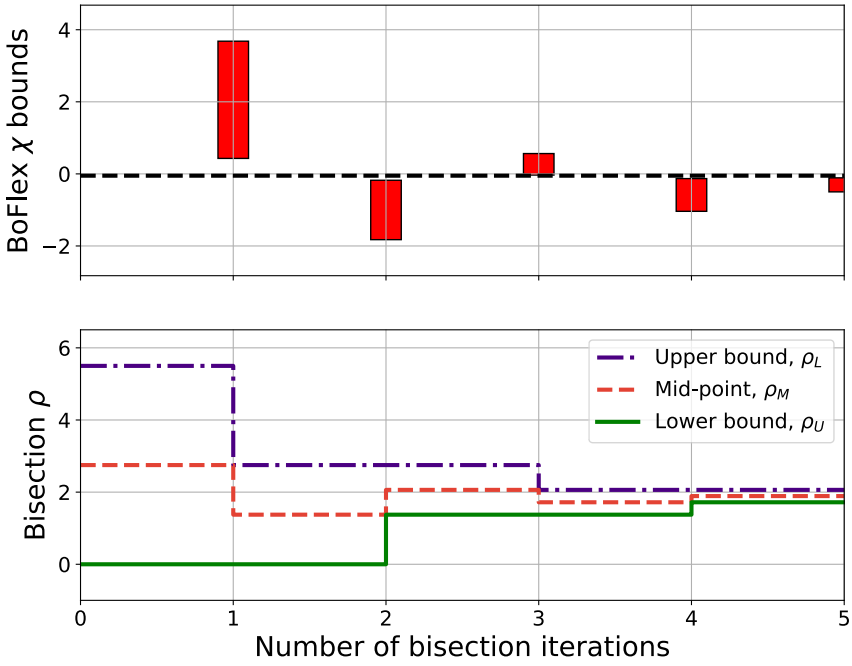

Figure 3: Illustration of the BoFlex-Index algorithm on a test problem. The top plot shows the range for the flexibility test parameter $\chi \in [\hat{\chi}_i^L, \hat{\chi}_i^U]$ estimated with BoFlex in Line 3 of Algorithm 2 for every bisection iteration i. The bottom plot shows the upper, lower, and midpoint value for bisection parameter ρ at every iteration. As the number of iterations increases, we see that the bounds converge toward the true flexibility index value.

Under the aforementioned constraints on θ and z, this system can be verified to be flexible, as shown in Figure 5. This exercise is straightforward when we assume that $f_1, \ldots, f_4$ are known, however, it becomes much more difficult when we do not have access to (23) but instead can only evaluate these functions at specific (θ, z) values. Since BoFlex is designed for such cases, we will assume that these functions are completely unknown for testing purposes.

To demonstrate the importance of each component of BoFlex, we perform a type of ablation study wherein we modify certain parts of the algorithm to see its impact on the overall performance. We consider the following three modifications to the search process:

- **Random Sampling:** In this case, the pair $(\boldsymbol{\theta}_{t+1}, \mathbf{z}_{t+1})$ are selected by uniform random sampling from $\Theta \times \mathcal{Z}$. We can interpret this case as neglecting information from past samples to make our current selection.

- **No Alternating Bounds:** In this case, we use the upper confidence bound to select both the uncertain parameters and the recourse variables. This amounts to replacing Line 11 of Algorithm 1 with the following

$$\mathbf{z}_{t+1} \in \operatorname*{argmin}_{\mathbf{z} \in \mathcal{Z}} \max_{j \in \mathcal{J}} u_t(\boldsymbol{\theta}_{t+1}, \mathbf{z}, j).$$

  This case effectively neglects the fact that we are solving problems with competing objectives (max versus min) and so is less likely to explore potentially promising values for the recourse variables.

- **Constraint Aggregation:** This case corresponds to a previous algorithm that we developed based on KS aggregation (7) in [33]. The idea here is to

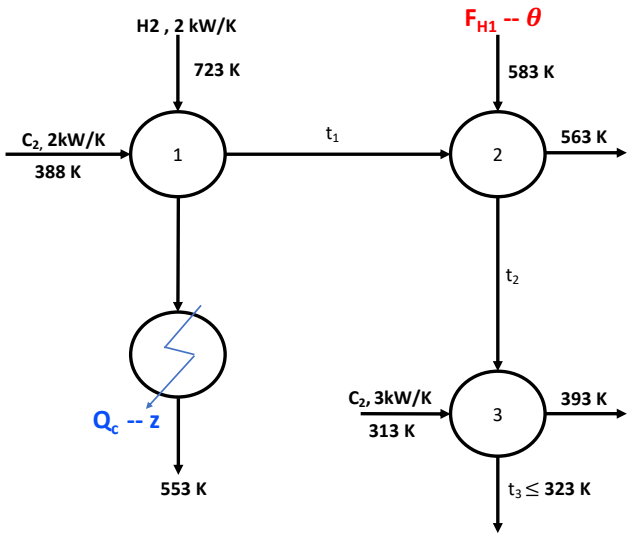

Figure 4: Diagram of heat exchanger network problem described in Example 2.

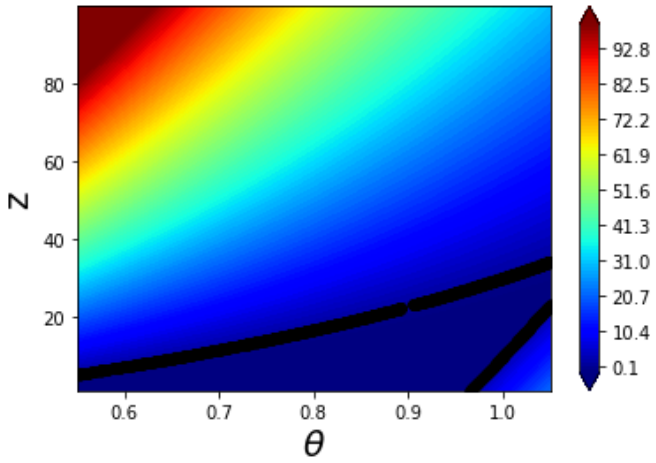

Figure 5: Contour of the worst-case constraint value $\max_{j \in \mathcal{J}} f_j(\theta, z)$ for heat exchanger network problem described in Example 2 as a function of θ and z. The black line represents the level set corresponding to a value of zero.

ignore the data for each individual inequality and instead build a single GP model for the aggregated worst-case constraint value. As such, this method neglects valuable information provided for each constraint relative to BoFlex.

