Anticipating the long-term effect of online learning in control

Alexandre Capone and Sandra Hirche

Abstract— **Control schemes that learn using measurement data collected online are increasingly promising for the control of complex and uncertain systems. However, in most approaches of this kind, learning is viewed as a side effect that passively improves control performance, e.g., by updating a model of the system dynamics. Determining how improvements in control performance due to learning can be actively exploited in the control synthesis is still an open research question. In this paper, we present AntLer, a design algorithm for learning-based control laws that anticipates learning, i.e., that takes the impact of future learning in uncertain dynamic settings explicitly into account. AntLer expresses system uncertainty using a non-parametric probabilistic model. Given a cost function that measures control performance, AntLer chooses the control parameters such that the expected cost of the closed-loop system is minimized approximately. We show that AntLer approximates an optimal solution arbitrarily accurately with probability one. Furthermore, we apply AntLer to a nonlinear system, which yields better results compared to the case where learning is not anticipated.**

I. Introduction

Control design often requires an accurate model of the system dynamics. However, obtaining a mathematical model is often prohibitive due to system intricacy or lack of expertise. Moreover, erroneously assuming that a model is correct can lead to poor control performance. These issues have been increasingly addressed by employing online learning-based strategies, i.e., algorithms that employ system measurements collected online to improve control performance. This is typically achieved either by learning a model of the system, e.g., with Bayesian modeling tools [1]–[7], or by directly learning the optimal control law, e.g., by applying online reinforcement learning [8]. Despite belonging to the broader category of adaptive control, the intricacy of online learning-based control algorithms often does not allow a formal assessment of the resulting control performance, as opposed to many classical adaptive control strategies [9], [10].

Even though online learning-based approaches adapt over time using measurement data, they often include parameters that are *data-independent*, i.e., parameters that are fixed a priori and do not depend on the collected data. Examples include control gains [1], [6], [11] and safety-relevant parameters [4], [12]. Most of these methods choose the data-independent parameters such that system safety and stability is guaranteed after an arbitrary model update [4], [6], [12], while others omit guarantees altogether [1]–[3], [7], [8], [11]. Hence, although learning is an integral part of the control loop, much the same as the control law itself, it only improves the control law in a passive fashion. In other words, the control law is not designed with future learning in mind. This can cause the control to be overly conservative, leading to excessively costly state trajectories.

Efficiently choosing data-independent parameters in a learning-based setting requires accurately assessing how the control law will perform, which is generally achieved by leveraging any prior knowledge about the system. To this end, we introduce a novel algorithm for optimizing data-independent parameters that quantifies how system uncertainty is expected to be reduced over time due to learning. In other words, the proposed algorithm anticipates the impact that online learning will have on future control performance.

Within the control community, the idea of anticipating and exploiting learning effects in control design has been explored in the form of dual control [13], [14]. So far, dual control has been investigated mostly within the context of structured models with parametric uncertainties, with few exceptions [15], [16]. However, [16] requires the true system to be affine in the control, and both [16] and [15] employ approximations that yield no theoretical guarantees. Hence, developing a general method that provably approximates data-independent parameters arbitrarily accurately remains an open research question.

In this paper, we present AntLer (anticipating learning), a sampling-based algorithm that approximates optimal data-independent parameters of online learning-based control laws in uncertain settings. Our approach accounts for a broad class of model uncertainties by using a probabilistic Gaussian process model. Given a cost function that quantifies control performance over a finite-time horizon, AntLer is able to express the expected cost for an online learning-based control law. Minimizing the resulting expression with respect to the control law's data-independent parameters corresponds to a stochastic optimal control problem, which AntLer solves approximately using sample average approximation. AntLer is applicable to a wide class of dynamical systems that include an additive uncertainty, as well as process noise. We show that, under reasonable assumptions, AntLer approximates the optimal solution arbitrarily accurately given a large enough number of samples.

The remainder of this paper is organized as follows. Section II describes the general problem setting and the assumptions used in this paper. In Section III the probabilistic approach used to quantify model uncertainty is discussed. Section IV contains our main result. Therein, we introduce the AntLer algorithm and provide a corresponding theoretical

This work was supported by the ERC Starting Grant "Control based on Human Models" under grant agreement no. 337654.

The authors are with the Department of Electrical and Computer Engineering, Technical University of Munich, 80333 Munich, Germany (e-mail: alexandre.capone@tum.de; hirche@tum.de).

analysis. In Section V AntLer is applied to a numerical system. We then provide some concluding remarks in Section VI.

Notation: Let $\mathbb{N}$ denote the natural numbers, $\mathbb{R}$ the real numbers, and $\mathbb{R}_+$ the non-negative real numbers. We employ bold lowercase and uppercase letters to denote vectors and matrices, respectively. For $\mu, \sigma \in \mathbb{R}_+$, a normal distribution with mean μ and variance σ^2 is denoted as $\mathcal{N}(\mu, \sigma^2)$. For $d \in \mathbb{N}$, we denote the space of continuously differentiable functions on $\mathbb{R}^d$ as $\mathcal{C}^1(\mathbb{R}^d)$, and the d-dimensional identity matrix as $\boldsymbol{I}_d$. Moreover, for matrices $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{d \times d}$, we use $[\boldsymbol{A}, \boldsymbol{B}]$ to denote the horizontal concatenation of $\boldsymbol{A}$ followed by $\boldsymbol{B}$. The entry in the i-th row and j-th column of $\boldsymbol{A}$ is denoted by $[\boldsymbol{A}]_{ij}$. The symbol $\cup$ denotes the union of two sets. We use $\mathrm{E}_{\boldsymbol{a}_1,\dots,\boldsymbol{a}_d}[\,\cdot\,]$ to denote the expected value operator with respect to the probability distribution of the random variables $\boldsymbol{a}_1, \dots, \boldsymbol{a}_d \in \mathbb{R}^d$.

II. Problem Statement

We consider a discrete-time system of the form

$$\begin{aligned} \boldsymbol{x}_{t+1} &= \boldsymbol{f}(\boldsymbol{x}_t, \boldsymbol{u}_t) + \boldsymbol{g}(\boldsymbol{x}_t, \boldsymbol{u}_t) + \boldsymbol{w}_t \\ &=: \boldsymbol{f}(\tilde{\boldsymbol{x}}_t) + \boldsymbol{g}(\tilde{\boldsymbol{x}}_t) + \boldsymbol{w}_t \end{aligned} \tag{1}$$

where $\boldsymbol{x}_t \in \mathcal{X} \subseteq \mathbb{R}^{N_x}$ and $\mathbf{u}_t \in \mathcal{U} \subseteq \mathbb{R}^{N_u}$ are the system's state vector and control vector at the t-th time step, respectively. The initial state $\boldsymbol{x}_0 \in \mathcal{X}$ is assumed to be fixed and known. The vector of augmented states $\tilde{\boldsymbol{x}}_t := (\boldsymbol{x}_t, \boldsymbol{u}_t) \in \tilde{\mathcal{X}}$, where $\tilde{\mathcal{X}} := \mathcal{X} \times \mathcal{U}$, concatenates the state vector $\boldsymbol{x}_t$ and the vector of control inputs $\boldsymbol{u}_t$, and is henceforth employed for the sake of simplicity. The system is disturbed by multivariate normally distributed process noise $\boldsymbol{w}_t \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_w^2)$. Here $\boldsymbol{\Sigma}_w = \mathrm{diag}(\sigma_{w_1}, \dots, \sigma_{w_{N_x}})$ is a nonnegative diagonal matrix, which we assume to know. The function $\boldsymbol{f} \in \mathcal{C}^1(\tilde{\mathcal{X}})$, corresponds to the prior model of the system dynamics, whereas $\boldsymbol{g} \in \mathcal{C}^1(\tilde{\mathcal{X}})$ is unknown and is assumed to be drawn from a Gaussian process (GP). This is described thoroughly in Section III.

Remark 1: In this paper, we assume that $\boldsymbol{x}_0$ is fixed and known solely to avoid cumbersome notation. The algorithm proposed in this work extends straightforwardly to the more general case where only the probability distribution of $\boldsymbol{x}_0$ is known.

Remark 2: This constellation can be assumed for a wide variety of settings. For example, if no prior system knowledge is available, then this is reflected by choosing $\boldsymbol{f}(\boldsymbol{x}_t, \boldsymbol{u}_t) = \boldsymbol{x}_t$.

