ORIGINAL ARTICLE
AIChE Journal

Adversarially Robust Bayesian Optimization for Efficient Auto-Tuning of Generic Control Structures under Uncertainty

Joel A. Paulson[1*] | Georgios Makrygiorgos[2†] | Ali Mesbah[2†]

[1]Department of Chemical and Biomolecular Engineering at The Ohio State University, Columbus, OH 43210, USA

[2]Department of Chemical and Biomolecular Engineering at the University of California, Berkeley, CA 94720, USA

Correspondence
Ali Mesbah
Email: mesbah@berkeley.edu

Funding information
This material is in part based upon work supported by the National Aeronautics and Space Administration (NASA) under grant number NNX17AJ31G. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of NASA.

The performance of optimization- and learning-based controllers critically depends on the selection of several tuning parameters that can affect the closed-loop control performance and constraint satisfaction in highly nonlinear and nonconvex ways. Due to the black-box nature of the closed-loop performance measures, there has been a significant interest in automatic calibration (i.e., auto-tuning) of complex control structures using derivative-free optimization methods, including Bayesian optimization (BO) that can handle expensive unknown cost functions. However, an open challenge when applying BO to auto-tuning is how to effectively deal with uncertainties in the closed-loop system that cannot be attributed to a lumped, small-scale noise term. This paper addresses this challenge by developing an adversarially robust BO (ARBO) method that is particularly suited to auto-tuning problems with significant time-invariant uncertainties in an expensive "high fidelity" system model used for closed-loop simulations. ARBO relies on a Gaussian process model that jointly describes the effect of the tuning parameters and uncertainties on the closed-loop performance. From this joint Gaussian process model, ARBO uses an alternating confidence-bound procedure to simultaneously select the next candidate tuning and uncertainty realizations, implying only one expensive closed-loop simulation is needed at each iteration. The advantages of ARBO are demonstrated on two case studies, including an illustrative problem and auto-tuning of a nonlinear model predictive controller for an uncertain bioreactor.

KEYWORDS
Data-driven optimization; Robust Bayesian optimization; Controller auto-tuning; Model uncertainty

1 | INTRODUCTION

Recent years have witnessed significant progress in the design and application of optimization- and learning-based controllers that can deal with multivariable dynamics, constraints, and uncertainties that appear in the system and/or the environment. However, the design of such advanced controllers hinges on the selection of several *tuning parameters* that may strongly affect closed-loop performance and constraint satisfaction. Additionally, these tuning parameters can come in a variety of different forms including continuous (e.g., weight parameters), discrete (e.g., logical switching conditions such as adaptive tuning), and categorical (e.g., type of numerical discretization scheme) representations, which implies their impact on performance can be highly nonlinear and non-convex. Therefore, in practice, these tuning parameters are usually selected via trial-and-error experimentation or heuristic-based strategies that rely on expensive closed-loop simulations or experiments, which can become prohibitive when the effects of system uncertainties are accounted for [1].

To mitigate the expensive nature of tuning of advanced controllers, there has been an increasing interest in automatic calibration (aka *auto-tuning* [2, 3, 4, 5, 6]) of complex control structures to achieve desired closed-loop performance. To this end, data-driven optimization methods have been found to be particularly promising since auto-tuning can be interpreted as a black-box problem in which the objective function is expensive to evaluate, potentially non-convex and multi-modal, and whose derivatives either do not exist or cannot be determined. Bayesian optimization (BO) [7, 8] has emerged as a powerful approach for handling these types of black-box problems, even when the measured objective value is corrupted by noise. Several recent works have successfully demonstrated BO for model learning and auto-tuning of model predictive control (MPC) [5, 9, 10, 11] and other complex control structures [12, 13].

Standard BO approaches for auto-tuning rely on non-parametric Gaussian process (GP) models [14], constructed from closed-loop simulation or experimental data, to describe the impact of controller tuning parameters on the closed-loop performance measures; these GP models can be interpreted as probabilistic “surrogate models” for the performance measures of interest. Although GP models are able to account for the effect of system uncertainties (e.g., exogenous disturbances, measurement noise, and/or time-invariant uncertainties in process models used for closed-loop simulations) by optimizing an “effective noise” hyperparameter, this representation can lead to poor predictions when uncertainties are relatively large. That is, the GP model yields such a large variance in predictions that the mean prediction is dominated by noise, suggesting the GP model is uninformative. In such cases, the BO procedure will become quite fragile and thus will lead to poor overall results. We addressed this challenge in our recent work by introducing an auto-tuning approach, referred to as probabilistically robust Bayesian optimization (PRBO), that provides a probabilistic robustness certificate at every iteration (i.e., every time a new set of tuning parameters is tested) [15]. The key idea in PRBO is to use sample-based estimates of the worst-case performance measures at each iteration. We show how many samples are required — *independent* of the number and probability distribution of the uncertainties — to ensure these worst-case estimates are not violated by other randomly sampled uncertainties within a prespecified probability level. However, since PRBO provides this certification at every iteration, it generally requires a fairly large number of closed-loop simulations/experiments to be performed in order to establish accurate estimates of the worst-case performance measures. This can limit the applicability of PRBO especially when expensive "high-fidelity" process models (or experiments) are utilized for generating closed-loop data.

In this paper, we present an alternative robust BO approach to PRBO that is well-suited for auto-tuning problems that rely on expensive closed-loop simulations with significant time-invariant uncertainties. This type of problem setting appears in a wide variety of applications that use complex process models and model predictive controllers (MPC), including advanced manufacturing and energy systems [16], among many other applications. As opposed to measuring an estimate of the worst-case performance directly (as done in PRBO), the proposed approach, referred to as *adversarially robust BO* (ARBO), looks to construct a GP model that *simultaneously* describes the effect of the controller tuning parameters and system uncertainties on the closed-loop performance. In this way, we can directly use the joint GP model to predict the location of a minimax solution to the robust auto-tuning problem. We show, however, that using a naive mean-based GP approximation of the performance measure will yield overall poor tuning results, as it lacks the ability to tradeoff between exploration of unknown parts of the *design-uncertainty space* and exploitation of the current estimate of the best tuning parameters. Instead, the proposed ARBO method uses a GP confidence bound-based procedure suggested in [17] to realize a tradeoff between the exploration and exploitation of the design-uncertainty space. In this approach, we alternate between an optimistic prediction of the performance measure to select the next best set of tuning parameters and a pessimistic prediction of the performance measure to select the most likely worst-case uncertainty for the suggested best tuning parameters. By applying this two-step procedure, we only require one (expensive) closed-loop simulation at each iteration of ARBO, which is significantly less than alternatives such as PRBO. Building upon the theory in [17], we also discuss the rate of convergence of the ARBO method, and provide an explicit upper bound on the distance from the best suggested tuning parameters and the true minimax optimal solution, which decays to zero as the number of iterations increases. We demonstrate the value of the proposed ARBO method on two case studies; an illustrative problem to highlight the key steps and advantages of ARBO and a challenging auto-tuning problem in which a highly nonlinear bioreactor with several unknown parameters is controlled using nonlinear MPC with multiple constraint backoffs that must be tuned.

2 | PROBLEM STATEMENT

We are interested in the auto-tuning problem for a general class of controllers, i.e., we want to select the unknown tuning parameters such that we achieve the best possible closed-loop performance, while protecting against potentially adversarial effects of some "external" source of uncertainty. Let $\theta \in \mathbb{R}^{p_1}$ denote the vector of controller tuning parameters and $\delta \in \mathbb{R}^{p_2}$ denote the uncertainty vector. Given some scalar measure of the closed-loop performance $f : \mathbb{R}^{p_1} \times \mathbb{R}^{p_2} \rightarrow \mathbb{R}$ whose structure is unknown, we formulate the auto-tuning problem as the following robust black-box optimization problem

$$\min_{\theta \in \Theta} \max_{\delta \in \Delta} f(\theta, \delta), \tag{1}$$

where $\Theta \subset \mathbb{R}^{p_1}$ and $\Delta \subset \mathbb{R}^{p_2}$ are the compact sets of possible tuning parameters and uncertainty realizations, respectively. The controller tuning parameters θ can represent any manipulable value including discrete structural choices (e.g., turning on/off a component) that are modeled with binary variables, as well as parametric choices that are modeled by continuous variables (e.g., increasing a weight value between lower and upper bounds). To account for the effects of uncertainty on controller tuning, we must quantify the impact of different realizations of δ on the performance measure f. Thus, throughout this work, we assume that a *high-fidelity simulator* of the process is available for simulating the effect of specific controller configurations and uncertainty realizations on the closed-loop performance measure f.[1] This allows f to be flexibly specified by the user in terms of any finite-time metric; some common examples include total operating cost or setpoint tracking error, average or maximum constraint violation, and end-of-batch product quality.

We aim to find the (approximate) global solution to the controller auto-tuning problem (1). The specific algorithm chosen to solve (1) will depend on its underlying characteristics. Thus, we assume that the following characteristics hold, which is generally the case in simulation-based tuning of advanced controllers under uncertainty [1].

