2021 60th IEEE Conference on Decision and Control (CDC)
December 13-15, 2021. Austin, Texas

Probabilistically Robust Bayesian Optimization for Data-Driven Design of Arbitrary Controllers with Gaussian Process Emulators

Joel A. Paulson†, Ketong Shao†, and Ali Mesbah

***Abstract*— A fundamental challenge in control applications stems from the lack of a sufficiently detailed system model that can be used for systematic controller design and tuning under uncertainty. This paper presents a fully data-driven robust controller design approach based on any finite set of closed-loop performance measures. We first present a method for emulating the unknown plant dynamics using a Gaussian process (GP) model learned from input-output data. By running closed-loop simulations under realizations of the GP model using posterior sampling, the impact of system uncertainties on the closed-loop performance measures is quantified. We then formulate the robust controller design problem as a constrained Bayesian optimization (CBO) problem defined in terms of the GP-emulated performance measures. To ensure a sufficient number of samples are used to estimate the worst-case performance measures, we derive a bound on the joint probability of violation that is independent of the number or probability distribution of the uncertainties. The advantages of the proposed approach are illustrated on a benchmark control problem, which demonstrates guaranteed probabilistic estimates on the worst-case performance measures are provided at every iteration of CBO.**

I. Introduction

Systematic design and validation of arbitrary control structures is generally a challenging task. Controller tuning typically relies on extensive trial-and-error simulations or experimentation [1]. Design parameters of a controller often affect multiple closed-loop performance measures in non-convex and non-smooth ways, implying closed-form expressions relating these parameters to performance are not readily available. In addition, the effects of system uncertainties must be accounted for in controller tuning to ensure robust and satisfactory closed-loop performance is attained in the face of uncertainties [2].

To address these challenges, there has been a growing interest in applying *data-driven* optimization methods to automate the tuning procedure. Bayesian optimization (BO) [3] has been used for solving black-box controller tuning optimization problems [4]–[8]. In particular, constrained BO (CBO) [9] has been shown to effectively deal with the black-box, expensive-to-evaluate and noisy nature of closed-loop performance measures, while explicitly accounting for (nonlinear) constraints on design specifications [10].

This work was supported by the National Science Foundation under Grant 1839527.

K. Shao and A. Mesbah are with the Department of Chemical and Biomolecular Engineering at the University of California, Berkeley, CA 94720, USA. {ketong_shao,mesbah}@berkeley.edu

J. A. Paulson is with the Department of Chemical and Biomolecular Engineering at The Ohio State University, Columbus, OH 43210, USA. paulson.82@osu.edu

†J. Paulson and K. Shao contributed equally to this work.

Nonetheless, these works do not provide formal guarantees on the quality of the resulting "optimal" set of tuning parameters. Solution guarantees are especially useful when designing controllers, e.g., for safety-critical applications in which it is imperative to ensure the closed-loop system does not violate constraints, or other performance requirements, given the best available model of uncertainty. In [11], we presented an approach for providing probabilistic guarantees on the optimally designed controller based on recent theoretical results for non-convex scenario optimization [12]. The proposed approach involved two key stages: (i) first generating a set of "good" candidate designs using CBO and (ii) selecting the best design from this set using non-convex scenario optimization that provides a *distribution-independent* bound on the probability of constraint violation.

In our previous work [11], we assumed that a sufficiently detailed system model is available for controller tuning purposes. However, construction of such a simulator may be difficult in practice. Thus, the first contribution of this work is to develop a fully data-driven *robust* controller design approach based on any finite set of closed-loop performance measures. This is achieved by constructing a Gaussian process (GP) emulator of the system, i.e., a probabilistic model for the state transition function, that can be applied recursively to simulate the closed-loop state distribution over time. Yet, an important challenge with the GP representation of the dynamics is its infinite-dimensional nature, meaning it is not possible to exactly compute the worst-case objective or constraint violations appearing in a robust controller design problem. As such, our second contribution is the development of a novel joint probabilistic performance level, which simultaneously estimates the worst-case performance measures using a set of independent random samples that can be generated using a GP posterior sampling method [13]. We establish a bound on the number of samples needed to ensure the worst-case performance estimates jointly achieve a pre-specified probability of accuracy. The derived bound, which is an extension of [14], is completely independent of the number of uncertainties as well as their underlying probability distribution. Using this derived bound, along with the posterior GP sampling method, worst-case estimates of closed-loop performance measures are incorporated into the CBO framework to *a priori* ensure the worst-case estimates satisfy desired probabilistic properties at every iteration. We refer to the proposed approach as probabilistically robust BO (PRBO), as it alleviates the need for the post verification of the optimally designed controller needed in [11]. Advantages of PRBO are demonstrated on a benchmark problem.

II. Problem Statement

Consider uncertain, discrete-time systems of the form

$$\mathbf{x}^+ = \mathbf{f}(\mathbf{x}, \mathbf{u}, \mathbf{d}), \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^{n_x}$ is the current system state, $\mathbf{x}^+ \in \mathbb{R}^{n_x}$ is the successor state, $\mathbf{u} \in \mathbb{R}^{n_u}$ is the control input, and $\mathbf{d} \in \mathbb{R}^{n_d}$ is an unknown disturbance. The dynamics of the system, defined by the function $\mathbf{f} : \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \times \mathbb{R}^{n_d} \to \mathbb{R}^{n_x}$, are *unknown* and are to be learned from data. We assume that noisy measurements of the state are available

$$\mathbf{y} = \mathbf{f}(\mathbf{x}, \mathbf{u}, \mathbf{d}) + \mathbf{v}, \tag{2}$$

where $\mathbf{y} \in \mathbb{R}^{n_x}$ is the measurement of the successor state perturbed by Gaussian noise $\mathbf{v} \sim \mathcal{N}(\mathbf{0}, \mathbf{\Sigma_v})$ with zero mean and diagonal covariance matrix $\mathbf{\Sigma_v} = \text{diag}(\sigma^2_{v_1}, \ldots, \sigma^2_{v_{n_x}})$.

Assumption 1: The state $\mathbf{x}$ and disturbance $\mathbf{d}$ are measurable at the current time step for all times. ◁

Remark 1: The measured state assumption can be satisfied by defining the state vector in terms of past inputs and outputs [15]. In addition, $\mathbf{d}$ can be interpreted as a measured time-varying disturbance that is unknown in the future, which is included for the sake of generality. The effect of all other unmeasured disturbances is lumped into $\mathbf{v}$ in (2).

