MITSUBISHI ELECTRIC RESEARCH LABORATORIES
https://www.merl.com

Online Constrained Bayesian Inference and Learning of Gaussian-Process State-Space Models

Berntorp, Karl; Menner, Marcel

TR2022-066 June 11, 2022

Abstract

Recent research has shown that it is possible to perform online learning of nonlinear dynamical systems. Furthermore, the results suggest that combining approximate Gaussian-process (GP) regression with model-based estimators, such as Kalman filters and particle filters (PFs), leads to efficient learners under the GP-state-space model (GP-SSM) framework. Here, we analyze how learning of GP-SSMs can be done when there are constraints on the system to be learned. Our analysis is based on a recently developed online PF-based learning method, where the GP-SSM is expressed as a basis-function expansion. We show that the method by adaptation of the basis functions can satisfy several constraints, such as symmetry, antisymmetry, Neumann boundary conditions, and linear operator constraints. A Monte-Carlo simulation study indicates reduced estimation errors with more than 50%.

American Control Conference (ACC) 2022

© 2022 MERL. This work may not be copied or reproduced in whole or in part for any commercial purpose. Permission to copy in whole or in part without payment of fee is granted for nonprofit educational and research purposes provided that all such whole or partial copies include the following: a notice that such copying is by permission of Mitsubishi Electric Research Laboratories, Inc.; an acknowledgment of the authors and individual contributions to the work; and all applicable portions of the copyright notice. Copying, reproduction, or republishing for any other purpose shall require a license with payment of fee to Mitsubishi Electric Research Laboratories, Inc. All rights reserved.

Mitsubishi Electric Research Laboratories, Inc.
201 Broadway, Cambridge, Massachusetts 02139

Online Constrained Bayesian Inference and Learning of Gaussian-Process State-Space Models

Karl Berntorp and Marcel Menner

***Abstract*— Recent research has shown that it is possible to perform online learning of nonlinear dynamical systems. Furthermore, the results suggest that combining approximate Gaussian-process (GP) regression with model-based estimators, such as Kalman filters and particle filters (PFs), leads to efficient learners under the GP-state-space model (GP-SSM) framework. Here, we analyze how learning of GP-SSMs can be done when there are constraints on the system to be learned. Our analysis is based on a recently developed online PF-based learning method, where the GP-SSM is expressed as a basis-function expansion. We show that the method by adaptation of the basis functions can satisfy several constraints, such as symmetry, antisymmetry, Neumann boundary conditions, and linear operator constraints. A Monte-Carlo simulation study indicates reduced estimation errors with more than 50%.**

I. Introduction

We consider online Bayesian learning of Gaussian-process state-space models (GP-SSMs)

$$\boldsymbol{x}_{k+1} = \boldsymbol{f}(\boldsymbol{x}_k) + \boldsymbol{w}_k, \tag{1a}$$

$$\boldsymbol{y}_k = \boldsymbol{h}(\boldsymbol{x}_k) + \boldsymbol{e}_k, \tag{1b}$$

where the (latent) state $\boldsymbol{x}_k \in \mathbb{R}^{n_x}$ at each time step k is observed through the measurement $\boldsymbol{y}_k \in \mathbb{R}^{n_y}$. The nonlinear functions $\boldsymbol{f} : \mathbb{R}^{n_x} \to \mathbb{R}^{n_x}$ and $\boldsymbol{h} : \mathbb{R}^{n_x} \to \mathbb{R}^{n_y}$ are assumed to be realizations from GPs. The process noise $\boldsymbol{w}_k$ and measurement noise $\boldsymbol{e}_k$ are Gaussian distributed with covariance $\boldsymbol{Q}$ and $\boldsymbol{R}$ according to $\boldsymbol{w}_k \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{Q})$ and $\boldsymbol{e}_k \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{R})$, respectively. Note that the formulation (1) by a coordinate transformation also includes partially unknown systems. GPs are nonparametric modeling tools for learning models from data [1], which imply flexibility in the ability to model general nonlinear functions without a priori enforcing an explicit parametric structure. Originally, GPs were mostly used for learning of systems akin to (1b). However, during the last decade, numerous succesful methods have emerged for offline GP-based learning of nonlinear dynamical systems [2]–[5], where the GP-SSM (1) is an interesting subset.

