Cautious Nonlinear Covariance Steering using Variational Gaussian Process Predictive Models

Alexandros Tsolovikos * **Efstathios Bakolas** *

* *The University of Texas at Austin, Austin, Texas 78712-1221 (tsolovikos@utexas.edu, bakolas@austin.utexas.edu)*

Abstract: In this work, we consider the problem of steering the first two moments of the uncertain state of an *unknown* discrete-time stochastic nonlinear system to a given terminal distribution in finite time. Toward that goal, first, a non-parametric predictive model is learned from a set of available training data points using stochastic variational Gaussian process regression: a powerful and highly scalable machine learning tool for learning distributions over arbitrary nonlinear functions. Second, we formulate a tractable nonlinear covariance steering algorithm that utilizes the learned Gaussian process predictive model to compute a feedback policy that will drive the distribution of the state of the system close to the goal distribution. In a greedy approach, we linearize the Gaussian process model at each time step around the latest predicted mean and covariance, solve the linear covariance steering problem, and propagate the state statistics to the next time step using the unscented transform. This process is then repeated in a shrinking-horizon model predictive control fashion. The cautiousness of the Gaussian process predictive model, which captures both the process noise and modeling errors, is demonstrated in numerical simulations.

Keywords: nonparametric methods, nonlinear system identification, stochastic system identification, covariance steering, stochastic optimal control problems

1. INTRODUCTION

In this paper, we consider the finite-horizon covariance steering problem for discrete-time stochastic nonlinear systems described by *non-parametric Gaussian process* models. In particular, we consider the problem of 1) learning sparse stochastic variational Gaussian process (SVGP) predictive models for stochastic nonlinear systems from training data and 2) using the SVGP models for computing feedback control policies that steer the mean and covariance of the uncertain state of the underlying system 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) [Rasmussen (2003)] 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 [Grimes et al. (2006); Ko et al. (2007a)] and observation models for state estimation [Ko et al. (2007b); Ko and Fox (2009)], as well as trajectory optimization [Pan and Theodorou (2015)] and motion planning [Mukadam et al. (2016); Hewing et al. (2020)]. Inference using GP models is inherently dependent on the training data and the cost of inference with exact GPs scales with the cube of the number of training points. For that reason, a number of sparse approximations of GPs have been proposed in the literature, the most common ones using a set of "inducing variables" [Quinonero-Candela and Rasmussen (2005); Titsias (2009)]. Further scalability can be achieved by using stochastic variational inference [Hoffman et al. (2013)], leading to sparse GP models that can be trained on large datasets with thousands or millions of data points using stochastic gradient descent, while retaining a small inference cost [Hensman et al. (2013)].

The covariance steering (or covariance control) problem for both continuous and discrete-time linear Gaussian systems has been studied extensively for the infinite-horizon [Hotz and Skelton (1987); Xu and Skelton (1992)] as well as the finite-horizon case [Chen et al. (2016a,b); Bakolas (2016); Goldshtein and Tsiotras (2017)]. Nonlinear density steering problems for feedback linearizable nonlinear systems were recently studied in Caluya and Halder (2019), while an iterative covariance steering algorithm for nonlinear systems based on a linearization of the system along reference trajectories was presented in Ridderhof et al. (2019). Stochastic nonlinear model predictive control with probabilistic constraints can also be found in Mesbah et al. (2014); Sehr and Bitmead (2017).

Main Contribution: In this work, non-parametric state predictive models of discrete-time stochastic nonlinear systems with *unknown dynamics* are learned using *stochastic variational GP regression* from a set of training samples obtained by measuring the underlying stochastic nonlinear system. Then, the learned SVGP model is used to control the mean and covariance of the state of the unknown systems in a *cautious* greedy nonlinear covariance steering algorithm that takes into account both process noise and model uncertainties. Similar to Bakolas and Tsolovikos (2020), the SVGP-based greedy nonlinear covariance steering algorithm linearizes the learned model around the

⋆ This research has been supported in part by NSF awards ECCS-1924790, CCMI-1753687 and CMMI-1937957. Alexandros Tsolovikos acknowledges support by the A. Onassis Foundation scholarship.

⋆⋆The code for this work is available at github.com/alextsolovikos/greedyGPCS

latest mean state prediction or estimate, solves the linear covariance steering problem from the current to the target distribution and then propagates the state statistics to the next step using the unscented transform [Julier and Uhlmann (2004)], modified to take into account the uncertainty estimates provided by the GP predictive model [Ko et al. (2007b)]. This three-step process is repeated in a shrinking-horizon model predictive control fashion 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 2, the process of learning a predictive model from sample data points using SVGP regression is presented. The greedy nonlinear covariance steering problem for non-parametric GP predictive models is formulated in Section 3. Section 4 presents numerical simulations and comparisons of the GP model with the analytic one. We conclude with remarks and directions for future research in Section 5.

Notation: Given a random vector $\mathbf{x}$, $\mathbb{E}[\mathbf{x}]$ denotes its expected value (mean) and $\mathrm{Cov}[\mathbf{x}]$ its covariance. The space of real symmetric $n \times n$ matrices will be denoted by $\mathbb{S}_n$. The convex cone of $n \times n$ (symmetric) positive semi-definite and (symmetric) positive definite matrices will be denoted by $\mathbb{S}_n^+$ and $\mathbb{S}_n^{++}$, respectively. Finite-length sequences are denoted as $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\} = \{\mathbf{x}_i\}_{i=1}^N$. The i-th element of a vector $\mathbf{x}$ is denoted by $[\mathbf{x}]_i$. Similarly, the i-th element of the j-th column of a matrix M is denoted as $[M]_{ij}$. For a scalar-valued function $f(\cdot) : \mathbb{R}^n \to \mathbb{R}$ and a sequence of vectors $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^N$, we define $f(\mathbf{X})$ as the vector with $[f(\mathbf{X})]_i = f(\mathbf{x}_i)$. Similarly, if $k(\cdot,\cdot) : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$, then $k(\mathbf{X},\mathbf{X})$ is the matrix with elements $[k(\mathbf{X},\mathbf{X})]_{ij} = k(\mathbf{x}_i,\mathbf{x}_j)$. Finally, if $\mathbf{x} \in \mathbb{R}^n$ and $\mathbf{y} \in \mathbb{R}^m$, then $[\mathbf{x};\mathbf{y}] = [\mathbf{x}^\top, \mathbf{y}^\top]^\top \in \mathbb{R}^{n+m}$ will denote the vertical concatenation of $\mathbf{x}$ and $\mathbf{y}$.