For notational simplicity, we define the concatenated vector $\mathbf{z} = (\mathbf{x}, \mathbf{u}, \mathbf{d}) \in \mathbb{R}^{n_z}$ with $n_z = n_x + n_u + n_d$. It is assumed a finite number M of noisy measurements are available from (2). This data can be represented by

$$\mathbf{Z} = [\mathbf{z}^{(1)}, \ldots, \mathbf{z}^{(M)}]^\top \in \mathbb{R}^{M \times n_z}, \tag{3a}$$

$$\mathbf{Y} = [\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(M)}]^\top \in \mathbb{R}^{M \times n_y}, \tag{3b}$$

where $\mathbf{z}^{(i)}$ and $\mathbf{y}^{(i)}$ denote the corresponding ith input and output data point, respectively. Thus, we consider multiple sources of uncertainty, including the initial condition $\mathbf{x}_0$, the future disturbance sequence $\{\mathbf{d}_0, \mathbf{d}_1, \ldots\}$, and the unknown dynamics $\mathbf{f}(\mathbf{x}, \mathbf{u}, \mathbf{d})$. We look to train a probabilistic model using the available training data $\mathcal{D} = \{\mathbf{Z}, \mathbf{Y}\}$ from (3). Gaussian process (GP) regression models are one such example, which generalizes the Gaussian probability distribution to distributions over functions [16]. Let $\mathbf{f}|\mathcal{D}$ denote the random function with posterior distribution $p(\mathbf{f}|\mathcal{D})$; Section III discusses how such a GP state-space model can be learned.

Given a controller $\mathbf{u} = \kappa(\mathbf{x}, \mathbf{d}; \theta)$, parametrized by *design variables* $\theta \in \Theta$ in a bounded set $\Theta \subset \mathbb{R}^{n_\theta}$, and substituting this expression into (1), we arrive at the closed-loop system

$$\mathbf{x}_{k+1} = \mathbf{f}(\mathbf{x}_k, \kappa(\mathbf{x}_k, \mathbf{d}_k; \theta), \mathbf{d}_k), \tag{4}$$

where $\mathbf{x}_k$ and $\mathbf{d}_k$ are the state and disturbance at time step k, respectively. The only restriction on the control law $\kappa(\cdot)$ is that it is time-invariant and defined for every $(\mathbf{x}, \mathbf{d}; \theta) \in \mathbb{R}^{n_x} \times \mathbb{R}^{n_d} \times \Theta$; it can otherwise be a non-convex and/or implicitly-defined function. Note that a single trajectory of the closed-loop system over a finite number of time steps T is completely determined by the choice of θ and the realization of the uncertainties $\mathbf{w} = \{\mathbf{x}_0, \mathbf{d}_0, \ldots, \mathbf{d}_{T-1}, \mathbf{f}|\mathcal{D}\}$.

Assumption 2: The uncertain variables $\mathbf{w} \in \mathcal{W}$ follow a probability distribution $\Pr_{\mathcal{W}}$ defined over a bounded support $\mathcal{W}$, from which independent and identically distributed (i.i.d.) samples can be drawn. ◁

We now define the robust controller design problem as

$$\min_{\theta \in \Theta} \max_{\mathbf{w} \in \mathcal{W}} \ \phi_1(\theta, \mathbf{w}), \tag{5a}$$

$$\text{s.t. } \phi_i(\theta, \mathbf{w}) \leq 0, \ \forall \mathbf{w} \in \mathcal{W}, \ \forall i \in \{2, \ldots, P\}, \tag{5b}$$

where $\phi_1 : \Theta \times \mathcal{W} \to (-\infty, \infty)$ is a measurable function that defines the controller objective, $\phi_i : \Theta \times \mathcal{W} \to \mathbb{R}$ encodes the i^{th} performance constraint for a given choice of design variables and uncertainty realization, and P is the total number of performance functions. The functions $\{\phi_i(\cdot)\}_{i=1}^P$ will typically be related to, e.g., tracking error, violation of critical safety or quality constraints, or economic costs.

The robust design problem (5) falls into the class of *semi-infinite programming* problems, which are challenging to solve, especially when $\{\phi_i(\cdot, \mathbf{w})\}_{i=1}^P$ are non-convex for any $\mathbf{w} \in \mathcal{W}$. Even when $\theta \in \Theta$ is fixed, we cannot exactly compute worst-case performance and constraint violations

$$\Phi_i(\theta) = \max_{\mathbf{w} \in \mathcal{W}} \phi_i(\theta, \mathbf{w}), \quad \forall i = 1, \ldots, P, \tag{6}$$

because this requires the exact solution to generally non-convex maximization problems [1]. Although a variety of guaranteed reachability methods, such as Taylor models [17], are available, they not only require additional assumptions about the knowledge and structure of the uncertainty $\mathbf{w}$ and the functions $\{\phi_i(\theta, \mathbf{w})\}_{i=1}^P$, but may also produce highly conservative bounds. We look to overcome these challenges by employing *randomized algorithms* [14] that approximate the worst-case measures (6) via random sampling

$$\Phi_i(\theta) \approx \hat{\Phi}^i_N(\theta) = \max_{j=1,\ldots,N} \phi_i(\theta, \mathbf{w}^{(j)}), \tag{7}$$

where $\mathbf{w}^{(1)}, \mathbf{w}^{(2)}, \ldots, \mathbf{w}^{(N)}$ are N i.i.d. samples from $\Pr_{\mathcal{W}}$. Using this randomized approach, we can cast (5) as a simulation optimization (SO) problem that involves an "outer" optimization over θ and an "inner" stochastic simulation to approximate $\{\Phi_i(\theta)\}_{i=1}^P$. To this end, two challenges must be addressed: (i) how can uncertainty samples $\{\mathbf{w}^{(i)}\}_{i=1}^N$ be generated such that they include random multivariate functions $\mathbf{f}|\mathcal{D}$?; and (ii) how should N be selected to provide joint probabilistic guarantees on the accuracy of $\{\hat{\Phi}^i_N(\theta)\}_{i=1}^P$?

We address the first challenge by using a posterior sampling approach for GPs (Section III). We then address the second challenge by deriving a guaranteed bound on N that is independent of the number of uncertain parameters, the size of the support set $\mathcal{W}$, and the probability distribution $\Pr_{\mathcal{W}}$ (Section IV). Lastly, these reliable scenario estimates are incorporated into a constrained SO framework for robust controller design (Section V).

