Greedy Covariance Control for Stochastic Nonlinear Systems using Gaussian Process Prediction Models and the Unscented Transform

Alexandros Tsolovikos

Efstathios Bakolas

Abstract—In this work, the problem of steering the first two moments of the uncertain state of an unknown discrete-time stochastic system to a given terminal distribution in finite time is considered. Toward that goal, first, a non-parametric prediction model is learned from a set of available training data points using Gaussian process regression: a powerful machine learning tool for learning distributions over arbitrary nonlinear functions. Second, a tractable nonlinear covariance steering algorithm that utilizes the Gaussian process prediction model to compute a feedback policy that will drive the probability density function of the state of the system close to the goal density is formulated. In particular, a greedy covariance steering control policy is implemented that linearizes the Gaussian process prediction model around the latest mean and covariance predictions at each time step and solves the linear covariance steering control problem, which can be formulated as a tractable, finite-dimensional convex program. Then, only the first control law of the solution to the linear problem is applied. At each step, the information on the state statistics is updated by computing approximations of the predicted state mean and covariance of the resulting closed-loop nonlinear system step using the unscented transform and the learned Gaussian process prediction model. Numerical simulations illustrating the main ideas of this paper are also presented.

I. INTRODUCTION

In this paper, we consider the finite-horizon covariance steering problem for discrete-time stochastic nonlinear (DTSN) systems described by non-parametric Gaussian process models. In particular, we consider the problem of learning Gaussian process prediction models for DTSN systems from training data and then steering the mean and covariance of the uncertain state of such systems to desired quantities at a given (finite) terminal time. This problem will be referred to as the Gaussian process-based nonlinear covariance steering problem.

Literature Review: Gaussian Processes (GP) [1] are non-parametric regression models that describe distributions over functions and are ideal for learning predictive models for arbitrary nonlinear stochastic systems due to their flexibility and inherent ability to provide uncertainty estimates that capture both model uncertainties and process noise. GP regression models have been used extensively for learning predictive state models for dynamical systems [2]–[4] and observation models for state estimation [5], [6], as well as trajectory optimization [7], [8] and motion planning [9].

The infinite-horizon covariance steering (or covariance control) problem for both continuous-time and discrete-time linear Gaussian systems has been studied extensively [10]–[14]), while the finite-horizon problem has been addressed in [15], [16] for the continuous-time and in [17]–[19] for the discrete-time case. Covariance control problems for incomplete and imperfect state information have also been studied in [20]–[22]. Nonlinear density steering problems for feedback linearizable nonlinear systems were recently studied in [23], while an iterative covariance steering algorithm for nonlinear systems based on a linearization of the system along reference state and input trajectories was presented in [24]. Stochastic nonlinear model predictive control with probabilistic constraints can also be found in [25], [26].

Main Contribution: In this work, non-parametric state prediction models of discrete-time stochastic nonlinear systems with unknown dynamics are learned using Gaussian process regression and are used to control the mean and covariance of the state of the unknown systems in a greedy algorithm similar to the one presented in [27]. To the best of our knowledge, this is the first paper to utilize Gaussian process prediction models for the purpose of discrete-time nonlinear covariance steering.

First, we present the process of learning a non-parametric GP prediction state model from a set of training samples obtained by measuring the stochastic nonlinear system of interest. Then, the non-parametric prediction model is used in a greedy finite-horizon covariance steering algorithm.

The Gaussian process-based greedy nonlinear covariance steering algorithm consists of three steps. In the first step, the non-parametric GP model is linearized around the latest state prediction or estimation. This linearization is repeated at each time step as new information becomes available. In the second step, the feedback control policy that solves the linear Gaussian covariance steering for the linearized system is computed, but only the first control law is executed. In the third step, the state mean and covariance of the closed-loop system that results by applying the feedback control policy computed at the previous step are propagated to the next time step using the unscented transform [28], [29], modified to take into account the uncertainty estimates provided by the GP prediction model [5]. This three-step process is repeated until the final time step, when the terminal state mean and covariance should sufficiently approximate the
goal quantities.

Structure of the paper: The rest of the paper is organized as follows. In Section II the process of learning a prediction model from sample data points using GP regression is presented. The nonlinear covariance steering problem is formulated in Section III. In Section IV the greedy algorithm of [27] is reformulated to incorporate the non-parametric GP prediction model. Section V presents numerical simulations. We conclude with remarks and directions for future research in Section VI.

II. System Identification Using Gaussian Process Regression

A. Notation

We denote by $\mathbb{R}^n$ the set of $n$-dimensional real vectors. Given integers $\alpha, \beta$ with $\alpha \leq \beta$, the discrete interval from $\alpha$ to $\beta$ is denoted by $[\alpha, \beta]$. $E[\cdot]$ is the expectation operator. Given a random vector $x$, $E[x]$ denotes its mean and $\text{Cov}[x]$ its covariance, where $\text{Cov}[x] := E[(x - E[x])(x - E[x])^T]$. The space of real symmetric $n \times n$ matrices will be denoted by $S_n$. Furthermore, the convex cone of $n \times n$ (symmetric) positive semi-definite and (symmetric) positive definite matrices will be denoted by $S^n_+$ and $S^{++}_n$, respectively. Given a matrix $\Sigma \in S^n_+$, a vector $\bar{x} \in \mathbb{R}^n$ and a positive scalar $\alpha$, we denote by $E_{\alpha}(\bar{x}, \Sigma)$ the ellipsoid $\{x \in \mathbb{R}^n : (x - \bar{x})^{T}\Sigma^{-1}(x - \bar{x}) \leq \alpha\}$. In addition, $b\text{diag}(A_1, \ldots, A_l)$ is the block diagonal matrix formed by the matrices $A_i$, $i \in \{1, \ldots, l\}$. Finite-length sequences are denoted as $\{x_1, \ldots, x_N\} = \{x_i\}_{i=1}^N$. Finally, if $x \in \mathbb{R}^n$ and $y \in \mathbb{R}^m$, then $[x; y] = [x^T, y^T]^T \in \mathbb{R}^{n+m}$ will denote the vector formed by stacking $x$ and $y$ together.