We assume that a parametric online learning-based control law of the form $\boldsymbol{u} : \Gamma \times \boldsymbol{\Theta} \times \mathcal{X} \mapsto \mathcal{U}$ is employed to control (1), where $\boldsymbol{\Theta}$ denotes the space of data-independent control parameters, and $\Gamma := \{\{\tilde{\boldsymbol{x}}_0, \dots, \tilde{\boldsymbol{x}}_t\} \in \tilde{\mathcal{X}}^t \mid t \in \mathbb{N}\}$ is the set of all finite subsets of $\tilde{\mathcal{X}}$. At every time step, the control law $\boldsymbol{u}(\cdot, \cdot, \cdot)$ takes as arguments the system measurement data $\mathcal{D}_t = \{\tilde{\boldsymbol{x}}_0, \dots, \tilde{\boldsymbol{x}}_{t-1}\} \in \Gamma$ collected up to time step t, the data-independent control parameters $\boldsymbol{\vartheta} \in \boldsymbol{\Theta}$, and the current state $\boldsymbol{x}_t$. The collected data $\mathcal{D}_t$ is employed to update the control law at every time step, e.g., by learning a model of the system. The control parameters $\boldsymbol{\vartheta}$ correspond to the data-independent components of the control law, e.g., multiplicative scalars used to scale confidence regions and thereby guarantee operational safety [17], or linear feedback gains [6]. This formulation encompasses most discrete-time online learning-based control strategies. We henceforth write $\boldsymbol{u}_t(\boldsymbol{\vartheta}) := \boldsymbol{u}(\mathcal{D}_t, \boldsymbol{\vartheta}, \boldsymbol{x}_t)$ to denote the online learning-based control law at time step t.

Remark 3: In order to anticipate the effect of online learning, we aim to predict which data set $\mathcal{D}_t$ will be collected over time and how it will affect the overall control performance. As a baseline, we consider the case where predictions are carried out without anticipating learning, which amounts to predicting the closed-loop behavior under the data-independent counterpart $\boldsymbol{u}_t^0(\boldsymbol{\vartheta}) := \boldsymbol{u}(\mathcal{D}_0, \boldsymbol{\vartheta}, \boldsymbol{x}_t)$. Here we assume $\mathcal{D}_0 := \{\}$ without loss of generality. In Section V, we compare predictions made with both control laws using a simple example.

Remark 4: The method presented in this paper extends straightforwardly to a setting where the system measurements $\mathcal{D}_t$ used to update the control law are corrupted by normally distributed observation noise. However, for notational convenience, we focus solely on the case without observation noise.

Our goal is to minimize a finite horizon cost function

$$C(\boldsymbol{\vartheta}) := \mathrm{E}_{\tilde{\boldsymbol{x}}_1,\dots,\tilde{\boldsymbol{x}}_T}\left[\sum_{t=0}^{T} c_t(\boldsymbol{x}_t, \boldsymbol{u}_t(\boldsymbol{\vartheta}))\right] \tag{2}$$

over the data-independent control parameters $\boldsymbol{\vartheta}$, where $c_t : \tilde{\mathcal{X}} \mapsto \mathbb{R}_+$ are continuously differentiable functions that express the immediate cost. The probability distribution of $\tilde{\boldsymbol{x}}_1, \dots, \tilde{\boldsymbol{x}}_T$ captures both the effect of process noise $\boldsymbol{w}_t$, as well as the model uncertainty $\boldsymbol{g}(\cdot)$. This is discussed in Section III. We denote the minimal value of (2) as $C^* := \min_{\boldsymbol{\vartheta} \in \boldsymbol{\Theta}} C(\boldsymbol{\vartheta})$ and the corresponding set of minimizing parameters as $\boldsymbol{\Theta}^* := \{\boldsymbol{\vartheta} \in \boldsymbol{\Theta} \mid C(\boldsymbol{\vartheta}) = C^*\}$.

If no assumptions about the online learning-based control law $\boldsymbol{u}_t(\cdot)$ are made, then it is generally impossible to reliably predict the closed-loop behavior of (1). Hence, we need to impose some restrictions on the type of control law considered.

Assumption 1: There exists a compact subset $\tilde{\boldsymbol{\Theta}} \subseteq \boldsymbol{\Theta}$, such that $\sum_{t=0}^{T} c_t(\boldsymbol{x}_t, \boldsymbol{u}_t(\boldsymbol{\vartheta})) > C^*$ holds for all $\boldsymbol{\vartheta} \in \boldsymbol{\Theta} \setminus \tilde{\boldsymbol{\Theta}}$ and arbitrary $\boldsymbol{x}_0, \dots, \boldsymbol{x}_T \in \mathcal{X}$.

Assumption 1 is less restrictive than assuming that $\boldsymbol{\Theta}$ is compact, which is often the case in learning-based applications, e.g., in settings where safety-relevant constraints are an issue [18], [19]. Furthermore, Assumption 1 does not impose strong limitations in practice, as $\tilde{\boldsymbol{\Theta}}$ may still be very large.

In order to be able to find a minimizer $\boldsymbol{\vartheta}^* \in \boldsymbol{\Theta}^*$ of (2), we additionally require the control law $\boldsymbol{u}(\cdot, \cdot, \cdot)$ to satisfy some regularity conditions. In this paper, we restrict the control law to the broad and practically relevant class of continuously differentiable functions, as described in the following.

Assumption 2: The control law $\boldsymbol{u}(\mathcal{D}_t, \boldsymbol{\vartheta}, \boldsymbol{x}_t)$ is continuously differentiable with respect to its arguments, where continuous differentiability with respect to the data is defined

as follows. For every fixed $\mathcal{D} \in \Gamma$, $\boldsymbol{x}_t \in \mathcal{X}$ and $\boldsymbol{\vartheta} \in \boldsymbol{\Theta}$, the function

$$\boldsymbol{u}_{\mathcal{D},\boldsymbol{\vartheta},\boldsymbol{x}_t}(\tilde{\boldsymbol{x}}) \coloneqq \boldsymbol{u}(\mathcal{D} \cup \tilde{\boldsymbol{x}}, \boldsymbol{\vartheta}, \boldsymbol{x}_t)$$

is continuously differentiable with respect to $\tilde{\boldsymbol{x}}$ for all $\tilde{\boldsymbol{x}} \in \tilde{\mathcal{X}}$.

Many commonplace control laws are continuously differentiable with respect to the state $\boldsymbol{x}_t$ and parameters $\boldsymbol{\vartheta} \in \boldsymbol{\Theta}$, e.g., linear feedback gains and neural networks. Furthermore, control update rules are often continuously differentiable with respect to the data, e.g., if a model of the system is learned online [5].

III. Probabilistic system model

In this section, we provide a brief introduction to GPs, and describe how we use them to capture model uncertainty and predict control performance.

A. Predictions using Gaussian processes

In order to assess how the learning-based control law will perform in an uncertain environment, we require a probabilistic model that expresses model uncertainty given prior system measurements. To this end, we model (1) using a *Gaussian process* (GP), a probabilistic modeling tool that captures model uncertainty. We opt to employ GPs in this work because they often exhibit good generalization behavior in practice. However, we note that other probabilistic modeling frameworks can be employed, e.g., Bayesian neural networks.

We introduce GPs for the case where the state is a scalar, i.e., $N_x = 1$, and then explain how one-dimensional GPs are extended to the multivariate case. A GP is a collection of dependent random variables, for which any finite subset is jointly normally distributed [20]. It is fully specified by a mean function $m : \tilde{\mathcal{X}} \mapsto \mathbb{R}$ and a positive definite kernel function $k : \tilde{\mathcal{X}} \times \tilde{\mathcal{X}} \mapsto \mathbb{R}$. In this paper, since our prior knowledge is captured by $f(\cdot)$, we set $m \equiv 0$ without loss of generality [20]. The kernel $k(\cdot,\cdot)$ is a similarity measure for evaluations of $g(\cdot)$, and encodes function properties such as smoothness and periodicity. Throughout this paper, we assume that the kernel $k(\cdot,\cdot)$ is continuously differentiable, which reflects the assumption that $g(\cdot)$ is continuously differentiable [20]. Given $m(\cdot)$ and $k(\cdot,\cdot)$, we denote a GP by $\mathcal{GP}(m,k)$. By modeling an unknown function $g(\cdot)$ with a GP, we implicitly assume that any finite set of function evaluations $\boldsymbol{y}_{\mathcal{D}_t} \coloneqq (g(\tilde{\boldsymbol{x}}_0), \dots, g(\tilde{\boldsymbol{x}}_{t-1}))$ at arbitrary points $\mathcal{D}_t \coloneqq \{\tilde{\boldsymbol{x}}_0, \dots, \tilde{\boldsymbol{x}}_{t-1}\}$ is jointly normally distributed,

$$\boldsymbol{y}_{\mathcal{D}_t} \sim \mathcal{N}\left(\boldsymbol{0}, \boldsymbol{K}_{D_t}\right), \tag{3}$$

where the entries of the covariance matrix $\boldsymbol{K}_{\mathcal{D}_t}$ are given by $[\boldsymbol{K}_{\mathcal{D}_t}]_{ij} = k(\tilde{\boldsymbol{x}}_{i-1}, \tilde{\boldsymbol{x}}_{j-1})$, $i,j = 1,\dots,t$.