Learning in GP-SSMs amounts to estimating the posterior distributions of the state-transition function $\boldsymbol{f}$, measurement function $\boldsymbol{h}$, and possibly also the associated covariance matrices $\boldsymbol{Q}$, $\boldsymbol{R}$, and the hyperparameters associated with the GPs. This problem is difficult to solve for several reasons. First, the state is implicitly observed through the measurement model (1b), and the quality of the state estimates affects the model identification, and vice versa, which necessitates including state inference in the learning process. Second, since the GP-SSM (1) is nonlinear and at least partially unknown, the state inference problem is nonlinear and cannot be solved with analytic methods. Offline, several iterative procedures exist, for example, using particle Markov-chain Monte-Carlo (PMCMC) [5], [6]. However, online, the inference and learning problem must be addressed simultaneously.

Recently, several methods for GP-based online learning have been proposed; for example, [7] proposes a particle-filter (PF) based method in combination with a Gaussian kernel basis-function expansion whose weights are learned using MCMC, based on the maximum-likelihood state estimate from the PF. The work [8] uses online sparse GP regression, and [9] uses time-varying inducing points and importance sampling akin to the PF. In [10], we developed a PF-based method that leverages a reduced-rank formulation [5], [11] of the GP-SSM, in which connections between GPs and a finite basis-function expansion of the unknown system dynamics is made by introducing certain priors on the basis-function coefficients. The method in [10] is highly accurate and shown to provide computation times suitable for real-time execution of low-dimensional problems such as tire-friction estimation for vehicle-control applications. However, it scales unsatisfactory with the state dimensions. To remedy this, the work in [12] combines an extended Kalman filter (EKF) with a radial basis-function expansion to provide a computationally efficient way to perform online learning in SSMs, even for high-dimensional systems.

All of the abovementioned papers consider learning without addressing structural properties (constraints) of the system. However, in practice, the efficacy of, for example, the choice of basis functions or kernel, is tightly connected to the particular system being learned. In this paper, we investigate the ability of our previously developed online state inference and learning method [10] for learning of systems with constraints, which can be motivated either by physics (e.g., the tire–road friction is antisymmetric) or some other prior knowledge (e.g., we only consider tire–road friction when the vehicle is braking). For simplicity, we focus on $\boldsymbol{f}$ in (1a), but the extension to $\boldsymbol{h}$ in (1b) is analogous. We show that our method by construction can incorporate constraints such as symmetry ($\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{f}(-\boldsymbol{x})$) and antisymmetry ($\boldsymbol{f}(\boldsymbol{x}) = -\boldsymbol{f}(-\boldsymbol{x})$). Furthermore, inspired by [13], we can also improve estimation performance for linearly constrained systems. Note that we refer to constraints as limitations on the system $\boldsymbol{f}$ and not on the state $\boldsymbol{x}$.

Notation: For a vector $\boldsymbol{x}$, $\boldsymbol{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ indicates that $\boldsymbol{x} \in \mathbb{R}^{n_x}$ is Gaussian distributed with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$ and x_n denotes the nth component of $\boldsymbol{x}$. Matrices are indicated in capital bold font as $\boldsymbol{X}$. The determinant of $\boldsymbol{X}$ is $|\boldsymbol{X}|$

Mitsubishi Electric Research Laboratories (MERL), 02139 Cambridge, MA, USA. Email: karl.o.berntorp@ieee.org

and the trace of $\boldsymbol{X}$ is $\mathrm{Tr}(\boldsymbol{X})$. With $p(\boldsymbol{x}_{0:k}|\boldsymbol{y}_{0:k})$, we mean the posterior density function of the state trajectory $\boldsymbol{x}_{0:k}$ from time step 0 to time step k given the measurement sequence $\boldsymbol{y}_{0:k} := \{\boldsymbol{y}_0, \ldots, \boldsymbol{y}_k\}$, and $\boldsymbol{x}_{0:k}^i$ is the ith realization of $\boldsymbol{x}_{0:k}$. The notation $f \sim \mathcal{GP}(0, \kappa(x, x'))$ means that the function $f(x)$ is a realization from a GP prior with a given covariance function $\kappa(x, x')$ subject to some hyperparameters $\boldsymbol{\vartheta}$, and $\mathcal{IW}(\nu, \boldsymbol{\Lambda})$ is the inverse-Wishart distribution with degree of freedom ν and scale matrix $\boldsymbol{\Lambda}$. We write $\mathcal{MN}(\boldsymbol{M}, \boldsymbol{Q}, \boldsymbol{V})$ to denote the Matrix-Normal distribution with mean $\boldsymbol{M}$, right covariance (scale) $\boldsymbol{Q}$, and left covariance (scale) $\boldsymbol{V}$. Finally, the compund distribution $\mathcal{MNIW}(\boldsymbol{A}, \boldsymbol{Q}|\boldsymbol{M}, \boldsymbol{V}, \boldsymbol{\Lambda}, \nu) = \mathcal{MN}(A|\boldsymbol{M}, \boldsymbol{Q}, \boldsymbol{V})\mathcal{IW}(\boldsymbol{Q}|\nu, \boldsymbol{\Lambda})$.