III. Scenario Sampling for Dynamic Gaussian Process Emulators

A. Gaussian process regression

Here, we use GPs to learn unknown latent functions $f : \mathbb{R}^{n_x} \to \mathbb{R}$ from noisy measurements $y = f(\mathbf{z}) + v$, where

$v \sim \mathcal{N}(0, \sigma_v^2)$ is a zero-mean Gaussian noise with variance σ_v^2. Let a function $f(\mathbf{z})$ be distributed as a GP with mean function $m(\mathbf{z})$ and covariance function $k(\mathbf{z}, \mathbf{z}')$

$$f(\mathbf{z}) \sim \mathcal{GP}(m(\mathbf{z}), k(\mathbf{z}, \mathbf{z}')). \tag{8}$$

The GP prior distribution can then be defined by an initial choice of $m(\mathbf{z})$ and $k(\mathbf{z}, \mathbf{z}')$, which depend generally on a set of hyperparameters $\mathbf{\Psi}_c$, i.e., $m(\mathbf{z}|\mathbf{\Psi}_c)$ and $k(\mathbf{z}, \mathbf{z}'|\mathbf{\Psi}_c)$. Often the mean is specified to be zero $m(\mathbf{z}|\mathbf{\Psi}_c) = 0$ [18], [19], which can be achieved by normalizing the data before training. When the function is modeled as a member of the space of smooth functions C^∞, the squared exponential (SE) covariance function can be selected

$$k(\mathbf{z}, \mathbf{z}'|\mathbf{\Psi}_c) = \zeta^2 \exp\left(-\frac{1}{2}(\mathbf{z} - \mathbf{z}')^\top \mathbf{\Lambda}^{-2}(\mathbf{z} - \mathbf{z}')\right), \tag{9}$$

where $\mathbf{z}, \mathbf{z}' \in \mathbb{R}^{n_z}$ are arbitrary inputs, ζ^2 is the covariance magnitude, and $\mathbf{\Lambda} = \text{diag}(\lambda_1, \ldots, \lambda_{n_z})$ is a diagonal scaling matrix. Under the assumption $m(\mathbf{z}) = 0$ and an SE covariance function in (9), the hyperparameters consist of $\mathbf{\Psi}_c = [\zeta, \lambda_1, \ldots, \lambda_{n_z}]^\top$. Due to the additive property of Gaussian distributions, we can derive a GP for the measurement

$$y \sim \mathcal{GP}(0, k(\mathbf{z}, \mathbf{z}'|\mathbf{\Psi}_c) + \sigma_v^2 \delta_{\mathbf{z}\mathbf{z}'}), \tag{10}$$

where $\delta_{\mathbf{z}\mathbf{z}'}$ is the Kronecker delta function that is equal to 1 when $\mathbf{z} = \mathbf{z}'$ and zero otherwise. Whenever the noise variance σ_v^2 is unknown, it can be included in the joint set of hyperparameters of the prior denoted by $\mathbf{\Psi} = [\mathbf{\Psi}_c^\top, \sigma_v^2]^\top$.

Once the hyperparameters are estimated, the combined data $\mathcal{D}$ can be used to infer the posterior distribution $f(\mathbf{z})|\mathcal{D}$. The training data and $f(\cdot)$ at any arbitrary test input $\mathbf{z}$ must be jointly Gaussian. Hence, we can use the conditional distribution rule for multivariate normal distributions to derive

$$f(\mathbf{z})|\mathcal{D} \sim \mathcal{N}(\mu_f(\mathbf{z}; \mathcal{D}), \sigma_f^2(\mathbf{z}; \mathcal{D})), \tag{11}$$

where

$$\mu_f(\mathbf{z}; \mathcal{D}) = \mathbf{k}^\top(\mathbf{z}) \mathbf{\Sigma}_\mathbf{Y}^{-1} \mathbf{Y}, \tag{12a}$$

$$\sigma_f^2(\mathbf{z}; \mathcal{D}) = k(\mathbf{z}, \mathbf{z}) - \mathbf{k}^\top(\mathbf{z}) \mathbf{\Sigma}_\mathbf{Y}^{-1} \mathbf{k}(\mathbf{z}), \tag{12b}$$

and $\mathbf{k}(\mathbf{z}) = [k(\mathbf{z}, \mathbf{z}^{(1)}), \ldots, k(\mathbf{z}, \mathbf{z}^{(M)})]^\top$. The posterior mean $\mu_f(\mathbf{z}; \mathcal{D})$ represents our best prediction of the unknown function $f(\mathbf{z})$ at any $\mathbf{z}$, whereas the posterior variance $\sigma_f^2(\mathbf{z}; \mathcal{D})$ is a measure of uncertainty in this prediction.

A useful feature of GPs is that they can be recursively updated in a straightforward manner when new data becomes available. Let the new input and output be denoted by $\mathbf{z}^+$ and y^+, respectively, and let $\mathcal{D}^+ = (\mathcal{D}, (\mathbf{z}^+, y^+))$. Then, the updated mean and variance function are

$$\mu_f^+(\mathbf{z}; \mathcal{D}^+) = \mathbf{k}^{+\top}(\mathbf{z}) \mathbf{\Sigma}_\mathbf{Y}^{+-1} \mathbf{Y}^+, \tag{13a}$$

$$\sigma_f^{2+}(\mathbf{z}; \mathcal{D}^+) = k^+(\mathbf{z}, \mathbf{z}) - \mathbf{k}^{+\top}(\mathbf{z}) \mathbf{\Sigma}_\mathbf{Y}^{+-1} \mathbf{k}^+(\mathbf{z}), \tag{13b}$$

where

$$\mathbf{k}^+(\mathbf{z}) = [\mathbf{k}^\top(\mathbf{z}), k(\mathbf{z}, \mathbf{z}^+)]^\top, \tag{14a}$$

$$\mathbf{Y}^+ = [\mathbf{Y}^\top, y^+]^\top, \quad \mathbf{Z}^+ = [\mathbf{Z}^\top, \mathbf{z}^{+\top}]^\top, \tag{14b}$$

$$\mathbf{\Sigma}_\mathbf{Y}^{+-1} = \begin{bmatrix} \mathbf{\Sigma}_\mathbf{Y} & \mathbf{k}^\top(\mathbf{z}^+) \\ \mathbf{k}(\mathbf{z}^+) & k(\mathbf{z}^+, \mathbf{z}^+) + \sigma_v^2 \end{bmatrix}^{-1}. \tag{14c}$$

B. State-space model construction

The GP regression in Section III-A can be used to learn unknown dynamic models of the form (1) using the system measurements (2), as long as separate (independent) GPs are constructed for each output dimension of $\mathbf{f}(\cdot)$, i.e.,

$$\mathbf{f}(\mathbf{x}, \mathbf{u}, \mathbf{d}) = \mathbf{f}(\mathbf{z}) = [f_1(\mathbf{z}), \ldots, f_{n_x}(\mathbf{z})]^\top. \tag{15}$$