Note that these cases only modify the selection of $(\boldsymbol{\theta}_{t+1}, \mathbf{z}_{t+1})$ in Lines 10 and 11 of Algorithm 1; we keep the termination process the same since our theoretical results show it is robust to the chosen design rule (as long as β_t are sufficiently large).

The results of BoFlex and the other comparison methods for a maximum of $T = 30$ iterations are shown in Figure 6. We provide all algorithms with a set of $N_{\text{init}} = 10$ random samples to train the hyperparameters of the kernel, as discussed in Section 4.4. Since the results depend on these random initial samples, we repeat the methods 100 times to get an estimate of the average performance (shown with a line) and corresponding confidence bounds (shown with the shaded region) in Figure 6 for the upper $\hat{\chi}_t^U$ and lower $\hat{\chi}_t^L$ bound on χ. BoFlex clearly outperforms

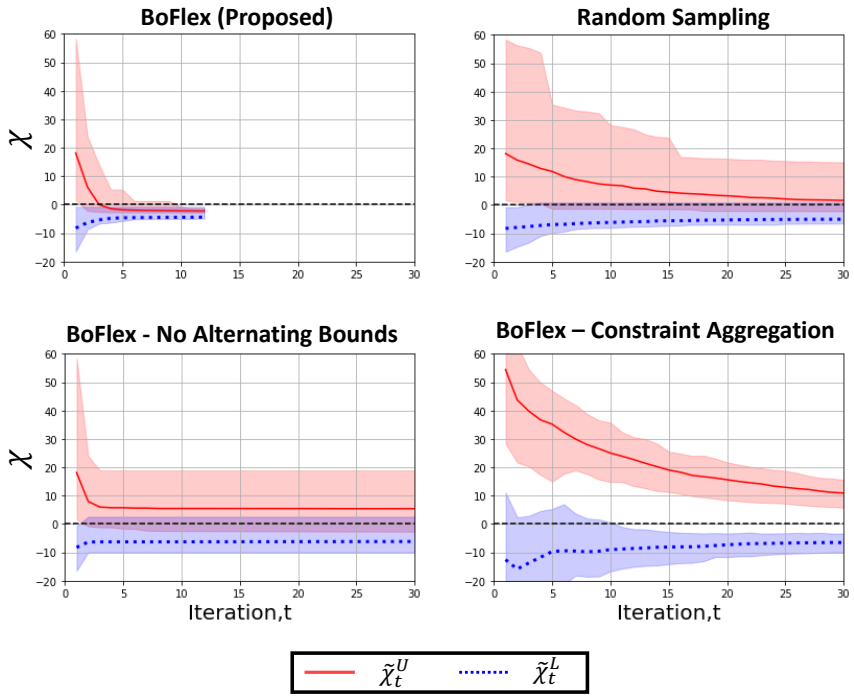

Figure 6: Comparison of the quality of the upper and lower bounds on the flexibility test parameter χ as a function of number of iterations for BoFlex and three modifications that remove an essential piece of the algorithm, which result in performance losses. The red (blue) cloud shows the best and worst case estimates for the upper (lower) bound of χ over 100 replicates with the median value shown with a solid red (blue) line. The dashed black line shows the zero value, which is the cutoff point between a flexible and inflexible system.

all other tested methods, consistently terminating by iteration 13 in all 100 trials. On the other hand, none of the other three cases converge (on average) within the 30 iterations. The "no alternating bound" case makes good initial progress, however, it quickly gets stuck and is unable to make progress likely due to lack of exploration in the z space. Furthermore, the "constraint aggregation" case shows much slower convergence, which is likely due to sharp changes in the KS function that make the GP model more difficult to train. Interestingly, "random sampling" appears to be the second best method, though it still demonstrates a substantially slower rate of convergence when compared to BoFlex on this problem.

6.3. Example 3: Large Heat Exchanger Network

As a third example, we consider the larger HEN system from [3, Example 1], which is illustrated in Figure 7. We slightly modified the structure of the equations by having the heat capacity depend nonlinearly on temperature to further increase the complexity of the problem. The resulting inequality constraints that this HEN must satisfy are given by

$$f_1(\boldsymbol{\theta}, z) = -0.67Q_c + T_3 - 350 \leq 0, \tag{24a}$$
$$f_2(\boldsymbol{\theta}, z) = -T_5 - 0.75C_1(T_1)T_1 + 0.5Q_c - T_3 + 1388.5 \leq 0, \tag{24b}$$
$$f_3(\boldsymbol{\theta}, z) = -T_5 - 1.5C_2(T_1)T_1 + Q_c - 2T_3 + 2044 \leq 0, \tag{24c}$$
$$f_4(\boldsymbol{\theta}, z) = -T_5 - 1.5C_2(T_1)T_1 + Q_c - 2T_3 - 2T_8 + 2830 \leq 0, \tag{24d}$$
$$f_5(\boldsymbol{\theta}, z) = T_5 + 1.5C_2(T_1)T_1 - Q_c + 2T_3 + 3T_8 - 3153 \leq 0, \tag{24e}$$

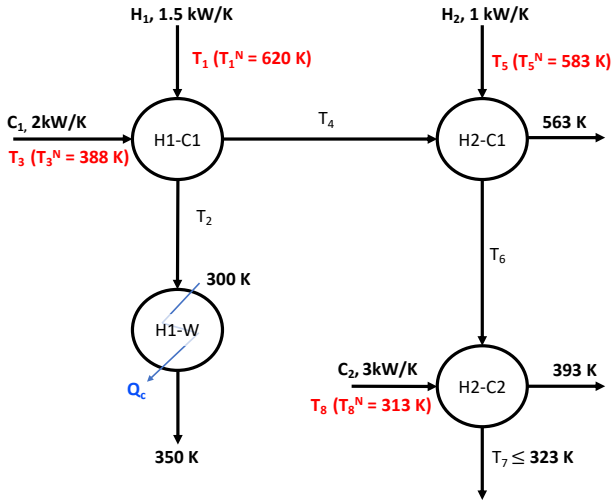

Figure 7: Diagram of heat exchanger network problem described in Example 3.

where $\boldsymbol{\theta} = (T_1, T_3, T_5, T_8)$ are the uncertain inlet temperatures, $z = Q_c$ is the recourse cooling rate, $C_1(T_1) = 1 + 0.02\cos\left(\frac{T_1}{4}\right)$, and $C_2(T_1) = 1 + 0.01\cos\left(\frac{T_1}{4}\right)$ are nonlinear perturbation to the heat capacity. In addition to the nonlinear perturbation, we also assume that the observations of each of these inequality functions are subject to random noise $\varepsilon_j \sim \text{Uniform}(-0.5, 0.5)$. The range of possible cooler heat duties is $\{z : 3 \leq z \leq 150\}$ kW. In this case, we will assume an uncertainty set of the form (20) with an expected deviation $\Delta\boldsymbol{\theta} = (1, 1, 1, 1)$ K for all temperature values (the nominal values are reported in Figure 7).