2. STOCHASTIC VARIATIONAL GAUSSIAN PROCESSES FOR DISCRETE-TIME DYNAMICS

2.1 Sparse Variational Gaussian Process Regression

Consider the vector $\mathbf{y}$, where y_i is a noisy observation of an unknown scalar-valued function $f(\cdot) : \mathbb{R}^n \to \mathbb{R}$ at a known location $\mathbf{x}_i$, for all $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^N$, and the measurement likelihood $p(y_i \mid f(\mathbf{x}_i))$ is known. Let $\mathbf{f}$ be the (unknown) vector containing the values of $f(\cdot)$ at the points $\mathbf{X}$. We introduce a Gaussian prior on $f(\cdot)$, i.e. $f(\mathbf{x}) \sim \mathcal{N}(f(\mathbf{x}) \mid m(\mathbf{x}), k(\mathbf{x},\mathbf{x}))$, where $m(\cdot) : \mathbb{R}^n \to \mathbb{R}$ is a chosen *mean function* (e.g. zero, constant, or linear) and $k(\cdot,\cdot) : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ is the *kernel function* that measures the closeness between two input points and specifies the smoothness and continuity properties of the underlying function $f(\cdot)$. Now, the *prior* over the vector $\mathbf{f}$ can be written as

$$p(\mathbf{f};\mathbf{X}) = \mathcal{N}(\mathbf{f} \mid m(\mathbf{X}), k(\mathbf{X},\mathbf{X})), \tag{1}$$

where the mean vector is defined as $[m(\mathbf{X})]_i = m(\mathbf{x}_i)$ and the covariance is $[k(\mathbf{X},\mathbf{X})]_{ij} = k(\mathbf{x}_i,\mathbf{x}_j)$.

Exact GP Inference: The joint density of $\mathbf{y}$ and $\mathbf{f}$ is

$$p(\mathbf{y},\mathbf{f};\mathbf{X}) = p(\mathbf{y} \mid \mathbf{f};\mathbf{X})p(\mathbf{f};\mathbf{X}). \tag{2}$$

If the likelihood function is chosen to be Gaussian, e.g. $p(\mathbf{y} \mid \mathbf{f};\mathbf{X}) = \mathcal{N}(\mathbf{y} \mid \mathbf{f}, \sigma_\epsilon^2 I)$, then, the marginal likelihood

$$\begin{aligned} p(\mathbf{y};\mathbf{X}) &= \int p(\mathbf{y} \mid \mathbf{f};\mathbf{X})p(\mathbf{f};\mathbf{X})d\mathbf{f} \\ &= \mathcal{N}(\mathbf{y} \mid m(\mathbf{X}), k(\mathbf{X},\mathbf{X}) + \sigma_\epsilon^2 I) \end{aligned} \tag{3}$$

is analytically computed and the *hyperparameters* $\boldsymbol{\Theta} = \{\theta_m, \theta_k, \sigma_\epsilon\}$ that define the Gaussian process mean, kernel, and likelihood functions can be directly optimized by minimizing the negative log-likelihood of the training data:

$$\boldsymbol{\Theta}_{\text{opt}} = \arg\min_{\boldsymbol{\Theta}} \left(-\log\, p(\mathbf{y};\mathbf{X})\right). \tag{4}$$

Prediction of y_* on a new location $\mathbf{x}_*$ is done by conditioning on the training data,

$$\begin{aligned} p(y_*;\mathbf{x}_*,\mathbf{y},\mathbf{X}) &= \int p(y_*,\mathbf{y};\mathbf{x}_*,\mathbf{X})d\mathbf{y} \\ &= \mathcal{N}(y_* \mid \mu_*, \sigma_*), \end{aligned} \tag{5}$$

where

$$\mu_* = m(\mathbf{x}_*) + k(\mathbf{x}_*,\mathbf{X})\left[k(\mathbf{X},\mathbf{X}) + \sigma_\epsilon^2 I\right]^{-1}(\mathbf{y} - m(\mathbf{X}))$$
$$\sigma_* = k(\mathbf{x}_*,\mathbf{x}_*) - k(\mathbf{x}_*,\mathbf{X})\left[k(\mathbf{X},\mathbf{X}) + \sigma_\epsilon^2 I\right]^{-1}k(\mathbf{X},\mathbf{x}_*).$$

The inference cost is $\mathcal{O}(N^3)$, which can be expensive when the number of training points N is large. In order to reduce the inference cost of a GP model, we can use sparse approximations of Gaussian processes.

Sparse Variational GP Inference: Define a set of M inducing locations $\mathbf{Z} = \{\mathbf{z}_i\}_{i=1}^M$, with $M \ll N$, where M and $\mathbf{Z}$ are parameters to be chosen. Also, define the vector $\mathbf{u}$ as $[\mathbf{u}]_i = f(\mathbf{z}_i)$. The joint density of $\mathbf{y}$, $\mathbf{f}$, and $\mathbf{u}$ is

$$p(\mathbf{y},\mathbf{f},\mathbf{u}) = p(\mathbf{y} \mid \mathbf{f};\mathbf{X})p(\mathbf{f} \mid \mathbf{u};\mathbf{X},\mathbf{Z})p(\mathbf{u};\mathbf{Z}), \tag{6}$$

where $p(\mathbf{u};\mathbf{Z}) = \mathcal{N}(\mathbf{u} \mid m(\mathbf{Z}), k(\mathbf{Z},\mathbf{Z}))$ is the Gaussian prior on $\mathbf{u}$ (similar to (1)) and $p(\mathbf{f} \mid \mathbf{u};\mathbf{X},\mathbf{Z}) = \mathcal{N}\left(\mathbf{f} \mid \tilde{\boldsymbol{\mu}}, \tilde{\boldsymbol{\Sigma}}\right)$, with

$$[\tilde{\boldsymbol{\mu}}]_i = m(\mathbf{x}_i) + k(\mathbf{x}_i,\mathbf{Z})k(\mathbf{Z},\mathbf{Z})^{-1}(\mathbf{u} - m(\mathbf{Z})),$$
$$[\tilde{\boldsymbol{\Sigma}}]_{ij} = k(\mathbf{x}_i,\mathbf{x}_j) - k(\mathbf{x}_i,\mathbf{Z})k(\mathbf{Z},\mathbf{Z})^{-1}k(\mathbf{Z},\mathbf{x}_j).$$