To build a GP for each function $f_i(\mathbf{z})$ for $i = 1, \ldots, n_x$, we use a subset of the measurements $\mathbf{Y}_i = [y_i^{(1)}, \ldots, y_i^{(M)}]^\top$, with $\mathbf{Y} = [\mathbf{Y}_1, \ldots, \mathbf{Y}_{n_x}]$ representing the full observation matrix. Following the notation introduced in (11) and (12), the posterior Gaussian distribution of the multi-input, multi-output $\mathbf{f}(\cdot)$ at any test input $\mathbf{z}$ is specified as [20]

$$\mathbf{f}(\mathbf{z})|\mathcal{D} \sim \mathcal{N}(\boldsymbol{\mu}_\mathbf{f}(\mathbf{z}; \mathcal{D}), \mathbf{\Sigma}_\mathbf{f}(\mathbf{z}; \mathcal{D})), \tag{16}$$

where

$$\boldsymbol{\mu}_\mathbf{f}(\mathbf{z}; \mathcal{D}) = [\mu_{f_1}(\mathbf{z}; \mathcal{D}_1), \ldots, \mu_{f_{n_x}}(\mathbf{z}; \mathcal{D}_{n_x})]^\top, \tag{17a}$$

$$\mathbf{\Sigma}_\mathbf{f}(\mathbf{z}; \mathcal{D}) = \text{diag}(\sigma_{f_1}^2(\mathbf{z}; \mathcal{D}_1), \ldots, \sigma_{f_{n_x}}^2(\mathbf{z}; \mathcal{D}_{n_x})), \tag{17b}$$

and $\mu_{f_i}(\mathbf{z}; \mathcal{D}_i)$ and $\sigma_{f_i}^2(\mathbf{z}; \mathcal{D}_i)$ denote the posterior GP mean and variance functions for $f_i(\cdot)$, respectively, built from the datasets $\mathcal{D}_i = \{\mathbf{Z}, \mathbf{Y}_i\}$ for all $i = 1, \ldots, n_x$.

Expression (16) can be interpreted as a fully data-driven "emulator" of the process that can be used for closed-loop simulation. If a prior system model is available, it can be incorporated by using the GP to describe the models' prediction error, as discussed, e.g., in [21]. As noted in Remark 1, we can apply the proposed emulation procedure to input-output models through a proper choice of state definition. Although larger state dimensions will increase the complexity of GP training, the recently developed GPyTorch [22] package is able to train GPs with $M > 10^6$ data points.

C. Monte Carlo posterior sampling

Since GPs are distributions over functions, a random sample of a GP must yield a *deterministic* function. Generating a full Monte Carlo (MC) sample requires sampling an infinite-dimensional stochastic process, which cannot be done exactly and thus one must resort to, e.g., spectral sampling [23]. On the other hand, if the GP MC sample only needs to be known at a *finite* number of points, we can derive an *exact i.i.d. sampling procedure*. The latter case corresponds to dynamic state-space models over a finite-time horizon. As such, we follow a similar procedure to that introduced in [13] to generate the corresponding closed-loop state and input sequences, which is summarized in Algorithm 1.

In Algorithm 1, we start with a random sample of the initial state $\mathbf{x}_0^{(i)}$ and the disturbance $\mathbf{d}_0^{(i)}$. Given these samples, the control law can be evaluated to obtain $\mathbf{u}_0^{(i)}$, which allows us to define the posterior distribution $p(\mathbf{x}_1|\mathbf{z}_0^{(i)}, \mathcal{D})$ in terms of the GP (16) evaluated at $\mathbf{z}_0^{(i)} = (\mathbf{x}_0^{(i)}, \mathbf{u}_0^{(i)}, \mathbf{d}_{k-1}^{(i)})$. Since the posterior is evaluated at a single test point $\mathbf{z}_0^{(i)}$, a realization $\mathbf{x}_1^{(i)}$ can be obtained by sampling from the aforementioned normal distribution. To obtain a realization of $\mathbf{x}_2^{(i)}$, however, we must now condition on the realization

Algorithm 1 Trajectory sampling for GP state-space models.

Input: The probability distribution $\Pr_{\mathcal{W}}$, the initial GP mean $\boldsymbol{\mu}_{\mathbf{f}}(\mathbf{z};\mathcal{D})$ and covariance $\boldsymbol{\Sigma}_{\mathbf{f}}(\mathbf{z};\mathcal{D})$, the dataset $\mathcal{D}$, the controller $\kappa(\cdot)$, and the number of time steps T.

Initialize: Draw $\mathbf{x}_0^{(i)}$ according to $\Pr_{\mathcal{W}}$ and set $\mathcal{D}_0 = \mathcal{D}$.

1: **for** $k = 1$ to T **do**
2: Draw $\mathbf{d}_{k-1}^{(i)}$ according to $\Pr_{\mathcal{W}}$.
3: Evaluate $\mathbf{u}_{k-1}^{(i)} = \kappa(\mathbf{x}_{k-1}^{(i)}, \mathbf{d}_{k-1}^{(i)}; \theta)$.
4: Concatenate $\mathbf{z}_{k-1}^{(i)} = (\mathbf{x}_{k-1}^{(i)}, \mathbf{u}_{k-1}^{(i)}, \mathbf{d}_{k-1}^{(i)})$.
5: Draw $\mathbf{x}_k^{(i)} \sim \mathcal{N}(\boldsymbol{\mu}_{\mathbf{f}}(\mathbf{z}_{k-1}^{(i)}; \mathcal{D}_{k-1}), \boldsymbol{\Sigma}_{\mathbf{f}}(\mathbf{z}_{k-1}^{(i)}; \mathcal{D}_{k-1}))$.
6: Update dataset $\mathcal{D}_k = \{[\mathbf{Z}^\top, \mathbf{z}_{k-1}^{(i)\top}]^\top, [\mathbf{Y}^\top, \mathbf{x}_k^{(i)\top}]^\top\}$.

Output: MC sample of the state and control sequences $\mathbf{X}^{(i)} = [\mathbf{x}_0^{(i)}, \ldots, \mathbf{x}_T^{(i)}]^\top$ and $\mathbf{U}^{(i)} = [\mathbf{u}_0^{(i)}, \ldots, \mathbf{u}_{T-1}^{(i)}]^\top$.

$\mathbf{x}_1^{(i)}$ because this is part of the sampled function path. That is, if we happen to return to the input value $\mathbf{z}_t^{(i)} = \mathbf{z}_0^{(i)}$ at a later time point $t > 0$, this GP sample should produce the same state $\mathbf{x}_{t+1}^{(i)} = \mathbf{x}_1^{(i)}$. This can be achieved by treating $\mathbf{x}_1^{(i)}$ as a new perfectly measured training point. This process is thus repeated until the final time step T is reached, as described in Algorithm 1 via $\mathcal{D}_k$ in lines 5 and 6, which is recursively updated using the sampled state at each step. The recursive operations for a scalar GP function are shown in (13), which can be straightforwardly extended to vector functions using the procedure outlined in Section III-B.

Remark 2: The GP model for $\mathbf{f}(\mathbf{z})|\mathcal{D}$ has infinite support and, thus, does not satisfy Assumption 2. To ensure the uncertainty is bounded in practice, we truncate the samples in line 5 of Algorithm 1 within a specified confidence region. A similar strategy has been used in [21]. The truncation changes the distribution of $\mathbf{f}(\mathbf{z})|\mathcal{D}$ so that it is no longer a true GP, which correspondingly changes $\Pr_{\mathcal{W}}$ so that it satisfies Assumption 2. However, since the results developed in this paper apply to any $\Pr_{\mathcal{W}}$, this change does to impact our subsequent theoretical analysis.