II. Online Bayesian Inference and Learning

We give a brief description of the online PF-based inference and learning method for GP-SSMs in [10].

A. Approximate Formulation of a GP

The learning method in [10] expresses $\boldsymbol{f} = [f_1 \ \cdots \ f_{n_x}]^\top$ by a basis-function expansion according to

$$\hat{f}_i(\boldsymbol{x}) = \sum_{j=1}^{M} \gamma_{ij} \phi_j(\boldsymbol{x}) \tag{2}$$

for each $i = 1, 2, \ldots, n_x$, where the weights γ_{ij} are to be determined. To enable tractability in the learning problem and not introduce too much flexibility, the order n_x of $\boldsymbol{f}$ is known and the number of basis functions M is set a priori. The basis-function expansion (2) leads to a joint inference and learning approach that is linear in the weights γ_{ij}.

To make connections to GPs, we choose the eigenfunctions with associated eigenvalues λ_j to the Laplace operator, which for the domain $[-L_1, L_1] \times \cdots \times [-L_{n_x}, L_{n_x}] \in \mathbb{R}^{n_x}$ are

$$\phi_{j_1,\ldots,j_{n_x}} = \prod_{n=1}^{n_x} \frac{1}{\sqrt{L_n}} \sin\left(\frac{\pi j_n (x_n + L_n)}{2L_n}\right), \tag{3a}$$

$$\lambda_{j_1,\ldots,j_{n_x}} = \sum_{n=1}^{n_x} \left(\frac{\pi j_n}{2L_n}\right)^2. \tag{3b}$$

From (3a), it is clear that the number of weights to learn nominally increases exponentially with the state dimension as $M = m^{n_x}$, where m is the number of weights for $n_x = 1$. This is unwanted for many basis-function expansions, but possible to alleviate. In [5], two possibilities are pointed out. First, to assume independence between the different dimensions. Second, to choose another set of basis function.

For covariance functions that are a function of $r = |\boldsymbol{x} - \boldsymbol{x}'|$, [11] provides a connection between GPs and a basis-function expansion of a function f with (3). The connection between the basis-function expansion (2) with basis functions (3a) is

$$\boldsymbol{f}(\boldsymbol{x}) \sim \mathcal{GP}(\boldsymbol{0}, \boldsymbol{\kappa}(\boldsymbol{x}, \boldsymbol{x}')) \Leftrightarrow \boldsymbol{f}(\boldsymbol{x}) \approx \sum_{j=1}^{M} \boldsymbol{\gamma}_j \phi_j(\boldsymbol{x}), \tag{4}$$

with $\boldsymbol{\gamma}_j = [\gamma_{1j} \ \cdots \ \gamma_{n_x j}]^\top$ and

$$\gamma_{ij} \sim \mathcal{N}(0, \mathcal{S}(\lambda_j)). \tag{5}$$

We use the squared exponential covariance function,

$$\kappa(r) = \sigma^2 \exp\left(-\frac{r^2}{2\ell^2}\right), \tag{6}$$

with hyperparameters $\boldsymbol{\vartheta} = \{\sigma, \ell\}$, where for simplicity we assume the same hyperparameters for each dimension. The covariance (6) has the spectral density

$$\mathcal{S}(\omega) = \sigma^2 \sqrt{2\pi\ell^2} \exp\left(-\frac{\pi^2 \ell^2 \omega^2}{2}\right). \tag{7}$$

With the connection (4), (7) can be used to assign suitable priors on the weights. With the basis-function expansion (2) and basis functions (3), a reduced-rank GP-SSM (1a) is

$$\boldsymbol{x}_{k+1} = \underbrace{\begin{bmatrix} \gamma_{11} & \cdots & \gamma_{1m} \\ \vdots & & \vdots \\ \gamma_{n_x 1} & \cdots & \gamma_{n_x m} \end{bmatrix}}_{\boldsymbol{A}} \underbrace{\begin{bmatrix} \phi_1(\boldsymbol{x}_k) \\ \vdots \\ \phi_M(\boldsymbol{x}_k) \end{bmatrix}}_{\boldsymbol{\varphi}(\boldsymbol{x}_k)} + \boldsymbol{w}_k. \tag{8}$$

In the limit, (8) converges to (1) [11].