To demonstrate the results presented in Theorem 2, we apply BoFlex to this problem for three different ρ values: $\rho = 2$, $\rho = 4$, and $\rho = 8$, which correspond to true χ values for -0.84, 2.28, and 7.40, respectively. Similarly to the previous example, we generate 10 random initial data points for hyperparameter tuning and perform 100 replicates to compute an approximate distribution of termination iterations. The empirical cumulative distribution function (CDF) for these three different cases is reported in Figure 8. We see that, for $\rho = 8$ and $\rho = 4$, all 100 replicates terminate within 5 and 11 iterations, respectively. For $\rho = 2$, we have that 90% of the replicates have terminated by iteration 20. These results exactly match the results of Theorem 2, which states that the number of iterations until termination should decrease as the magnitude $\eta = |\chi|$ increases. Furthermore, we see that BoFlex can confidently provide solutions to this challenging 5-dimensional problem in a small number of iterations.

6.4. Example 4: Bubble Column Reactor

For the final case study, we consider the simulation-based bubble column reactor model originally developed by [55], which simulates the system illustrated in Figure 9. The model consists of two major components: (i) a set of multiphase convection-dispersion equations that govern the transport of species in the column and (ii) a genome-scale reconstruction of the metabolism of the bacteria identified using the flux balance analysis method. We have previously used this case study to test a novel robust optimization method in [51]. However, our prior work focused on the case that the design variables cannot adapt to the uncertainty realizations, so here we modify the problem to look at the flexibility of the system to uncertainty.

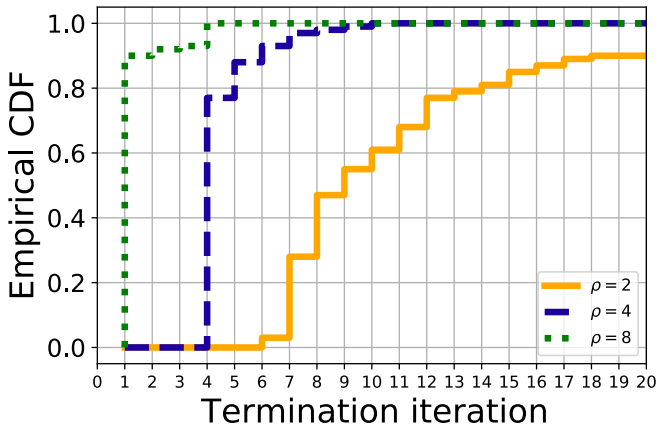

Figure 8: The empirical cumulative distribution function (CDF) for the termination iteration of the proposed BoFlex algorithm applied to Example 3 under different uncertainty sets. We see that larger ρ values (more uncertainty) actually result in faster convergence in this case since the magnitude of χ is larger in these cases, as expected based on the results shown in Theorem 2.

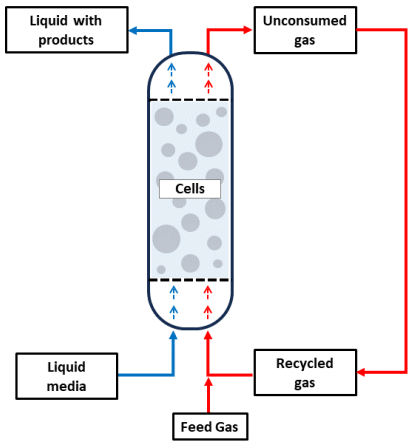

Figure 9: Schematic of the bubble column reactor system from [55].

The system is required to satisfy two inequality constraints that are both related to product quality requirements

$$f_1(\theta, z) = 13.5 \text{ g/L} - C_E(\theta, z) \leq 0, \tag{25a}$$
$$f_2(\theta, z) = C_A(\theta, z) - 8.5 \text{ g/L} \leq 0, \tag{25b}$$

where C_E and C_A, respectively, denote the steady-state concentration of ethanol and acetate in the liquid product stream. These concentrations are functions of the operating temperature θ (which is an uncertain parameter) and the superficial gas velocity z (which represents a recourse variable). The temperature can vary within the interval $\theta \in [308, 312]$ K and the superficial gas velocity can be manipulated between the following bounds $z \in [10, 14]$ m/hr. Similarly to both previous case studies, we assume that 10 random initial samples are available for hyperparameter tuning. Since this is an expensive simulator, we only run BoFlex a single time for a total of $T = 10$ iterations. BoFlex returns the following bounds on $\chi \in [\hat{\chi}_T^L, \hat{\chi}_T^U]$

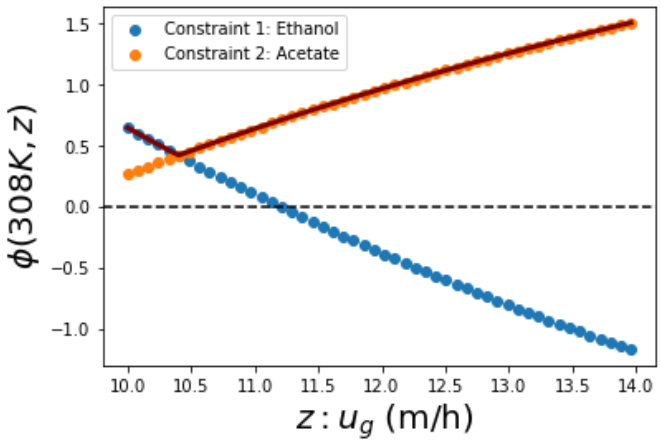

Figure 10: Constraints $f_1(\theta, z)$, $f_2(\theta, z)$, and $\phi(\theta, z) = \max_{j \in \mathcal{J}} f_j(\theta, z)$ values for the bubble column reactor case study for $\theta = 308$ K (worst-case value identified by BoFlex after 10 iterations). These results were obtained by running 50 high-fidelity simulations on a grid of z values. We can see that no feasible z value results in the worst-case constraint being less than 0 such that the true system is inflexible.