Assumption 1 ***(1)*** *The worst-case uncertainty* $\delta^\star(\theta) \in \text{argmax}_{\delta \in \Delta} f(\theta, \delta)$ *cannot be determined from prior knowledge.*
(2) *The feasible sets* Θ *and* Δ *are known and compact.*
(3) *The closed-loop performance measure* $f(\theta, \delta)$ *is fully black-box in nature such that no closed-form expression exists for* f *and it does not have any known special structure such as convexity or linearity.*
(4) *The total dimension of the inputs* $p = p_1 + p_2$ *is typically not too large;* $p \leq 20$ *is a good rule-of-thumb.*
(5) *When the closed-loop performance performance is evaluated, we only observe* $f(\theta, \delta)$*, meaning that first- or second-order derivatives cannot be evaluated.*
(6) *The observations of* $f(\theta, \delta)$ *are corrupted by noise. That is,* $y = f(\theta, \delta) + \epsilon$*, where* $\epsilon \in \mathcal{N}(0, \sigma_\epsilon^2)$.

Characteristics (1)-(3) in Assumption 1 imply minimal restrictions on the structure of the to-be-designed controller such that the proposed method for controller auto-tuning can be applied even when the control law is defined implicitly—for example, as is the case in model predictive control (MPC). Characteristic (5) prevents application of derivative-based optimization methods for solving (1). For simplicity, characteristic (6) assumes the effective noise ϵ leading to noisy observations y of the closed-loop performance measure is normally distributed with zero mean. The variance of noise can be treated as a hyperparmaeter, as discussed in Section 4. Notice that the closed-loop performance measure $f(\theta, \delta)$ is quantified through possibly expensive simulations of the closed-loop system using a process simulator. As such, the performance measure can be queried a limited number of times; often on the order of a few hundred of closed-loop simulations.

Remark Although δ can in principle represent any source of uncertainty, this may lead to a high-dimensional representation of δ due to the time-varying nature of control problems. As such, this may not satisfy characteristic (4) in Assumption 1. Instead, δ should represent the key time-invariant uncertainties (e.g., sensitive model parameters and/or initial conditions) that have the most dominant influence on the performance measure f. If prior knowledge about the dominant time-invariant uncertainties is not available, it can be obtained via global sensitivity analysis [18, 19], which can be facilitated via surrogate modeling [20]. Notice that, although not included in δ, the effect of time-varying process and measurement noise is accounted for through noisy observations of f; see characteristic (6) in Assumption 1.

---

[1] We refer to the process simulator as "high-fidelity" to denote the fact that it can be a computationally-expensive model, such as a multiscale model, built from a collection of software codes/packages.

The most direct way to solve Problem (1) would be via a nested optimization approach wherein an inner maximization is performed for each iteration of an outer minimization algorithm [21]. This approach, however, will expend excessive effort computing the worst-case closed-loop performance for every selected design variables θ, which is not appropriate when dealing with expensive evaluations of f using a high-fidelity process simulator. This also precludes the use of evolutionary algorithms [22], which are popular techniques when the objective function can be evaluated a large number of times. Alternatively, we look to reformulate (1) as a *bandit feedback* problem [23]. The main idea is to sequentially select $(\theta_t, \delta_t) \in \Theta \times \Delta$ at every iteration $t = 1, 2, \dots$ (here, "iteration" refers to a single closed-loop simulation), and receive the corresponding noisy observations of the cost $y_t = f(\theta_t, \delta_t) + \epsilon_t$. Our regret in this decision can be quantified in terms of the *instantaneous robust-regret* r_t^δ, which is defined as

$$r_t^\delta = \max_{\delta \in \Delta} f(\theta_t, \delta) - \max_{\delta \in \Delta} f(\theta^\star, \delta), \tag{2}$$

where $\theta^\star \in \operatorname{argmin}_{\theta \in \Theta} \max_{\delta \in \Delta} f(\theta, \delta)$ is any global solution to (1). Ideally, we could derive an algorithm that minimizes the *cumulative robust-regret* after T iterations $R_T^\delta = \sum_{t=1}^T r_t^\delta$; however, these quantities cannot be revealed to the algorithm since they require perfect knowledge of the global solution. A viable alternative is to select an algorithm that has *no robust-regret*, i.e., $\lim_{T \to \infty} \frac{1}{T} R_T^\delta = 0$ [17]. The only way that the average robust-regret can approach zero is for the instantaneous robust-regret to approach zero, since $r_t^\delta \geq 0$ must be non-negative. This implies that there exists a $t > 0$ such that $\max_{\delta \in \Delta} f(\theta_t, \delta)$ is arbitrarily close to $\max_{\delta \in \Delta} f(\theta^\star, \delta)$ and the algorithm converges as long as R_T^δ grows sublinearliy with T. In the absence of uncertainty (i.e., the nominal setting of $\Delta = \{\hat{\delta}\}$), we can easily find the point $\{\theta_1, \dots, \theta_T\}$ that minimizes the (non-robust) regret by selecting the point that produces the smallest value of $f(\theta_t, \hat{\delta})$. This is no longer true in the robust case, however, due to the inclusion of the max operator in (2). Therefore, we require a new recommendation procedure in addition to the selection policy for (θ_t, δ_t). In the next section, we present a variant of the sequential learning algorithm in [17], referred to as adversarially robust Bayesian optimization, that can achieve the desired no robust-regret property using a combined Gaussian process (GP) model for $f(\theta, \delta)$, which simultaneously models the effect of the design variables and uncertainty realizations on the closed-loop performance measure.

3 | ADVERSARIALLY ROBUST BAYESIAN OPTIMIZATION

In this section, we first review Gaussian process (GP) regression for data-driven modeling of the closed-loop performance measure. We will then present the adversarially robust Bayesian optimization (ARBO) algorithm, followed by an overview of established theoretical results [17] related to the robust-regret when solving (1) under the conditions specified in Assumption 1.

3.1 | Gaussian Process Regression

Let $\mathbf{x} = [\theta^\top, \delta^\top]^\top \in \mathcal{X}$ denote the concatenated vector of design variables and uncertainties, where $\mathcal{X} = \Theta \times \Delta \subset \mathbb{R}^p$ and $p = p_1 + p_2$. We interchangeably denote $f(\theta, \delta)$ as $f(\mathbf{x})$ (and vice versa) throughout the paper. Since the structure of f is not known, we cannot make rigid parametric assumptions for f. However, without further assumptions, it would be impossible to achieve sublinear robust-regret for (1); for example, f could be discontinuous at every input $\mathbf{x} \in \mathcal{X}$ in the worst-case. Therefore, we assume that a certain degree of smoothness holds in practice, such that we can leverage GP models that enforce smoothness implicitly without making any parametric assumptions. The basic idea underpinning GPs is that the function values $f(\mathbf{x})$, associated with different values of $\mathbf{x}$, are random variables and any finite collection of these random variables have a joint Gaussian distribution [14]. A GP distribution, denoted by $f(\mathbf{x}) \sim \mathcal{GP}(\mu(\mathbf{x}), k(\mathbf{x}, \mathbf{x}'))$, is parametrized by a prior mean function $\mu(\mathbf{x})$ and a covariance (or kernel) function $k(\mathbf{x}, \mathbf{x}')$. Without loss of generality, we assume that the prior is zero mean, i.e., $\mu(\mathbf{x}) = 0$ for all $\mathbf{x} \in \mathcal{X}$.[2] The chosen class of covariance functions determines the properties of the fitted functions. In this work, we will focus on stationary covariance functions from the Matern class, defined as

$$k(\mathbf{x}, \mathbf{x}'; \nu, \Psi) = \zeta^2 \frac{2^{1-\nu}}{\Gamma(\nu)} (\sqrt{2\nu} r(\mathbf{x}, \mathbf{x}')) B_\nu \left(\sqrt{2\nu} r(\mathbf{x}, \mathbf{x}') \right), \tag{3}$$

where $r(\mathbf{x}, \mathbf{x}') = \sqrt{(\mathbf{x} - \mathbf{x}') L^{-2} (\mathbf{x} - \mathbf{x}')}$ is the scaled Euclidean distance, $L = \text{diag}(l_1, \ldots, l_p)$ is a diagonal scaling matrix composed of length-scale parameters $l_1, \ldots, l_p > 0$, ν is a parameter that dictates smoothness (i.e., the corresponding function is $\lceil \nu/2 - 1 \rceil$ times differentiable), ζ^2 is a scaling factor for the output variance, Γ and B_ν are the Gamma and modified Bessel functions, respectively, and $\Psi = \{l_1, \ldots, l_p, \zeta\}$ are the hyperparameters of the kernel for fixed ν. Training a GP model corresponds to calibrating $\{\Psi, \sigma_\epsilon\}$ to the available data. For now, we assume the kernel hyperparameters are known and discuss the training procedure further in Section 4.