The method considers approximate joint state inference and learning of the GP-SSM (1a) with known measurement equation (1b). Instead of targeting (1a), the method learns (8). This implies estimating the posterior distributions of $\boldsymbol{x}$, $\boldsymbol{A}$, and $\boldsymbol{Q}$, at each time step k. Since the method is aimed at online applications, it approximates the marginal (filtering) distributions. The involved distributions cannot be resolved analytically, and the estimation of $\boldsymbol{x}$, $\boldsymbol{A}$, and $\boldsymbol{Q}$ is therefore based on a tailored PF for approximating $p(\boldsymbol{x}_k|\boldsymbol{y}_{0:k})$ and $p(\boldsymbol{\theta}_k|\boldsymbol{y}_{0:k})$, where $\boldsymbol{\theta} = \{\boldsymbol{A}, \boldsymbol{Q}\}$. Specifically, the method approximates the joint posterior density $p(\boldsymbol{x}_{0:k+1}, \boldsymbol{\theta}_k|\boldsymbol{y}_{0:k})$, from which marginal densities $p(\boldsymbol{x}_k|\boldsymbol{y}_{0:k})$ and $p(\boldsymbol{\theta}_k|\boldsymbol{y}_{0:k})$ can be computed. We decompose the joint posterior as

$$p(\boldsymbol{x}_{0:k+1}, \boldsymbol{\theta}_k|\boldsymbol{y}_{0:k}) = p(\boldsymbol{\theta}_k|\boldsymbol{x}_{0:k+1}, \boldsymbol{y}_{0:k}) p(\boldsymbol{x}_{0:k+1}|\boldsymbol{y}_{0:k}). \tag{9}$$

The two densities on the right-hand side of (9) are estimated recursively. Given the state trajectory, we update the sufficient statistics of the unknown parameters. The second distribution on the right-hand side of (9) is approximated with the PF, while the first distribution on the right-hand side is updated analytically, for each particle.

B. Estimating the Parameter Posterior

The distribution of $\boldsymbol{\theta}_k$ is computed conditioned on the realization of the state and measurement trajectories for each particle. For a specific realization $\boldsymbol{x}_{0:k+1}$, the posterior density of $\boldsymbol{\theta}_k$ can be written according to

$$p(\boldsymbol{\theta}_k|\boldsymbol{x}_{0:k+1}, \boldsymbol{y}_{0:k}) = p(\boldsymbol{\theta}_k|\boldsymbol{x}_{0:k+1}), \tag{10}$$

since (1b) is not dependent on $\boldsymbol{\theta}_k$. Using Bayes' rule, (10) can be decomposed into a likelihood and prior as

$$p(\boldsymbol{\theta}_k|\boldsymbol{x}_{0:k+1}) \propto p(\boldsymbol{x}_{k+1}|\boldsymbol{\theta}_k, \boldsymbol{x}_{0:k}) p(\boldsymbol{\theta}_k|\boldsymbol{x}_{0:k}). \tag{11}$$

The first term on the right-hand side of (11) is the transition density of the reduced-rank model (8),

$$p(\boldsymbol{x}_{k+1}|\boldsymbol{\theta}_k, \boldsymbol{x}_k) = \mathcal{N}(\boldsymbol{x}_{k+1}|\boldsymbol{A}_k \boldsymbol{\varphi}(\boldsymbol{x}_k), \boldsymbol{Q}_k). \tag{12}$$

To get a recursive expression for updating (11), the method expresses the prior on the coefficients γ_{ij} in (5) at time step $k = 0$ as a zero-mean $\mathcal{MN}$ distribution [14] over $\boldsymbol{A}$,

$$\boldsymbol{A} \sim \mathcal{MN}(\boldsymbol{0}, \boldsymbol{Q}, \boldsymbol{V}), \tag{13}$$

with right covariance $\boldsymbol{Q}$ and left covariance $\boldsymbol{V}$ with diagonal elements $\mathcal{S}(\lambda_j)$, which incorporates the prior (5). Assuming the covariance prior to be $\mathcal{IW}$ distributed is common in covariance estimation due to its beneficial properties [5], [15], [16]. Using the $\mathcal{MNIW}$ as a prior distribution, [10] formulates a recursive expression for analytically updating the sufficient statistics.

C. Particle Filtering for State Inference

PFs approximate the posterior density $p(\boldsymbol{x}_{0:k}|\boldsymbol{y}_{0:k})$ by a set of N weighted trajectories,

$$p(\boldsymbol{x}_{0:k}|\boldsymbol{y}_{0:k}) \approx \sum_{i=1}^{N} q_k^i \delta_{\boldsymbol{x}_{0:k}^i}(\boldsymbol{x}_{0:k}), \tag{14}$$

where q_k^i is the importance weight of the ith state trajectory $\boldsymbol{x}_{0:k}^i$ and $\delta(\cdot)$ is the Dirac delta mass. The PF algorithm iterates between prediction and weight update, combined with a resampling step that removes particles with low weights and replaces them with more likely particles. The particle weights are typically updated as

$$q_k^i \propto q_{k-1}^i p(\boldsymbol{y}_k|\boldsymbol{x}_k^i). \tag{15}$$

III. Learning with Constraints

The efficiency of GP-based models is tightly connected to how prior knowledge about the system is incorporated into the GP modeling. Perhaps the most obvious way to enforce system constraints in learning with GPs is to add artifical measurements that obey the constraints at given points in the state space. However, this leads to an increased problem dimension, and the constraints are only guaranteed to be satisfied at the points in the state space corresponding to the artifical measurements. We focus on satisfying the constraints by construction of the basis functions.