Using (3), we are able to condition the GP on any measurements taken prior to the control design. In the following, for the sake of notational simplicity, we assume that no prior measurement data is available, and describe how to recursively draw and condition the GP on samples. However, conditioning the GP on system measurement data is identical to conditioning on samples up to an additive term that represents noise covariance [20].

In order to predict the control performance of $\boldsymbol{u}_t(\cdot)$, we aim to draw sample trajectories that satisfy (3). We henceforth distinguish sample evaluations of the GP model, which are drawn using (3), from evaluations of the true system (1) by denoting samples using the superscript s. A sample system trajectory is computed by sequentially sampling from the one-step prediction of the unknown dynamics at time step t

$$g^s(\tilde{\boldsymbol{x}}_t^s) \sim \mathcal{N}\left(\mu_t^s\left(\tilde{\boldsymbol{x}}_t^s\right), \left(\sigma_t^s\left(\tilde{\boldsymbol{x}}_t^s\right)\right)^2\right), \tag{4}$$

and subsequently computing the next sample state

$$\begin{aligned} \tilde{\boldsymbol{x}}_{t+1}^s &= \left(x_{t+1}^s, u_t^s(\boldsymbol{\vartheta})\right) \coloneqq \left(x_{t+1}^s, u(\mathcal{D}_t^s, \boldsymbol{\vartheta}, x_{t+1}^s)\right), \\ x_{t+1}^s &= f(\tilde{\boldsymbol{x}}_t^s) + g_t^s(\tilde{\boldsymbol{x}}_t^s) + w_t^s, \end{aligned} \tag{5}$$

where $\mathcal{D}_t^s \coloneqq \{\tilde{\boldsymbol{x}}_0^s, \dots, \tilde{\boldsymbol{x}}_{t-1}^s\}$ and $w_t^s \sim \mathcal{N}(0, \sigma_w)$. Here $\tilde{\boldsymbol{x}}_0^s \coloneqq (\boldsymbol{x}_0, \boldsymbol{u}_0(\boldsymbol{\vartheta}))$ is introduced for simplicity of exposition. The mean and variance of (4) are computed using

$$\mu_t^s\left(\tilde{\boldsymbol{x}}_t^s\right) \coloneqq \mu\left(\tilde{\boldsymbol{x}}_t^s | \mathcal{D}_t^s, \boldsymbol{y}_{\mathcal{D}_t^s}\right) = \boldsymbol{k}^{\mathrm{T}}\left(\tilde{\boldsymbol{x}}_t^s\right) \boldsymbol{K}_{\mathcal{D}_t^s}^{-1} \boldsymbol{y}_{\mathcal{D}_t^s}^{\mathrm{T}} \tag{6}$$

$$\begin{aligned} \left(\sigma_t^s\left(\tilde{\boldsymbol{x}}_t^s\right)\right)^2 &\coloneqq \sigma^2\left(\tilde{\boldsymbol{x}}_t^s | \mathcal{D}_t^s, \boldsymbol{y}_{\mathcal{D}_t^s}\right) \\ &= k\left(\tilde{\boldsymbol{x}}_t^s, \tilde{\boldsymbol{x}}_t^s\right) - \boldsymbol{k}^{\mathrm{T}}\left(\tilde{\boldsymbol{x}}_t^s\right) \boldsymbol{K}_{\mathcal{D}_t^s}^{-1} \boldsymbol{k}\left(\tilde{\boldsymbol{x}}_t^s\right), \end{aligned} \tag{7}$$

respectively. Here the vector

$$\boldsymbol{y}_{\mathcal{D}_t^s} \coloneqq (g^s(\tilde{\boldsymbol{x}}_0^s), \dots, g^s(\tilde{\boldsymbol{x}}_{t-1}^s)) \tag{8}$$

concatenates previously drawn sample states $\tilde{\boldsymbol{x}}_i^s \in \mathcal{D}_t^s$, and

$$\boldsymbol{k}\left(\tilde{\boldsymbol{x}}_t^s\right) = \left(k(\tilde{\boldsymbol{x}}_0^s, \tilde{\boldsymbol{x}}_t^s),\ \dots,\ k(\tilde{\boldsymbol{x}}_{t-1}^s, \tilde{\boldsymbol{x}}_t^s)\right)^{\mathrm{T}} \tag{9}$$

consists of kernel evaluations at $\tilde{\boldsymbol{x}}_t^s$ and $\tilde{\boldsymbol{x}}_i^s \in \mathcal{D}_t^s$.

Remark 5: Here we abuse notation slightly by employing $g^s(\cdot)$ to refer to a function sampled from the GP. As can be seen from (6)-(9), $g^s(\cdot)$ depends on previously sampled function evaluations. In fact, a sample function evaluation is computed as

$$g^s(x_{t+1}) = \mu_t^s\left(\tilde{\boldsymbol{x}}_t^s\right) + \sigma_t^s\left(\tilde{\boldsymbol{x}}_t^s\right) \zeta^s, \tag{10}$$

where $\zeta^s \in \mathcal{N}(0,1)$. In Section IV, we use rigorous notation by referring to sample function evaluations as in (10).

Remark 6: It is necessary that the GP samples $g^s(\tilde{\boldsymbol{x}}_t^s)$ and process noise samples σ_w^s be drawn separately in order for the vector $\boldsymbol{y}_{\mathcal{D}_t^s}$ to be uniquely defined. This in turn guarantees that the sampled function $g^s(\cdot)$ exhibits deterministic behavior at points where samples were previously drawn [20]. We require this to reflect the fact that $g(\cdot)$ is unknown but deterministic. Hence, we draw sample trajectories that satisfy (4) and (5) as

$$x_{t+1}^s = f(\tilde{\boldsymbol{x}}_t^s) + \mu_t^s\left(\tilde{\boldsymbol{x}}_t^s\right) + \sigma_t^s\left(\tilde{\boldsymbol{x}}_t^s\right) \zeta_1^s + \sigma_w \zeta_2^s, \tag{11}$$

where $\zeta_1^s, \zeta_2^s \sim \mathcal{N}(0,1)$ are sampled separately.

Remark 7: Typically, multiple samples can be drawn from the same GP simultaneously [20]. However, since we are interested in samples that satisfy the system dynamics, we need to draw a sample and compute the resulting state sequentially.

In the case where the state is multidimensional, we model each state transition using a separate GP, i.e.,

$$\boldsymbol{x}_{t+1}^s \sim \mathcal{N}\left(\boldsymbol{f}(\tilde{\boldsymbol{x}}_t^s) + \boldsymbol{\mu}_t^s(\tilde{\boldsymbol{x}}_t^s), (\boldsymbol{\Sigma}_t^s(\tilde{\boldsymbol{x}}_t^s))^2 + \boldsymbol{\Sigma}_w^2\right), \quad (12)$$

where

$$\boldsymbol{\mu}_t^s(\tilde{\boldsymbol{x}}_t^s) := \left(\mu(\tilde{\boldsymbol{x}}_t^s|\boldsymbol{y}_{1,\mathcal{D}_t^s}),\ \ldots,\ \mu(\tilde{\boldsymbol{x}}_t^s|\boldsymbol{y}_{N_x,\mathcal{D}_t^s})\right),$$
$$(\boldsymbol{\Sigma}_t^s(\tilde{\boldsymbol{x}}_t^s))^2 := \text{diag}\Big(\sigma^2(\tilde{\boldsymbol{x}}_t^s|\boldsymbol{y}_{1,\mathcal{D}_t^s}),\ \ldots,\ \sigma^2(\tilde{\boldsymbol{x}}_t^s|\boldsymbol{y}_{N_x,\mathcal{D}_t^s})\Big).$$

Here $\boldsymbol{y}_{i,\mathcal{D}_t^s} := \left(g_i^s(\tilde{\boldsymbol{x}}_0^s),\ \ldots,\ g_i^s(\tilde{\boldsymbol{x}}_{t-1}^s)\right)$ concatenates samples of the i-th component of the GP model for every $i = 1, \ldots, N_x$.