IV. Sample Complexity Bound for Joint Probabilistic Performance Levels

Using the procedure described in Section III, we can construct the required multi-sample $\{\mathbf{w}^{(1)}, \ldots, \mathbf{w}^{(N)}\}$ to approximate the performance functions using (7). However, a theoretical challenge arises from selecting the number of samples N. To this end, we present a novel *joint* probabilistic validation procedure. Let us first define a set of joint probabilistic performance levels.

Definition 1 (Joint probabilistic performance levels): We define $\gamma_1, \ldots, \gamma_P \in \mathbb{R}$ as joint probabilistic performance levels with violation probability $\epsilon \in (0,1)$ for $\{\phi_i(\cdot)\}_{i=1}^P$ if

$$\Pr_{\mathcal{W}}\left\{\bigcup_{i=1}^{P}\{\phi_i(\hat{\theta}, \mathbf{w}) > \gamma_i\}\right\} \leq \epsilon \tag{18}$$

for any given feasible design parameter $\hat{\theta} \in \Theta$. ◁

Expression (18) implies that, for any set of P performance functions, $\{\gamma_i\}_{i=1}^P$ are chosen such that all functions are simultaneously below these levels with probability at least $1-\epsilon$. Theorem 1 provides a way to compute joint probabilistic performance levels with high confidence. This theorem is a generalization of the result presented in [24] for the particular case of $P = 1$.

Theorem 1: Let $\{\phi_1(\theta, \mathbf{w}), \ldots, \phi_P(\theta, \mathbf{w})\}$ be a finite set of performance functions and $\{\mathbf{w}^{(1)}, \ldots, \mathbf{w}^{(N)}\}$ be a set of N i.i.d. samples drawn from $\mathcal{W}$ with probability $\Pr_{\mathcal{W}}$. Also, define $\gamma_N^i = \max_{j=1,\ldots,N} \phi_i(\hat{\theta}, \mathbf{w}^{(j)})$ for any given design parameter $\hat{\theta} \in \Theta$ and for all $i = 1, \ldots, P$. If

$$N \geq \frac{\log\left(\frac{P}{\delta}\right)}{\log\left(\frac{1}{1-\frac{\epsilon}{P}}\right)}, \tag{19}$$

then $\{\gamma_N^i\}_{i=1}^P$ are joint probabilistic performance levels with probability no smaller than $1-\delta$, i.e.,

$$\Pr_{\mathcal{W}^N}\left\{\Pr_{\mathcal{W}}\left\{\bigcup_{i=1}^{P}\{\phi_i(\hat{\theta}, \mathbf{w}) > \gamma_N^i\}\right\} \leq \epsilon\right\} \geq 1-\delta, \tag{20}$$

where $\Pr_{\mathcal{W}^N} = \Pr_{\mathcal{W}} \times \cdots \times \Pr_{\mathcal{W}}$ is the product of the individual probability measures.

Proof: Given any $\gamma \in \mathbb{R}$, the probability of event $\phi_i(\hat{\theta}, \mathbf{w}) > \gamma$ is denoted by $E_i(\gamma) = \Pr_{\mathcal{W}}\{\phi_i(\hat{\theta}, \mathbf{w}) > \gamma\}$. Using Theorem 3.1 of [24], we have

$$\Pr_{\mathcal{W}^N}\{E_i(\gamma_N^i) > \tilde{\epsilon}\} \leq (1-\tilde{\epsilon})^N,$$

for any given $\tilde{\epsilon} \in (0,1)$ and $i \in \{1, \ldots, P\}$. Now consider the probability δ_F that the empirical performance levels $\{\gamma_N^i\}_{i=1}^P$ are *not* joint probabilistic performance levels with violation probability ϵ. We can establish an upper bound on δ_F using the probability bounds on $\{E_i(\gamma_N^i) > \tilde{\epsilon}\}$

$$\begin{aligned}
\delta_F &= \Pr_{\mathcal{W}^N}\left\{\Pr_{\mathcal{W}}\left\{\bigcup_{i=1}^{P}\{\phi_i(\hat{\theta}, \mathbf{w}) > \gamma_N^i\}\right\} > \epsilon\right\}, \\
&\leq \Pr_{\mathcal{W}^N}\left\{\sum_{i=1}^{P} E_i(\gamma_N^i) > \epsilon\right\}, \\
&\leq \Pr_{\mathcal{W}^N}\left\{\bigcup_{i=1}^{P}\left\{E_i(\gamma_N^i) > \frac{\epsilon}{P}\right\}\right\}, \\
&\leq \sum_{i=1}^{P} \Pr_{\mathcal{W}^N}\left\{E_i(\gamma_N^i) > \frac{\epsilon}{P}\right\}, \\
&\leq P\left(1 - \frac{\epsilon}{P}\right)^N \leq \delta.
\end{aligned}$$

To derive the second line, we have used Boole's inequality on the inner joint violation probability. The third line follows from the fact that the sum of P random variables can only be greater than ϵ if at least one of them is greater than ϵ/P, which is a necessary but not a sufficient condition. The fourth line follows from another application of Boole's inequality, and the fifth line is the direct application of the previously derived inequality with $\tilde{\epsilon} = \epsilon/P$, $\forall i = 1, \ldots, P$. Provided that (19) holds, δ_F must be less than or equal to δ, which establishes the stated claim. ■

The derived bound (19) is *independent* of the number of uncertainties, the size of the support set $\mathcal{W}$, and the probability distribution $\Pr_{\mathcal{W}}$. As such, Theorem 1 can be applied to infinite-dimensional uncertainties such as $\mathbf{w}$ and truncated GP models in accordance with Remark 2. The effect of the cardinality P of the performance functions on sample complexity N can be large, as it divides ϵ in the logarithm in the denominator. However, N remains substantially smaller than even single performance ($P = 1$) alternatives such as the Chernoff bound.

V. Controller Design via Probabilistically Robust Bayesian Optimization (PRBO)

In this section, we address the controller design problem (5) via the SO paradigm on the equivalent problem

$$\min_{\theta\in\Theta} \Phi_1(\theta) \quad \text{s.t.} \quad \Phi_i(\theta) \leq 0, \quad \forall i \in \{2,\dots,P\} \tag{21}$$

using the probabilistic estimates provided in (7). Since no assumptions are made about the structure of (21), this problem cannot be solved with established gradient-based optimization algorithms. Instead, we rely on a particularly data-efficient black-box optimization strategy referred to as Bayesian optimization (BO) [3], which only requires the ability to sample the objective (and constraints when they are present) at arbitrary query points $\theta \in \Theta$. A major advantage of BO is that these objective and/or constraint evaluations can produce stochastic outputs, meaning it can be applied under the stochastic estimates (7). Whenever the number of samples is selected to satisfy Theorem 1, we refer to the method as probabilistically robust BO (PRBO) since every iteration produces joint probabilistic performance levels.