However, $\mathbf{u}$ *is unknown*, since $f(\cdot)$ is also unknown. Following Hensman et al. (2013), we choose a *variational* posterior

$$q(\mathbf{f},\mathbf{u}) = p(\mathbf{f} \mid \mathbf{u};\mathbf{X},\mathbf{Z})q(\mathbf{u}), \tag{7}$$

where $q(\mathbf{u}) = \mathcal{N}(\mathbf{u} \mid \mathbf{m}, \mathbf{S})$ and $\mathbf{m}$, $\mathbf{S}$ are the parameters defining the variational distribution (along with $\mathbf{Z}$). Since both terms in (7) are Gaussian, we can get rid of $\mathbf{u}$ by marginalizing over it, that is,

$$\begin{aligned} q(\mathbf{f} \mid \mathbf{m},\mathbf{S};\mathbf{X},\mathbf{Z}) &= \int p(\mathbf{f} \mid \mathbf{u};\mathbf{X},\mathbf{Z})q(\mathbf{u})d\mathbf{u} \\ &= \mathcal{N}(\mathbf{f} \mid \boldsymbol{\mu}, \boldsymbol{\Sigma}), \end{aligned} \tag{8}$$

where, if we define the functions

$$\mu_f(\mathbf{x}_i) := m(\mathbf{x}_i) + k(\mathbf{x}_i,\mathbf{Z})\left[k(\mathbf{Z},\mathbf{Z})\right]^{-1}(\mathbf{m} - m(\mathbf{Z})),$$
$$\boldsymbol{\Sigma}_f(\mathbf{x}_i,\mathbf{x}_j) := k(\mathbf{x}_i,\mathbf{x}_j) - k(\mathbf{x}_i,\mathbf{Z})k(\mathbf{Z},\mathbf{Z})^{-1}\left[k(\mathbf{Z},\mathbf{Z}) - \mathbf{S}\right]k(\mathbf{Z},\mathbf{Z})^{-1}k(\mathbf{Z},\mathbf{x}_j),$$

then $[\boldsymbol{\mu}]_i = \mu_f(\mathbf{x}_i)$ and $[\boldsymbol{\Sigma}]_{ij} = \boldsymbol{\Sigma}_f(\mathbf{x}_i,\mathbf{x}_j)$.

The variational parameters ($\mathbf{Z}$, $\mathbf{m}$, and $\mathbf{S}$), along with the hyperparameters $\boldsymbol{\Theta} = \{\theta_m, \theta_k, \sigma_\epsilon\}$, can be found by maximizing the lower bound $\mathcal{L}$ on the marginal likelihood,

$$\log p(\mathbf{y} \mid \mathbf{X}) \geq \mathbb{E}_{q(\mathbf{f},\mathbf{u})}\left[\log \frac{p(\mathbf{y},\mathbf{f},\mathbf{u})}{q(\mathbf{f},\mathbf{u})}\right] = \mathcal{L}. \tag{9}$$

The lower bound can be factorized as

$$\mathcal{L} = \sum_{i=1}^N \mathbb{E}_{q(f_i|\mathbf{m},\mathbf{S};\mathbf{x}_i,\mathbf{Z})}\left[\log p(y_i \mid f_i)\right] - \mathrm{KL}\left[q(\mathbf{u}) \| p(\mathbf{u})\right],$$

where KL denotes the Kullback-Leibler divergence. Note that the expectation can be computed analytically if the likelihood $p(y_i \mid f_i)$ is Gaussian. An immediate consequence of that choice is that, since the bound is the sum over the training data, we can perform stochastic inference through minibatch subsampling. This allows inference on large datasets and, more importantly, *online learning* of the variational parameters.

Once the variational parameters have been trained, *predicting* the distribution of y_* on a test location $\mathbf{x}_*$ is simply

$$p(y_*; \mathbf{x}_*, \mathbf{m}, \mathbf{S}, \mathbf{Z}) = \mathcal{N}\left(y_* \mid \mu_f(\mathbf{x}_*), \mathbf{\Sigma}_f(\mathbf{x}_*, \mathbf{x}_*) + \sigma_\epsilon^2\right).$$

Now, only an $M \times M$ matrix needs to be inverted.

Multiple Outputs: So far, the output $y_i \in \mathbb{R}$ has been a scalar. In the case of multiple outputs $\mathbf{y}_i \in \mathbb{R}^D$, we can define the matrices $\mathbf{Y}$, $\mathbf{F}$, and $\mathbf{U}$ as the matrices containing the observation $\mathbf{y}_i$ and function values $\mathbf{f}(\mathbf{x}_i)$ and $\mathbf{f}(\mathbf{z}_i)$ as their i-th rows. The latent functions are now $f_d(\cdot) : \mathbb{R}^n \to \mathbb{R}$, for $d = 1, \ldots, D$, and an independent sparse GP is learned for each function by maximizing a lower bound similar to the scalar case, but with $p(\mathbf{Y}, \mathbf{F}, \mathbf{U}) = \Pi_{d=1}^D p(\mathbf{y}_d, \mathbf{f}_d, \mathbf{u}_d)$ and $q(\mathbf{F}, \mathbf{U}) = \Pi_{d=1}^D q(\mathbf{f}_d, \mathbf{u}_d)$ in place of $p(\mathbf{y}, \mathbf{f}, \mathbf{u})$ and $p(\mathbf{f}, \mathbf{u})$, respectively.

2.2 SVGP for Discrete-time Dynamics

Consider a discrete-time dynamical system of the form

$$\mathbf{z}_{t+1} = \mathbf{g}(\mathbf{z}_t, \mathbf{u}_t) + \epsilon_t, \tag{10}$$

where $\mathbf{z}_t \in \mathbb{R}^{n_z}$ is the state at time step t, $\mathbf{u}_t \in \mathbb{R}^{n_u}$ is the control input, and $\epsilon_t \in \mathbb{R}^{n_z}$ the i.i.d. additive Gaussian white noise, with $\mathbf{w}_t \sim \mathcal{N}\left(\epsilon_t \mid 0, \sigma_\epsilon^2 I\right)$.