A key advantage of GPs, in addition to their non-parametric nature, is the availability of simple analytic expressions for the posterior distribution of $f(\mathbf{x})$ for any input $\mathbf{x} \in \mathcal{X}$. Let us assume that we have t previous observations of the objective $\mathbf{y}_t = [y_1, \ldots, y_t]^\top$ at inputs $\mathbf{X}_t = \{\mathbf{x}_1, \ldots, \mathbf{x}_t\}$. The GP model can account for the fact that these measurements are noisy, i.e., $y_t = f(\mathbf{x}_t) + \epsilon_t$ where $\epsilon_t \sim \mathcal{N}(0, \sigma_\epsilon^2)$. Given that the noise ϵ_t obeys a normal distribution, the posterior $f | \mathbf{X}_t, \mathbf{y}_t$ remains a GP $\mathcal{GP}(\mu_t(\mathbf{x}), k_t(\mathbf{x}, \mathbf{x}'))$ with the following expressions for the mean μ_t, covariance k_t, and variance σ_t^2 [14]

$$\mu_t(\mathbf{x}) = \mathbf{k}_t^\top(\mathbf{x}) \left(\mathbf{K}_t + \sigma_\epsilon^2 \mathbf{I}_t \right)^{-1} \mathbf{y}_t, \tag{4a}$$

$$k_t(\mathbf{x}, \mathbf{x}') = k(\mathbf{x}, \mathbf{x}') - \mathbf{k}_t^\top(\mathbf{x}) \left(\mathbf{K}_t + \sigma_\epsilon^2 \mathbf{I}_t \right)^{-1} \mathbf{k}_t(\mathbf{x}'), \tag{4b}$$

$$\sigma_t^2(\mathbf{x}) = k_t(\mathbf{x}, \mathbf{x}), \tag{4c}$$

where $\mathbf{k}_t(\mathbf{x}) = [k_j(\mathbf{x}_1, \mathbf{x}), \ldots k_j(\mathbf{x}_t, \mathbf{x})]^\top$ contains the covariances between the input $\mathbf{x}$ and observed data points $\mathbf{X}_t$, the covariance matrix $\mathbf{K}_t$ has entries $[\mathbf{K}_t]_{ij} = k(\mathbf{x}_i, \mathbf{x}_j)$ for all $i, j \in \{1, \ldots, t\}$, and $\mathbf{I}_t$ is the $t \times t$ identify matrix. The main advantage of the posterior GP expressions in (4) is that they can be used to generate *confidence bounds* on the prediction of $f(\theta, \delta)$ for any choice of input. Both the upper and lower confidence bounds will be leveraged in the development of the ARBO algorithm, as described next.

3.2 | ARBO Algorithm

Given a so-called *exploration parameter* β_t, we can define the following upper and lower confidence bounds on f

$$\text{ucb}_t(\theta, \delta) = \mu_t(\theta, \delta) + \beta_t^{1/2} \sigma_t(\theta, \delta), \tag{5a}$$

$$\text{lcb}_t(\theta, \delta) = \mu_t(\theta, \delta) - \beta_t^{1/2} \sigma_t(\theta, \delta), \tag{5b}$$

---

[2] This can easily be achieved by normalizing the data before training, as discussed in, e.g., [24].

which are readily determined from the posterior GP in (4). For sufficiently large choices of β_t, these confidence bounds will be large enough to ensure the no robust-regret property with high probability (see Theorem 1). The ARBO algorithm [17], which relies on the lower and upper confidence bounds (5), is presented in Algorithm 1. The suggested θ_t at each iteration is the one that has the minimum "robust" *lower* confidence bound, as given in (8). For this choice of θ_t, we must select a feasible uncertainty sample. According to (9), we select the uncertainty value δ_t that maximizes the *upper* confidence bound. We can interpret these opposite choices as: (i) optimistic selections under uncertainty for θ_t and (ii) pessimistic selections under uncertainty for the anticipated worst-case point δ_t. While the choice (i) is common to traditional BO algorithms that utilize confidence bounds, the choice (ii) is unique to ARBO to mitigate any possible negative effects caused by the uncertainty. Once the main loop in Algorithm 1 has been completed, a final "recommended" point must be selected from the sequence $\{\theta_1, \ldots, \theta_T\}$. Although there are many potential choices, we choose the one that minimizes a pessimistic bound on the robust-regret in (10). To this end, let us assume $f(\theta, \delta) \leq \text{ucb}_{t-1}(\theta, \delta)$ for all $(\theta, \delta) \in \Theta \times \Delta$; this condition will be more formally stated later. Then, we can define the following pessimistic estimate of r_t^δ

$$\bar{r}_t^\delta = \max_{\delta \in \Delta} \text{ucb}_{t-1}(\theta_t, \delta) - f^\star, \tag{6}$$

where $f^\star = \max_{\delta\in\Delta} f(\theta^\star, \delta) = \min_{\theta\in\Theta} \max_{\delta\in\Delta} f(\theta, \delta)$, which must satisfy $r_t^\delta \leq \bar{r}_t^\delta$ for all $t \geq 1$ under the above-stated assumption. The main difference between (2) and (6) is that the algorithm has enough information to identify the index $t^\star$ that minimizes $\bar{r}_t^\delta$ since the global solution does not depend on t. Yet, $\bar{r}_t^\delta$ is related to another important quantity in bandit optimization termed the *simple robust-regret* after T iterations, which is denoted by S_T^δ and defined as

$$S_T^\delta = \min_{t\in\{1,\ldots,T\}} r_t^\delta = \min_{t\in\{1,\ldots,T\}} \max_{\delta\in\Delta} f(\theta_t, \delta) - f^\star. \tag{7}$$

It is evident that $S_T^\delta \leq \bar{r}_{t^\star}^\delta$ for all $T \geq 1$. This in turn implies that bounds established on $\bar{r}_{t^\star}^\delta$ immediately transfer to the simple robust-regret S_T^δ, as discussed in the next section. Notice that Algorithm 1 relies on only a single expensive closed-loop simulation run to be performed at every iteration, which is significantly fewer than the vast majority of available alternatives, such as [21, 25].

3.3 | Upper Bound on Simple Robust-Regret

The ARBO Algorithm 1 requires selection of the exploration parameters $\{\beta_t\}_{t\geq 1}$ that specify the width of the confidence intervals on f. To this end, we rely on a simple result from [23] to select this sequence. We will focus on the case of a finite set $\mathcal{X} = \Theta \times \Delta$ for simplicity, and discuss the extension to a compact and convex set later.

Lemma 1 (Confidence bounds [23]) *Let* $f(\mathbf{x}) \sim \mathcal{GP}(0, k(\mathbf{x}, \mathbf{x}'))$ *be a sample of a GP for which noisy observations* $y_t = f(\mathbf{x}_t) + \epsilon_t$ *with* $\epsilon_t \sim \mathcal{N}(0, \sigma_\epsilon^2)$ *are available. Let* $\beta_t = 2\log(|\mathcal{X}|t^2\pi^2/(6\alpha))$ *for a specified failure probability* $\alpha \in (0, 1)$ *and finite discrete set* $|\mathcal{X}| < \infty$*. Then, the following bounds on the objective function* $f(\mathbf{x})$

$$f(\mathbf{x}) \in [\text{lcb}_{t-1}(\mathbf{x}), \text{ucb}_{t-1}(\mathbf{x})], \quad \forall \mathbf{x} \in \mathcal{X}, \forall t \geq 1, \tag{11}$$

hold with probability (over the GP posterior at every iteration) at least $1 - \alpha$.

Next, we define the *maximum information gain* (MIG), which provides a measure of the informativeness of any finite set of sampling points $\mathcal{A} \subset \mathcal{X}$ [26].

Algorithm 1 The robust sequential learning algorithm for ARBO.

Input: The set of the design variables Θ and the uncertainty Δ; kernel k corresponding to GP prior; exploration parameters $\{\beta_t\}_{t\geq 1}$; and total number of iterations T.

```
1: Initialize the mean and standard deviation $(\mu_0, \sigma_0) \leftarrow (0, k^{1/2})$.
2: for $t = 1$ to $T$ do
3:     Solve the following min-max optimization problem for $\theta_t$
           $\theta_t = \operatorname*{argmin}_{\theta\in\Theta} \max_{\delta\in\Delta} \text{lcb}_{t-1}(\theta, \delta).$   (8)
4:     Solve the following maximization problem for $\delta_t$
           $\delta_t = \operatorname*{argmax}_{\delta\in\Delta} \text{ucb}_{t-1}(\theta_t, \delta).$   (9)
5:     Run a closed-loop simulation at $\mathbf{x}_t = [\theta_t^\top, \delta_t^\top]^\top$ to compute performance measure $y_t = f(\theta_t, \delta_t) + \epsilon_t$.
6:     Perform Bayesian posterior update to estimate $\mu_t$, $\sigma_t$, $\text{lcb}_t$ and $\text{ucb}_t$ using (4) and (5) by including the latest
       query of the closed-loop performance measure $\{\mathbf{x}_t, y_t\}$.
7: end for
8: Return the point $\theta_{t^\star}$ with the smallest upper confidence bound (our best guess of the optimal design variables)
       $t^\star = \operatorname*{argmin}_{t\in\{1,\ldots,T\}} \max_{\delta\in\Delta} \text{ucb}_{t-1}(\theta_t, \delta).$   (10)
```

Definition 1 *Let $\mathcal{A} \subset \mathcal{X}$ denote any subset of sampling points from $\mathcal{X}$ and let f be a sample of a GP model with the same sampling conditions stated in Lemma 1. The maximum information gain for f under t measurements is defined as*

$$\gamma_t = \max_{\mathcal{A}\subset\mathcal{X}:|\mathcal{A}|=t} \frac{1}{2} \log\det(\mathbf{I}_t + \sigma_\epsilon^{-2}\mathbf{K}_{\mathcal{A}}), \tag{12}$$

where $\mathbf{K}_{\mathcal{A}} = [k(\mathbf{x}, \mathbf{x}')]_{\mathbf{x},\mathbf{x}'\in\mathcal{A}}$ is the kernel matrix. Note that the term inside of the max in (12) *is the Shannon mutual information between f and the observations at points $\mathbf{x} \in \mathcal{A}$.* ◃

We can now state the main theorem that bounds the performance of the ARBO Algorithm 1. We give a brief sketch of the proof of this result, which is a slightly different version of that provided in [17, Supplementary Material].