where $\hat{\chi}_T^L = 0.41$ and $\hat{\chi}_T^U = 0.42$, suggesting the system is *inflexible*. To verify this, we perform 50 high-fidelity simulations over a grid of z values for $\theta = 308$ K (which is the predicted worst-case value). The results are shown in Figure 10 for which we see that there is no feasible z value that leads to the worst-case constraint $\phi(\theta, z) = \max_{j \in \mathcal{J}} f_j(\theta, z)$ evaluated at $\theta = 308$ K being greater than zero, i.e., $\psi(308) = \min_{z \in \mathcal{Z}} \phi(308, z) > 0$. It is important to note that, without any prior knowledge about the model structure, we were able to tightly verify that the system is inflexible using only 20 total expensive simulations. This is a direct consequence of BoFlex's ability to intelligently design samples that are informative for validating or invalidating the flexibility test.

7. Conclusions

This paper proposes a new algorithm, BoFlex, for efficient flexibility analysis of black-box systems that allows for both uncertain parameters and recourse variables to impact multiple feasibility constraints. The algorithm takes into account the decomposed, multi-level optimization structure of the flexibility problem, which we show enables efficiency gains over our previously developed constraint aggregation methods and can also handle sub-Gaussian measurement noise. We also theoretically characterize the performance of BoFlex on arbitrary (potentially non-convex) functions belonging to a reproducing kernel Hilbert space (RKHS) in two ways. First, we show that upper and lower bounds on flexibility test parameter are guaranteed to bound the true parameter value and that the gap between these bounds converges to zero under mild conditions with high probability. Second, we derive an explicit bound on the number of iterations until convergence to a correct result in terms of the maximum information gain of the data-driven surrogate model. This establishes that BoFlex converges to a correct result in a finite number of iterations (with high probability) under mild conditions on the complexity of the RKHS. We demonstrate BoFlex's superior performance to alternative methods on three separate case studies. In all cases, BoFlex efficiently and automatically identifies correct solutions to the flexibility test problems, thereby significantly reducing computational cost in practice.

There are several interesting directions for future work. Although easily applicable to low-dimensional problems, BoFlex relies on the construction of probabilistic surrogate models whose priors may be difficult to identify in high-dimensional problems. Furthermore, the solution of the tri-level subproblems defined in terms of these surrogate models become substantially more difficult in higher dimensions. Developing more scalable approaches to both the surrogate model construction and acquisition function optimization are thus worthy of future investigation. In this work, we derived upper bounds on performance, however, it is not clear how tight these bounds are in practice. Therefore, from a theoretical perspective, it would be interesting to develop *algorithm-independent lower bounds* bounds on performance to give some indication how much the upper bound can be improved. Results of this type have recently been obtained in the standard Bayesian optimization setting [56] but have yet to be extended to the tri-level optimization problems of interest in this work. Lastly, a couple of recent works have shown the value of exploiting prior physical knowledge in the development of acquisition functions in standard Bayesian optimization [57, 58]. Similar ideas can be incorporated into BoFlex, though this would further complicate the acquisition function subproblems. As such, specific real-world applications for which this type of physics-based knowledge can be exploited would be interesting to pursue.

8. Proofs

8.1. Proof of Lemma 1

Proof. [Lemma 1] Directly follows from the proof of Theorem 2.2 in [43], which is related to Theorem 2 in [40]. The main difference is that we obtain q measurements at every iteration such that the mutual information $I(\mathbf{y}_t; h)$ grows at a faster rate than in the traditional single measurement setting, which completes the proof. □

8.2. Proof of Theorem 1

Lemma 2. *Fix $t \geq 0$. If $|h(\boldsymbol{\theta}, \mathbf{z}, j) - \mu_t(\boldsymbol{\theta}, \mathbf{z}, j)| \leq \beta_t^{1/2} \sigma_t(\boldsymbol{\theta}, \mathbf{z}, j)$ for all $(\boldsymbol{\theta}, \mathbf{z}, j) \in \Theta \times \mathcal{Z} \times \mathcal{J}$, then $\hat{\chi}_t^U - \hat{\chi}_t^L \leq 2\beta_t^{1/2} \sigma(\boldsymbol{\theta}_{t+1}, \mathbf{z}_{t+1}, j_{t+1})$ where $\boldsymbol{\theta}_{t+1}$ and $\mathbf{z}_{t+1}$ are given in Lines 10 and 11 of Algorithm 1, respectively, and $j_{t+1} \in \operatorname{argmax}_{j \in \mathcal{J}} u_t(\boldsymbol{\theta}_{t+1}, \mathbf{z}_{t+1}, j)$.*

Proof. Given the definitions of $\hat{\chi}_t^U$ and $\hat{\chi}_t^L$, given in Lines 2 and 3 of BoFlex (Algorithm 1) we can establish the following sequence of inequalities