B. Problem setup

Consider a nonlinear stochastic system that evolves according to

$$x_{k+1} = f(x_k, u_k) + \varepsilon_k,$$

(1)

where $x_k, x_{k+1} \in \mathbb{R}^n$ are the states at time steps $k$ and $k+1$, respectively, $u_k \in \mathbb{R}^m$ is the input, $\varepsilon_k \in \mathbb{R}^n$ is the process noise, which is assumed to be an independent and identically distributed (i.i.d.) random variable with

$$E[\varepsilon_k] = 0, \quad E[\varepsilon_k \varepsilon_j^T] = \delta_{k,j} Q_k,$$

(2)

for all positive integers $k, j$, where $Q_k \in \mathbb{R}^n_+$ and $\delta_{k,j}$ is the Kronecker delta, i.e. $\delta_{k,j} := 1$ when $k = j$ and $\delta_{k,j} := 0$, otherwise.

Assume that the state transition function $f(\cdot)$ is unknown, but we have access to a set of (noisy, in general) training data $D = \{(x^{(i)}, u^{(i)}), y^{(i)}\}_{i=1}^M$, where $y^{(i)} = f(x^{(i)}, u^{(i)}) + \epsilon^{(i)}$ is the state transition vector and $M$ is the number of training data points. The first goal of this paper is to learn a non-parametric state transition model from the available data $D$ using Gaussian process regression. This non-parametric prediction model will then be used to design a control policy that will steer the mean and covariance of the state of the system from an initial distribution to a desired terminal one in finite time.

C. Gaussian Process Regression

Gaussian processes (GP) are a powerful and flexible tool for learning non-parametric regression models from training data. A GP can be thought of as defining a distribution over functions. One of the key advantages of GPs is that the predictions from GP models have the form of a full predictive distribution, i.e., predictions come with uncertainty estimates that take into account both process noise and regression uncertainty due to limited training data.

The standard regression problem is to approximate a function $f(\cdot)$ from a set of training data $D = \{(x^{(i)}, y^{(i)})\}_{i=1}^M$, drawn from the noisy process

$$y^{(i)} = f(x^{(i)}) + \epsilon^{(i)},$$

(3)

where $x^{(i)} \in \mathbb{R}^n$ is the input vector and $y^{(i)} \in \mathbb{R}$ is the observed scalar output that differs from the function value $f(x^{(i)})$ by the additive noise $\epsilon^{(i)}$ that is assumed to be independent and identically distributed Gaussian with zero mean and variance $\sigma^2_n$.

$$\epsilon^{(i)} \sim \mathcal{N}(0, \sigma^2_n).$$

(4)

Stacking the training inputs as $X = [x^{(1)}, \ldots, x^{(M)}]$ and $y = [y^{(1)}, \ldots, y^{(M)}]^T$, one can write the joint Gaussian distribution over the outputs $y$ as

$$p(y) = \mathcal{N}(0, K + \sigma^2_n I),$$

(5)

where $K$ is the kernel matrix with elements $K_{i,j} = k(x^{(i)}, x^{(j)})$, $i, j \in [1, M]$, and $k(x, x')$ is the kernel function, which measures the closeness between the inputs.

Given the training data set $D$ and a test input $x_*$, the Gaussian predictive distribution over the output $y_*$ is

$$p(y_* | x_*, D) = \mathcal{N}(GP_{\mu}(x_*, D), GP_{\Sigma}(x_*, D)),$$

(6)

where

$$GP_{\mu}(x_*, D) = K_*^{T} [K + \sigma^2_n I]^{-1} y,$$

(7)

$$GP_{\Sigma}(x_*, D) = k(x_*, x_*) - K_*^{T} [K + \sigma^2_n I]^{-1} K_*$$

(8)

are the posterior mean and covariance of $y_*$ conditioned on $x_*$ and $D$. Note that $k_*$ is the kernel vector between the training inputs, i.e., the columns of $X$, and the test input $x_*$. A common choice for the kernel function is the squared exponential kernel,

$k(x, x') = \sigma^2 \exp \left(-\frac{1}{2} (x - x')^{T} L^{-1} (x - x') \right),$

(9)

where $\sigma^2$ is the signal variance and $L = \text{diag}(l_1^2, \ldots, l_n^2)$ is the diagonal matrix containing the length scales for each input dimension. The squared exponential kernel will be used throughout this paper.

The parameters $\theta = [\sigma^2, L, \sigma^2_n]$ are the hyperparameters of the GP and can be learned by maximizing the log likelihood of the training outputs $y$ conditioned on the inputs $X$, 

$$\theta_{\text{max}} = \underset{\theta}{\text{argmax}} \{ \log (p(y|X, \theta)) \},$$

(10)
which can be solved using a number of numerical optimization methods [1].

D. Learning Prediction Models

We apply Gaussian process regression to learn a prediction model of the dynamical system (1). Assume that \( f(\cdot) \) is unknown, but a set of noisy training data \( D = \{([x(i); u(i)], y(i))\}_{i=1}^M \) is available, where \( y(i) = f(x(i), u(i)) + \epsilon(i) \) is the state transition vector. The goal is to learn a separate GP for each element of the \( n \)-dimensional state transition function \( f(\cdot) \), as well as the process noise covariance, \( Q_k \), defined in (13).