Theorem 1 (Upper ARBO Performance Bound [17]) *Fix $\alpha \in (0, 1)$, $\beta_t = 2\log(|\mathcal{X}|t^2\pi^2/(6\alpha))$, and $T \geq 1$. Running the ARBO algorithm for a sample f of a GP with zero mean and kernel $k(\mathbf{x}, \mathbf{x}')$, the simple robust-regret must satisfy*

$$Pr\left\{ S_T^\delta \leq \bar{r}_{t^\star}^\delta \leq \sqrt{\frac{C_1\beta_T\gamma_T}{T}} \right\} \geq 1 - \alpha, \tag{13}$$

where $C_1 = 8/\log(1 + \sigma_\epsilon^{-2})$.

Proof From Lemma 1, we know that $\text{lcb}_{t-1}(\mathbf{x}) \leq f(\mathbf{x}) \leq \text{ucb}_{t-1}(\mathbf{x})$ holds for all $\mathbf{x} \in \mathcal{X}$, $t \geq 1$ with probability greater than or equal to $1 - \alpha$. Given this, from the definitions in (2) and (6), we have

$$\begin{aligned} r_t^\delta = \max_{\delta \in \Delta} f(\theta_t, \delta) - \min_{\theta \in \Theta} \max_{\delta \in \Delta} f(\theta, \delta) \leq \bar{r}_t^\delta &= \max_{\delta \in \Delta} \text{ucb}_{t-1}(\theta_t, \delta) - \min_{\theta \in \Theta} \max_{\delta \in \Delta} f(\theta, \delta), \\ &= \text{ucb}_{t-1}(\theta_t, \delta_t) - \min_{\theta \in \Theta} \max_{\delta \in \Delta} f(\theta, \delta), \\ &\leq \text{ucb}_{t-1}(\theta_t, \delta_t) - \min_{\theta \in \Theta} \max_{\delta \in \Delta} \text{lcb}_{t-1}(\theta, \delta), \\ &= \text{ucb}_{t-1}(\theta_t, \delta_t) - \max_{\delta \in \Delta} \text{lcb}_{t-1}(\theta_t, \delta), \\ &\leq \text{ucb}_{t-1}(\theta_t, \delta_t) - \text{lcb}_{t-1}(\theta_t, \delta_t), \\ &= 2\beta_t^{1/2} \sigma_{t-1}(\theta_t, \delta_t), \end{aligned}$$

where the first line follows from the upper bound on f, the second line follows from the definition of δ_t in (9), the third line follows from the lower bound on f, the fourth line follows from the definition of θ_t in (8), the fifth line follows from the fact that $\max_{\delta \in \Delta} \text{lcb}_{t-1}(\theta_t, \delta) \geq \text{lcb}_{t-1}(\theta_t, \delta_t)$ for any feasible choice of $\delta_t \in \Delta$, and the sixth line follows from the difference between the confidence bounds in (5). Given this bound, we can also see that the following sequence of inequalities must hold with probability $\geq 1 - \alpha$

$$(R_T^\delta)^2 \leq T \textstyle\sum_{t=1}^T (r_t^\delta)^2 \leq 4\beta_T \sum_{t=1}^T \sigma_{t-1}^2(\theta_t, \delta_t),$$

where the first step follows from the Cauchy-Schwarz inequality and the second step follows from the monotonicity of the sequence $\{\beta_t\}_{t \geq 1}$. Next, we use a special case of [23, Lemma 5.4] to establish a bound on the sum of variances in terms of the MIG (12)

$$4 \textstyle\sum_{t=1}^T \sigma_{t-1}^2(\theta_t, \delta_t) \leq C_1 \gamma_T,$$

for $C_1 = 8/\log(1 + \sigma_\epsilon^{-2})$. From these results, it follows that $\Pr\{R_T^\delta \leq \sqrt{C_1 T \beta_T \gamma_T}\} \geq 1 - \alpha$. The assertion in (13) follows by noting that the minimum of a sequence must be less than or equal to the average, i.e., $S_T^\delta \leq \frac{1}{T} R_T^\delta$, in addition to the fact that the same inequalities hold for $\bar{r}_t^\delta$ in place of r_t^δ. ■

As the total number of iterations T increases in (13), we observe that the simple robust-regret gets closer to the desired value of zero, implying the global minimax solution has been found in the limit as $T \rightarrow \infty$, as long as the numerator $C_1 \beta_T \gamma_T \in o(T)$, where o is little-o notation that implies $C_1 \beta_T \gamma_T$ decays faster than T. The choice of β_T in Theorem 1 clearly shows logarithmic growth with respect to T. However, we also require bounds on the MIG γ_T to establish convergence. It was shown in [23] that γ_T has sublinear dependence with respect to T for many commonly used kernels, including the Mateŕn class, such that the ARBO algorithm converges to function evaluations near $\theta^\star$ with high probability for sufficiently small choices of α. This is a key advantage of the confidence bound-based ARBO algorithm compared to available alternatives whose theoretical properties have yet to be understood.

4 | PRACTICAL IMPLEMENTATION OF ARBO

In this section, we discuss some of the main aspects in practical implementation of the ARBO Algorithm 1, as also considered in the case studies presented in Section 5.

4.1 | Choice of Exploration Constant β_t

Lemma 1 and Theorem 1 only hold for discrete spaces $\mathcal{X}$. However, using the discretization technique introduced in [23], these results can be extended to continuous spaces that are compact and convex. The main added assumption is that the kernel function $k(\mathbf{x}, \mathbf{x}')$ must be chosen such that it ensures the following high probability bounds on the derivatives of f for some constants $a, b > 0$

$$\Pr\left\{\sup_{\mathbf{x}\in\mathcal{X}}\left|\frac{\partial f(\mathbf{x})}{\partial x_i}\right| > L\right\} \leq a e^{-(L/b)^2}, \ \forall i = 1, \dots, p, \forall L > 0. \tag{14}$$

Whenever this condition holds, the results in Lemma 1 and Theorem 1 can be generalized to any compact and convex set $\mathcal{X} \subset [0, r]^p$ by enlarging the exploration constant

$$\beta_t = 2\log\left(\frac{2\pi^2 t^2}{3\alpha}\right) + 2p\log\left(t^2 pbr \log^{\frac{1}{2}}(4pa/\alpha)\right). \tag{15}$$

To the best of our knowledge, these results have not yet been extended to arbitrary non-convex sets. However, this may not pose a challenge in practice since the choices of β_t are generally known to be conservative [27]. In the case studies in Section 5, we select $\beta_t = \beta_0 p \log(2t)$ to capture the dominant dependence of the exploration constant on t and p. A tyical value for β_0 is 0.1. An interesting direction for future work includes establishing a more robust way to select $\{\beta_t\}_{t\geq 1}$ for specific applications.

4.2 | Estimation of GP Hyperparameters

The results in Lemma 1 and Theorem 1 assume that the hyperparameters $\{\Psi, \sigma_\epsilon\}$ of the GP prior for f are known exactly. Since this is often not true in practice, we rely on the maximum likelihood estimation (MLE) framework to determine the optimal hyperparameters $\{\Psi_t^\star, \sigma_\epsilon^\star\}$ that, at every iteration t, maximize the log-likelihood $\mathcal{L}_t(\Psi, \sigma_\epsilon)$ [14]

$$\{\Psi_t^\star, \sigma_{\epsilon,t}^\star\} \in \underset{\Psi,\sigma_\epsilon}{\operatorname{argmax}}\ \mathcal{L}_t(\Psi, \sigma_\epsilon) = \log(p(\mathbf{y}_t|\mathbf{X}_t, \Psi, \sigma_\epsilon)). \tag{16}$$

Based on the GP prior, the measured data vector $\mathbf{y}_t$ must be distributed according to a multivariate Gaussian distribution of the following form

$$\mathbf{y}_t \sim \mathcal{N}(0, \Sigma_t(\Psi, \sigma_\epsilon)), \quad [\Sigma_t(\Psi, \sigma_\epsilon)]_{ij} = k(\mathbf{x}_i, \mathbf{x}_j|\Psi) + \sigma_\epsilon^2\delta_{ij}, \quad \forall i, j \in \{1, \dots, t\}. \tag{17}$$

Using this representation, an analytical expression for the log-likelihood function can be derived as

$$\mathcal{L}_t(\Psi, \sigma_\epsilon) = -\mathbf{y}_t^\top \Sigma_t^{-1}\mathbf{y}_t - \frac{1}{2}\log(\det(\Sigma_t)) - \frac{p}{2}\log(2\pi). \tag{18}$$

The optimization problem (16) is a nonlinear program that can be solved using gradient-based methods (e.g., IPOPT [28]) since (18) is a smooth, differentiable function. To ensure the optimizer does not get stuck in a local solution, it is useful to “warm-start” the local solver with the best solution found from a heuristic global optimization method such as the DIRECT solver [29]. Notice that the “warm-start” approach will introduce an additional step into Algorithm 1 that could be somewhat computationally expensive depending on the size of the optimization (16). A simple way to reduce the computational cost associated with hyperparameter estimation is to update the hyperparameters of the

GP model only periodically, instead of at every iteration. In this work, we exclusively use the Python package `GPy` [30] to train and make predictions with GP models.