BO is a sequential model-based approach for solving (21), in which we prescribe a prior belief over the set of possible objective and constraint functions that is refined using Bayesian posterior updating. Equipped with probabilistic models for the objectives and any unknown constraints, we can induce sequential acquisition functions $\alpha_n : \Theta \to \mathbb{R}$ that leverage uncertainty in the posterior distributions to guide exploration. The acquisition function should be chosen in a way that represents how promising each $\theta \in \Theta$ would be if it were evaluated next. Thus, the proposed PRBO algorithm consists of the following steps at every iteration $n \in \{1, 2, \dots\}$:

1. Select next design $\theta_{n+1} = \text{argmax}_{\theta\in\Theta}\alpha_n(\theta; \mathcal{M}_n)$;
2. Evaluate worst-case performance functions (7) to obtain $\{\hat{\Phi}^i_{N,n+1}\}_{i=1}^P$ with an N satisfying Theorem 1;
3. Augment data $\mathcal{M}_{n+1} = \{\mathcal{M}_n, (\hat{\Phi}^1_{N,n+1}, \dots, \hat{\Phi}^P_{N,n+1})\}$;
4. Update probabilistic models of performance functions.

A wide variety of probabilistic "surrogate" models can be used to represent the objective and constraint functions, including parametric and non-parametric models. The latter is often preferred due to their ability to represent any function given a sufficiently large dataset. GP regression models are the most popular models to use in BO, especially since they result in analytic expressions for the Bayesian posterior update, as shown in Section III.

Several different acquisition functions have been proposed, including lower confidence bound, Thompson sampling, knowledge gradient, and expected improvement (see, e.g., [3] for more details). Regardless of the choice of the unconstrained acquisition function, denoted by $\alpha^{\text{unc}}_n : \Theta \to \mathbb{R}$, the latter acquisition functions do not directly account for constraints. An intuitive extension proposed in [9] is to define improvement as occurring only when constraints are satisfied. This implies that $\alpha_n(\cdot)$ can be defined as

$$\alpha_n(\theta; \mathcal{M}_n) = \alpha^{\text{unc}}_n(\theta; \mathcal{M}^1_n) \prod_{i=2}^{P} \Pr\{\Phi_i(\theta) \leq 0 | \mathcal{M}^i_n\}, \tag{22}$$

where $\mathcal{M}^i_n$ denotes the collection of the i^{th} performance function evaluations at BO iteration n. The probability terms can be computed analytically for a GP model, as shown in [9]. Since the terms in (22) are cheap to evaluate, this maximization can be carried out efficiently using any of the readily available global optimization metaheuristics. Note that, whenever we do not have any feasible data points, it can be useful to neglect the $\alpha^{\text{unc}}_n(\theta; \mathcal{M}^1_n)$ factor and directly maximize the probability of constraint satisfaction. This search is purely exploitative and will either discover that a particular region of Θ is feasible, or its probability will drop and the algorithm will search for a more promising region.

VI. Numerical Example

The proposed PRBO approach for robust controller design is demonstrated on a modified benchmark double integrator problem from [21]. The system dynamics are given by

$$\mathbf{x}^+ = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \mathbf{x} + \begin{bmatrix} 1 \\ 1 \end{bmatrix} \mathbf{u} + \begin{bmatrix} 1 \\ 0 \end{bmatrix} \frac{1}{2} \sum_{i=1}^{3} \beta_i(\mathbf{x}), \tag{23}$$

where

$$\begin{aligned} \beta_1(\mathbf{x}) &= 1 - \cos\left(\tfrac{1.6}{\pi}[\mathbf{x}]_1\right), \\ \beta_2(\mathbf{x}) &= \sin\left(\tfrac{1.3}{\pi}[\mathbf{x}]_2\right), \\ \beta_3(\mathbf{x}) &= -\sin\left(\tfrac{0.7}{\pi}[\mathbf{x}]_1[\mathbf{x}]_2\right), \end{aligned}$$

which represents the true, but unknown dynamics $\mathbf{f}(\cdot)$ in (1). Note that no additional disturbances $\mathbf{d}$ are considered. We assume that the states and inputs are constrained by $(\mathbf{x}, \mathbf{u}) \in \mathcal{X} \times \mathcal{U}$, where $\mathcal{X} = \{\mathbf{x} \in \mathbb{R}^2 : -10 \leq [\mathbf{x}]_1 \leq 10,\ -10 \leq [\mathbf{x}]_2 \leq 10\}$ and $\mathcal{U} = \{u \in \mathbb{R} : -5 \leq \mathbf{u} \leq 5\}$.

Since the dynamics (23) are unknown, we must first construct a GP emulator of the system dynamics, as described in Section III using a Matern covariance function ($\nu = 3/2$). We assume that $M = 300$ measurements of the form (2)-(3), with $\sigma_v^2 = 10^{-4}$ known, have been collected using randomly generated input sequences. We adapted code from [23] to train the hyperparameters of the GP emulator. To demonstrate the emulator can capture the true behavior of the (unknown) system, we performed a set of validation runs for a randomly selected open-loop input sequence in Fig. 1. In particular, we show system trajectories for 1000 GP posterior realizations (red) and the true system behavior (blue). Although the GP state-space model produces a distribution that contains the

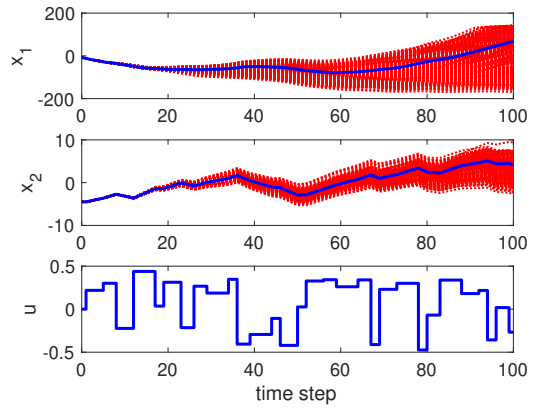

Fig. 1: Gaussian process emulation of the unknown system dynamics. The red profiles show 1000 Gaussian process posterior state realizations for the given input profile. The blue profile represents the true behavior of the system.

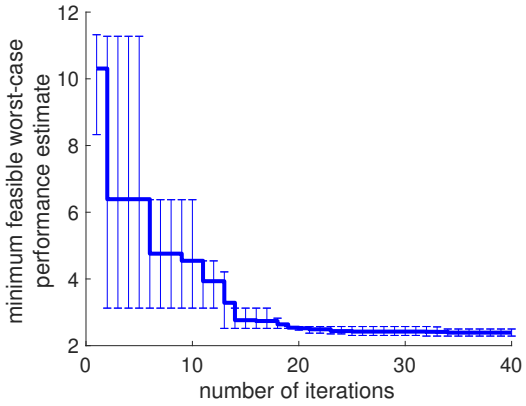

Fig. 2: The minimum feasible worst-case performance estimate as a function of the number of PRBO iterations. The thick blue line represents the mean value averaged over 5 independent PRBO runs, whereas the bars denote the corresponding minimum and maximum values.

true system, its uncertainty estimates grow substantially over time since (23) is not open-loop stable. This suggests the need for a well-designed controller to achieve desired levels of performance.