Assume that the underlying dynamics $\mathbf{g}(\cdot, \cdot)$ are unknown, but full-state measurements of the state transitions for given inputs are available for sampling (e.g. via experiments). In particular, assume that *observations*

$$\mathbf{y}_i = \mathbf{g}(\mathbf{z}_i, \mathbf{u}_i) + \epsilon_i \tag{11}$$

at known locations

$$\mathbf{x}_i = [\mathbf{z}_i; \mathbf{u}_i] \tag{12}$$

are available, that is, our dataset consists of N triplets, $\mathcal{D} = \{(\mathbf{y}_i, \mathbf{z}_i, \mathbf{u}_i)\}_{i=1}^N$. Following Subsection 2.1, we can fit a multitask (multi-output) sparse variational GP to the measurements, in order to get a non-parametric approximate model of the dynamics. The observations and corresponding inputs to the SVGP are the ones defined in (11) and (12), respectively, the number of outputs is $D = n_z$ and the number of inputs (features) is $n = n_z + n_u$. Choosing an appropriate mean $m(\cdot)$ (e.g. zero, constant, or linear) and a kernel function $k(\cdot, \cdot)$ (typically, a squared exponential), the variational parameters and hyperparameters of the SVGP are learned by minimizing the negative of the lower bound, $-\mathcal{L}$, via stochastic gradient descent on minibatches of $\mathcal{D}$.

The learned SVGP model can now be defined as

$$\mathbf{z}_{t+1} = G(\mathbf{z}_t, \mathbf{u}_t) + \mathbf{w}_t, \tag{13}$$

where

$$G(\mathbf{z}_t, \mathbf{u}_t) = \boldsymbol{\mu}_f([\mathbf{z}_t; \mathbf{u}_t]) \tag{14}$$

is the mean of the next state and

$$\mathbf{w}_t \sim \mathcal{N}\left(\mathbf{w}_t \mid 0, \Sigma_f([\mathbf{z}_t; \mathbf{u}_t], [\mathbf{z}_t; \mathbf{u}_t]) + \sigma_\epsilon^2\right) \tag{15}$$

is the additive noise, the covariance of which captures not only the process noise, but also the model uncertainties.

2.3 Linearization of the SVGP Dynamics

Given a trained SVGP model like the one in (13), if a linearization around a given state $\mathbf{z}_*$ and input $\mathbf{u}_*$ is necessary, it can be easily computed as

$$\mathbf{z}_{t+1} \approx A_* \mathbf{z}_t + B_* \mathbf{u}_t + \mathbf{d}_*, \tag{16}$$

where

$$A_* = \frac{\partial}{\partial \mathbf{z}} G(\mathbf{z}_*, \mathbf{u}_*) = \frac{\partial}{\partial [\mathbf{z}; \mathbf{u}]} \boldsymbol{\mu}_f([\mathbf{z}; \mathbf{u}]) \begin{bmatrix} I_{n_z} \\ 0 \end{bmatrix} \Bigg|_{\substack{\mathbf{z}=\mathbf{z}_* \\ \mathbf{u}=\mathbf{u}_*}}, \tag{17}$$

$$B_* = \frac{\partial}{\partial \mathbf{u}} G(\mathbf{z}_*, \mathbf{u}_*) = \frac{\partial}{\partial [\mathbf{z}; \mathbf{u}]} \boldsymbol{\mu}_f([\mathbf{z}; \mathbf{u}]) \begin{bmatrix} 0 \\ I_{n_u} \end{bmatrix} \Bigg|_{\substack{\mathbf{z}=\mathbf{z}_* \\ \mathbf{u}=\mathbf{u}_*}}, \tag{18}$$

and

$$\mathbf{d}_* = -A_* \mathbf{z}_* - B_* \mathbf{u}_* + G(\mathbf{z}_*, \mathbf{u}_*).$$

For compactness, denote the linearization operation as

$$\{A_*, B_*, \mathbf{d}_*\} = \mathrm{LIN}_G\{\mathbf{z}_*, \mathbf{u}_*\}. \tag{19}$$

Note that linearization with respect to the inputs to the GP will depend on the selected mean $m(\cdot)$ and kernel $k(\cdot, \cdot)$ functions. In practice, the above Jacobians can be easily computed via *automatic differentiation* (e.g. Autograd in PyTorch [Paszke et al. (2017)]).

3. GREEDY NONLINEAR COVARIANCE STEERING

3.1 Problem Formulation

Consider the finite-time evolution of the stochastic system (10). The goal of finite-time covariance steering is to find a control policy that will steer the state of (10) from a given initial distribution with mean $\boldsymbol{\mu}_0$ and covariance Σ_0 to a given terminal one with mean and covariance $\boldsymbol{\mu}_f$ and Σ_f, respectively, in a finite horizon of T time steps.

A *greedy* approach to finite-horizon covariance steering was presented in Bakolas and Tsolovikos (2020), where the dynamics (10) are linearized at each time step around the current mean, the *linear* covariance steering problem from the current to the target mean and covariance is solved, and only the first control law is applied. However, the exact dynamics in (10) are unknown and cannot be used in the model-based covariance steering algorithm of Bakolas and Tsolovikos (2020). Instead, the greedy algorithm will be adapted to be used with the approximate, non-parametric GP model that we learned in Section 2.

In particular, consider the learned model (13) for $t = 0, \ldots, T-1$, with an initial state $\mathbf{z}_0$ drawn from a distribution with $\mathbb{E}[\mathbf{z}_0] = \boldsymbol{\mu}_0$ and $\mathrm{Cov}[\mathbf{z}_0] = \Sigma_0$, where $\boldsymbol{\mu}_0 \in \mathbb{R}^{n_z}$ and $\Sigma_0 \in \mathbb{S}_{n_z}^{++}$ are given. The process noise, $\mathbf{w}_t$, is assumed to be a sequence of i.i.d. random variables drawn from (15). Furthermore, $\mathbf{z}_0$ is conditionally independent of $\mathbf{w}_t$, for all $t = 0, \ldots, T-1$.

Because the identified system in (13) is nonlinear, there is no guarantee that an initial state drawn from a normal distribution will lead to future states being Gaussian. Therefore, as explained in Bakolas and Tsolovikos (2020), it is more prudent to talk about steering the nonlinear system mean and covariance *close* to desired quantities (in the Loewner sense) rather than steering the state distribution to a goal distribution.