Remark 8: Modeling each state transition with a separate GP corresponds to assuming that the state transitions are conditionally independent. Alternatively, a generalization of GPs to multiple dimensions is also applicable [20]. However, the latter approach is significantly more computationally expensive than the former. Moreover, employing separate GPs for each state transition function has been shown to yield good results in practice [21].

Remark 9: For the sake of brevity, we only show here how to model a multidimensional $\boldsymbol{g}(\cdot)$ using a single kernel $k(\cdot,\cdot)$ for all entries of $\boldsymbol{g}(\cdot)$. However, the methods described herein extend straightforwardly to the case where different kernels are employed for each entry of $\boldsymbol{g}(\cdot)$.

We assume that the model uncertainty due to the unknown function $\boldsymbol{g}(\cdot)$ is faithfully captured by a GP with kernel $k(\cdot,\cdot)$. Formally, this is stated as follows.

Assumption 3: Let $\mathcal{GP}(m,k)$ be a GP with mean $m \equiv 0$ and known continuously differentiable kernel $k(\cdot,\cdot)$. Then the entries of the unknown function $\boldsymbol{g}(\cdot)$ are samples of $\mathcal{GP}(m,k)$, i.e., $g_i \sim \mathcal{GP}(m,k)$ holds for $i = 1, \ldots, N_x$.

Choosing an appropriate kernel $k(\cdot,\cdot)$ requires a priori knowledge of the system. However, the assumptions required for choosing a kernel are generally far less restrictive than for parametric models, since they only pertain to features such as smoothness and periodicity. Furthermore, in some cases, error bounds can be obtained if the kernel is poorly chosen [22].

B. Predicting control performance

Assumption 3 implies that, for a fixed set of parameters $\boldsymbol{\vartheta}$, the expected state of the true system (1) at an arbitrary time step t is given by

$$\mathrm{E}_{\tilde{\boldsymbol{x}}_1,\ldots,\tilde{\boldsymbol{x}}_T}[\boldsymbol{x}_t] = \int_{\mathcal{X}^t} \boldsymbol{x}_t^s \prod_{i=0}^{t-1} \mathrm{p}(\boldsymbol{\zeta}_i^s) d\boldsymbol{\zeta}_i^s, \quad (13)$$

where the integrand is computed recursively using

$$\boldsymbol{x}_{i+1}^s = \boldsymbol{f}(\tilde{\boldsymbol{x}}_i^s) + \boldsymbol{\mu}_i^s(\tilde{\boldsymbol{x}}_i^s) + [\boldsymbol{\Sigma}_i^s(\tilde{\boldsymbol{x}}_i^s) \quad \boldsymbol{\Sigma}_w]\,\boldsymbol{\zeta}_i^s, \quad (14)$$

and $\mathrm{p}(\boldsymbol{\zeta}_i^s) = \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_{2N_x})$.

The corresponding cost function is given by

$$C(\boldsymbol{\vartheta}) = \sum_{t=0}^{T} \int_{\mathcal{X}^t} c_t(\boldsymbol{x}_t^s, \boldsymbol{u}_t^s(\boldsymbol{\vartheta})) \prod_{i=0}^{t-1} p(\boldsymbol{\zeta}_i^s) d\boldsymbol{\zeta}_i^s. \quad (15)$$

Lemma 1: Let Assumptions 2 and 3 hold. Furthermore, let $\sum_{t=0}^{T} c_t(\boldsymbol{x}_t^s, \boldsymbol{u}_t^s(\boldsymbol{\vartheta}))$ be the integrand of (15), where the states are computed using (14) and $\boldsymbol{\zeta}_t^s \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_{2N_x})$ for all t. Then both $\sum_{t=0}^{T} c_t(\boldsymbol{x}_t^s, \boldsymbol{u}_t^s(\boldsymbol{\vartheta}))$ and (15) are continuously differentiable with respect to $\boldsymbol{\vartheta}$.

Proof: Since $k(\cdot,\cdot)$, $c_t(\cdot,\cdot)$, $\boldsymbol{u}_t^s(\cdot)$ are continuously differentiable with respect to their arguments, $\sum_{t=0}^{T} c_t(\boldsymbol{x}_t^s, \boldsymbol{u}_t^s(\boldsymbol{\vartheta}))$ is a composition of continuously differentiable functions. Hence it is continuously differentiable with respect to the control parameters $\boldsymbol{\vartheta}$. Due to Leibniz's rule, this implies that (15) is also continuously differentiable with respect to $\boldsymbol{\vartheta}$. □

IV. Sample average approximation

Computing the integral (15) is generally intractable. Hence, we compute an estimate of the minimizer of (15) by employing a *sample average approximation* (SAA) of (15),

$$C(\boldsymbol{\vartheta}) \simeq C_M(\boldsymbol{\vartheta}, \mathcal{Z}_M) := \frac{1}{M} \sum_{t=0}^{T} \left(\sum_{m=1}^{M} c_t\left(\boldsymbol{x}_t^{(m)}, \boldsymbol{u}_t^{(m)}(\boldsymbol{\vartheta})\right)\right). \quad (16)$$

Here $M \in \mathbb{N}$ is the number of sample trajectories. The set $\mathcal{Z}_M := \left\{\boldsymbol{\zeta}_0^{(1)}, \ldots, \boldsymbol{\zeta}_{T-1}^{(1)}, \ldots, \boldsymbol{\zeta}_0^{(M)}, \ldots, \boldsymbol{\zeta}_{T-1}^{(M)}\right\}$ subsumes MT samples from $\mathcal{N}(\boldsymbol{0}, \boldsymbol{I}_{2N_x})$, which are treated as fixed quantities during optimization. The superscript (m) denotes the m-th sample trajectory, which is computed recursively as

$$\boldsymbol{x}_{t+1}^{(m)} = \boldsymbol{f}\left(\tilde{\boldsymbol{x}}_t^{(m)}\right) + \boldsymbol{\mu}_i^s\left(\tilde{\boldsymbol{x}}_t^{(m)}\right) + \left[\boldsymbol{\Sigma}_i^s\left(\tilde{\boldsymbol{x}}_t^{(m)}\right) \quad \boldsymbol{\Sigma}_w\right]\boldsymbol{\zeta}_t^{(m)},$$

with $\tilde{\boldsymbol{x}}_t^{(m)} := (\boldsymbol{x}_t^{(m)}, \boldsymbol{u}_t^{(m)}(\boldsymbol{\vartheta}))$, $\mathcal{D}_t^{(m)} := \{\tilde{\boldsymbol{x}}_0^{(m)}, \ldots, \tilde{\boldsymbol{x}}_{t-1}^{(m)}\}$. We denote the minimum of the SAA (16) as $C_M^* := \min_{\boldsymbol{\vartheta}\in\boldsymbol{\Theta}} C_M(\boldsymbol{\vartheta}, \mathcal{Z}_M)$, and the corresponding set of minimizers as $\boldsymbol{\Theta}_M^* := \{\boldsymbol{\vartheta} \in \boldsymbol{\Theta} \mid C_M(\boldsymbol{\vartheta}, \mathcal{Z}_M) = C_M^*\}$.

The steps required to compute a minimizer of (16) yield the AntLer algorithm, which is presented in Algorithm 1.