4.3 | Minimax Optimization for lcb_{t-1}

Our analysis in Section 3 assumed that we could exactly optimize the acquisition functions defined in terms of the lower and upper confidence bounds in (8) and (9). The maximization problem (9) resembles the sub-problem that arises in the standard BO, suggesting that the same basic principles can be leveraged to develop a practical solution method for the ARBO Algorithm 1. Here, we propose to use a combination of derivative-free search with a local gradient-based solver for the min-max optimization (8) at each itertation. Note that since $\text{lcb}_{t-1}(\theta, \delta)$ may be non-convex with respect to θ and non-concave with respect to δ, we cannot use traditional alternating gradient descent-ascent methods, as they may not even locally converge [31].

The proposed approach partially exploits the differentiability of $\text{lcb}_{t-1}(\theta, \delta)$. Let us denote the optimal objective value for the inner maximization problem as $g_{t-1}(\theta) = \max_{\delta \in \Delta} \text{lcb}_{t-1}(\theta, \delta)$. We can then equivalently formulate (8) as

$$\min_{\theta \in \Theta} \; g_{t-1}(\theta), \tag{19}$$

where g_{t-1} is a black-box function that can only be evaluated by calling an internal algorithm to approximate $g_{t-1}(\theta)$ for any choice of $\theta \in \Theta$. Since, for any fixed θ, $\text{lcb}_{t-1}(\theta, \delta)$ is a smooth function whose derivatives can be efficiently computed, we can rely on gradient-based solvers (e.g., the well-known L-BFGS-B algorithm) to quickly converge to a local optimum. Since we need a good estimate of the *global solution* for the inner maximization, we need some type of globalization strategy. One approach is to apply a random multi-start approach wherein a large number of random samples of δ (10^5 in our numerical experiments) are used to initially evaluate $\text{lcb}_{t-1}(\theta, \delta)$ with θ fixed. The δ points that lead to the 10 highest $\text{lcb}_{t-1}(\theta, \delta)$ values are used to warm start the local solver, with the largest converged objective value being returned as our best approximation to $g_{t-1}(\theta)$. We then treat (19) as a black-box optimization problem that can be solved with any number of available derivative-free optimization methods. In this work, we rely on BOBYQA [32, 33], which is a local trust region-based approach. We again rely on a random multi-start procedure to protect against local solutions for this outer minimization problem; however, since g_{t-1} is fairly expensive to evaluate, we must carefully select the number of repeats to ensure a solution can be found in a reasonable amount of time.

5 | CASE STUDIES

In this section, we demonstrate the performance of the ARBO algorithm on two problems. The first case study is an illustrative example that is meant to showcase several implementation details of Algorithm 1. Since the exact knowledge of the function and its min-max solution is available, we can directly compute the key performance assessment measures, such as the simple robust-regret, in the illustrative example. The second case study, on the other hand, focuses on a challenging nonlinear MPC (NMPC) auto-tuning problem. Since this auto-tuning problem involves a nonlinear plant simulator, we do not have exact knowledge of the true solution and thus cannot use simple robust-regret as our performance measure. Instead, we evaluate the solution quality directly in terms of the closed-loop performance and constraint satisfaction profiles. The main goal of this section is to show that ARBO can more reliably find high-performance tuning parameters with significantly fewer closed-loop simulations than alternative methods.

5.1 | Illustrative Example

Consider a system in the form of (1), with the following analytic expression for f

$$f(\theta, \delta) = \sin(\theta\delta) + \sqrt{\delta}\theta^2 - 0.5\theta, \tag{20}$$

where $\Theta = [-1, 2]$ is the feasible set of decision variable and $\Delta = [2, 4]$ is the feasible set of the uncertainty. Throughout this section, (20) is unknown to any of the black-box algorithms, and is only used for assessing our the regret-based performance measures. Figure 1 shows a plot of $f(\theta, \delta^{(i)})$ versus θ for a large number of random samples $\delta^{(i)} \in \Delta$, with the worst-case function $g(\theta) = \max_{\delta \in \Delta} f(\theta, \delta)$ shown with a dotted black line. From this plot, we can see that $\theta^\star = -0.45$, which corresponds to an optimal minimax objective value of $f^\star = 0$.

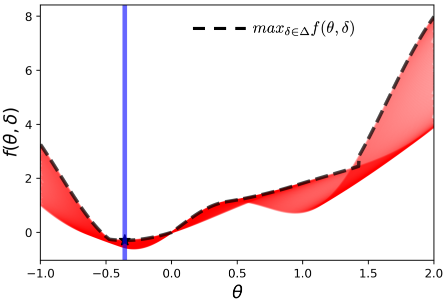


FIGURE 1 Objective function plots for various values of the uncertain parameter δ. The star symbol denotes the true minimax solution, while the blue line represents the best recommended discovered by ARBO.

We use the simple robust-regret S_T^δ as our metric since we aim to identify this robust solution in as few iterations as possible. Theorem 1 highlights the importance of the β_t sequence within Algorithm 1, as this is the main tool used to navigate the exploitation-exploration tradeoff in the joint $\{\theta, \delta\}$ space. To better illustrate this point, we compare ARBO to the a purely exploitative approach, namely a *Gaussian process-based robust optimization* (GP-RO) approach. In GP-RO, we completely ignore the variance information provided by the GP model for $f(\theta, \delta)$ and, instead, sample $\theta = \text{argmin}_{\theta \in \Theta} \max_{\delta \in \Delta} \mu_{t-1}(\theta, \delta)$ and $\delta_t = \text{argmax}_{\delta \in \Delta} \mu_{t-1}(\theta_t, \delta)$. Similarly, for the recommendation process, we also rely only on the mean function, i.e., $\theta_{t^\star}$ is returned with $t^\star = \text{argmin}_{t \in \{1,\ldots,T\}} \max_{\delta \in \Delta} \mu_{t-1}(\theta_t, \delta)$.

It is well-known that determining the hyperparameters of GP models, as discussed in Section 4, is often unreliable for very small datasets. Thus, as opposed to starting Algorithm 1 from iteration 1, it is usually preferred to select the first N_{init} points uniformly at random in any BO procedure to ensure a high-degree of exploration initially [34]. In this illustrative problem, we selected $N_{init} = p^2 - 1$ random points before running Algorithm 1. Since the simple robust-regret is a function of these randomly selected initial points, S_T^δ itself is a random quantity, so that showing results for a single initialization is not very informative. Instead, we repeated both the ARBO and GP-RO methods 10 times (under the same random seeds) to construct a sample average estimate for the expected simple robust-regret, i.e.,

$$\mathbb{E}\{S_T^\delta\} \approx \frac{1}{N_{repeat}} \sum_{i=1}^{N_{repeat}} S_T^{\delta,(i)}, \tag{21}$$

where $S_T^{\delta,(i)}$ denotes the simple robust-regret for the ith run of the algorithm starting from the ith set of N_{init} random initial points and $N_{repeat} = 10$. Since this estimate is constructed from a finite number of samples, we also report estimated confidence intervals computed as 1.96 times the standard deviation divided by the square root of the number of repeats (also known as the standard error formula).

The simple robust-regret plots for both ARBO and GP-RO are shown in Figure 2, with the estimated sample average and corresponding confidence-bound error bars on the left and the individual sample paths $S_T^{\delta,(i)}$ for all $i \in \{1, \dots, N_{repeat}\}$ shown on the right. We clearly see that ARBO consistently converges to within a tolerance of 10^{-6} of the true $f^\star$ for all considered initial points, which leads to $\mathbb{E}\{S_T^\delta\} \approx 0$ using only 15 function evaluations. GP-RO, on the other hand, shows considerably worse performance for the individual sample paths as well as the sample average. The right plot in Figure 2 is particularly informative, as we see that multiple runs of GP-RO get "stuck" and do not make progress for the entire 30 allotted function evaluations. This behavior is not unexpected, as it is well-known to occur in algorithms that lack any degree of exploration – in this context, there is no clear incentive for GP-RO to sample in unexplored regions of the $\Theta \times \Delta$ space.