If we take the class of admissible control policies to be the set of sequences of control laws that are measurable functions of the realization of the current state, the nonlinear covariance steering problem can be formulated as follows:

Problem 1. (nonlinear covariance steering problem). Let $\boldsymbol{\mu}_0, \boldsymbol{\mu}_\mathrm{f} \in \mathbb{R}^{n_z}$ and $\Sigma_0, \Sigma_\mathrm{f} \in \mathbb{S}^{++}_{n_z}$ be given. Find a control policy $\pi := \{\kappa_t(\cdot)\}_{t=0}^{T-1}$ that will steer the system (13) and, consequently, (10), from the initial state $\mathbf{z}_0$ with $\mathbb{E}[\mathbf{z}_0] = \boldsymbol{\mu}_0$ and $\mathrm{Cov}[\mathbf{z}_0] = \Sigma_0$ to a terminal state $\mathbf{z}_T$ with

$$\boldsymbol{\mu}_T = \boldsymbol{\mu}_\mathrm{f}, \quad (\Sigma_\mathrm{f} - \Sigma_T) \in \mathbb{S}^{+}_{n_z}. \tag{20}$$

3.2 Finite-Horizon Linearized Covariance Steering Problem

Next, we formulate a linearized covariance steering problem for the system described by the linearization

$$\mathbf{z}_{j+1|t} \approx A_t \mathbf{z}_{j|t} + B_t \mathbf{u}_{j|t} + \mathbf{d}_t, \tag{21}$$

of (13) around the mean state $\boldsymbol{\mu}_t$ and corresponding (previous) control policy,

$$\{A_t, B_t, \mathbf{d}_t\} = \mathrm{LIN}_G\{\boldsymbol{\mu}_t, \phi^*_{t|t-1}\left(\{\boldsymbol{\mu}_i\}_{i=t-1}^{t}\right)\}, \tag{22}$$

for $j = t, \ldots, T-1$. For the latter problem, consider the class $\mathcal{U}$ of admissible control policies that consist of the sequence of control laws $\{\phi_{j|t}(\cdot)\}_{j=t}^{T-1}$ that are affine functions of the histories of states, that is,

$$\phi_{j|t}(\{\mathbf{z}_{i|t}\}_{i=t}^{j}) = \boldsymbol{\upsilon}_{j|t} + \sum_{i=t}^{j} K_{j,i|t} \mathbf{z}_{i|t}, \tag{23}$$

for $j = t, \ldots, T-1$. The linearized covariance steering problem at time step t is formulated as follows:

Problem 2. (t-th linearized covariance steering problem). Let $\boldsymbol{\mu}_t, \boldsymbol{\mu}_\mathrm{f} \in \mathbb{R}^{n_z}$ and $\Sigma_t, \Sigma_\mathrm{f} \in \mathbb{S}^{++}_{n_z}$ be given. Among all admissible control policies $\varpi_t := \{\phi_{j|t}(\cdot)\}_{j=t}^{T-1} \in \mathcal{U}$, with $\phi_{j|t}(\cdot)$ of the form (23), find a control policy ϖ_t^* that minimizes the performance index

$$J_t(\varpi_t) := \mathbb{E}\Big[\sum_{j=t}^{T-1} \phi_{j|t}(\{\mathbf{z}_{i|t}\}_{i=t}^{j})^\top \phi_{j|t}(\{\mathbf{z}_{i|t}\}_{i=t}^{j})\Big] \tag{24}$$

subject to the recursive dynamic constraints (21) and the boundary conditions

$$\mathbb{E}[\mathbf{z}_{t|t}] = \boldsymbol{\mu}_t, \qquad \mathrm{Cov}[\mathbf{z}_{t|t}] = \Sigma_t, \tag{25a}$$

$$\mathbb{E}[\mathbf{z}_{T|t}] = \boldsymbol{\mu}_\mathrm{f}, \qquad (\Sigma_\mathrm{f} - \mathrm{Cov}[\mathbf{z}_{T|t}]) \in \mathbb{S}^{+}_{n_z}. \tag{25b}$$

The performance index ensures that the control input will have finite energy, without excessive actuation. State and / or input constraints can also be incorporated in the above optimization-based solution (see Bakolas (2018)).

Problem 2 can be formulated as a convex *semi-definite program* (SDP) and, thus, can be solved efficiently using any available conic solver. The formulation of the SDP is ommitted in the interest of space, but can be found in Bakolas (2018) and Bakolas and Tsolovikos (2020).

For compactness, denote the solution to the t-th linearized covariance steering problem as

$$\{\phi^*_{j|t}(\cdot)\}_{j=t}^{T-1} = \mathrm{LCS}_{t,T}\{A_t, B_t, \mathbf{d}_t, \boldsymbol{\mu}_t, \Sigma_t, \boldsymbol{\mu}_\mathrm{f}, \Sigma_\mathrm{f}\}. \tag{26}$$

3.3 Gaussian Process-Based Unscented Transform for Uncertainty Propagation

Let $\pi = \{\kappa_t(\cdot)\}_{t=0}^{T-1}$ be an admissible control policy for Problem 1. Then, the closed-loop dynamics become

$$\mathbf{z}_{t+1} = G(\mathbf{z}_t, \kappa_t(\mathbf{z}_t)) + \mathbf{w}_t. \tag{27}$$

The mean and covariance of the uncertain state of the nonlinear system described by (27) is propagated using the *unscented transform* [Julier and Uhlmann (2004)]. To this aim, assume that the mean $\boldsymbol{\mu}_t := \mathbb{E}[\mathbf{z}_t]$ and covariance $\Sigma_t := \mathrm{Cov}[\mathbf{z}_t]$ of the state of (13) (or estimates of these quantities) are known at time step t.