Remark 10: Despite being mainly designed with online learning-based control laws in mind, AntLer can also be

Algorithm 1 Anticipating learning (AntLer)

Input: $\boldsymbol{x}_0$, $\boldsymbol{u}(\cdot,\cdot,\cdot)$, T, M, $\boldsymbol{\Sigma}_w$, $\boldsymbol{f}(\cdot)$, $\boldsymbol{\zeta}_0^{(1)}, \ldots, \boldsymbol{\zeta}_{T-1}^{(M)}$

Solve

$$\begin{aligned}
\boldsymbol{\vartheta}_M^* = \arg\min_{\boldsymbol{\vartheta}} & \sum_{t=0}^{T} \left(\frac{1}{M} \sum_{m=1}^{M} c_t\left(\boldsymbol{x}_t^{(m)}, \boldsymbol{u}_t^{(m)}(\boldsymbol{\vartheta})\right)\right) \\
\text{s.t.} \quad & \boldsymbol{x}_{t+1}^{(m)} = \boldsymbol{f}\left(\tilde{\boldsymbol{x}}_t^{(m)}\right) + \boldsymbol{\mu}_i^s\left(\tilde{\boldsymbol{x}}_t^{(m)}\right) \\
& \qquad + \left[\boldsymbol{\Sigma}_i^s\left(\tilde{\boldsymbol{x}}_t^{(m)}\right) \quad \boldsymbol{\Sigma}_w\right]\boldsymbol{\zeta}_t^{(m)} \\
& \tilde{\boldsymbol{x}}_t^{(m)} = (\boldsymbol{x}_t^{(m)}, \boldsymbol{u}_t^{(m)}(\boldsymbol{\vartheta})) \\
& \tilde{\boldsymbol{x}}_0^{(m)} = (\boldsymbol{x}_0, \boldsymbol{u}_0(\boldsymbol{\vartheta})) \\
& \mathcal{D}_t^{(m)} = \{\tilde{\boldsymbol{x}}_0^{(m)}, \ldots, \tilde{\boldsymbol{x}}_{t-1}^{(m)}\} \\
& \forall\, t \in \{0, \ldots, T-1\},\ m \in \{1,\ \ldots, M\}
\end{aligned}$$

Output: $\boldsymbol{\vartheta}_M^*$

employed in the special case where the control law does not change based on the data collected online. In such settings, AntLer becomes similar in principle to model-based reinforcement learning approaches, e.g., [21].

Remark 11: The algorithm proposed in this paper can also be applied to the infinite-horizon case, e.g., by implementing it in a receding horizon fashion. This would generally require a terminal constraint to be considered, for which probabilistic guarantees can be derived, e.g., as in [23].

We now aim to prove that a solution $\boldsymbol{\vartheta}_M^*$ obtained with AntLer approximates an optimum $\boldsymbol{\vartheta}^* \in \boldsymbol{\Theta}^*$ of the exact problem arbitrarily accurately for a sufficiently high number of samples M. To achieve this, we show that both the approximate and exact cost functions $C_M(\cdot,\cdot)$, $C(\cdot)$, satisfy some regularity conditions.

Lemma 2: Let Assumptions 1–3 hold, and choose $\tilde{\boldsymbol{\Theta}}$ as in Assumption 1. Then $C(\cdot)$ is finite-valued and continuously differentiable on $\tilde{\boldsymbol{\Theta}}$, and $C_M(\cdot, \mathcal{Z}_M)$ converges to $C(\cdot)$ with probability 1 uniformly in $\tilde{\boldsymbol{\Theta}}$ as $M \to \infty$.

To prove Lemma 2, we make use of the following result, which corresponds to [24, Theorem 7.48]:

Lemma 3 ([24]): Let $\tilde{\boldsymbol{\Theta}}$ be a nonempty compact subset of $\boldsymbol{\Theta}$ and suppose that

i) For any $\boldsymbol{\vartheta} \in \tilde{\boldsymbol{\Theta}}$, the function $\sum_{t=0}^{T} c_t(\boldsymbol{x}_t^s, \boldsymbol{u}_t^s(\boldsymbol{\vartheta}))$ is continuously differentiable at $\boldsymbol{\vartheta}$ for almost every sample $\boldsymbol{\zeta}_T^{(m)}$,
ii) The absolute value of $\sum_{t=0}^{T} c_t(\boldsymbol{x}_t^s, \boldsymbol{u}_t^s(\boldsymbol{\vartheta}))$ is upper bounded by an integrable function on the subset $\tilde{\boldsymbol{\Theta}}$,
iii) The samples $\boldsymbol{\zeta}_0^{(m)}, \dots, \boldsymbol{\zeta}_{T-1}^{(m)}$ are i.i.d.

Then $C(\cdot)$ is finite-valued and continuously differentiable on $\tilde{\boldsymbol{\Theta}}$, and $C_M(\cdot, \mathcal{Z}_M)$ converges to $C(\cdot)$ with probability 1 uniformly in $\tilde{\boldsymbol{\Theta}}$ as $M \to \infty$.

Proof of Lemma 2: We show that the conditions of Lemma 3 hold for the compact subset $\tilde{\boldsymbol{\Theta}}$ from Assumption 1.

Since $\tilde{\boldsymbol{\Theta}}$ is bounded, Lemma 1 implies that $\sum_{t=0}^{T} c_t(\boldsymbol{x}_t^s, \boldsymbol{u}_t^s(\boldsymbol{\vartheta}))$ satisfies conditions i) and ii) of Lemma 3. Moreover, the samples $\boldsymbol{\zeta}_1^{(m)}, \dots, \boldsymbol{\zeta}_T^{(m)}$ are i.i.d., i.e., condition iii) of Lemma 3 is also satisfied. □

Using Lemma 2, we are able prove that Algorithm 1 approximates an optimal solution $\boldsymbol{\vartheta}^* \in \boldsymbol{\Theta}^*$ arbitrarily accurately with probability 1 for large enough M. This corresponds to our main result, and is stated in the following theorem.

Theorem 1: Let $\boldsymbol{\zeta}_t^{(m)} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_{N_x})$, $t \in \{0, \dots, T-1\}$, $m \in \{1, \dots, \infty\}$ be a fixed sequence of random samples. For every M, let $\boldsymbol{\vartheta}_M^*$ denote a vector of approximate optimal parameters obtained with Algorithm 1 and the samples $\boldsymbol{\zeta}_0^{(1)}, \dots, \boldsymbol{\zeta}_{T-1}^{(M)}$. Moreover, let Assumptions 1–3 hold. Then, for every $\epsilon > 0$, there exists an $M_\epsilon \in \mathbb{N}$, such that $|C_M^* - C^*| \leq \epsilon$ and $\min_{\boldsymbol{\vartheta}^* \in \boldsymbol{\Theta}^*} \|\boldsymbol{\vartheta}_M^* - \boldsymbol{\vartheta}^*\|_2 \leq \epsilon$ holds for all $M \geq M_\epsilon$ with probability 1.

We prove Theorem 1 by employing [24, Theorem 5.3], which we now state.

Lemma 4 ([24]): Suppose there exists a compact subset $\tilde{\boldsymbol{\Theta}} \subseteq \boldsymbol{\Theta}$, such that

i) $\boldsymbol{\Theta}^*$ is non-empty and $\boldsymbol{\Theta}^* \subseteq \tilde{\boldsymbol{\Theta}}$,
ii) The function $C(\boldsymbol{\vartheta})$ is finite-valued and continuously differentiable on $\tilde{\boldsymbol{\Theta}}$,
iii) $C_M(\boldsymbol{\vartheta}, \mathcal{Z}_M)$ converges to $C(\boldsymbol{\vartheta})$ with probability 1 as $M \to \infty$, uniformly in $\boldsymbol{\vartheta} \in \tilde{\boldsymbol{\Theta}}$,
iv) With probability 1, for M large enough, the set $\boldsymbol{\Theta}_M^*$ is nonempty and $\boldsymbol{\Theta}_M^* \subseteq \tilde{\boldsymbol{\Theta}}$.

Then $C_M^* \to C^*$, $\max_{\boldsymbol{\vartheta}_M^* \in \boldsymbol{\Theta}_M^*} \min_{\boldsymbol{\vartheta}^* \in \boldsymbol{\Theta}^*} \|\boldsymbol{\vartheta}_M^* - \boldsymbol{\vartheta}^*\|_2 \to 0$ holds with probability 1 as $M \to \infty$.

Proof of Theorem 1: We show that the conditions of Lemma 4 hold for the compact subset $\tilde{\boldsymbol{\Theta}}$ from Assumption 1.

Conditions ii) and iii) are satisfied due to Lemma 2. Hence, it remains to be shown that i) and iv) hold.