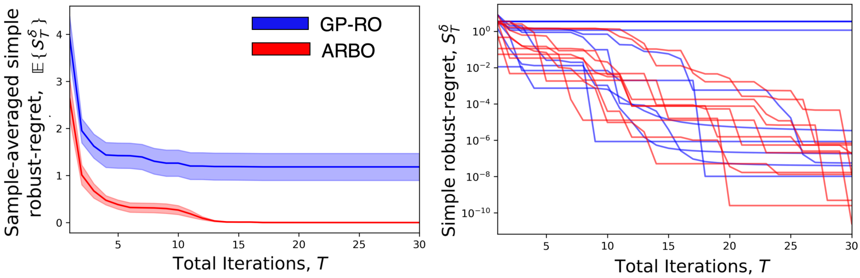
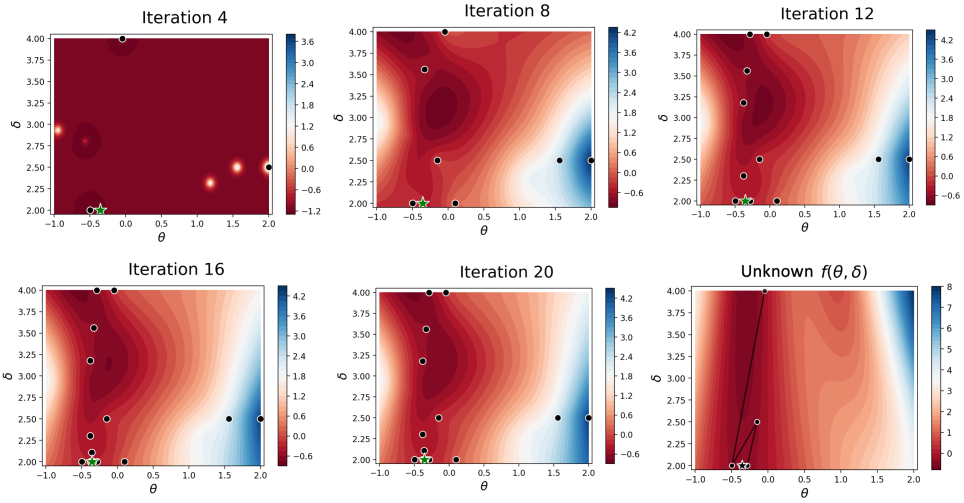


FIGURE 2 The simple robust-regret for ARBO and GP-RO. The runs are repeated for 10 times and the average simple robust regret is shown along with the 95% confidence intervals (left). Individual simple robust-regret sample paths for different uncertainty realizations (right).

To provide additional insights into the improved performance of ARBO over GP-RO, we plot the lower confidence bound contour plots for various iterations of a single run of ARBO in Figure 3. We see in the early iterations (top left) that the lower confidence bound gives low predictions in most of the $\Theta \times \Delta$ space since most of it is unexplored. As more of the samples suggested by ARBO have been incorporated, we see that the lower confidence bound is able to filter out regions of the space that are not likely to be near the global minimax solution (e.g., ARBO no longer samples near $\theta = 2$ after it sees large values there). In the later iterations (bottom middle), we see that the lower confidence bound provides a good approximation of the unknown true function (bottom right) in the region around the global minimax solution denoted with a star; however, it provides an optimistic prediction of $f(\theta, \delta)$ elsewhere. Since this optimistic prediction is still worse than our known, tested evaluation, we can adaptively exclude regions of our search space without wasting the computationally expensive samples. This highlights a fundamentally important point about the BO perspective: it is easier (i.e., fewer samples are needed) to find a globally optimal solution than building a globally accurate surrogate model. Additionally, this is the key missing component in GP-RO (which can be interpreted as ARBO with $\beta_t = 0$), as the mean predictions alone do not posses enough information about the quality of the predictions. This can lead to repeated evaluations at the same uninformative points.


Iteration 4

Iteration 8

Iteration 12

Iteration 16

Iteration 20

Unknown $f(\theta, \delta)$

FIGURE 3 Contour plots showing the convergence of ARBO. The contour plots show the lower confidence bound and the sequence of optimal points to be queried $\{\theta_t, \delta_t\}$. Lower plots show the current recommended point.

5.2 | Auto-tuning of NMPC for a Bioreactor

Having verified the practical implementation and important theoretical results of ARBO on an illustrative example, we now look to apply ARBO to the auto-tuning problem of an NMPC strategy for an uncertain bioreactor.

5.2.1 | High-fidelity process model

We consider a benchmark continuous bioreactor problem originally presented in [35]. The dynamics of the bioreactor can be modeled by a set of three nonlinear ordinary differential equations given by

$$\dot{X}(t) = -D(t)X(t) + \mu(t)X(t), \qquad X(0) = X_0, \tag{22a}$$

$$\dot{S}(t) = D(t)(S_f(t) - S(t)) - \frac{1}{Y_{X/S}}\mu(t)X(t), \qquad S(0) = S_0, \tag{22b}$$

$$\dot{P}(t) = -D(t)P(t) + (\alpha\mu(t) + \beta)X(t), \qquad P(0) = P_0, \tag{22c}$$

where $X(t)$, $S(t)$, and $P(t)$ denote the biomass, substrate, and product concentration (units of g/L), respectively, with initial conditions $X_0 = 0.3$ g/L, $S_0 = 0.2$ g/L, and $P_0 = 0$ g/L; $D(t)$ is the dilution rate (units of hr^{-1}); $S_f(t)$ is concentration of substrate in the feed (units of g/L); $Y_{X/S}$ is the cell-mass yield (units of g/g); $\mu(t)$ is the specific growth rate (units of hr^{-1}); and α and β are parameters related to the product yield. The specific growth rate is assumed to follow a modified Monod kinetic law that takes into account both substrate and product inhibition

$$\mu(t) = \frac{\mu_{\max}\left(1 - \frac{P(t)}{P_m}\right)S(t)}{K_m + S(t) + \frac{S^2(t)}{K_i}}. \tag{23}$$

We consider $\mu_{\max}$ and P_m to be the dominant time-invariant uncertainties in the plant simulator, as explored in several previous case studies [36, 37]. Here, we assume that $\delta = (\mu_{\max}, P_m) \in \Delta = [0.9, 1.4]$ hr^{-1} $\times$ $[1.3, 1.6]$ g/L. The other model parameters are assumed to be constant and are listed in Table 1.

TABLE 1 Known model parameters for the bioreactor case study.

Fixed Parameters	Values	Units
$Y_{X/S}$	0.2	g/g
α	2.5	g/g
β	0.7	hr^{-1}
K_m	1.2	g/L
K_i	20	g/L

The states of the bioreactor model (22) are given by $z(t) = (X(t), S(t), P(t))$, while $u(t) = (D(t), S_f(t))$ denote the two manipulated inputs. As such, we can write (22) in the following state-space representation

$$\dot{z}(t) = \mathcal{F}(z(t), u(t), \delta), \quad z(0) = z_0, \tag{24}$$

where $\mathcal{F} : \mathbb{R}^3 \times \mathbb{R}^2 \times \mathbb{R}^2 \to \mathbb{R}^3$ is a function that represents the dynamics of the bioreactor in (22) and (23). The control objective is to maximize the amount of product extracted from the bioreactor over a finite processing time of $t_f = 5$ hr, while satisfying minimum and maximum constraints on the biomass concentration. We can generally denote such state (or path) constraints as $\mathcal{G}(z(t), u(t)) \leq 0$, which reduce to the following in this case study

$$\mathcal{G}(z(t), u(t)) = [X_{LB} - X(t), X_{UB} - X(t)] \leq 0, \tag{25}$$

where $X_{LB} = 0.265$ g/L and $X_{UB} = 0.385$ g/L are the lower and upper bounds on the biomass concentration, respectively. The manipulated inputs must also satisfy hard input constraints $u(t) \in \mathbb{U} = [0.05, 1]$ hr^{-1} $\times$ $[10, 20]$ g/L. We assume that the manipulated inputs can be updated every $\delta t = 0.1$ hr in the simulation, such that there are a total of $N_{sim} = 50$ simulation steps to compute the behavior for each uncertainty realization.

5.2.2 | Control-relevant model and NMPC formulation

Although a dynamic model of the bioreactor is available here (i.e., we have access to $\mathcal{F}$), this is not always the case in practice. Furthermore, even when plant simulators are available, they may be excessively complex to use for MPC design and implementation. Thus, a more practical approach is often to construct a *control-relevant model* using system identification methods based on either plant simulation data or real plant data. Here, we used a residual neural network approach [38] that learns the "flow-map" function for time-invariant dynamic systems. In particular, this approach learns a transition function $\tilde{\mathcal{F}}(z_k, u_k)$ that can be applied recursively to predict the forward evolution of the states

$$z_{k+1} = \tilde{\mathcal{F}}(z_k, u_k) \tag{26}$$

from some initial condition z_0 given a future input sequence. The weights and bias parameters of the neural network representing $\tilde{\mathcal{F}}$ are trained such that the successor state at the next discrete time can be predicted given the current states and control inputs that have been determined from the plant simulator described in Section 5.2.1. We only collected simulation data for the nominal parameter values $\mu_{\max} = 1.0$ hr^{-1} and $P_m = 1.2$ g/L, though one could treat the unknown parameters as additional inputs to the model during training. A deep neural network with 4 layers, 15 nodes per layer, and ReLU activation functions was used to represent $\tilde{\mathcal{F}}$ in (26). The training was efficiently carried out using `Tensorflow` via the `Keras` API [39]. Standard best practices regarding the selection of batch size, weight/bias initialization, and stochastic gradient descent optimizer settings were utilized. As such, not only this case study considers time-invariant parametric uncertainty in the plant simulator, but also plant-model mismatch with respect to the control-relevant model used for the NMPC design at hand.