A. Boundary Conditions

Boundary-condition constraints arise in a number of applications, for example, in simultaneous localization and mapping and spatial data analysis. The eigenfunctions and corresponding eigenvalues in (3) are given by the solution to the Laplace operator with Dirichlet boundary conditions [11] on a rectangular grid. As such, the basis-function expansion (2) with eigenfunctions and eigenvalues in (3) directly satisfy boundary conditions of the form $\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{0},\ \boldsymbol{x} \in \Omega\boldsymbol{x}$, where $\Omega\boldsymbol{x}$ is the boundary of the domain of $\boldsymbol{x}$.

With a different choice of eigenvalues, other boundary conditions can be accommodated; for instance, Neumann boundary conditions, that is,

$$\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{n}}(\boldsymbol{x}) = \boldsymbol{0},\ \boldsymbol{x} \in \Omega\boldsymbol{x}, \tag{16}$$

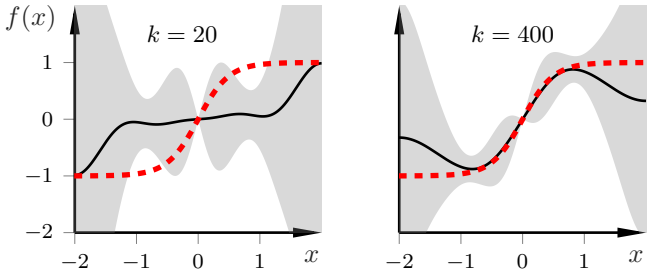

Fig. 1. Illustration of Neumann boundary conditions at $x = \pm 2$. The red dashed line is the true function, weighted mean estimate of our approach is in black solid, and the gray area indicates the 2σ confidence bounds.

where $\boldsymbol{n}$ is the tangent of the boundary, can be satisfied by the eigenfunctions and eigenvalues

$$\phi_{j_1,\ldots,j_{n_x}} = \prod_{n=1}^{n_x} \frac{1}{\sqrt{L_n}} \cos\left(\frac{\pi j_n (x_n + L_n)}{2L_n}\right), \tag{17a}$$

$$\lambda_{j_1,\ldots,j_{n_x}} = \sum_{n=1}^{n_x} \left(\frac{\pi j_n}{2L_n}\right)^2. \tag{17b}$$

1) Illustrative Example: To illustrate the Neumann conditions (16), consider the system

$$x_{k+1} = \tanh(2x_k) + w_k, \quad w_k \sim \mathcal{N}(0, 0.1), \tag{18a}$$

$$y_k = x_k + e_k, \quad e_k \sim \mathcal{N}(0, 0.1), \tag{18b}$$

where the objective is to learn $f(x_k) = \tanh(2x_k)$ and $Q_w = 0.1$, as well as estimating x_k. The function f in (18a) has the property that $\frac{\mathrm{d}f}{\mathrm{d}x} \to 0$ as $|x|$ grows large. We enforce the boundary condition at $L = 2$ in (17), where $\frac{\mathrm{d}f}{\mathrm{d}x} \approx 0$, and use $M = 16$ basis functions.

Fig. 1 shows an illustration of Neumann boundary conditions, where the boundaries are at $x = \pm 2$. With the choice of basis functions (17a), the estimated function has zero gradient for all time steps at $x = \pm 2$. Since boundary conditions are inherently enforced in the approach, we do not elaborate further here but refer to [5], [10], [11] for some illustrations of boundary conditions.

B. Symmetry

Symmetry constraints is another set of constraints that can be easily incorporated. Examples of systems that include symmetry constraints are double integrators, a pendulum when controlling the torque at the base, and vehicle lane-change control. For the method in Sec. II, symmetry can be enforced by a simple reindexing of the basis-function expansion (3a). By only choosing odd indices $j = 1, 3, \ldots$, the method satisifies symmetry constraints.