We begin by showing that the set $\boldsymbol{\Theta}^*$ is nonempty. To this end, consider an arbitrary sequence of control parameters $\boldsymbol{\vartheta}_i$, $i = 1, \dots, \infty$, with $\lim_{i\to\infty} C(\boldsymbol{\vartheta}_i) = C^*$. Due to Assumption 1 and the continuity of $C(\cdot)$ (i.e., Lemma 1), there exists an $I \in \mathbb{N}$, such that $\boldsymbol{\vartheta}_i \in \tilde{\boldsymbol{\Theta}}$ holds for all $i \geq I$. Since $\boldsymbol{\vartheta}_1, \dots, \boldsymbol{\vartheta}_I$ are finite-valued, this implies that the sequence $\boldsymbol{\vartheta}_i$, $i = 1, \dots, \infty$ belongs to a compact set. Due to the Bolzano-Weierstrass theorem, $\boldsymbol{\vartheta}_i$ contains a convergent subsequence with limit $\boldsymbol{\vartheta}^* \in \tilde{\boldsymbol{\Theta}}$. Hence, $\boldsymbol{\Theta}^*$ is nonempty and $\boldsymbol{\Theta}^* \subseteq \tilde{\boldsymbol{\Theta}}$, i.e., Condition i) of Lemma 4 is satisfied. Using the same argument we can show that $\boldsymbol{\Theta}_M^*$ is nonempty. Moreover, Assumption 1 implies that $\boldsymbol{\Theta}_M^* \subseteq \tilde{\boldsymbol{\Theta}}$ holds for all $\mathcal{Z}_M$, i.e., Condition iv) is satisfied. □

Hence, the AntLer algorithm approximates an optimal vector of data-independent parameters $\boldsymbol{\vartheta}^*$ with arbitrary accuracy for large enough M with probability 1.

For a control law $\boldsymbol{u}_t(\boldsymbol{\vartheta})$ that potentially improves its performance through online learning, Theorem 1 implies that AntLer guarantees superior control performance for large enough M compared to the case where learning is not anticipated. This is shown by comparing predictions for $\boldsymbol{u}_t(\cdot)$ to predictions for its data independent counterpart $\boldsymbol{u}_t^0(\cdot) = (\mathcal{D}_0, \cdot, \boldsymbol{\vartheta})$. We state this formally in the following.

Assumption 4: Let

$$C^0(\boldsymbol{\vartheta}) \coloneqq \mathrm{E}_{\tilde{\boldsymbol{x}}_1, \dots, \tilde{\boldsymbol{x}}_T} \left[\sum_{t=0}^{T} c_t(\boldsymbol{x}_t, \boldsymbol{u}(\mathcal{D}_0, \boldsymbol{x}_t, \boldsymbol{\vartheta})) \right], \quad (17)$$

be the cost function under the data-independent counterpart $\boldsymbol{u}_t^0(\boldsymbol{\vartheta}) = \boldsymbol{u}(\mathcal{D}_0, \boldsymbol{\vartheta}, \boldsymbol{x}_t)$, and let $C^{0,*} \coloneqq \min_{\boldsymbol{\vartheta}} C^0(\boldsymbol{\vartheta})$ be its minimum. Then $C^{0,*} < C^*$, where C^* is the minimum of (2).

This amounts to assuming that $\boldsymbol{u}_t(\boldsymbol{\vartheta})$ potentially improves its performance as new data is gathered.

Corollary 1: Let Assumptions 1–4 hold, and let $C(\cdot)$ be given as in (2). Furthermore, let $C^{0,*}$ be the optimal cost under the data-independent counterpart, as given in Assumption 4, and let $\boldsymbol{\zeta}_t^{(m)} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}_{N_x})$, $t \in \{0, \dots, T-1\}$, $m \in \{1, \dots, \infty\}$, be a fixed sequence of random samples. For every $M \in \mathbb{N}$, let $\boldsymbol{\vartheta}_M^*$ denote the approximate optimal solution obtained with Algorithm 1 and the samples $\boldsymbol{\zeta}_0^{(1)}, \dots, \boldsymbol{\zeta}_{T-1}^{(M)}$. Then

there exists an M^0, such that $C(\boldsymbol{\vartheta}_M^*) < C^{0,*}$ holds for all $M \geq M^0$ with probability 1.

Proof: This follows directly from Theorem 1. □

V. Numerical example

We now illustrate the proposed algorithm using a simple nonlinear trajectory tracking problem. We demonstrate the convergence of the approximate optimal parameters computed by AntLer as the number of samples M grows, and compare the computed parameters and predictions to those obtained without anticipating learning. Furthermore, by preforming Monte Carlo simulations of the true system, we showcase the superior performance of the approximate optimal parameters compared to the case where learning is not anticipated.

The source code of the experiments presented in this section is available at https://git.lsr.ei.tum.de/acapone/antler.

A. System description

We consider the one-dimensional system

$$x_{t+1} = f(\tilde{x}_t) + g(\tilde{x}_t) + w_t, \tag{18}$$

with initial state $x_0 = 0$, process noise $w_t \sim \mathcal{N}(0, 0.01^2)$, and state transition functions

$$f(\tilde{x}_t) = x_t + u_t, \tag{19}$$

$$g(\tilde{x}_t) = 0.85\sin(12x_t) + x_t^2(\exp(-0.2x_t^2)). \tag{20}$$

We aim to design an online learning-based control law that tracks the trajectory $x_t^{\text{ref}} = 4\sin(t/2\pi)$ as accurately as possible, while simultaneously accounting for any potential tracking errors due to the unknown function $g(\cdot)$. To this end we choose the control law

$$u_t(\boldsymbol{\vartheta}) = -\mu_t(x_t) - \vartheta_1(x_t - \vartheta_2 x_t^{\text{ref}}), \tag{21}$$

where ϑ_1 acts as a control gain, and ϑ_2 scales the reference trajectory and enables to avoid regions of high model uncertainty. The term $\mu_t(x_t)$ is a GP mean, which is updated online as new data points are collected. We compute $\mu_t(x_t)$ using the same kernel as for AntLer, which we specify in the sequel. Employing the same kernel both for predictions and control is reasonable, since we assume that it faithfully represents the unknown function $g(\cdot)$.

We quantify control performance by employing the cost function

$$C(\boldsymbol{\vartheta}) = \mathrm{E}_{\tilde{\boldsymbol{x}}_1,\ldots,\tilde{\boldsymbol{x}}_{150}}\left[\sum_{t=0}^{150} c_t\right], \tag{22}$$

where the immediate cost terms $c_t := (x_t - x_t^{\text{ref}})^2$ penalize deviations from the reference trajectory.

We now describe the kernel used for AntLer predictions and the online learning-based control law (21). We assume to know that $g(\cdot)$ depends only on the state x_t, and that it corresponds to a smooth function. This information is encoded into the GP by employing a squared exponential kernel that takes only the state as argument, i.e.,

$$k(\tilde{x}_i, \tilde{x}_j) =: k(x_i, x_j) = \sigma_k^2 \exp\left(-\frac{(x_i - x_j)^2}{2l}\right), \tag{23}$$

where the signal variance $\sigma_k^2 \in \mathbb{R}_+$ and length scale $l \in \mathbb{R}_+$ are obtained by training the GP using log marginal likelihood optimization [20]. To this end, we assume to have 100 measurements of (18), which were obtained using a control law that attempts to minimize the distance of the true system (18) to the origin. Squared exponential kernels are dense within the space of continuous functions on compact sets, i.e., they can approximate any continuous function uniformly and arbitrarily well on compact subsets of $\mathcal{X}$ [25]. Moreover, the posterior mean $\mu_t(\cdot)$ of a GP obtained with a squared exponential kernel exhibits smooth behavior [20]. Hence, (23) is an appropriate choice for this setting.

It can easily be shown that, in a setting where $g(x_t)$ is known, i.e., $\mu_t(x_t) = g(x_t)$, the system trajectory is optimal for $\vartheta_1 = \vartheta_2 = 1$. Since the control law (21) learns $g(\cdot)$ online, it is reasonable to expect that the optimal parameters $\boldsymbol{\vartheta}^*$ for unknown $g(\cdot)$ lie within a neighborhood of $\vartheta_1 = \vartheta_2 = 1$, provided that $g(\cdot)$ is learned correctly. Hence, we assume that the optimal parameters lie within the compact subset $\tilde{\boldsymbol{\Theta}} = [-1, -1] \times [2, 2]$. In the following, we employ this assumption to restrict the feasible region of the optimization problem to $\tilde{\boldsymbol{\Theta}}$.