Given this control-relevant model and the control objective described in Section 5.2.1, we formulate the following NMPC problem

$$\begin{aligned} \min_{z_{i|k}, u_{i|k}, \varepsilon_{i|k}} \quad & \textstyle\sum_{i=0}^{N-1} \mathcal{L}(z_{i|k}, u_{i|k}) + \rho \| \varepsilon_{i|k} \|_1, && && (27) \\ \text{s.t.} \quad & z_{i+1|k} = \tilde{\mathcal{F}}(z_{i|k}, u_{i|k}), && \forall i = 0, \ldots, N-1, \\ & \mathcal{G}(z_{i|k}, u_{i|k}) + \theta \le \varepsilon_{i|k}, && \forall i = 0, \ldots, N-1, \\ & \varepsilon_{i|k} \ge 0, && \forall i = 0, \ldots, N-1, \\ & u_{i|k} \in \mathbb{U}, && \forall i = 0, \ldots, N-1, \\ & z_{0|k} = z(t_k), \end{aligned}$$

where N is the prediction horizon; $z_{i|k}$ and $u_{i|k}$ are the predicted state and control inputs i steps ahead of current time k; $z(t_k)$ is the measured state at time t_k (from the plant simulator); $\mathcal{L}(z(t), u(t)) = -VD(t)P(t)\delta t$ is the stage cost with reactor volume $V = 10$ L; $\varepsilon_{i|k}$ are slack variables for the state constraints; ρ is a large penalty weight for state constraint violations; and $\theta \in \Theta = [0, 0.125]^2$ are the tunable backoff parameters that can be selected to improve the inherent robustness guarantees in NMPC [40]. Note the stage cost is defined as the negative of the amount of product extracted from the bioreactor over each δt period – the negative arises since we want to maximize product, but formulated our problem in terms of a minimization. Let $u^{\star}_{0|k}(z(t_k), \theta)$ denote the first element of the solution to (27). We can then define the closed-loop system as the combination of (24) and the NMPC law

$$u(t_k) = u^{\star}_{0|k}(z(t_k), \theta), \tag{28}$$

where the control inputs are constant during each time interval $[t_k, t_{k+1}), \forall k \in \{0, \ldots, N_{sim} - 1\}$.

5.2.3 | Formulation of auto-tuning problem

Given the closed-loop simulation described in Section 5.2.2, we can now formulate the auto-tuning problem as selecting backoffs $\theta \in \Theta$ such that the worst-case mass of product (with respect to uncertainties $\delta \in \Delta$) is maximized while the biomass constraints (25) are not (significantly) violated. To formulate the auto-tuning problem as in (1), we must select $f(\theta, \delta)$ to be a weighted combination of productivity and constraint violations. Thus, we mathematically represent the overall control objective in terms of the closed-loop simulation outputs as follows

$$f(\theta, \delta) = \sum_{k=1}^{N_{sim}} \mathcal{L}(z(t_k), u(t_k)) + w \| [\mathcal{G}(z(t_k), u(t_k))]^+ \|_1, \tag{29}$$

where $[a]^+ = \max\{a, 0\}$ denotes the element-wise positive part operator and $w = 20$ is a weight parameter chosen to have a significant penalty associated with constraint violations.

5.2.4 | Results and performance comparison

Now, we look to compare the performance of ARBO and GP-RO to the previously described NMPC auto-tuning under uncertainty problem. We first examine the evolution of the recommended tuning $\theta_{t^\star}$ over 60 iterations of the ARBO and GP-RO algorithms, as shown in Figure 4. Similarly to the illustrative example, we use Latin Hypercube Sampling [41] to generate the first 15 samples in $\Theta \times \Delta$ to ensure sufficient initial coverage of the search space. ARBO appears to make more informed transitions among recommended tuning, which can be attributed to systematically trading off exploration and exploitation of the $\Theta \times \Delta$ space. On the other hand, GP-RO does not take into account the uncertainty in the GP predictions and, as a result, visits many possible optima as it attempts to learn a better surrogate for the closed-loop performance measure (29).

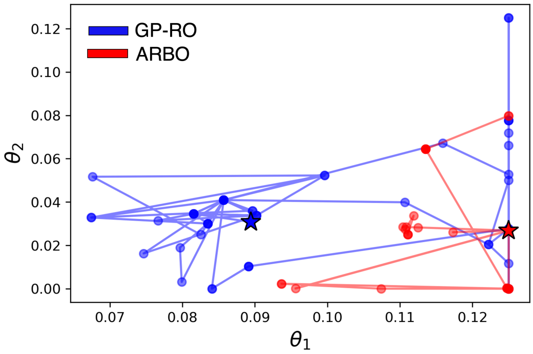


FIGURE 4 Recommended backoffs for the auto-tuning problem. The circles represent the points that the GP-RO (blue) and ARBO (red) algorithm suggested as "optimal" throughout the iterations. The stars denote the final point at iteration 60.

Next, we analyze the closed-loop performance (referred to as *Total Cost* below) for the final recommended point of the ARBO and GP-RO algorithms by evaluating $\{f(\theta_{t^\star}, \delta^{(i)})\}_{i=1}^{N_{samp}}$ at $N_{samp} = 100$ uniformly randomly sampled uncertainty values $\delta^{(i)} \in \Delta$. The resulting histograms are shown in Figure 5 – an additional comparison to the nominal setting in which the backoffs are set to zero (labeled *No-Backoff*) is provided to better highlight the advantages of auto-tuning. As seen from Figure 5, the estimated worst-case Total Cost for ARBO is significantly lower than that for the No-Backoff case and GP-RO. We also see that the characteristics of the Total Cost distribution are more favorable for ARBO, as it is clearly skewed toward lower values (resulting in a better mean performance as well). Figure 6 shows the closed-loop state profiles for the three cases considered in Figure 5. From Figure 6, we see that ARBO leads to significantly less violation of the upper biomass bound X_{UB}, while not sacrificing on the amount of product obtained throughout the processing time of interest. It is important to note that achieving this behavior

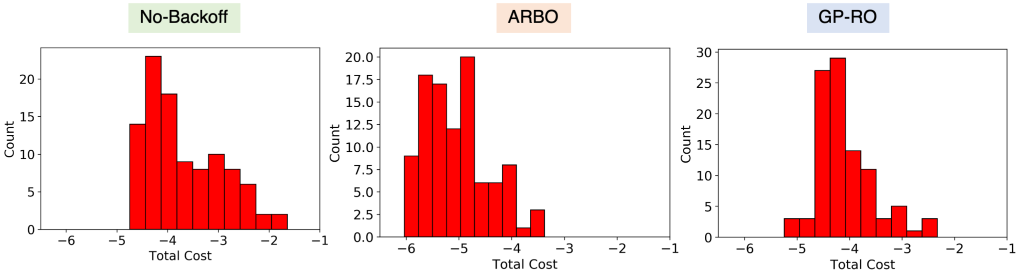


FIGURE 5 Histograms of the Total Cost given in (29) established under 100 realizations of the parametric uncertainties in the plant simulator for the cases of NMPC with No-Backoff, NMPC auto-tuned with ARBO, and NMPC auto-tuned with GP-RO.

required careful *simultaneous* tuning of θ_1 and θ_2, as we need to allow for some level of X_{LB} violation to obtain a large enough amount of product. It would be difficult for a practitioner to infer this careful balance without running an impractically large number of closed-loop simulations. In fact, ARBO was able to uncover this desired balance in an automated fashion using 60 total simulations; this is fewer than the 100 simulations we used to estimate the worst-case performance for the final recommended point.

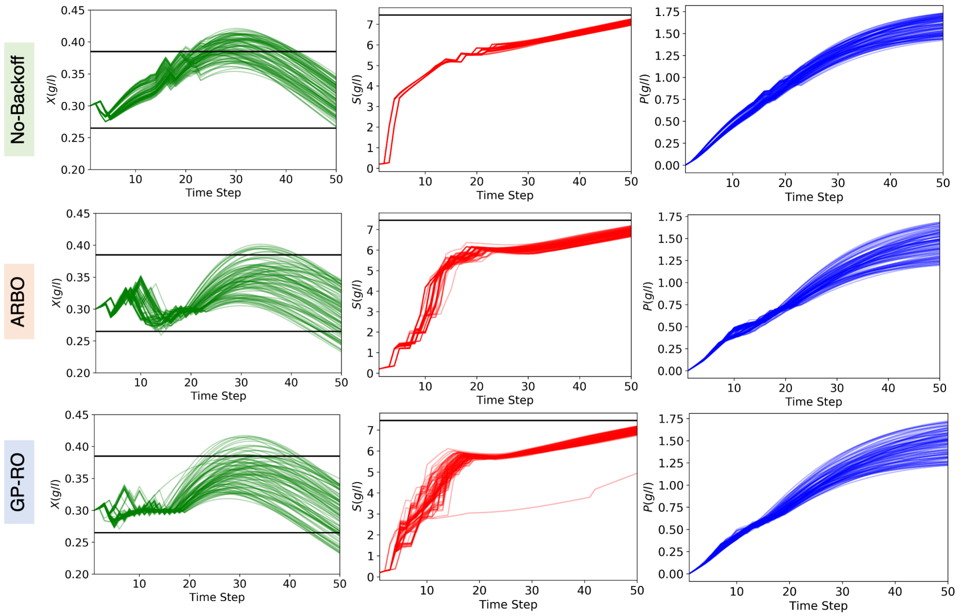


FIGURE 6 Closed-loop state profiles for 100 realizations of parametric uncertainties in the plant simulator for the cases of NMPC with No-Backoff, NMPC auto-tuned with ARBO, and NMPC auto-tuned with GP-RO.