$$\begin{aligned}
\hat{\chi}_t^U - \hat{\chi}_t^L &= \max_{\boldsymbol{\theta} \in \Theta} \min_{\mathbf{z} \in \mathcal{Z}} \max_{j \in \mathcal{J}} u_t(\boldsymbol{\theta}, \mathbf{z}, j) - \max_{\boldsymbol{\theta} \in \Theta} \min_{\mathbf{z} \in \mathcal{Z}} \max_{j \in \mathcal{J}} l_t(\boldsymbol{\theta}, \mathbf{z}, j), && \text{(26a)} \\
&= \min_{\mathbf{z} \in \mathcal{Z}} \max_{j \in \mathcal{J}} u_t(\boldsymbol{\theta}_{t+1}, \mathbf{z}, j) - \max_{\boldsymbol{\theta} \in \Theta} \min_{\mathbf{z} \in \mathcal{Z}} \max_{j \in \mathcal{J}} l_t(\boldsymbol{\theta}, \mathbf{z}, j), && \text{(26b)} \\
&\leq \min_{\mathbf{z} \in \mathcal{Z}} \max_{j \in \mathcal{J}} u_t(\boldsymbol{\theta}_{t+1}, \mathbf{z}, j) - \min_{\mathbf{z} \in \mathcal{Z}} \max_{j \in \mathcal{J}} l_t(\boldsymbol{\theta}_{t+1}, \mathbf{z}, j), && \text{(26c)} \\
&= \min_{\mathbf{z} \in \mathcal{Z}} \max_{j \in \mathcal{J}} u_t(\boldsymbol{\theta}_{t+1}, \mathbf{z}, j) - \max_{j \in \mathcal{J}} l_t(\boldsymbol{\theta}_{t+1}, \mathbf{z}_{t+1}, j), && \text{(26d)} \\
&\leq \max_{j \in \mathcal{J}} u_t(\boldsymbol{\theta}_{t+1}, \mathbf{z}_{t+1}, j) - \max_{j \in \mathcal{J}} l_t(\boldsymbol{\theta}_{t+1}, \mathbf{z}_{t+1}, j), && \text{(26e)} \\
&= u_t(\boldsymbol{\theta}_{t+1}, \mathbf{z}_{t+1}, j_{t+1}) - \max_{j \in \mathcal{J}} l_t(\boldsymbol{\theta}_{t+1}, \mathbf{z}_{t+1}, j), && \text{(26f)} \\
&\leq u_t(\boldsymbol{\theta}_{t+1}, \mathbf{z}_{t+1}, j_{t+1}) - l_t(\boldsymbol{\theta}_{t+1}, \mathbf{z}_{t+1}, j_{t+1}), && \text{(26g)} \\
&= 2\beta_t^{1/2} \sigma_t(\boldsymbol{\theta}_{t+1}, \mathbf{z}_{t+1}, j_{t+1}), && \text{(26h)}
\end{aligned}$$

where (26b) follows from the definition of $\boldsymbol{\theta}_{t+1}$; (26c) follows from $F(\boldsymbol{\theta}_{t+1}) \leq \max_{\boldsymbol{\theta} \in \Theta} F(\boldsymbol{\theta})$ for any choice of $\boldsymbol{\theta}_{t+1} \in \Theta$ and any function F by definition of the max operator; (26d) follows from the definition of $\mathbf{z}_{t+1}$, which minimizes the worst-case constraint violation predicted by the lower bound; (26e) follows from the fact

that the minimum over $\mathbf{z} \in \mathcal{Z}$ must be less than or equal to the value for a specific $\mathbf{z}_{t+1} \in \mathcal{Z}$; (26f) follows from the definition of j_{t+1}; (26g) follows from the same arguments as (26c); and (26h) follows from the definitions of the upper and lower confidence bounds, which are separated by $2\beta_t^{1/2}\sigma_t(\cdot)$ at any specific point. □

Lemma 3. *The sum of the GP predicted standard deviations evaluated at the sample points can be bounded by the maximum information gain as follows*

$$2\sum_{t=1}^{T}\sigma_{t-1}(\boldsymbol{\theta}_t, \mathbf{z}_t, j_t) \leq \sqrt{C_1 T \gamma_{Tq}}, \tag{27}$$

where $C_1 = 8/\ln(1+\lambda^{-1})$.

Proof. The proof is analogous to [31, Lemma 5.4] with a couple of important differences due to having several functions. First, we apply Cauchy-Schwarz inequality on a vector of all ones and all standard deviations:

$$2\sum_{t=1}^{T}\sigma_{t-1}(\boldsymbol{\theta}_t, \mathbf{z}_t, j_t) \leq \sqrt{4T\sum_{t=1}^{T}\sigma_{t-1}^2(\boldsymbol{\theta}_t, \mathbf{z}_t, j_t)}. \tag{28}$$

Next, we bound the individual variance terms as follows

$$\begin{aligned}\sigma_{t-1}^2(\boldsymbol{\theta}_t, \mathbf{z}_t, j_t) &= \lambda(\lambda^{-1}\sigma_{t-1}^2(\boldsymbol{\theta}_t, \mathbf{z}_t, j_t)) \\ &\leq \lambda C_2 \ln(1+\lambda^{-1}\sigma_{t-1}^2(\boldsymbol{\theta}_t, \mathbf{z}_t, j_t)),\end{aligned} \tag{29}$$

with $C_2 = \lambda^{-1}/(1+\lambda^{-1}) \geq 1$. This inequality holds since $s^2 \leq C_2\ln(1+s^2)$ for all $s \in [0, \lambda^{-1}]$ and $\lambda^{-1}\sigma_{t-1}^2(\boldsymbol{\theta}_t, \mathbf{z}_t, j_t) \leq \lambda^{-1}k((\boldsymbol{\theta}_t, \mathbf{z}_t, j_t), (\boldsymbol{\theta}_t, \mathbf{z}_t, j_t)) \leq \lambda^{-1}$ by Assumption 2. By [31, Lemma 5.3], the mutual information (13) can be expressed in terms of the predictive GP variances

$$I(\mathbf{y}_t; h) = \frac{1}{2}\sum_{t=1}^{T}\sum_{j=1}^{q}\ln\left(1+\lambda^{-1}\sigma_{t-1}^2(\boldsymbol{\theta}_t, \mathbf{z}_t, j)\right). \tag{30}$$

We can combine the previous result with this one to establish the following bound on the sum of the variances

$$\begin{aligned}\sum_{t=1}^{T}\sigma_{t-1}^2(\boldsymbol{\theta}_t, \mathbf{z}_t, j_t) &\leq \lambda C_2 \sum_{t=1}^{T}\ln(1+\lambda^{-1}\sigma_{t-1}^2(\boldsymbol{\theta}_t, \mathbf{z}_t, j_t)), \\ &\leq 2\lambda C_2 I(\mathbf{y}_t; h), \\ &\leq 2\lambda C_2 \gamma_{Tq},\end{aligned} \tag{31}$$

where the second line follows from the monotonicity of the mutual information. We can complete the proof by combining (31) with (28) and noting that $C_1 = 8\lambda C_2$. □

We are now in a position to prove the main result:

Proof. [Theorem 1] Since the minimum of a sequence is upper bounded by the

average of the sequence, it follows

$$G_T^\star = \min_{t\in\{1,\dots,T\}} G_t \leq \frac{1}{T}\sum_{t=1}^{T} G_t = \frac{1}{T}\sum_{t=1}^{T}\left(\hat{\chi}_{t-1}^U - \hat{\chi}_{t-1}^L\right), \tag{32a}$$

$$\leq \frac{1}{T}\sum_{t=1}^{T} 2\beta_{t-1}^{1/2}\sigma_{t-1}(\boldsymbol{\theta}_t, \mathbf{z}_t, j_t), \tag{32b}$$

$$\leq \frac{2\beta_T^{1/2}}{T}\sum_{t=1}^{T}\sigma_{t-1}(\boldsymbol{\theta}_t, \mathbf{z}_t, j_t), \tag{32c}$$

$$\leq \sqrt{\frac{C_1\beta_T\gamma_{Tq}}{T}}, \tag{32d}$$

where (32a) uses the definition of the bound gap, (32b) follows from Lemma 2, (32c) follows from the monotonicity of the β_t sequence, and (32d) uses (27) established by Lemma 3. Note that these inequalities hold with probability $\geq 1-\delta$ by Lemma 1. Using (32d), we deduce that $G_T^\star \leq \epsilon$ if we can pick a pair of values ϵ and T that satisfies $\frac{C_1\beta_T\gamma_{Tq}}{T} \leq \epsilon^2$. The stated claims in the theorem then follow by rearranging this inequality, thus completing the proof. □

8.3. Proof of Theorem 2

Proof. [Theorem 2] We split this proof into two parts based on $\chi = \eta$ (not flexible) or $\chi = -\eta$ (flexible). We will then prove by contradiction that, under these disjoint cases, the algorithm must terminate given the bounds established in Lemma 1, which imply for all $(\boldsymbol{\theta}, \mathbf{z}, j) \in \Theta \times \mathcal{Z} \times \mathcal{J}$ and $t \geq 0$

$$f_j(\boldsymbol{\theta}, \mathbf{z}) \in [l_t(\boldsymbol{\theta}, \mathbf{z}, j), u_t(\boldsymbol{\theta}, \mathbf{z}, j)], \tag{33}$$

must hold with probability $\geq 1-\delta$.

Let us start with $\chi = \eta > 0$. Assume, for the sake of contradiction, that the algorithm never terminates. This immediately implies $\hat{\chi}_t^L \leq 0$ for all $t \geq 0$ such that the test on Line 7 of Algorithm 1 never passes. Let $j_{t+1} \in \operatorname{argmax}_{j\in\mathcal{J}} u_t(\boldsymbol{\theta}_{t+1}, \mathbf{z}_{t+1}, j)$ for $\boldsymbol{\theta}_{t+1}$ and $\mathbf{z}_{t+1}$ defined in Lines 10 and 11 of Algorithm 1, respectively, then we can also establish the following sequence of inequalities for $\hat{\chi}_t^L$

$$0 \geq \hat{\chi}_t^L = \max_{\boldsymbol{\theta}\in\Theta}\min_{\mathbf{z}\in\mathbf{Z}}\max_{j\in\mathcal{J}} l_t(\boldsymbol{\theta}, \mathbf{z}, j), \tag{34a}$$

$$\geq \min_{\mathbf{z}\in\mathbf{Z}}\max_{j\in\mathcal{J}} l_t(\boldsymbol{\theta}_{t+1}, \mathbf{z}, j), \tag{34b}$$

$$= \max_{j\in\mathcal{J}} l_t(\boldsymbol{\theta}_{t+1}, \mathbf{z}_{t+1}, j), \tag{34c}$$

$$\geq l_t(\boldsymbol{\theta}_{t+1}, \mathbf{z}_{t+1}, j_{t+1}), \tag{34d}$$

where (34b) is a consequence of $\boldsymbol{\theta}_{t+1} \in \Theta$ being feasible so must be less than or equal to the maximum; (34c) follows from the definition of $\mathbf{z}_{t+1}$; and (34d) follows from the fact that $j_{t+1} \in \mathcal{J}$ so it must be less than or equal to the maximum. Since $\chi = \eta$, we also have

$$u_t(\boldsymbol{\theta}_{t+1}, \mathbf{z}_{t+1}, j_{t+1}) = \max_{j\in\mathcal{J}} u_t(\boldsymbol{\theta}_{t+1}, \mathbf{z}_{t+1}, j), \tag{35a}$$

$$\geq \min_{\mathbf{z}\in\mathbf{Z}}\max_{j\in\mathcal{J}} u_j(\boldsymbol{\theta}_{t+1}, \mathbf{z}, j), \tag{35b}$$

$$= \max_{\boldsymbol{\theta}\in\Theta}\min_{\mathbf{z}\in\mathbf{Z}}\max_{j\in\mathcal{J}} u_t(\boldsymbol{\theta}, \mathbf{z}, j), \tag{35c}$$

$$\geq \max_{\boldsymbol{\theta}\in\Theta}\min_{\mathbf{z}\in\mathbf{Z}}\max_{j\in\mathcal{J}} f_j(\boldsymbol{\theta}, \mathbf{z}) = \chi = \eta, \tag{35d}$$

where (35a) follows from the definition of j_{t+1}; (35b) is a consequence of $\mathbf{z}_{t+1} \in \mathcal{Z}$ being feasible so must be greater than or equal to the minimum; (35c) follows from the definition of $\boldsymbol{\theta}_{t+1}$; and (35d) follows from (33) and the assumption that $\chi = \eta$. Combining (34) and (35), we have

$$2\sum_{t=1}^{T} \beta_{t-1}^{1/2} \sigma_{t-1}(\boldsymbol{\theta}_t, \mathbf{z}_t, j_t) = \sum_{t=1}^{T} (u_{t-1}(\boldsymbol{\theta}_t, \mathbf{z}_t, j_t) - l_{t-1}(\boldsymbol{\theta}_t, \mathbf{z}_t, j_t)) \geq \sum_{t=1}^{T} \eta = T\eta. \tag{36}$$