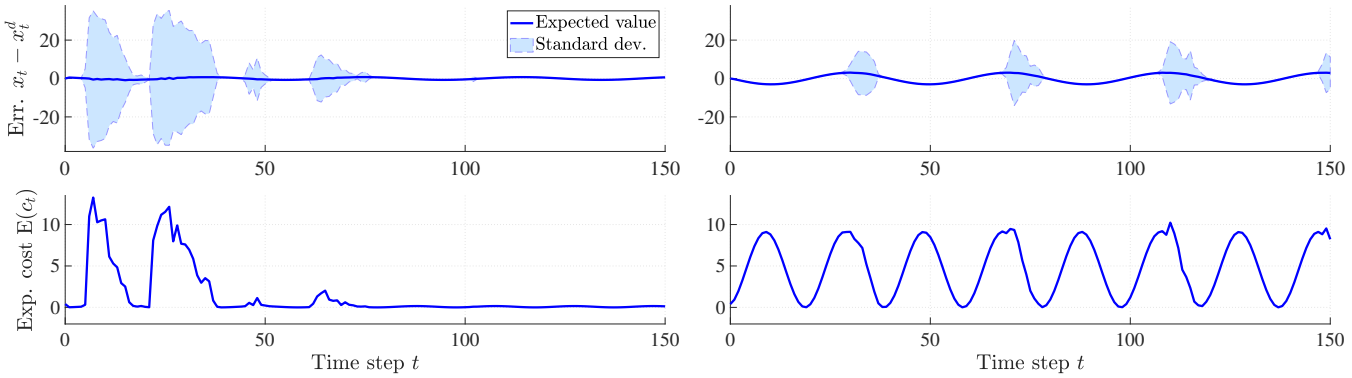
B. Approximate optimal solutions using AntLer

We demonstrate the convergence of the approximate optimal solution $\boldsymbol{\vartheta}_M^*$ to $\boldsymbol{\vartheta}^*$ as M grows by applying AntLer using $M \in \{2, 10, 50, 100, 200\}$ samples. Additionally, in order to illustrate Corollary 1, we make predictions and optimize the parameters $\boldsymbol{\vartheta}$ without anticipating learning, i.e., by using the data-independent counterpart of the control law $u_t^0(\boldsymbol{\vartheta}) = -\mu_0(x_t) - \vartheta_1(x_t - \vartheta_2 x_t^{\text{ref}})$. To optimize the parameters of $u_t^0(\boldsymbol{\vartheta})$, we employ AntLer with $M = 200$. We are able to do so, since $u^0(\cdot)$ is a special case of an online learning-based control law. Hence, approximate optimal parameters can also be obtained using AntLer. For simplicity of exposition, we henceforth refer to $\boldsymbol{\vartheta}^{*,0}$ as the optimum of the data-independent counterpart.

To solve the SAA problem in AntLer, we employ a gradient-based method with different starting values, which are sampled from the uniform distribution on $\tilde{\boldsymbol{\Theta}}$. A solution is found after at most 17 gradient-descent steps. In Table I, we display the approximate optimal parameters $\boldsymbol{\vartheta}_M^*$ computed by Antler. In Fig. 1a, we present AntLer predictions

TABLE I: Approximate optimal parameters $\boldsymbol{\vartheta}_M^*$ computed by AntLer for different M.

M	2	10	50	100	200
$\boldsymbol{\vartheta}_M^*$	$(0.9, 0.9)^{\mathrm{T}}$	$(1.1,\ 1)^{\mathrm{T}}$	$(1, 0.9)^{\mathrm{T}}$	$(1, 0.9)^{\mathrm{T}}$	$(1, 0.9)^{\mathrm{T}}$



(a) Predicted optimum when learning is anticipated.

(b) Predicted optimum when learning is not anticipated.

Fig. 1: Predicted optima (a) with and (b) without anticipating learning. Both predictions are carried out using AntLer and $M = 200$ samples. The top rows show the predicted optimal tracking error $x_t - x_t^{\mathrm{ref}}$, the bottow rows show the expected immediate cost $\mathrm{E}[c_t]$. (a) Prediction for approximate optimal online learning-based control law $u_t(\boldsymbol{\vartheta}_M^*)$, where $\boldsymbol{\vartheta}_M^* = (1, 0.9)^{\mathrm{T}}$; predicted cost is $C_M(\boldsymbol{\vartheta}_M^*, \mathcal{Z}_M) = 212$. (b) Prediction for approximate optimal data-independent counterpart $u_t^0(\boldsymbol{\vartheta}^{0,*})$, where $\boldsymbol{\vartheta}^{0,*} = (0.9, 0.2)^{\mathrm{T}}$; predicted cost is $C_M(\boldsymbol{\vartheta}_M^*, \mathcal{Z}_M) = 731$.

for $M = 200$ and the approximate optimal online learning-based law $u_t(\boldsymbol{\vartheta}_M^*)$. Furthermore, in Fig. 1b we show predictions for the optimal data-independent counterpart $u_t^0(\boldsymbol{\vartheta}^{0,*})$.

The value of the approximate optimal parameters is $\boldsymbol{\vartheta}_M^* \approx (1, 0.9)^{\mathrm{T}}$ for all $50 < M < 200$. This indicates that $\boldsymbol{\vartheta}_M^*$ has converged to a small neighborhood of the optimal parameters $\boldsymbol{\vartheta}^*$, as expected from Theorem 1.

AntLer predicts that, by scaling the reference trajectory with $\vartheta_2 = 0.9$, an optimal trade-off is achieved between the information of the collected data and the error caused by model uncertainty. In other words, if the control law were to attempt to fully enforce the reference trajectory, i.e., $\vartheta_2 = 1$, then AntLer predicts that too many measurements need to be collected before good tracking performance is achieved. However, if $\vartheta_2 = 0.9$ is chosen, then AntLer predicts that the unknown dynamics will be learned quickly enough to achieve good tracking performance within the time horizon $T = 150$. This becomes apparent in the predictions in Fig. 1a. Therein, the variance of the state x_t and the expected immediate cost c_t under the approximate optimal control law $u_t(\boldsymbol{\vartheta}_M^*)$ decrease over time. After $t \approx 70$, they become approximately zero.

The parameters $\boldsymbol{\vartheta}^{0,*} = (0.9, 0.2)^{\mathrm{T}}$ of the optimal data-independent counterpart $u_t(\boldsymbol{\vartheta}^{0,*})$ attempt to keep the system close to the origin. This is because predictions for $u_t^0(\boldsymbol{\vartheta})$ do not anticipate learning. In other words, they only yield low tracking errors in regions where model uncertainty is already low. As measurement data at the origin was collected prior to the control design, model uncertainty is high in the whole state space except for a neighborhood of the origin. Hence the approximate optimal parameters $\boldsymbol{\vartheta}^{0,*} = (0.9, 0.2)^{\mathrm{T}}$ attempt to keep the system within this region. This is reflected in the predictions in Fig. 1b, where the tracking error exhibits little variance compared to Fig. 1a.

For $M = 200$, the predicted cost $C(\boldsymbol{\vartheta}_M^*) = 212$ under the approximate optimal control law $u_t(\boldsymbol{\vartheta}_M^*)$ is lower than the predicted cost $C(\boldsymbol{\vartheta}^{0,*}) = 731$ under the data-independent counterpart $u_t(\boldsymbol{\vartheta}^{0,*})$. Assuming that the GP specified by the kernel (23) correctly captures the model uncertainty due to $g(\cdot)$, Corollary 1 implies that control performance will be superior if $\boldsymbol{\vartheta}_M^*$ is applied to the true system instead of $\boldsymbol{\vartheta}^{0,*}$. This indeed is the case, as shown in the following.

C. Monte Carlo simulations of true system

The parameters $\boldsymbol{\vartheta}_M^* = (1, 0.9)^{\mathrm{T}}$ computed by AntLer for $M = 200$ are employed to control the true system (18) in 100 Monte Carlo runs. Moreover, we compare the results to the Monte Carlo simulation using the optimal parameters obtained without anticipating learning $\boldsymbol{\vartheta}^{0,*} = (0.9, 0.2)^{\mathrm{T}}$. The respective results are shown in Fig. 2a and Fig. 2b.