Let \( X = [[x(1); u(1)], \ldots, [x(M); u(M)]] \in \mathbb{R}^{(n+m) \times M} \) and \( Y = [y(1), \ldots, y(M)]^T \in \mathbb{R}^{M \times n} \) be the concatenation of the training input and output data. Denote \( X_j \) and \( Y_j \) the \( j \)-th columns of \( X \) and \( Y \), respectively. Also, denote the data sets \( D_j = \{X_j, Y_j\} \), for \( j = 1, n, \ldots, M \). Then, the GP approximation of (1) can be written as

\[
x_{k+1} = \text{GP}_\mu([x_k; u_k]|D) + w_k, \tag{11}
\]

where

\[
\text{GP}_\mu([x_k; u_k]|D) = \begin{bmatrix}
GP^1_\mu([x_k; u_k]|D_1) \\
\vdots \\
GP^M_\mu([x_k; u_k]|D_M)
\end{bmatrix}
\]

is an \( n \)-dimensional vector of the separate GPs learned for each one of the \( n \) elements of the state transition function \( f(\cdot) \) (since each GP maps vector inputs to a scalar), and \( w_k \sim \mathcal{N}(0, \Sigma) \) models both the process noise and the modeling uncertainties due to the GP regression. With a rich set of training data, \( \text{GP}_\mu \) will approximate \( f \), while \( \text{GP}_\Sigma \) will quantify both process noise and modeling errors.

III. NONLINEAR COVARIANCE STEERING

A. Problem Formulation

Consider the finite-time evolution of system (1). The second goal of this paper is to find a control policy that will steer the state of (1) from a given initial distribution \( x_0 \sim \mathcal{N}(\mu_0, \Sigma_0) \) toward a given terminal one \( \mathcal{N}(\mu_t, \Sigma_t) \) in a finite horizon of \( N \) time steps. However, \( f(\cdot) \) is unknown. Instead, the non-parametric GP model (11) that was learned in section II-D will be used for the optimal control design.

In particular, consider (11) for \( k \in [0, N-1] \), where \( N \) is a positive integer, and the initial state \( x_0 \) is a random vector with \( E[x_0] = \mu_0 \) and \( \text{Cov}[x_0] = \Sigma_0 \), with \( \mu_0 \in \mathbb{R}^n \) and \( \Sigma_0 \in \mathbb{S}^n_+ \) being given quantities. Furthermore, \( x_{0:N} := \{x_k\}_{k=0}^N \) and \( u_{0:N-1} := \{u_k\}_{k=0}^{N-1} \) correspond to the state and input processes, respectively. In addition, \( w_{0:N-1} := \{w_k\}_{k=0}^{N-1} \) is the process noise, which is assumed to be a sequence of i.i.d. random variables with

\[
E[w_k] = 0, \quad E[w_k w_k^T] = \delta_{kk} W_k, \tag{13}
\]

for all \( k, j \in [0, N-1] \), where \( W_k = \text{GP}_\Sigma([x_k; u_k]|D) \in \mathbb{S}^n_+ \). Furthermore, \( x_0 \) is independent of \( u_{0:N-1} \), that is,

\[
E[x_0 w_k] = 0, \quad E[w_k w_0^T] = 0, \tag{14}
\]

for all \( k \in [0, N-1] \). For the problem of finding a control law that will steer the system to a goal distribution, if \( \mu_k \) and \( \Sigma_k \) denote the state mean and covariance at time step \( k \), that is,

\[
\mu_k := E[x_k], \quad \Sigma_k := \text{Cov}[x_k], \tag{15}
\]

and the class of admissible control policies is taken to be the set of sequences of control laws that are measurable functions of the realization of the current state of the system, then, the nonlinear covariance steering problem can be formulated as follows:

**Problem 1:** Let \( \mu_0, \mu_t \in \mathbb{R}^n \) and \( \Sigma_0, \Sigma_t \in \mathbb{S}^n_+ \) be given. Find a control policy \( \pi := \{u(k)\}_{k=0}^{N-1} \) that will steer the system (1) and, consequently, (11), from the initial state \( x_0 \) with \( E[x_0] = \mu_0 \) and \( \text{Cov}[x_0] = \Sigma_0 \), to a terminal state \( x_N \) with \( \mu_N = \mu_t \), \( (\Sigma_t - \Sigma_N) \in \mathbb{S}^n_+ \). \tag{16}

**Remark 1** Given that the system in (11) is nonlinear, enforcing the equality constraint \( \Sigma_k = \Sigma_t \) would be a difficult task in practice. Following [27], we consider instead the relaxed constraint given in (16) according to which, it suffices to achieve a terminal state covariance \( \Sigma_N \) that is “smaller” than \( \Sigma_t \), which corresponds to a situation in which the (desired) terminal mean \( \mu_t \) will be reached by representative samples of system’s trajectories with less uncertainty than the uncertainty corresponding to \( \Sigma_t \).

B. Linearization of the Gaussian Process Prediction Model

The DTSN system (11) described by the GP model can be associated at time step \( k \in [0, N-1] \) with a discrete-time stochastic linear system, which corresponds to the linearization of (11) around a given point \( (\mu_k, \nu_k) \) in \( \mathbb{R}^n \times \mathbb{R}^m \) which is given by

\[
z_{i+1} = A_k(z_{i|k} - \mu_k) + B_k(u_{i|k} - \nu_k) + r_k + w_{i|k}, \tag{17}
\]

for \( i \in [k, N-1] \) and \( z_{k|k} = z_k \), with \( E[z_k] = \mu_k \) and \( \text{Cov}[z_k] = \Sigma_k \), where \( \mu_k \in \mathbb{R}^n \), \( \Sigma_k \in \mathbb{S}^n_+ \), and \( \nu_k \in \mathbb{R}^m \). In addition, it is assumed that \( E[z_k w_{i|k}^T] = 0 \) for all \( i \in [k, N-1] \).