Meanwhile, we also have that

$$2\sum_{t=1}^{T} \beta_{t-1}^{1/2} \sigma_{t-1}(\boldsymbol{\theta}_t, \mathbf{z}_t, j_t) \leq 2\beta_T^{1/2} \sum_{t=1}^{T} \sigma_{t-1}(\boldsymbol{\theta}_t, \mathbf{z}_t, j_t) \leq \sqrt{C_1 \beta_T T \gamma_{Tq}}, \tag{37}$$

where the first inequality follows from the monotonicity of the β_t sequence and the second inequality follows from Lemma 3. Combining (36) and (37), we have

$$\sqrt{C_1 \beta_T T \gamma_{Tq}} \geq T\eta, \tag{38}$$

which implies $\eta \leq \sqrt{\frac{C_1 \beta_T \gamma_{Tq}}{T}}$. From the definition of β_t, we have that $\beta_T = \mathcal{O}(\gamma_{Tq})$ such that $\sqrt{\frac{C_1 \beta_T \gamma_{Tq}}{T}} = \mathcal{O}\left(\frac{\gamma_{Tq}}{\sqrt{T}}\right)$. However, since $\lim_{T\to\infty} \frac{\gamma_{Tq}}{\sqrt{T}} = 0$, this creates a contradiction with $\eta > 0$ such that BoFlex must correctly terminate with $\hat{\chi}_T^L > 0$ for some finite T whenever $\chi = \eta > 0$.

We can follow the same logical steps for the case of $\chi = -\eta < 0$, with the main difference being that we must switch the signs when going through the initial arguments. Eventually, we end up with the same inequality (38) such that we can follow the same arguments to conclude that $\hat{\chi}_T^U \leq 0$ for some finite T. Since these are mutually exclusive events, Boflex must finitely terminate to the correct results, which completes the proof. □

References

[1] K. Halemane, I. Grossmann, Optimal process design under uncertainty, AIChE Journal (1983) 425–433.

[2] I. E. Grossmann, B. A. Calfa, P. Garcia-Herreros, Evolution of concepts and models for quantifying resiliency and flexibility of chemical processes, Computers & Chemical Engineering 70 (2014) 22–34.

[3] I. E. Grossmann, C. A. Floudas, Active constraint strategy for flexibility analysis in chemical processes, Computers & Chemical Engineering 11 (6) (1987) 675–693.

[4] C. A. Floudas, Z. H. Gümüş, M. G. Ierapetritou, Global optimization in design under uncertainty: feasibility test and flexibility index problems, Industrial & Engineering Chemistry Research 40 (20) (2001) 4267–4282.

[5] E. Pistikopoulos, T. Mazzuchi, A novel flexibility analysis approach for processes with stochastic parameters, Computers & Chemical Engineering 14 (9) (1990) 991–1000. doi:10.1016/0098-1354(90)87055-t.
URL `https://doi.org/10.1016/0098-1354(90)87055-t`

[6] D. Straub, I. Grossmann, Integrated stochastic metric of flexibility for systems with discrete state and continuous parameter uncertainties, Computers &

Chemical Engineering 14 (9) (1990) 967–985. doi:10.1016/0098-1354(90)87053-r.
URL https://doi.org/10.1016/0098-1354(90)87053-r

[7] V. D. Dimitriadis, E. N. Pistikopoulos, Flexibility analysis of dynamic systems, Industrial & Engineering Chemistry Research 34 (12) (1995) 4451–4462.

[8] Q. Zhang, I. E. Grossmann, R. M. Lima, On the relation between flexibility analysis and robust optimization for linear systems, AIChE Journal 62 (9) (2016) 3109–3123.

[9] A. Ben-Tal, A. Nemirovski, Robust optimization–methodology and applications, Mathematical Programming 92 (2002) 453–480.

[10] N. V. Sahinidis, Optimization under uncertainty: State-of-the-art and opportunities, Computers & Chemical Engineering 28 (6-7) (2004) 971–983.

[11] D. Bertsimas, D. B. Brown, C. Caramanis, Theory and applications of robust optimization, SIAM Review 53 (3) (2011) 464–501.

[12] V. Gabrel, C. Murat, A. Thiele, Recent advances in robust optimization: An overview, European Journal of Operational Research 235 (3) (2014) 471–483.

[13] D. Bertsimas, V. Gupta, N. Kallus, Data-driven robust optimization, Mathematical Programming 167 (2018) 235–292.

[14] İ. Yanıkoğlu, B. L. Gorissen, D. den Hertog, A survey of adjustable robust optimization, European Journal of Operational Research 277 (3) (2019) 799–813.

[15] J. P. Slotnick, A. Khodadoust, J. Alonso, D. Darmofal, W. Gropp, E. Lurie, D. J. Mavriplis, Cfd vision 2030 study: a path to revolutionary computational aerosciences, Tech. rep. (2014).

[16] I. Banerjee, S. Pal, S. Maiti, Computationally efficient black-box modeling for feasibility analysis, Computers & Chemical Engineering 34 (9) (2010) 1515–1521.

[17] F. Boukouvala, M. G. Ierapetritou, Feasibility analysis of black-box processes using an adaptive sampling kriging-based method, Computers & Chemical Engineering 36 (2012) 358–368.

[18] A. Rogers, M. Ierapetritou, Feasibility and flexibility analysis of black-box processes part 2: Surrogate-based flexibility analysis, Chemical Engineering Science 137 (2015) 1005–1013.

[19] Z. Wang, M. Ierapetritou, A novel feasibility analysis method for black-box processes using a radial basis function adaptive sampling approach, AIChE Journal 63 (2) (2017) 532–550.

[20] F. Zhao, I. E. Grossmann, S. García-Muñoz, S. D. Stamatis, Flexibility index of black-box models with parameter uncertainty through derivative-free optimization, AIChE Journal 67 (5) (2021) e17189.

[21] J. Larson, M. Menickelly, S. M. Wild, Derivative-free optimization methods, Acta Numerica 28 (2019) 287–404.

[22] M. J. Powell, et al., The BOBYQA algorithm for bound constrained optimization without derivatives, Cambridge NA Report NA2009/06, University of Cambridge, Cambridge 26 (2009).

[23] E. Brochu, V. M. Cora, N. De Freitas, A tutorial on Bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning, arXiv preprint arXiv:1012.2599 (2010).

[24] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, N. De Freitas, Taking the human out of the loop: A review of Bayesian optimization, Proceedings of the IEEE 104 (1) (2015) 148–175.