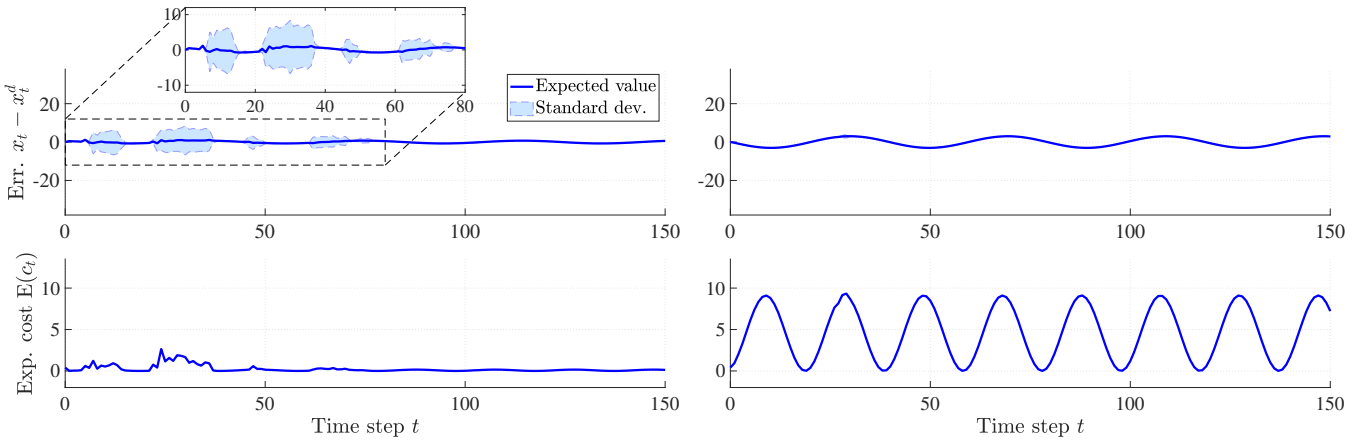
As shown in Fig. 2a, the variance of the state is high for $\boldsymbol{\vartheta} = \boldsymbol{\vartheta}_M^*$ at the beginning of the Monte Carlo simulation. This is due to the initially unknown system dynamics $g(\cdot)$. After approximately $t = 80$, enough measurement data has been gathered to adequately track the reference trajectory. Despite differences in overall variance and learning time, the results agree qualitatively with the AntLer prediction shown in Fig. 1a, which indicates that the kernel (23) was chosen adequately.

For $\boldsymbol{\vartheta} = \boldsymbol{\vartheta}^{0,*}$, the variance of the state is very low throughout the simulation. This is because the parameters $\boldsymbol{\vartheta} = \boldsymbol{\vartheta}^{0,*}$ steer the system to a region of low model uncertainty. These results are in agreement with the predictions presented in Fig. 1b.

The average cumulative cost for $\boldsymbol{\vartheta} = \boldsymbol{\vartheta}^{0,*}$ is 859. This is higher than for $\boldsymbol{\vartheta} = \boldsymbol{\vartheta}_M^*$, which achieves an average cost of 37. This was expected from the AntLer predictions and Corollary 1.

VI. Conclusion

We have presented AntLer, a control design approach that anticipates the effect of online learning and optimizes data-independent parameters accordingly. By expressing model



(a) Parameters optimized by anticipating learning.

(b) Parameters optimized without anticipating learning.

Fig. 2: Monte Carlo simulations of true system (18) consisting of 100 runs. The top rows show the tracking error $x_t - x_t^{\mathrm{ref}}$, the bottom rows show the average immediate cost $\mathrm{E}[c_t]$. (a) True system under approximate optimal online learning-based control law $u_t(\boldsymbol{\vartheta}_M^*)$, where $\boldsymbol{\vartheta}_M^* = (1,\ 0.9)^{\mathrm{T}}$ was obtained by anticipating learning; the average total cost is 37. (b) True system under $u_t(\boldsymbol{\vartheta}^{0,*})$, where $\boldsymbol{\vartheta}^{0,*} = (0.9,\ 0.2)^{\mathrm{T}}$ was obtained without anticipating learning; the average total cost is 859.

uncertainty with a Gaussian process model, we have formulated the parameter optimization problem as a stochastic optimal control problem, which AntLer solves approximately using sample average approximation. We have shown that AntLer approximates an optimal solution arbitrarily accurately with probability one for a sufficiently large number of samples. We have applied AntLer to a nonlinear system. The results have shown that model learning is correctly anticipated, which leads to a better choice of control parameters compared to the case where learning is not anticipated.

In future work, we aim to apply AntLer to complex online learning-based control laws, such as learning-based model predictive control and online reinforcement learning.

References

[1] E. D. Klenske, M. N. Zeilinger, B. Schölkopf, and P. Hennig, "Gaussian process-based predictive control for periodic error correction," *IEEE Transactions on Control Systems Technology*, vol. 24, no. 1, pp. 110–121, 2016.

[2] R. Murray-Smith and D. Sbarbaro, "Nonlinear adaptive control using nonparametric Gaussian process prior models," *IFAC Proceedings Volumes*, vol. 35, no. 1, pp. 325–330, 2002.

[3] S. Kamthe and M. Deisenroth, "Data-efficient reinforcement learning with probabilistic model predictive control," in *International Conference on Artificial Intelligence and Statistics*, 2018, pp. 1701–1710.

[4] T. Koller, F. Berkenkamp, M. Turchetta, and A. Krause, "Learning-based model predictive control for safe exploration," in *2018 IEEE Conference on Decision and Control (CDC)*, 2018, pp. 6059–6066.

[5] J. Umlauft and S. Hirche, "Feedback linearization based on Gaussian processes with event-triggered online learning," *IEEE Transactions on Automatic Control*, pp. 1–1, 2019.

[6] G. Chowdhary, H. A. Kingravi, J. P. How, P. A. Vela *et al.*, "Bayesian nonparametric adaptive control using Gaussian processes." *IEEE Trans. Neural Netw. Learning Syst.*, vol. 26, no. 3, pp. 537–550, 2015.

[7] D. Nguyen-Tuong and J. Peters, "Model learning for robot control: a survey," *Cognitive processing*, vol. 12, no. 4, pp. 319–340, 2011.

[8] B. Bakker, V. Zhumatiy, G. Gruener, and J. Schmidhuber, "Quasi-online reinforcement learning for robots," in *2006 IEEE International Conference on Robotics and Automation*, May 2006, pp. 2997–3002.

[9] K. J. Astrom and B. Wittenmark, *Adaptive Control*, 2nd ed. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1994.

[10] M. Krstic, I. Kanellakopoulos, and P. V. Kokotovic, *Nonlinear and adaptive control design*. Wiley, 1995.

[11] J. Kocijan, *Modelling and control of dynamic systems using Gaussian process models*. Springer, 2016.

[12] F. Berkenkamp and A. P. Schoellig, "Safe and robust learning control with Gaussian processes," in *2015 IEEE European Control Conference*, pp. 2496–2501.

[13] P. Dayan and T. J. Sejnowski, "Exploration bonuses and dual control," *Machine Learning*, vol. 25, no. 1, pp. 5–22, 1996.

[14] Y. Bar-Shalom and E. Tse, "Dual effect, certainty equivalence, and separation in stochastic control," *IEEE Transactions on Automatic Control*, vol. 19, no. 5, pp. 494–500, 1974.

[15] E. D. Klenske and P. Hennig, "Dual control for approximate Bayesian reinforcement learning." *Journal of Machine Learning Research*, vol. 17, no. 127, pp. 1–30, 2016.

[16] L. Král, J. Prüher, and M. Šimandl, "Gaussian process based dual adaptive control of nonlinear stochastic systems," in *22nd Mediterranean Conference on Control and Automation*. IEEE, 2014, pp. 1074–1079.

[17] F. Berkenkamp, R. Moriconi, A. P. Schoellig, and A. Krause, "Safe learning of regions of attraction for uncertain, nonlinear systems with Gaussian processes," in *2016 IEEE Conference on Decision and Control*, pp. 4661–4666.

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

[19] 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*, 2019.

[20] C. E. Rasmussen and C. K. Williams, "Gaussian processes for machine learning. 2006," *The MIT Press, Cambridge, MA, USA*, 2006.

[21] M. P. Deisenroth, D. Fox, and C. E. Rasmussen, "Gaussian processes for data-efficient learning in robotics and control," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 2, pp. 408–423, 2015.

[22] T. Beckers, J. Umlauft, and S. Hirche, "Mean square prediction error of misspecified gaussian process models," in *2018 IEEE Conference on Decision and Control*, Dec 2018, pp. 1162–1167.

[23] A. Lederer, H. Qing, and S. Hirche, "Confidence regions for simulations with learned probabilistic models," *American Control Conference*, pp. 1–1, 2020.

[24] A. Shapiro, D. Dentcheva, and A. Ruszczyński, *Lectures on stochastic programming: modeling and theory*. SIAM, 2009.

[25] C. A. Micchelli, Y. Xu, and H. Zhang, "Universal kernels," *Journal of Machine Learning Research*, vol. 7, no. Dec, pp. 2651–2667, 2006.