1) Illustrative Example: Consider

$$x_{k+1} = 10\mathrm{sinc}\left(\frac{x_k}{7}\right) + w_k, \quad w_k \sim \mathcal{N}(0, 4), \tag{19a}$$

$$y_k = x_k + e_k, \quad e_k \sim \mathcal{N}(0, 4), \tag{19b}$$

where the objective is to learn $f = 10\mathrm{sinc}\left(\frac{x_k}{7}\right)$ when the noise variance is known. We use $L = 30$ and $M = 35$, with $s_f = 50$ and $l = 3$, and the number of particles is $N = 50$.

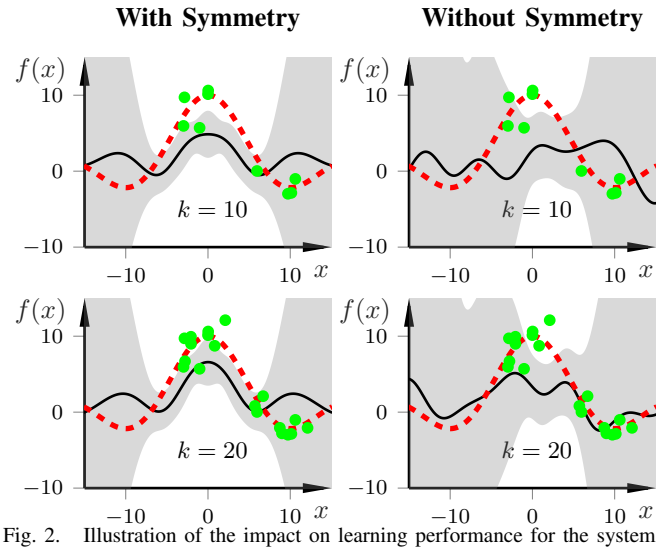

Fig. 2. Illustration of the impact on learning performance for the system (19) when explicitly enforcing symmetry (left column) and when using the standard basis functions (right column). Same notation as in Fig. 1.

Fig. 2 compares the estimation results when enforcing symmetry according to

$$\hat{f}(x) = \sum_{j=1,3,5,\ldots}^{35} \gamma_j \frac{1}{\sqrt{L}} \sin\left(\frac{\pi j(x+L)}{2L}\right) \tag{20}$$

and for the original basis-function expansion (3), $j = 1, \ldots, 36$, for $k = 10, 20, 30$. By enforcing symmetry, the estimation becomes more reliable in the sense that sensible estimates are achieved much faster and the uncertainty of the estimates, indicated by the gray area, is reduced.

C. Antisymmetry

Antisymmetry can be handled by choosing even indices of the basis functions (3a), that is, $j = 2, 4, \ldots$, which ensures that antisymmetry is enforced.

1) Application to Tire-Friction Estimation: The friction dependence between tire and road is highly nonlinear and varies heavily between different surfaces. The friction estimation problem is highly relevant for vehicle-control applications [16], [17]. The vehicle model relating the vehicle state to the tire friction is the *single-track model*, which depends on the lateral friction function μ^y and the subscripts f, r stand for front and rear wheel axle. In the estimation model, the tire-friction components μ_i^y, $i \in \{f, r\}$ are modeled as static functions of the slip quantities,

$$\mu_i^y = f_i^y(\alpha_i(\boldsymbol{x})),\ i \in \{f, r\}, \tag{21}$$

α_i is the slip angle, $\alpha_i = -\arctan(v_{y,i}/v_{x,i})$, where $v_{x,i}$ and $v_{y,i}$ are the longitudinal and lateral wheel velocities for wheel i with respect to an inertial system. The simulation uses the *Pacejka tire model* [18]

$$\mu_i = \bar{\mu}_i \sin(C_i \arctan(B_i(1-E_i)\alpha_i + E_i \arctan(B_i\alpha_i))), \tag{22}$$

where $\bar{\mu}$, B, C, and E are the peak, stiffness, shape, and curvature factor, respectively. For brevity, we define the vector $\boldsymbol{\alpha} = [\alpha_f\ \alpha_r]^{\mathrm{T}}$ and write (21) as $\boldsymbol{\mu} = [f_f^y\ f_r^y]^\top$, and model the friction vector as a realization from $\mathcal{GP}(\mathbf{0}, \kappa(\boldsymbol{\alpha}, \boldsymbol{\alpha}'))$.