First, we compute $2n_z + 1$ deterministic points, $\sigma_t^{(i)}$, $i = 1, \ldots 2n_z + 1$, which are also known as *sigma points*. Then, to each sigma point, we associate a pair of gains $(\gamma_t^{(i)}, \delta_t^{(i)})$. Subsequently, the sigma points $\{\sigma_t^{(i)}\}_{i=1}^{2n_z+1}$ are propagated to the next time step to obtain a new set of points $\{\hat{\sigma}_{t+1}^{(i)}\}_{i=1}^{2n_z+1}$, where

$$\hat{\sigma}_{t+1}^{(i)} = G(\sigma_t^{(i)}, \kappa_t(\sigma_t^{(i)})), \quad i = 1, \ldots, 2n_z. \tag{28}$$

Using this new point-set, one can approximate the (predicted) state mean and covariance at time step $t+1$ as

$$\hat{\boldsymbol{\mu}}_{t+1} = \sum_{i=0}^{2n_z} \gamma_t^{(i)} \hat{\sigma}_{t+1}^{(i)},$$

$$\hat{\Sigma}_{t+1} = \sum_{i=0}^{2n_z} \delta_t^{(i)} (\hat{\sigma}_{t+1}^{(i)} - \hat{\boldsymbol{\mu}}_{t+1})(\hat{\sigma}_{t+1}^{(i)} - \hat{\boldsymbol{\mu}}_{t+1})^\top + W_t.$$

Similar to Ko et al. (2007b), we set $W_t = \mathrm{Cov}[\mathbf{w}_t] = \Sigma_f([\mathbf{z}_t; \kappa_t(\mathbf{z}_t)], [\mathbf{z}_t; \kappa_t(\mathbf{z}_t)]) + \sigma_\epsilon^2 \in \mathbb{S}^{+}_{n_z}$ as the process noise covariance. Notice that W_t captures both the noise in the system as well as the model uncertainties resulting from the lack of training data points used in the learning phase, thus leading to conservative uncertainty estimates.

3.4 Greedy Nonlinear Covariance Steering for Gaussian Process Predictive Models

Now we have all the tools necessary to extend the *greedy nonlinear covariance steering* algorithm of Bakolas and Tsolovikos (2020) to Gaussian process predictive models. The greedy algorithm consists of three main steps. Consider the time step t, where $t = 0, \ldots, T-1$, and assume that estimates of the state mean, $\hat{\boldsymbol{\mu}}_t$, the state covariance, $\hat{\Sigma}_t$, as well as the input mean $\hat{\boldsymbol{\nu}}_t$, are known (starting from $\hat{\boldsymbol{\mu}}_0 = \boldsymbol{\mu}_0$, $\hat{\Sigma}_0 = \Sigma_0$, and $\hat{\boldsymbol{\nu}}_0 = 0$).

The *first step* is to linearize (13) around $(\hat{\boldsymbol{\mu}}_t, \hat{\boldsymbol{\nu}}_t)$:

$$\{A_t, B_t, \mathbf{d}_t\} = \mathrm{LIN}_G\{\hat{\boldsymbol{\mu}}_t, \hat{\boldsymbol{\nu}}_t\}, \tag{29}$$

where $\hat{\boldsymbol{\nu}}_t = \phi^*_{t|t-1}(\{\hat{\boldsymbol{\mu}}_i\}_{i=t-1}^{t})$. The linearization will have to be updated at each time step t since the estimates $\hat{\boldsymbol{\mu}}_t$ and $\hat{\boldsymbol{\nu}}_t$ will also be updated.

The *second step* is to solve the *t-th linearized covariance steering* problem (Problem 2) using the linearized model $\{A_t, B_t, \mathbf{d}_t\}$ and the predicted mean and covariance $(\hat{\boldsymbol{\mu}}_t, \hat{\Sigma}_t)$ estimated at time step t:

$$\{\phi^*_{j|t}(\cdot)\}_{j=t}^{T-1} = \mathrm{LCS}_{t,T}\{A_t, B_t, \mathbf{d}_t, \hat{\boldsymbol{\mu}}_t, \hat{\Sigma}_t, \boldsymbol{\mu}_\mathrm{f}, \Sigma_\mathrm{f}\}. \tag{30}$$

From $\{\phi^*_{j|t}(\cdot)\}_{j=t}^{T-1}$, we extract only the first control law,

$$\kappa_t(\mathbf{z}_t) := \phi^*_{t|t}(\mathbf{z}_t) = \boldsymbol{\upsilon}^*_{t|t} + K^*_{t|t}\mathbf{z}_t,$$

where $\mathbf{z}$ 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 t is then

$$\mathbf{z}_{t+1} = G(\mathbf{z}_t, \kappa_t(\mathbf{z}_t)) + \mathbf{w}_t. \tag{31}$$

In the *third step*, the estimates $(\hat{\boldsymbol{\mu}}_{t+1}, \hat{\Sigma}_{t+1})$ are computed by propagating the mean $\hat{\boldsymbol{\mu}}_t$ and covariance $\hat{\Sigma}_t$ of the closed-loop system to the next time step. The new mean and covariance, i.e., $\hat{\boldsymbol{\mu}}_{t+1}$ and $\hat{\Sigma}_{t+1}$, are computed using

the GP-based unscented transform described in Section 3.3. We write

$$(\hat{\boldsymbol{\mu}}_{t+1}, \hat{\Sigma}_{t+1}) := \mathrm{UT}\{\hat{\boldsymbol{\mu}}_t, \hat{\Sigma}_t; G(\cdot,\cdot), \kappa_t(\cdot)\}. \tag{32}$$

The three steps of the greedy covariance steering algorithm are repeated for all time steps $t = 0, \ldots, T-1$. At the end, 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 $\pi := \{\kappa_t(\mathbf{z})\}_{t=0}^{T-1}$ that solves Problem 1.

4. NUMERICAL RESULTS

In this section, the basic ideas of this paper are illustrated in numerical simulations. In particular, consider the following stochastic nonlinear system:

$$s_{x,t+1} = s_{x,t} + v_t \tau \cos\theta_t + \epsilon_t^{s_x}, \tag{33a}$$
$$s_{y,t+1} = s_{y,t} + v_t \tau \sin\theta_t + \epsilon_t^{s_y}, \tag{33b}$$
$$\theta_{t+1} = \theta_t + u_t^{\theta} v_t \tau + \epsilon_t^{\theta}, \tag{33c}$$
$$v_{t+1} = v_t + u_t^{v} \tau + \epsilon_t^{v}, \tag{33d}$$