The matrices \( A_k \) and \( B_k \) and the vector \( r_k \) are computed from linearizing the GP model (11) as follows:

\[
A_k := \frac{\partial}{\partial x_{k}} \text{GP}_\mu([x_k; u_k]|D) \bigg|_{x_k = \mu_k, u_k = \nu_k} \tag{18}
\]

\[
B_k := \frac{\partial}{\partial u_{k}} \text{GP}_\mu([x_k; u_k]|D) \bigg|_{x_k = \mu_k, u_k = \nu_k}
\]

for all \( k \in [0, N-1] \).
\[ B_k := \frac{\partial}{\partial u_k} \text{GP}_\mu^T([x_k; u_k]|D) \bigg|_{x_k = \mu_k, u_k = v_k} = \begin{bmatrix} \frac{\partial}{\partial u_k} \text{GP}_\mu^T([x_k; u_k]|D_1) \\ \vdots \\ \frac{\partial}{\partial u_k} \text{GP}_\mu^T([x_k; u_k]|D_n) \end{bmatrix} \bigg|_{x_k = \mu_k, u_k = v_k} \] (19)

with the Jacobians of the (scalar) GPs being
\[ \frac{\partial}{\partial x_k} \text{GP}_\mu^T([x_k; u_k]|D_j) = Y_j^T [K + \sigma_n^2 I]^{-1} \frac{\partial k_\mu}{\partial x_k} \] (20a)
\[ \frac{\partial}{\partial u_k} \text{GP}_\mu^T([x_k; u_k]|D_j) = Y_j^T [K + \sigma_n^2 I]^{-1} \frac{\partial k_\mu}{\partial u_k} \] (20b)

for \( j \in [1, n]_d \), where
\[ k_\mu = \{k([x_k; u_k], X_1), \ldots, k([x_k; u_k], X_M)\}^T \]
is the vector of kernel values between \([x_k; u_k]\) and the training inputs (the columns of \( X \)) and thus,
\[ \frac{\partial k}{\partial x_k} = \begin{bmatrix} \frac{\partial k([x_k; u_k], X_1)}{\partial x_{k,1}} & \cdots & \frac{\partial k([x_k; u_k], X_1)}{\partial x_{k,n}} \\ \vdots & \ddots & \vdots \\ \frac{\partial k([x_k; u_k], X_M)}{\partial x_{k,1}} & \cdots & \frac{\partial k([x_k; u_k], X_M)}{\partial x_{k,n}} \end{bmatrix}, \] (21a)
\[ \frac{\partial k}{\partial u_k} = \begin{bmatrix} \frac{\partial k([x_k; u_k], X_1)}{\partial u_{k,1}} & \cdots & \frac{\partial k([x_k; u_k], X_1)}{\partial u_{k,m}} \\ \vdots & \ddots & \vdots \\ \frac{\partial k([x_k; u_k], X_M)}{\partial u_{k,1}} & \cdots & \frac{\partial k([x_k; u_k], X_M)}{\partial u_{k,m}} \end{bmatrix}, \] (21b)

where \( M \) is the number of training data points, \( X_i = [x(i); u(i)] \) is the \( i \)-th training input, \( x_k,j, j \in [1, n]_d \), is the \( j \)-th element of vector \( x_k \in \mathbb{R}^n \) and \( u_k,j, j \in [1, m]_d \), is the \( j \)-th element of vector \( u_k \in \mathbb{R}^m \). For the squared exponential kernel \( \phi \), the partial derivative is
\[ \frac{\partial k(x_1, x_2)}{\partial x_{1,i}} = -\frac{\sigma_f^2}{\ell^2} (x_{1,i} - x_{2,i}) \exp\left(-\frac{1}{2}(x_1 - x_2)^T L^{-1} (x_1 - x_2)\right). \] (22)

In addition,
\[ r_k := \text{GP}_\mu^T([\mu_k; \nu_k]|D). \] (23)

Equivalently, we can write
\[ z_{i+1 | k} = A_k z_{i | k} + B_k u_{i | k} + d_k + w_{i | k}, \] (24)

where \( d_k := -A_k \mu_k + B_k \nu_k + r_k \).

The latter linear model will be referred to as the \( k \)-th linearized state space model, corresponding to the linearization of \( \Phi(k \cdot) \) around the latest available state at time step \( k \). Note that \( \Phi(k \cdot) \) is a linear invariant (LTI) system. However, for a different \( k \), one obtains a different linearized system with a different but time-invariant triplet \((A_k, B_k, r_k)\). An implicit assumption here is that the pair \((A_k, B_k)\) is controllable.

### C. Finite-Horizon Linearized Covariance Steering Problem

Next, we formulate a linearized covariance steering problem for the system described in \( \Phi(k \cdot) \) for a given \( k \in [0, N - 1]_d \). The class \( \mathcal{U} \) of admissible control policies for the latter problem consists of the sequence of control laws \( \{\phi_{i | k}(\cdot)\}_{i=k}^{N-1} \), where
\[ \phi_{i | k}(z) = v_{i | k} + K_{i | k} z, \quad i \in [k, N - 1]_d. \] (25)

The linearized covariance steering problem at time step \( k \) is formulated as follows:

**Problem 2 (\( k \)-th linearized covariance steering problem):** Let \( \mu_k, \mu_i \in \mathbb{R}^n \) and \( \Sigma_k, \Sigma_i \in \mathbb{S}_n^+ \) be given. Among all admissible control policies \( \omega_k := \{\phi_{i | k}(\cdot)\}_{i=k}^{N-1} \in \mathcal{U} \), find a control policy \( \omega_k \) that minimizes the performance index
\[ J_k(\omega_k) := \mathbb{E} \left[ \sum_{i=k}^{N-1} \phi_{i | k}(z_{i | k})^T \phi_{i | k}(z_{i | k}) \right] \] (26)
subject to the recursive dynamic constraints \( (24) \) and the boundary conditions
\[ \mathbb{E}[z_k] = \mu_k, \quad \text{Cov}[z_k] = \Sigma_k, \] (27a)
\[ \mathbb{E}[z_N] = \mu_i, \quad (\Sigma_i - \text{Cov}[z_N]) \in \mathbb{S}_n^+. \] (27b)

The choice of the performance index ensures that the control input will have finite energy, without excessive actuation. Note that the terminal positive semi-definite constraint \( (\Sigma_i - \text{Cov}[z_N]) \in \mathbb{S}_n^+ \) differentiates Problem \( 2 \) from the standard linear quadratic Gaussian (LQG) problem. Although no state or input constraints are considered in this formulation, the optimization-based solution presented here and in \( [27] \) is applicable to the general problem formulation that includes such constraints (refer to \( [19] \) for more details).

For the \( k \)-th linearized covariance steering problem, finding a policy \( \omega_k \) that solves Problem \( 2 \) is equivalent to finding a sequence \( \{v_{i | k}, K_{i | k}\}_{i=k}^{N-1} \). The main idea of the greedy nonlinear covariance steering algorithm is to solve the \( k \)-th linearized covariance steering problem (Problem \( 2 \)) at each time step \( k \) and apply only the first control law that solves Problem \( 2 \), i.e., \( (v_{i | k}, K_{i | k}) \), at that time step. This idea has to be applied iteratively and the feedback policy has to be updated accordingly as new state measurements become available.

### D. Solution to the \( k \)-th Linearized Covariance Steering Problem

Next, the main steps of the solution to the \( k \)-th linearized covariance steering problem (Problem \( 2 \)) will be presented. To this aim, Eq. \( (24) \) can be written in compact form as
\[ z_k := G^k z_k + G^k u + G^k w + d_k, \] (28)

where
\[ z := [z_{k | k}^T, \ldots, z_{N | k}^T]^T, \quad u := [u_{k | k}^T, \ldots, u_{N-1 | k}^T]^T, \quad w := [w_{k | k}^T, \ldots, w_{N-1 | k}^T]^T, \quad d_k := [d_{k | k}^T, \ldots, d_{N-1 | k}^T]^T. \]

In addition, \( G^k_u, G^k_w, \) and \( G^k_z \) are defined as follows:
\[ G^k_u := \begin{bmatrix} 0 & 0 & \cdots & 0 \\ A_k B_k & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ A_k^{N-1-k} B_k & A_k^{N-2-k} B_k & \cdots & B_k \end{bmatrix}, \]
\[ G^k_w := \begin{bmatrix} B_k \\ \vdots \\ B_k \end{bmatrix} \text{when } B_k = 1, \]
\[ G^k_z := \begin{bmatrix} I \\ A_k^T \\ \vdots \\ (A_k^{N-k})^T \end{bmatrix}. \]
In view of (25), an admissible control sequence can be written compactly as
\[ u = \mathcal{K}_k z + \nu_k, \] (29)
where
\[ \mathcal{K}_k := \text{bdig}(K_{kk}, \ldots, K_{N-1,k}), \]
\[ \nu_k := [v_{k}^\top, \ldots, v_{N-1,k}^\top]^\top. \]

After plugging (29) into (28), the closed-loop dynamics can be expressed in compact form as
\[ z = T^k_z z_k + T^k_z \nu_k + T^k_{w}(w + d_k), \] (30)
where
\[ T^k_z := (I - G^k_z \mathcal{K}_k)^{-1} G^k_z, \]
\[ T^k_w := (I - G^k_w \mathcal{K}_k)^{-1} G^k_w. \]

Note that \((I - G^k_z \mathcal{K}_k)^{-1}\) is well-defined, as explained in [19, 30].

In view of equation (30), (29) becomes
\[ u = H^k_z z_k + H^k_w \nu_k + H^k_{w}(w + d_k), \] (32)
where
\[ H^k_z := \mathcal{K}_k (I - G^k_z \mathcal{K}_k)^{-1} G^k_z, \]
\[ H^k_w := I + \mathcal{K}_k (I - G^k_w \mathcal{K}_k)^{-1} G^k_w. \]

Plugging (32) in (26), one can obtain an expression for the expected cost as a function of the decision variables \(\mathcal{K}_k\) and \(\nu_k\). In particular,
\[ J_k(z_k) = \mathbb{E}[u^\top u] = \text{trace}(\mathbb{E}[uu^\top]) \]
\[ = \text{trace}(H^k_z \Sigma_k + \mu_k \mu_k^\top) \]
\[ + 2H^k_z \mu_k \text{trace}(H^k_z) + 2H^k_z \mu_k d_k^\top (H^k_z)^\top \]
\[ + H^k_w \nu_k \text{trace}(H^k_w) + 2H^k_w \nu_k d_k^\top (H^k_w)^\top \]
\[ =: J_k(\mathcal{K}_k, \nu_k), \] (34)
where \(W_{k-1} := \text{bdig}(W_k, \ldots, W_{N-1})\). In the previous derivation, we have used the available information about the statistics of \(z_k\) and in particular, that \(\mathbb{E}[z_k] = \mu_k, \mathbb{E}[z_k z_k^\top] = \Sigma_k + \mu_k \mu_k^\top\).