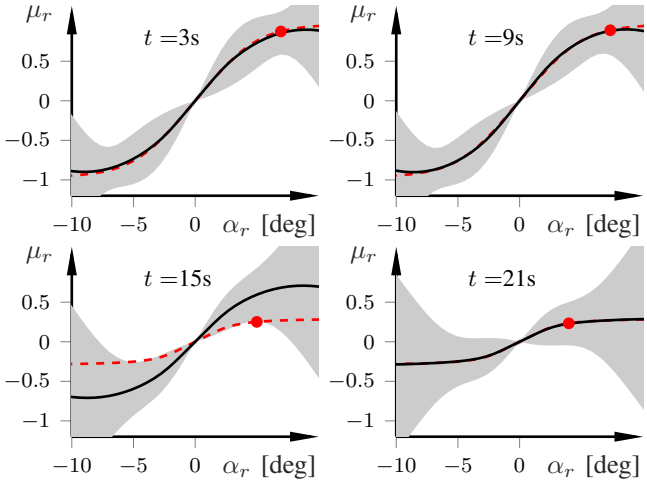

Fig. 3. Antisymmetry-enforced tire-friction estimates (black solid) and estimated 2σ confidence (gray area) at different time steps for one realization. The true tire-friction function as modeled by (22) is in red dashed.

The resulting estimation model can be written in the form (1a). The measurement model is based on a setup commonly available in production cars, namely the lateral acceleration a_m^Y and the yaw rate $\dot{\psi}_m$, forming the measurement vector $\boldsymbol{y} = [a_m^Y\ \dot{\psi}_m]^\top$. We model the measurement noise $\boldsymbol{e}_k$ as zero-mean Gaussian distributed noise with known covariance $\boldsymbol{R}$ according to $\boldsymbol{e}_k \sim \mathcal{N}(\mathbf{0}, \boldsymbol{R})$, and the resulting measurement model is on the form (1b).

We use 10 basis functions each for the front and rear tire. From expert knowledge (i) the friction between the front and rear wheel can be estimated independently, (ii) the tire friction is antisymmetric. This gives $M = 10$ basis functions in total if we enforce antisymmetry and $M = 20$ if we do not, whereas without using the expert knowledge $M = 10^2 = 100$ basis functions would have been needed. The number of particles is $N = 100$. In (7), $s_f = 50$, which allows the initial uncertainty to cover the range of possible friction values. We set $L = 30\pi/180$, since α usually is restricted to $-15 \lessapprox \alpha \lessapprox 15$ deg, and $\ell = 2\pi/180$.

Fig. 3 shows the estimation results at different timesteps when enforcing antisymmetry, and Fig. 4 shows the same when we do not enforce antisymmetry. In the simulation, there is a surface change from asphalt to snow at $t = 14.5$s, simulated by an instant change of the tire parameters in (22). By visual inspection, it is clear that the results when enforcing antisymmetry are more accurate, and the uncertainty is reduced. Also, convergence is reached faster when enforcing antisymmetry, since excitation of the system for $\alpha_j > 0$ also gives information for $\alpha_j < 0$, and vice versa.

D. Linear Operator Constraints

In this section we consider learning of (1) when $\boldsymbol{f}$ is subject to linear operator constraints, which means that

$$\mathcal{F}_{\boldsymbol{x}}(\boldsymbol{f}) = \mathbf{0} \tag{23}$$

for an operator $\mathcal{F}_{\boldsymbol{x}}$ fulfilling

$$\mathcal{F}_{\boldsymbol{x}}(\lambda_1 \boldsymbol{f}_1 + \lambda_2 \boldsymbol{f}_2) = \lambda_1 \mathcal{F}_{\boldsymbol{x}}(\boldsymbol{f}_1) + \lambda_2 \mathcal{F}_{\boldsymbol{x}}(\boldsymbol{f}_2), \tag{24}$$

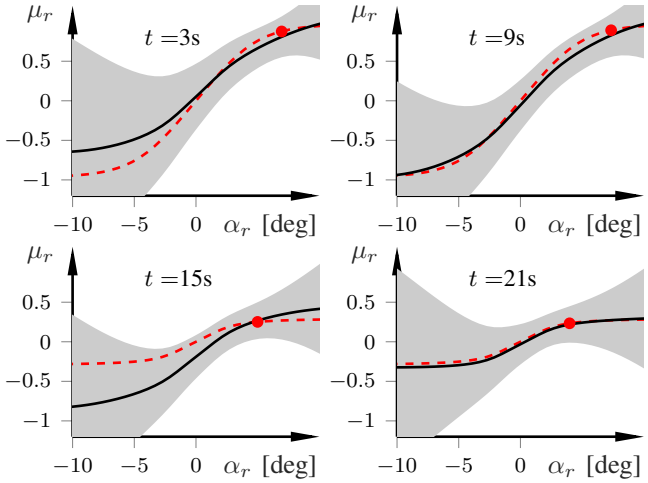

Fig. 4. Tire-friction estimates (black solid) and estimated 2σ confidence (gray area) at different time steps for one realization, without enforcing antisymmetry. The true tire-friction function as modeled by (22) is in red dashed.

for two functions $\boldsymbol{f}_1, \boldsymbol{f}_2$, where $\lambda_1, \lambda_2 \in \mathbb{R}$. Some examples of constraints are $\boldsymbol{A}\boldsymbol{f} = \boldsymbol{0}$; $\nabla \boldsymbol{f} = \boldsymbol{0}$; $\int \boldsymbol{f}(\boldsymbol{x})\mathrm{d}\boldsymbol{x}$; and differentiation.