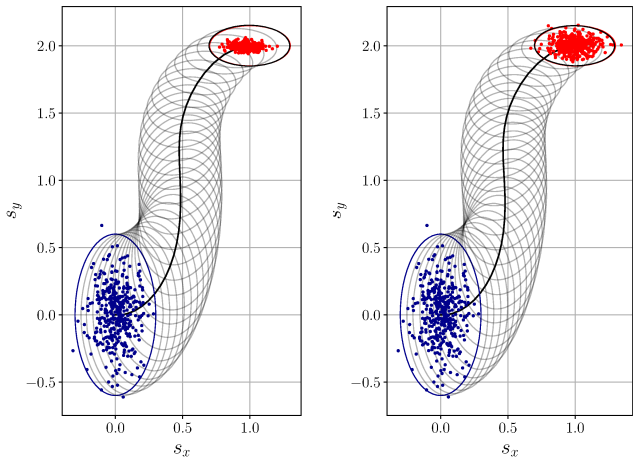
which is a discrete-time realization of a unicycle car model with state $\mathbf{z} = [s_x \ s_y \ \theta \ v]^\top$ and input $\mathbf{u} = [u^\theta \ u^v]^\top$. In order to train the SVGP model, we assume that a "black-box" simulator of the dynamics (33) is available for sampling full-state transitions $\mathbf{y}_i$ for given states $\mathbf{z}_i$ and inputs $\mathbf{u}_i$ (see (11) and (12), respectively). We run the simulator and collect a set of training data points that are then used to learn a stochastic variational GP model of the system dynamics, as presented in Section 2. Then, the learned model is used to steer the mean and covariance of the state of the underlying system from a given initial distribution to a prescribed terminal one. For our simulations, we consider (33) with time step $\tau = 0.05$, while the white noise $\epsilon_t \sim \mathcal{N}\left(\epsilon_t \mid 0, \mathrm{diag}(0.02, 0.02, 0.04, 0.04)^2\right)$.

Training: The SVGP model with zero mean, $m(\mathbf{x}) = 0$, squared-exponential kernel,

$$k(\mathbf{x}, \mathbf{x}') = \sigma_f^2 \exp\left(-1/2\,(\mathbf{x}-\mathbf{x}')^\top L^{-1} (\mathbf{x}-\mathbf{x}')\right)$$

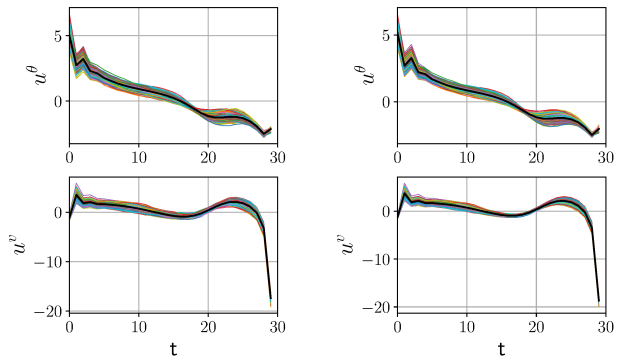
with separate length scales $L = \mathrm{diag}(l_1^2, \ldots, l_n^2)$ for each input dimension, and $M = 256$ inducing locations is setup using GPyTorch [Gardner et al. (2018)]. A set of $N = 9000$ data points $(\mathbf{x}_i, \mathbf{y}_i)$ are collected from randomly sampled states $\mathbf{z}$ and inputs $\mathbf{u}$ between $\mathbf{z}_{\min} = [-20 \ -20 \ -6\pi \ -10]^\top$, $\mathbf{u}_{\min} = [-20 \ -20]^\top$ and $\mathbf{z}_{\max} = [20 \ 20 \ 6\pi \ 20]^\top$, $\mathbf{u}_{\max} = [20 \ 20]^\top$, respectively. The variational parameters $\mathbf{Z}$, $\mathbf{m}$, and $\mathbf{S}$, along with the hyperparameters $\boldsymbol{\Theta} = \{\theta_m, \theta_t, \sigma_\epsilon\}$, are optimized by minimizing the negative lower bound, $-\mathcal{L}$, using the Adam optimizer [Kingma and Ba (2014)].

GP-Based Greedy Covariance Steering: Assume an initial state $\mathbf{z}_0$ drawn from a distribution with mean $\boldsymbol{\mu}_0 = [0 \ 0 \ 0 \ 1]^\top$ and covariance $\Sigma_0 = \mathrm{diag}(0.1, 0.2, 0.1, 0.1)^2$. The *target* terminal state mean and covariance are taken to be $\boldsymbol{\mu}_\mathrm{f} = [1 \ 2 \ 0 \ 1]^\top$ and $\Sigma_\mathrm{f} = \mathrm{diag}(0.1, 0.05, 0.05, 0.05)^2$, respectively. The greedy covariance steering algorithm of Section 3 is run with the identified SVGP model with a time horizon of $T = 30$ time steps. The Jacobians (18) for the model linearizations are computed using automatic differentiation in PyTorch [Paszke et al. (2017)]. For comparison, the *exact* model (33) is used with the greedy algorithm of Bakolas and Tsolovikos (2020) for the same initial and target distributions.



(a) SVGP Model (b) Exact Model

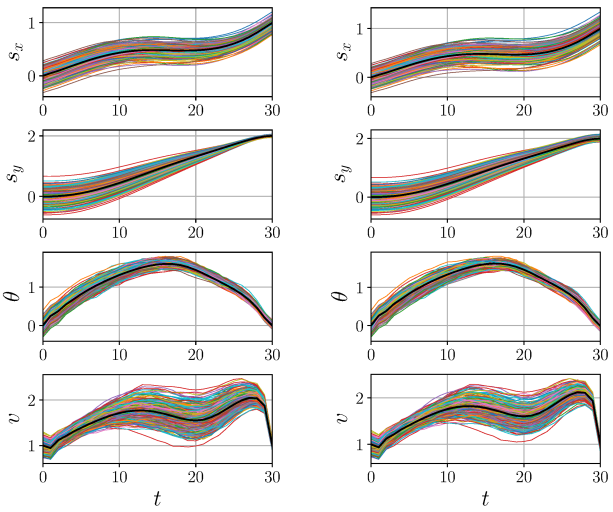
Fig. 1. Position (s_x, s_y) uncertainties. Blue: initial distribution. Red: target distribution. Black: actual terminal distribution. Gray: intermediate distributions. Solid black line: mean trajectory.



(a) SVGP Model (b) Exact Model

Fig. 2. Inputs $\mathbf{u}_t$. Solid black: mean input $\boldsymbol{\nu}_t$. Colored: inputs of Monte Carlo realizations.