To design a controller for the unknown system (23), we consider $P = 2$ performance measures. The first measure quantifies the control performance in terms of the average setpoint tracking error of the first state

$$\phi_1(\theta, \mathbf{w}) = \frac{1}{T}\sum_{k=1}^{T} \left\| [\mathbf{x}_k]_1 - r_k \right\|_1, \tag{25}$$

where $r_k \in \mathbb{R}$ is the reference value for the first state at time k over a finite-horizon of $T = 20$ steps. The second performance measure is the worst-case constraint violation over the entire simulation time, i.e.,

$$\phi_2(\theta, \mathbf{w}) = \max_{k\in\{0,\ldots,T\}} \max_{l\in\{1,2\}} g_l(\mathbf{x}_k), \tag{26}$$

where $g_1(\mathbf{x}_k) = [\mathbf{x}_k]_1 - 10$ and $g_2(\mathbf{x}_k) = -10 - [\mathbf{x}_k]_1$ are the violation of the upper and lower bounds for the first state at time k, respectively. Given these performance measures, we selected the controller $\kappa(\cdot)$ to be a model predictive control (MPC) law that uses the mean GP prediction model. The MPC controller solves the following nonlinear optimization problem at every sample time $k \in \{0, \ldots, T-1\}$

$$\begin{aligned} \min_{\mathbf{x}_{i|k}, \mathbf{u}_{i|k}} \quad & \textstyle\sum_{i=0}^{N_p-1} \ell_k(\mathbf{x}_{i|k}, \mathbf{u}_{i|k}; \theta_\ell) + \ell_f(\mathbf{x}_{N_p|k}), \\ \text{s.t.} \quad & \mathbf{x}_{i+1|k} = \boldsymbol{\mu}_{\mathbf{f}}([\mathbf{x}_{i|k}^\top, \mathbf{u}_{i|k}^\top]^\top; \mathcal{D}), \\ & (\mathbf{x}_{i+1|k}, \mathbf{u}_{i|k}) \in \mathcal{X}_{\text{mpc}}(\theta_b) \times \mathcal{U}, \\ & \mathbf{x}_{0|k} = \mathbf{x}_k, \quad \forall i = 0, \ldots, N_p - 1, \end{aligned} \tag{27}$$

where $N_p = 4$ is the prediction horizon; $x_{i|k}$ and $u_{i|k}$ are, respectively, the predicted state and control input at i time steps ahead of current time k; $\ell_f(\mathbf{x}) = 0$ is the terminal cost; $\ell_k(\mathbf{x}, \mathbf{u}; \theta_l) = \|[\mathbf{x}]_1 - r_k\|_2^2 + \frac{1}{\theta_\ell}\|\mathbf{u}\|_2^2$ is the stage cost with θ_ℓ representing a tunable weight coefficient; and

$$\mathcal{X}_{\text{mpc}}(\theta_b) = \{\mathbf{x} \in \mathbb{R}^2 : -10 + \theta_b \leq [\mathbf{x}]_1 \leq 10 - \theta_b\}$$

is the state constraints parametrized by a backoff tuning parameter θ_b. Let $\{\mathbf{u}^\star_{0|k}(\mathbf{x}_k;\theta), \ldots, \mathbf{u}^\star_{N_p-1|k}(\mathbf{x}_k;\theta)\}$ denote the set of optimal control inputs for (27), then the MPC law is defined by $\kappa(\mathbf{x}_k;\theta) = \mathbf{u}^\star_{0|k}(\mathbf{x}_k;\theta)$, where $\theta = (\theta_\ell, \theta_b) \in \Theta = [10^{-3}, 10^2] \times [0, 20]$ can influence both performance measures (25) and (26). The MPC problem (27) was formulated in CasADi [25] and solved using IPOPT [26].

The closed-loop system takes the form of (4), with a fixed initial condition $\mathbf{x}_0 = [-7.5, -4.5]^\top$. Since the feasible region of (5) cannot be determined exactly, we instead use the probabilistic feasibility result established in Theorem 1. In particular, we assert a controller design is *feasible* if (18) is satisfied for $\epsilon = 0.1$ and $\delta = 10^{-4}$. Thus, according to (19), a particular $\theta \in \Theta$ results in a feasible controller whenever $\gamma_N^2 = \max_{j=1,\ldots,N} \phi_2(\theta, \mathbf{w}^{(j)}) \leq 0$ over a set of $N = 194$ i.i.d. samples. Given these specifications, the PRBO approach in Section V was applied to find the optimal set of tuning parameters. To solve the PRBO problem, we used the ADMM-Bayesian Optimization algorithm [27]. The performance of the PRBO approach over 40 iterations, averaged over five independent runs, is shown in Fig. 2. We observe that the worst-case performance consistently improves as the number of iterations increases, suggesting PRBO can efficiently determine high-performance designs.

Next, we verify the "optimal" tuning parameters obtained via PRBO, i.e., $\theta^\star = (4.87, 0.8)$, produce a controller that meets the performance requirements (25) and (26). Fig. 3 shows the closed-loop state trajectories for $N = 194$ GP posterior realizations of the closed-loop dynamics. Not only has the MPC controller significantly reduced uncertainty compared to the previous open-loop runs (Fig. 1), but all of the GP posterior realizations satisfy constraints and yield satisfactory setpoint tracking. For comparison purposes, we also demonstrate the closed-loop state trajectories for a nominal tuning $\theta_0 = (10, 0)$, which results in significant constraint violations. It is evident that PRBO consistently finds "optimal" tuning parameters in approximately 20 se-

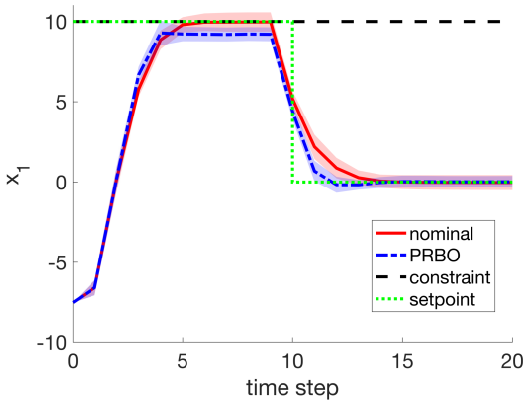

Fig. 3: The closed-loop state profiles based on $N = 194$ Gaussian process emulations of the closed-loop dynamics. The red profile corresponds to the MPC controller with nominal tuning, whereas the blue profile corresponds to the optimally tuned MPC via PRBO. The shaded regions are +/- two standard deviations around the mean.

quential iterations. These results illustrate the effectiveness of the proposed PRBO approach.