The terminal constraints are also expressed in terms of the decision variables \((\mathcal{K}_k, \nu_k)\). In particular,
\[ E[z_N | z_{<N}] = E[P_N z] = P_N E[z] \]
\[ = P_N (T^k_z \mu_k + T^k_w \nu_k + T^k_{w} d_k) \]
\[ =: f(\mathcal{K}_k, \nu_k), \] (35)
where \(P_N := [0, \ldots, 0, I]\). Then, the constraint \(E[z_N | z_{<N}] = \mu_k\) can be written as
\[ C_1(\mathcal{K}_k, \nu_k) = 0, \quad C_1(\mathcal{K}_k, \nu_k) := f(\mathcal{K}_k, \nu_k) - \mu_k. \] (36)
Furthermore,
\[ \text{Cov}[z_N | z_{<N}] = E[z_N z_N^\top | z_{<N}] - \mu_k \mu_k^\top, \] (37)
where
\[ E[z_N | z_{<N}] = E[P_N z z^\top P_N^\top] = P_N E[z z^\top] P_N^\top \]
\[ = P_N (T^k_z \Sigma_k + \mu_k \mu_k^\top)(T^k_z)^\top + T^k_z \mu_k d_k^\top (T^k_z)^\top \]
\[ + T^k_w d_k \nu_k (T^k_w)^\top + T^k_w \nu_k d_k^\top (T^k_w)^\top \]
\[ + T^k_w \nu_k d_k (T^k_w)^\top + T^k_w d_k \nu_k (T^k_w)^\top \]
\[ =: g(\mathcal{K}_k, \nu_k). \] (38)

Therefore, the terminal state covariance constraint, \((\Sigma_z - \Sigma_z(N)) \in S^+_n\), can be written as a positive semi-definite constraint
\[ C_2(\mathcal{K}_k, \nu_k) := \Sigma_z - g(\mathcal{K}_k, \nu_k) + \mu_k \mu_k^\top. \] (39)

**Problem 3:** Find a pair \((\mathcal{K}_k, \nu_k)\) that minimizes the predicted cost \(\tilde{J}_k(\mathcal{K}_k, \nu_k)\) subject to the constraints
\[ C_1(\mathcal{K}_k, \nu_k) = 0, \quad C_2(\mathcal{K}_k, \nu_k) \in S^+_n, \] (40)
where \(C_1(\mathcal{K}_k, \nu_k)\) and \(C_2(\mathcal{K}_k, \nu_k)\) are defined in (36) and (39), respectively.

**Remark 2** Problem 3 is not convex as explained in [19]. However, it can be easily expressed as a tractable convex program by applying a transformation to the pair of decision variables \((\mathcal{K}_k, \nu_k)\), as described in [19, 30].

**E. Propagation of State Mean and Covariance using the Gaussian Process-Based Unscented Transform**

Let \(\sigma = \{\kappa_k(\cdot)\}^{N-1}_{k=1}\) be an admissible control policy for Problem 1. Then, the closed-loop dynamics become
\[ x_{k+1} = \mathbf{GP}_\mu([x_k; \kappa_k(x_k)] | D) + w_k. \] (41)

The mean and covariance of the uncertain state of the nonlinear system described by (11), (12) is propagated using the unscented transform [28, 29]. To this aim, assume that the mean \(\mu_k := \mathbb{E}[x(k)]\) and covariance \(\Sigma_k := \text{Cov}[x(k)]\) of the system (11) (or estimates of these quantities) are known at time step \(k\).

First, we compute \(2n + 1\) deterministic points, \(\sigma^{(i)}_k, i \in [1, 2n+1]\), which are also known as *sigma points*, according to [28, 29]. Then, to each sigma point, we associate a pair of gains \((\gamma^{(i)}_k, \delta^{(i)}_k)\), according to [29, 31]. Subsequently, the sigma points \(\{\sigma^{(i)}_k\}^N_{i=1}\) are propagated to the next time step to obtain a new set of points \(\{\tilde{x}^{(i)}_{k+1}\}^N_{i=1}\), where
\[ \tilde{x}^{(i)}_{k+1} = \mathbf{GP}_\mu([\sigma^{(i)}_k; \kappa_k(\sigma^{(i)}_k)] | D), \quad i \in [0, 2n)_d. \] (42)

Using this new point-set, one can approximate the (predicted)
state mean and covariance at time step \(k+1\) as
\[
\dot{\mu}_{k+1} = 2L \sum_{i=0}^{2L} \gamma_{i}^{(i)} \hat{\sigma}_{k+1}^{(i)},
\]
\[
\dot{\Sigma}_{k+1} = 2L \sum_{i=0}^{2L} \delta_{i}^{(i)} \left( \hat{\sigma}_{k+1}^{(i)} - \dot{\mu}_{k+1} \right) \left( \hat{\sigma}_{k+1}^{(i)} - \dot{\mu}_{k+1} \right)^{T} + W_{k}.
\]
(S3a)

(S3b)

Similar to [5], we set \(W_{k} = \text{GP}_{Z}^{f}([\dot{\mu}_{k}; \kappa_{k}(\dot{\mu}_{k})])\) as the process noise covariance. Notice that \(W_{k}\) captures both the noise in the system as well as the model uncertainties resulting from the lack of training data points used in the system identification step.

IV. GAUSSIAN PROCESS-BASED GREEDY NONLINEAR COVARIANCE STEERING

The Gaussian process-based greedy nonlinear covariance steering algorithm, similar to the one proposed in [27], consists of three main steps. Consider the time step \(k\), where \(k \in [0, N-1]_{\mathbb{N}}\), and assume that estimates of the state mean, \(\hat{\mu}_{k}\), the state covariance, \(\hat{\Sigma}_{k}\), and the input mean \(\hat{v}_{k}\) are known (starting from \(\hat{\mu}_{0} = \mu_{0}, \hat{\Sigma}_{0} = \Sigma_{0}\), and \(\hat{v}_{0} = 0\)).