[25] P. I. Frazier, A tutorial on Bayesian optimization, arXiv preprint arXiv:1807.02811 (2018).

[26] C. K. Williams, C. E. Rasmussen, Gaussian Processes for Machine Learning, Vol. 2, MIT press Cambridge, MA, 2006.

[27] C. G. Raspanti, J. A. Bandoni, L. Biegler, New strategies for flexibility analysis and design under uncertainty, Computers & Chemical Engineering 24 (9-10) (2000) 2193–2209.

[28] J. K. Scott, M. D. Stuber, P. I. Barton, Generalized McCormick relaxations, Journal of Global Optimization 51 (4) (2011) 569–606.

[29] I. Bogunovic, J. Scarlett, S. Jegelka, V. Cevher, Adversarially robust optimization with Gaussian processes, Advances in Neural Information Processing Systems 31 (2018).

[30] J. A. Paulson, G. Makrygiorgos, A. Mesbah, Adversarially robust Bayesian optimization for efficient auto-tuning of generic control structures under uncertainty, AIChE Journal 68 (6) (2022) e17591.

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

[32] J. Mockus, Application of Bayesian approach to numerical methods of global and stochastic optimization, Journal of Global Optimization 4 (1994) 347–365.

[33] A. Kudva, J. A. Paulson, A Bayesian optimization approach for data-efficient flexibility analysis of expensive black-box models, In Proceedings of the Foundations of Computer Aided Process Operations / Chemical Process Control, 2023.

[34] G. A. Wrenn, An indirect method for numerical optimization using the Kreisselmeir-Steinhauser function, Tech. rep., NASA (1989).

[35] F. Berkenkamp, A. Krause, A. P. Schoellig, Bayesian optimization with safety constraints: safe and automatic parameter tuning in robotics, Machine Learning (2021) 1–35.

[36] M. M. Noack, J. A. Sethian, Advanced stationary and nonstationary kernel designs for domain-aware gaussian processes, Communications in Applied Mathematics and Computational Science 17 (1) (2022) 131–156.

[37] A. Berlinet, C. Thomas-Agnan, Reproducing kernel Hilbert spaces in probability and statistics, Springer Science & Business Media, 2011.

[38] Y. Abbasi-Yadkori, D. Pál, C. Szepesvári, Improved algorithms for linear stochastic bandits, Advances in Neural Information Processing Systems 24 (2011).

[39] S. Agrawal, N. Goyal, Thompson sampling for contextual bandits with linear payoffs, in: International Conference on Machine Learning, PMLR, 2013, pp. 127–135.

[40] S. R. Chowdhury, A. Gopalan, On kernelized multi-armed bandits, in: International Conference on Machine Learning, PMLR, 2017, pp. 844–853.

[41] F. Berkenkamp, A. P. Schoellig, A. Krause, No-regret Bayesian optimization with unknown hyperparameters, arXiv preprint arXiv:1901.03357 (2019).

[42] M. Kanagawa, P. Hennig, D. Sejdinovic, B. K. Sriperumbudur, Gaussian processes and kernel methods: A review on connections and equivalences, arXiv preprint arXiv:1807.02582 (2018).

[43] A. Durand, O.-A. Maillard, J. Pineau, Streaming kernel regression with provably adaptive mean, variance, and regularization, The Journal of Machine Learning Research 19 (1) (2018) 650–683.

[44] S. Vakili, K. Khezeli, V. Picheny, On information gain and regret bounds in Gaussian process bandits, in: International Conference on Artificial Intelligence and Statistics, PMLR, 2021, pp. 82–90.

[45] C. Fiedler, C. W. Scherer, S. Trimpe, Practical and rigorous uncertainty bounds for Gaussian process regression, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 35, 2021, pp. 7439–7447.

[46] E. Schulz, M. Speekenbrink, A. Krause, A tutorial on Gaussian process regression: Modelling, exploring, and exploiting functions, Journal of Mathematical Psychology 85 (2018) 1–16.

[47] A. D. Bull, Convergence rates of efficient global optimization algorithms., Journal of Machine Learning Research 12 (10) (2011).

[48] I. Bogunovic, A. Krause, Misspecified Gaussian process bandit optimization, Advances in Neural Information Processing Systems 34 (2021) 3004–3015.

[49] A. Capone, A. Lederer, S. Hirche, Gaussian process uniform error bounds with unknown hyperparameters for safety-critical applications, in: International Conference on Machine Learning, PMLR, 2022, pp. 2609–2624.

[50] D. Bertsimas, O. Nohadani, K. M. Teo, Robust optimization for unconstrained simulation-based problems, Operations Research 58 (1) (2010) 161–178.

[51] A. Kudva, F. Sorourifar, J. A. Paulson, Constrained robust Bayesian optimization of expensive noisy black-box functions with guaranteed regret bounds, AIChE Journal 68 (12) (2022) e17857.

[52] G. R. Wood, The bisection method in higher dimensions, Mathematical programming 55 (1992) 319–337.

[53] M. Balandat, B. Karrer, D. Jiang, S. Daulton, B. Letham, A. G. Wilson, E. Bakshy, BoTorch: A framework for efficient Monte-Carlo Bayesian optimization, Advances in Neural Information Processing Systems 33 (2020) 21524–21538.

[54] A. Kudva, J. A. Paulson, BoFlex (2023).
URL `https://github.com/PaulsonLab/BoFlex`

[55] J. Chen, J. A. Gomez, K. Höffner, P. Phalak, P. I. Barton, M. A. Henson, Spatiotemporal modeling of microbial metabolism, BMC Systems Biology 10 (1) (Mar. 2016). doi:10.1186/s12918-016-0259-2.
URL https://doi.org/10.1186/s12918-016-0259-2

[56] J. Scarlett, I. Bogunovic, V. Cevher, Lower bounds on regret for noisy gaussian process bandit optimization, in: Conference on Learning Theory, PMLR, 2017, pp. 1723–1742.

[57] J. A. Paulson, C. Lu, COBALT: COnstrained Bayesian optimizAtion of computationaLly expensive grey-box models exploiting derivaTive information, Computers & Chemical Engineering 160 (2022) 107700.

[58] C. Lu, J. A. Paulson, No-regret constrained Bayesian optimization of noisy and expensive hybrid models using differentiable quantile function approximations, arXiv preprint arXiv:2305.03824 (2023).