VII. Conclusions

This paper presented a data-driven approach for the robust design of arbitrary controllers. The main features of the proposed approach include: (i) accounting for system uncertainties in controller tuning through worst-case estimation of performance measures, and (ii) providing guarantees that the worst-case performance estimates jointly achieve a pre-specified probability of accuracy. Incorporation of these features into a constrained Bayesian optimization framework yields a fully data-driven and probabilistically robust controller design approach that circumvents the need for additional post verification of the designed controllers. Our future work will focus on investigating multi-objective formulations of the robust controller design problem.

References

[1] V. D. Blondel and J. N. Tsitsiklis, "A survey of computational complexity results in systems and control," *Automatica*, vol. 36, pp. 1249–1274, 2000.

[2] J. A. Paulson and A. Mesbah, "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*, Miami, 2018, pp. 6307–6313.

[3] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas, "Taking the human out of the loop: A review of Bayesian optimization," *Proceedings of the IEEE*, vol. 104, no. 1, pp. 148–175, 2016.

[4] F. Berkenkamp, A. P. Schoellig, and A. Krause, "Safe controller optimization for quadrotors with Gaussian processes," in *Proceedings of the IEEE International Conference on Robotics and Automation*, Stockholm, 2016, pp. 491–496.

[5] S. Bansal, R. Calandra, T. Xiao, S. Levine, and C. J. Tomiin, "Goal-driven dynamics learning via Bayesian optimization," in *Proceedings of the 56th IEEE Conference on Decision and Control*, Melbourne, 2017, pp. 5168–5173.

[6] M. Neumann-Brosig, A. Marco, D. Schwarzmann, and S. Trimpe, "Data-efficient autotuning with Bayesian optimization: An industrial control study," *IEEE Transactions on Control Systems Technology*, vol. 28, no. 3, pp. 730–740, 2019.

[7] D. Piga, M. Forgione, S. Formentin, and A. Bemporad, "Performance-oriented model learning for data-driven MPC design," *IEEE Control Systems Letters*, vol. 3, no. 3, pp. 577–582, 2019.

[8] M. Khosravi, V. Behrunani, P. Myszkorowski, R. S. Smith, A. Rupenyan, and J. Lygeros, "Performance-driven cascade controller tuning with Bayesian optimization," *arXiv preprint arXiv:2007.12536*, 2020.

[9] J. M. Hernández-Lobato, M. A. Gelbart, R. P. Adams, M. W. Hoffman, and Z. Ghahramani, "A general framework for constrained Bayesian optimization using information-based search," *Journal of Machine Learning Research*, vol. 17, no. 160, pp. 1–53, 2016.

[10] F. Sorourifar, G. Makrygiorgos, A. Mesbah, and J. A. Paulson, "A data-driven automatic tuning method for MPC under uncertainty using constrained Bayesian optimization," Venice, 2021, pp. 1–6.

[11] J. A. Paulson and A. Mesbah, "Data-driven scenario optimization for automated controller tuning with probabilistic performance guarantees," *IEEE Control Systems Letters*, vol. 5, pp. 1477–1482, 2020.

[12] M. C. Campi, S. Garatti, and F. A. Ramponi, "A general scenario theory for nonconvex optimization and decision making," *IEEE Transactions on Automatic Control*, vol. 63, no. 12, pp. 4067–4078, 2018.

[13] J. Umlauft, T. Beckers, and S. Hirche, "Scenario-based optimal control for Gaussian process state space models," in *Proceedings of the European Control Conference*, Limassol, 2018, pp. 1386–1392.

[14] R. Tempo, G. Calafiore, and F. Dabbene, *Randomized algorithms for analysis and control of uncertain systems: with applications*. London, UK: Springer Science & Business Media, 2012.

[15] M. Maiworm, D. Limon, and R. Findeisen, "Online learning-based model predictive control with Gaussian process models and stability guarantees," *International Journal of Robust and Nonlinear Control*, In Press, 2021.

[16] C. E. Rasmussen and C. K. I. Williams, *Gaussian processes for machine learning*. Cambridge, MA: MIT Press, 2006.

[17] M. Althoff, O. Stursberg, and M. Buss, "Reachability analysis of nonlinear systems with uncertain parameters using conservative linearization," in *Proceedings of the 47th IEEE Conference on Decision and Control*, Cancun, 2008, pp. 4042–4048.

[18] J. Kocijan, R. Murray-Smith, C. E. Rasmussen, and B. Likar, "Predictive control with Gaussian process models," in *Proceedings of the IEEE Region 8 EUROCON*. IEEE, 2003, pp. 352–356.

[19] G. Gregorčič and G. Lightbody, "Gaussian process approach for modelling of nonlinear systems," *Engineering Applications of Artificial Intelligence*, vol. 22, pp. 522–533, 2009.

[20] E. Bradford, L. Imsland, D. Zhang, and E. A. del Rio Chanona, "Stochastic data-driven model predictive control using Gaussian processes," *Computers & Chemical Engineering*, vol. 139, p. 106844, 2020.

[21] A. D. Bonzanini, J. A. Paulson, and A. Mesbah, "Safe learning-based model predictive control under state-and input-dependent uncertainty using scenario trees," in *Proceedings of the 59th IEEE Conference on Decision and Control*, Jeju Island, Republic of Korea, 2020, pp. 2448–2454.

[22] J. R. Gardner, G. Pleiss, D. Bindel, K. Q. Weinberger, and A. G. Wilson, "Gpytorch: Blackbox matrix-matrix gaussian process inference with GPU acceleration," *arXiv preprint arXiv:1809.11165*, 2018.

[23] E. Bradford, A. M. Schweidtmann, and A. Lapkin, "Efficient multiobjective optimization employing Gaussian processes, spectral sampling and a genetic algorithm," *Journal of Global Optimization*, vol. 71, pp. 407–438, 2018.

[24] R. Tempo, E.-W. Bai, and F. Dabbene, "Probabilistic robustness analysis: Explicit bounds for the minimum number of samples," in *Proceedings of the 35th IEEE Conference on Decision and Control*, 1996, pp. 3424–3428.

[25] J. A. E. Andersson, J. Gillis, G. Horn, J. B. Rawlings, and M. Diehl, "CasADi – A software framework for nonlinear optimization and optimal control," *Mathematical Programming Computation*, vol. 11, no. 1, pp. 1–36, 2019.

[26] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical programming*, vol. 106, pp. 25–57, 2006.

[27] S. Ariafar, J. Coll-Font, D. H. Brooks, and J. G. Dy, "ADMMBO: Bayesian optimization with unknown constraints using admm." *Journal of Machine Learning Research*, vol. 20, no. 123, pp. 1–26, 2019.