Discussion: With the chosen target covariance, the goal is to shrink the uncertainty in the coordinate s_y, the angle θ, and the velocity v, while retaining the uncertainty in s_x. The mean trajectory of (s_x, s_y) along with the corresponding $3-\sigma$ uncertainty ellipsoids are plotted in Fig. 1. The GP-based algorithm (Fig. 1a) is compared with the greedy algorithm that uses the exact (analytic) model of (33) (Fig. 1b). The first thing we notice is that the uncertainty predicted by the SVGP model is overestimated. This is expected, since GPs provide conservative uncertainty estimates that include not only the uncertainties due to process noise, but also due to modeling errors. The terminal distribution estimated by the Unscented Transform (solid black ellipsoid) reaches almost perfectly the desired one (red ellipsoid). However, since the uncertainties provided by the SVGP model are conservative, the actual distribution – which is visualized as particles (red) from 400 Monte Carlo realizations – is shrinked compared to the target one. In comparison, the estimated terminal distribution for the exact model reaches almost perfectly the target one (see red particles in Fig. 1b). Thus, the use of SVGP model results in a more *cautious* covariance steering.



(a) SVGP Model (b) Exact Model

Fig. 3. States $\mathbf{z}_t$. Solid black: mean state $\mathbf{z}_t$. Colored: states of Monte Carlo realizations.

5. CONCLUSION

In this work, a greedy covariance steering algorithm that uses scalable Gaussian process predictive models for discrete-time stochastic nonlinear systems with unknown dynamics has been proposed. First, a non-parametric predictive model is learned from a set of training data points using stochastic variational 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. More practical cases, such as that of systems with incomplete state measurements and active model learning, will be explored in the future by the authors.

REFERENCES

Bakolas, E. (2016). Optimal covariance control for discrete-time stochastic linear systems subject to constraints. In *IEEE CDC (2016)*, 1153–1158.

Bakolas, E. (2018). Finite-horizon covariance control for discrete-time stochastic linear systems subject to input constraints. *Automatica*, 91, 61–68.

Bakolas, E. and Tsolovikos, A. (2020). Greedy finite-horizon covariance steering for discrete-time stochastic nonlinear systems based on the unscented transform. In *ACC (2020)*, 3595–3600.

Caluya, K.F. and Halder, A. (2019). Finite horizon density control for static state feedback linearizable systems. *arXiv preprint arXiv:1904.02272*.

Chen, Y., Georgiou, T., and Pavon, M. (2016a). Optimal steering of a linear stochastic system to a final probability distribution, Part I. *IEEE Trans. on Autom. Control*, 61(5), 1158 – 1169.

Chen, Y., Georgiou, T., and Pavon, M. (2016b). Optimal steering of a linear stochastic system to a final probability distribution, Part II. *IEEE Trans. Autom. Control*, 61(5), 1170–1180.

Gardner, J.R., Pleiss, G., Bindel, D., Weinberger, K.Q., and Wilson, A.G. (2018). GPyTorch: Blackbox matrix-matrix gaussian process inference with GPU acceleration. *arXiv preprint arXiv:1809.11165*.

Goldshtein, M. and Tsiotras, P. (2017). Finite-horizon covariance control of linear time-varying systems. In *IEEE CDC (2017)*, 3606–3611.

Grimes, D.B., Chalodhorn, R., and Rao, R.P. (2006). Dynamic imitation in a humanoid robot through nonparametric probabilistic inference. In *Robotics: science and systems*, 199–206. Cambridge, MA.

Hensman, J., Fusi, N., and Lawrence, N.D. (2013). Gaussian processes for big data. *arXiv preprint arXiv:1309.6835*.

Hewing, L., Kabzan, J., and Zeilinger, M.N. (2020). Cautious model predictive control using Gaussian process regression. *IEEE Transactions on Control Systems Technology*, 28(6), 2736–2743.

Hoffman, M.D., Blei, D.M., Wang, C., and Paisley, J. (2013). Stochastic variational inference. *Journal of Machine Learning Research*, 14(5).

Hotz, A. and Skelton, R.E. (1987). Covariance control theory. *Int. J. Control*, 16(1), 13–32.

Julier, S.J. and Uhlmann, J.K. (2004). Unscented filtering and nonlinear estimation. *Proc. IEEE*, 92(3), 401–422.

Kingma, D.P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Ko, J. and Fox, D. (2009). GP-Bayesfilters: Bayesian filtering using Gaussian process prediction and observation models. *Autonomous Robots*, 27(1), 75–90.

Ko, J., Klein, D.J., Fox, D., and Haehnel, D. (2007a). Gaussian processes and reinforcement learning for identification and control of an autonomous blimp. In *ICRA (2007)*, 742–747.

Ko, J., Klein, D.J., Fox, D., and Haehnel, D. (2007b). GP-UKF: Unscented Kalman filters with Gaussian process prediction and observation models. In *IROS (2007)*, 1901–1907.

Mesbah, A., Streif, S., Findeisen, R., and Braatz, R.D. (2014). Stochastic nonlinear model predictive control with probabilistic constraints. In *ACC (2014)*, 2413–2419.

Mukadam, M., Yan, X., and Boots, B. (2016). Gaussian process motion planning. In *ICRA (2016)*, 9–15.

Pan, Y. and Theodorou, E.A. (2015). Data-driven differential dynamic programming using Gaussian processes. In *ACC (2015)*, 4467–4472.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. (2017). Automatic differentiation in PyTorch.

Quinonero-Candela, J. and Rasmussen, C.E. (2005). A unifying view of sparse approximate Gaussian process regression. *The Journal of Machine Learning Research*, 6, 1939–1959.

Rasmussen, C.E. (2003). Gaussian processes in machine learning. In *Summer School on Machine Learning*, 63–71. Springer.

Ridderhof, J., Okamoto, K., and Tsiotras, P. (2019). Nonlinear uncertainty control with iterative covariance steering. In *CDC (2020)*, 3484–3490.

Sehr, M.A. and Bitmead, R.R. (2017). Particle model predictive control: Tractable stochastic nonlinear output-feedback MPC. In *20th IFAC World Congress*, 15361 – 15366.

Titsias, M. (2009). Variational learning of inducing variables in sparse Gaussian processes. In *Artificial intelligence and statistics*, 567–574. PMLR.

Xu, J.H. and Skelton, R.E. (1992). An improved covariance assignment theory for discrete systems. *IEEE Trans. Autom. Control*, 37(10), 1588–1591.