The first step, which is referred to as the recursive linearization step (or RL step), is to construct a linearization \((A_{k}, B_{k}, r_{k})\) of (11) around the point \((\hat{\mu}_{k}, \hat{v}_{k})\) using (13)–(19) and (23). The linearization will have to be updated at each time step since the estimates \(\hat{\mu}_{k}\) and \(\hat{v}_{k}\) will also be updated. We write
\[
(A_{k}, B_{k}, r_{k}) = \Lambda(\hat{\mu}_{k}, \hat{v}_{k}; \text{GP}_{\mu}^{f}(|D|)).
\]
(44)

The second step is to solve the \(k\)-th linearized Gaussian covariance steering problem (LGCS step) and compute the feedback control policy that solves Problem 2. The latter problem is solved using the linearized model \((A_{k}, B_{k}, r_{k})\) obtained in the RL step and the estimates of the predicted mean and covariance \((\hat{\mu}_{k}, \hat{\Sigma}_{k})\) at time step \(k\).

Let \(\omega_{k}^{\ast}\) be the policy that solves Problem 2 starting from \(E[z_{k}|k] = \hat{\mu}_{k}\) and \(\text{Cov}[z_{k}|k] = \hat{\Sigma}_{k}\). We write
\[
\omega_{k}^{\ast} := S_{k}(A_{k}, B_{k}, r_{k}, \hat{\mu}_{k}, \hat{v}_{k}, \hat{\Sigma}_{k}),
\]
where \(\omega_{k}^{\ast} := \{\phi_{i}^{*}(\cdot)\}_{i=1}^{N-1}\). The computation of \(\omega_{k}^{\ast}\) can be done in real-time by means of robust and efficient convex optimization techniques [17], [19]. Then, we extract from \(\omega_{k}^{\ast}\) only its first control law. That is,
\[
\phi_{1}^{*}(z) := \mathcal{P}_{1}(\omega_{k}^{\ast}) = v_{k}^{\ast} + K_{k}^{\ast} z,
\]
where \(\mathcal{P}_{1}(\cdot)\) denotes the truncation operator that returns only the first element of a sequence. Then, we set the \(k\)-th control law of the feedback control policy \(\pi^{\ast}\) for the original nonlinear covariance steering problem (Problem 1) to be
\[
\kappa_{k}^{\ast}(x) := \phi_{1}^{*}(x) = v_{k}^{\ast} + K_{k}^{\ast} x,
\]
where \(x\) is the state of the original nonlinear system. The one-time-step transition map for the closed-loop dynamics based on information available at time step \(k\) is then described by
\[
x_{k+1} = \text{GP}_{\mu}^{\ast}(\cdot|D) + w_{k}
\]
\[
\text{GP}_{\mu}^{\ast}(\cdot|D) + w_{k}.
\]
(47)

Note that in this work, the estimates \((\hat{\mu}_{k}, \hat{\Sigma}_{k})\) are computed at time step \(k - 1\) by executing the third step of this algorithm, which is to propagate the mean \(\hat{\mu}_{k}\) and covariance \(\hat{\Sigma}_{k}\) of the closed-loop dynamics to the next time step. The new mean and covariance, i.e. \(\hat{\mu}_{k+1}\) and \(\hat{\Sigma}_{k+1}\), are computed using the GP-based unscented transform described in Section III-E. This is the predictive normalization step (PN step). We write
\[
(\hat{\mu}_{k+1}, \hat{\Sigma}_{k+1}) := \mathcal{F}_{k}(\hat{\mu}_{k}; \hat{\Sigma}_{k}; \text{GP}_{\mu}^{f}(|D|), \kappa_{k}^{\ast}(\cdot)),
\]
(48)

The three steps of the greedy covariance steering algorithm are repeated for all time steps \(k \in [0, N-1]_{\mathbb{N}}\). At the end of the process, the predicted approximations of the state mean and covariance should be sufficiently close to their corresponding goal quantities. The output of this iterative process is a control policy \(\pi^{\ast}_{0:N-1} := \{\kappa_{k}^{\ast}(x)\}_{k=0}^{N-1}\) that solves Problem 1.

V. NUMERICAL SIMULATIONS

In this section, the basic ideas of this paper are illustrated in numerical simulations. In particular, consider the following DTSN system:
\[
x_{k+1,1} = x_{k,1} + \tau x_{k,2},
\]
\[
x_{k+1,2} = x_{k,2} - \tau (\delta x_{k,1} + \zeta x_{k,1}^{3} + \gamma x_{k,2}) + \tau u_{k} + w_{k},
\]
(49a)
(49b)

which is a discrete-time realization of the Duffing oscillator. Assume that the dynamics (49) are unknown, but a “black-box” simulator for system (49) is available. The simulator will be run to collect a set of training data points that will be used to learn a non-parametric GP model of the system dynamics. Then, the learned model will be used to steer the mean and covariance of the state of the system from a given initial distribution toward a prescribed terminal one. For our simulations, we consider (49) with time step \(\tau = 0.01, \gamma = 0.05, \delta = -1, \zeta = 0.05\), while the white noise \(w_{k} \sim \mathcal{N}(0, 0.04^{2})\).