6 | CONCLUSIONS AND FUTURE WORK

We have presented a robust BO method for auto-tuning of arbitrary complex control structures using a “high-fidelity” plant simulator with significant time-invariant uncertainties. The proposed ARBO method uses a probabilistic Gaussian process surrogate model to jointly describe the effect of the tuning parameters and plant model uncertainties on the closed-loop performance. The Gaussian process model allows for using an alternating confidence-bound procedure to simultaneously select the next candidate tuning and uncertainty realizations. As such, ARBO requires only one (expensive) closed-loop simulation in each iteration, as compared to alternative robust BO approaches to auto-tuning that rely on vastly more closed-loop simulations in each iteration. Our results on two simulation case studies demonstrate the advantages of the confidence bound-based procedure of ARBO in systematically realizing a tradeoff between the exploration and exploitation of the design-uncertainty space relative to Gaussian process surrogate-based robust optimization that lacks an exploration mechanism. Our future work will mainly focus on using more complex non-Gaussian noise models for robust BO, as well as extending the proposed auto-tuning approach to handle multi-objective performance measures.

references

[1] Paulson JA, Mesbah A. Shaping the closed-loop behavior of nonlinear systems under probabilistic uncertainty using arbitrary polynomial chaos. In: Proceedings of the IEEE Conference on Decision and Control; 2018. p. 6307–6313.

[2] Garriga JL, Soroush M. Model predictive control tuning methods: A review. Industrial & Engineering Chemistry Research 2010;49(8):3505–3515.

[3] Tran QN, Scholten J, Ozkan L, Backx A. A model-free approach for auto-tuning of model predictive control. IFAC Proceedings Volumes 2014;47(3):2189–2194.

[4] Neumann-Brosig M, Marco A, Schwarzmann D, Trimpe S. Data-efficient autotuning with Bayesian optimization: An industrial control study. IEEE Transactions on Control Systems Technology 2019;28(3):730–740.

[5] Paulson JA, Mesbah A. Data-Driven scenario optimization for automated controller tuning with probabilistic performance guarantees. IEEE Control Systems Letters 2020;5(4):1477–1482.

[6] Zhu M, Bemporad A, Piga D. Preference-based MPC calibration. arXiv preprint arXiv:200311294 2020;.

[7] Snoek J, Larochelle H, Adams RP. Practical Bayesian optimization of machine learning algorithms. Advances in Neural Information Processing Systems 2012;25.

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

[9] Piga D, Forgione M, Formentin S, Bemporad A. Performance-oriented model learning for data-driven MPC design. IEEE Control Systems Letters 2019;3(3):577–582.

[10] Lu Q, Kumar R, Zavala VM. MPC controller tuning using Bayesian optimization techniques. arXiv preprint arXiv:200914175 2020;.

[11] Sorourifar F, Makrygiorgos G, Mesbah A, Paulson JA. A data-driven automatic tuning method for MPC under uncertainty using constrained Bayesian optimization. Venice; 2021. p. 1–6.

[12] Fiducioso M, Curi S, Schumacher B, Gwerder M, Krause A. Safe contextual Bayesian optimization for sustainable room temperature PID control tuning. arXiv preprint arXiv:190612086 2019;.

[13] Khosravi M, Behrunani V, Myszkorowski P, Smith RS, Rupenyan A, Lygeros J. Performance-driven cascade controller tuning with Bayesian optimization. IEEE Transactions on Industrial Electronics 2021;.

[14] Rasmussen CE, Williams CKI. Gaussian processes for machine learning. Cambridge, MA: MIT Press; 2006.

[15] Paulson JA, Shao K, Mesbah A. Probabilistically Robust Bayesian Optimization for Data-Driven Design of Arbitrary Controllers with Gaussian Process Emulators. In: Proceedings of the IEEE Conference on Decision and Control; 2021. p. Accepted.

[16] Edgar TF, Pistikopoulos EN. Smart manufacturing and energy systems. Computers & Chemical Engineering 2018;114:130–144.

[17] Bogunovic I, Scarlett J, Jegelka S, Cevher V. Adversarially robust optimization with Gaussian processes. In: Proceedings of the 32nd International Conference on Neural Information Processing Systems; 2018. p. 5765–5775.

[18] Sudret B. Global sensitivity analysis using polynomial chaos expansions. Reliability Engineering & System Safety 2008;93(7):964–979.

[19] Paulson JA, Martin-Casas M, Mesbah A. Fast uncertainty quantification for dynamic flux balance analysis using non-smooth polynomial chaos expansions. PLoS Computational Biology 2019;15(8):e1007308.

[20] Makrygiorgos G, Maggioni GM, Mesbah A. Surrogate modeling for fast uncertainty quantification: Application to 2D population balance models. Computers & Chemical Engineering 2020;138:106814.

[21] Marzat J, Walter E, Piet-Lahanier H. A new expected-improvement algorithm for continuous minimax optimization. Journal of Global Optimization 2016;64(4):785–802.

[22] Mukhopadhyay DM, Balitanas MO, Farkhod A, Jeon SH, Bhattacharyya D. Genetic algorithm: A tutorial review. International Journal of Grid and Distributed Computing 2009;2(3):25–32.

[23] Srinivas N, Krause A, Kakade SM, Seeger M. Gaussian process optimization in the bandit setting: No regret and experimental design. In: Proceedings of the International Conference on Machine Learning; 2015. p. 2171–2180.

[24] Bradford E, Schweidtmann AM, Lapkin A. Efficient multiobjective optimization employing Gaussian processes, spectral sampling and a genetic algorithm. Journal of Global Optimization 2018;71(2):407–438.

[25] Wabersich KP, Toussaint M. Automatic testing and minimax optimization of system parameters for best worst-case performance. In: IEEE/RSJ International Conference on Intelligent Robots and Systems IEEE; 2015. p. 5533–5539.

[26] Dani V, Hayes TP, Kakade SM. Stochastic linear optimization under bandit feedback 2008;.

[27] Kandasamy K, Schneider J, Póczos B. High dimensional Bayesian optimisation and bandits via additive models. In: Proceedings of the International Conference on Machine Learning; 2015. p. 295–304.

[28] Wächter A, Biegler LT. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. Mathematical programming 2006;106(1):25–57.

[29] Finkel D. DIRECT optimization algorithm user guide. North Carolina State University. Center for Research in Scientific Computation; 2003.

[30] A Gaussian processes framework in python;. `https://sheffieldml.github.io/GPy/`.

[31] Jin C, Netrapalli P, Jordan M. What is local optimality in nonconvex-nonconcave minimax optimization? In: Proceedings of the International Conference on Machine Learning; 2020. p. 4880–4889.

[32] Cartis C, Fiala J, Marteau B, Roberts L. Improving the flexibility and robustness of model-based derivative-free optimization solvers. ACM Transactions on Mathematical Software (TOMS) 2019;45(3):1–41.

[33] Powell MJ. The BOBYQA algorithm for bound constrained optimization without derivatives. Cambridge NA Report NA2009/06, University of Cambridge, Cambridge 2009;p. 26–46.

[34] Bull AD. Convergence rates of efficient global optimization algorithms. Journal of Machine Learning Research 2011;12(10).

[35] Agrawal P, Koshy G, Ramseier M. An algorithm for operating a fed-batch fermentor at optimum specific-growth rate. Biotechnology and Bioengineering 1989;33(1):115–125.

[36] Henson MA, Seborg DE. Nonlinear control strategies for continuous fermenters. Chemical Engineering Science 1992;47(4):821–835.

[37] Paulson JA, Martin-Casas M, Mesbah A. Input design for online fault diagnosis of nonlinear systems with stochastic uncertainty. Industrial & Engineering Chemistry Research 2017;56(34):9593–9605.

[38] Qin T, Wu K, Xiu D. Data driven governing equations approximation using deep neural networks. Journal of Computational Physics 2019;395:620–635.

[39] Chollet F, et al., Keras. GitHub; 2015. `https://github.com/fchollet/keras`.

[40] Paulson JA, Mesbah A. Nonlinear model predictive control with explicit backoffs for stochastic systems under arbitrary uncertainty. IFAC-PapersOnLine 2018;51(20):523–534.

[41] Loh WL. On Latin hypercube sampling. The annals of statistics 1996;24(5):2058–2080.