A. Learning a Gaussian Process Prediction Model

Before learning a GP prediction model, we run the simulator of system (49) and collect a set of training data points. \(D = \{(x_{i}^{(i)}; u_{i}^{(i)}; y_{i}^{(i)})\}_{i=1}^{M}\) where \(y_{i}^{(i)} = f(x_{i}^{(i)}; u_{i}^{(i)}; \varepsilon_{i}^{(i)})\). Starting from an initial condition \(x_{0} = \begin{bmatrix} 1, -1 \end{bmatrix}^{T}\), we feed the simulator with the input \(u_{k} = 10 \sin(5 (\frac{k}{500})^{2}) + 10 (\frac{k}{500})^{2}\) for \(k \in [0, 5000]_{\mathbb{N}}\), collect data pairs \((x_{i}^{(i)}; u_{i}^{(i)}; y_{i}^{(i)})\), and randomly sample \(M = 100\) of the 5000 available pairs. The trajectory of the system during the training simulation is shown in Figure 1.
The training data set $D$ is used to learn a GP prediction model of the form (11). The GP is optimized using the Python library scikit-learn [32]. The learned GP is validated on a test data set (different from $D$). Figure 2 compares the prediction of the GP model with the “true” states of the system, indicating good agreement between the two.

B. Solving the Nonlinear Covariance Steering Problem

Now that a model of the system dynamics is available, assume an initial state $x_0 \sim N(\mu_0, \Sigma_0)$ with $\mu_0 = [-1, 3]^T$ and $\Sigma_0 = \begin{bmatrix} 0.0175 & 0.013 \\ 0.013 & 0.0325 \end{bmatrix}$. The desired terminal state mean and covariance are taken to be $\mu_f = [0, 0]^T$ and $\Sigma_f = \begin{bmatrix} 0.0775 & 0.0217 \\ 0.0217 & 0.0525 \end{bmatrix}$, respectively. The finite-horizon covariance steering is executed for a horizon of $N = 30$ time steps.

![Fig. 1. Phase plot of the training simulation of system (19). The simulation consists of 5001 time steps. A set of $M = 100$ data pairs is randomly sampled from the available data and is used to learn a GP prediction model.](image1)

![Fig. 2. Real state vs mean and covariance predicted by the learned GP model on a test data set.](image2)

![Fig. 3. Time evolution of the sequence $\{E_k\}_{k=0}^N$. The vertical axis in this 3D graph corresponds to the time-step axis.](image3)

![Fig. 4. Sample trajectories of the closed loop system.](image4)

Figure 3 illustrates the time evolution of the predicted state covariance $\hat{\Sigma}_k$ in terms of the evolution of the sequence of ellipsoids $\{E_k\}_{k=0}^N$, where $E_k := \mathcal{E}_1(\mu_k, \Sigma_k)$, for $k \in [0, N]$. The desired terminal state covariance $\Sigma_f$ is associated with $E_f$, where $E_f := \mathcal{E}_1(\mu_f, \Sigma_f)$. In Fig. 3 the vertical axis corresponds to the time axis.

Sample trajectories of the closed loop system are illustrated in Fig. 4. The projection on the $x_1 - x_2$ plane of the 3D graph given in Fig. 3 is illustrated in Fig. 5. In these three figures, the black/gray ellipses correspond to $E_0$ and $E_f$. We observe that $E_N$ does not exactly match $E_f$, but is contained in $E_f$, which is expected given the semi-definite constraint (16), which corresponds to a relaxation of the hard equality constraint $\Sigma_N = \Sigma_f$.

VI. CONCLUSION

In this work, a greedy covariance steering algorithm that uses Gaussian process prediction models for discrete-time stochastic nonlinear systems with unknown dynamics has been proposed. First, a non-parametric prediction model is
learned from a set of training data using Gaussian process regression. Then, a set of linearized covariance steering problems is solved and the mean and covariance of the closed-loop system is predicted using the unscented transform.

The present work has utilized the standard Gaussian process regression method for learning the unknown dynamics. However, computationally efficient methods, such as sparse Gaussian processes, can be utilized for this task. Furthermore, this work has considered the case of perfect full-state information. However, a more practical case would be that of incomplete information, where the states of the system have to be estimated from partial measurements. Both of these as well as similar enhancements will be considered by the authors in future work.

REFERENCES

[1] C. E. Rasmussen, “Gaussian processes in machine learning,” in Summer School on Machine Learning, pp. 63–71, Springer, 2003.
[2] A. Girard, C. E. Rasmussen, J. Q. Candela, and R. Murray-Smith, “Gaussian process priors with uncertain inputs application to multiple-step ahead time series forecasting,” in NIPS (2003), pp. 545–552, 2003.
[3] D. B. Grimes, R. Chalodhorn, and R. P. Rao, “Dynamic imitation in a humanoid robot through nonparametric probabilistic inference,” in Robotics: science and systems, pp. 199–206, Cambridge, MA, 2006.
[4] J. Ko, D. J. Klein, D. Fox, and D. Haehnel, “Gaussian processes and reinforcement learning for identification and control of an autonomous blimp,” in ICRA (2007), pp. 742–747, 2007.
[5] J. Ko, D. J. Klein, D. Fox, and D. Haehnel, “GP-UKF: Unscented kalman filters with gaussian process prediction and observation models,” in IROS (2007), pp. 1901–1907, 2007.
[6] J. Ko and D. Fox, “GP-Bayesfilters: Bayesian filtering using gaussian process prediction and observation models,” Autonomous Robots, vol. 27, no. 1, pp. 75–90, 2009.
[7] Y. Pan and E. Theodorou, “Probabilistic differential dynamic programming,” in MPS (2014), pp. 1907–1915, 2014.
[8] Y. Pan and E. A. Theodorou, “Data-driven differential dynamic programming using gaussian processes,” in ACC (2015), pp. 4467–4472, 2015.
[9] M. Mukadam, X. Yan, and B. Boots, “Gaussian process motion planning,” in ICRA (2016), pp. 9–15, 2016.
[10] A. Hotz and R. E. Skelton, “Covariance control theory,” Int. J. Control, vol. 16, pp. 13–32, Oct 1987.