The work [13] presents a method to transform a given GP into a new GP that satisfies linear operator constraints (23), by choosing a GP mean and covariance function by construction satisfying the constraints. Unlike [13], since we use the reduced-rank formulation of a GP, we can design the basis functions such that they satisfy the constraints.

The approach in [13] uses that GPs are closed under linear operators. It guarantees that a sample drawn from the resulting GP obeys the constraints. To do this, [13] relates $\boldsymbol{f}$ with a function $\boldsymbol{g}$ via some other linear operator $\mathcal{G}_{\boldsymbol{x}}$,

$$\boldsymbol{f} = \mathcal{G}_{\boldsymbol{x}}(\boldsymbol{g}). \tag{25}$$

With a matrix-multiplication interpretation, the constraint (23) can from (25) be written as a constraint on $\mathcal{G}_{\boldsymbol{x}}$,

$$\mathcal{F}_{\boldsymbol{x}}\mathcal{G}_{\boldsymbol{x}} = \boldsymbol{0}. \tag{26}$$

The approach in [13] uses (26) to find a GP mean and covariance function prior for $\boldsymbol{f}$ that fulfills the constraints (23). One strength with this approach is that the prior for $\boldsymbol{g}$ can be designed independent from the constraint satisfaction, which means that other desired properties (e.g., smoothness) can be handled by the choice of $\boldsymbol{g}$. Our approach is based on [13]. However, by virtue of our basis-function expansion, we can instead proceed as in Algorithm 1 for generating a *basis-function approximation* of a GP satisfying the constraints (23). Line 1 in Algorithm 1 is a standard design step. The

Algorithm 1 $\boldsymbol{f}$ fulfilling linear operator constraints

1: Choose a basis-function expansion for $\boldsymbol{g}$.
2: Find an operator $\mathcal{G}_{\boldsymbol{x}}$ that fulfills (26).
3: Find $\hat{\boldsymbol{f}}$ according to (25).

work [13] includes an ansatz-based algorithm for finding the operator $\mathcal{G}_{\boldsymbol{x}}$ that can be applied also to our setting.

1) Divergence-Free Example: To illustrate, we use the artifical example of a divergence-free field from [13],

$$\boldsymbol{x}_{k+1} = \underbrace{\begin{bmatrix} e^{-ax_1x_2}\left(ax_1\sin(x_1x_2) - x_1\cos(x_1x_2)\right) \\ e^{-ax_1x_2}\left(x_2\cos(x_1x_2) - ax_2\sin(x_1x_2)\right) \end{bmatrix}}_{\boldsymbol{f}} + \boldsymbol{w}_k,$$
$$\boldsymbol{y}_k = \boldsymbol{x}_k + \boldsymbol{e}_k,$$

where $a = 0.01$, $\boldsymbol{w}_k \sim \mathcal{N}(\boldsymbol{0}, 0.1\boldsymbol{I})$, $\boldsymbol{e}_k \sim \mathcal{N}(\boldsymbol{0}, 0.01\boldsymbol{I})$. In this example, $\boldsymbol{f}$ is a divergence-free vector field, that is, $\boldsymbol{f}$ satisfies $\nabla \boldsymbol{f} = \boldsymbol{0}$. The constraint can be written in the form (23) with $\mathcal{F}_{\boldsymbol{x}} = \begin{bmatrix} \frac{\partial}{\partial x_1} & \frac{\partial}{\partial x_2} \end{bmatrix}$. Using Algorithm 1, we first choose a basis-function expansion g with weight prior (5),

$$g(\boldsymbol{x}) = \sum_{j=1}^{M} \gamma_j \phi_j(\boldsymbol{x}), \tag{27}$$

with

$$\phi_j(\boldsymbol{x}) = \prod_{n=1}^{2} \frac{1}{\sqrt{L_n}} \sin\left(\frac{\pi j(x_n + L_n)}{2L_n}\right). \tag{28}$$

Next, we choose $\mathcal{G}_{\boldsymbol{x}} = \begin{bmatrix} \frac{\partial}{\partial x_2} & -\frac{\partial}{\partial x_1} \end{bmatrix}^\top$, which satisfies (26). Finally, we determine $\hat{\boldsymbol{f}}$ from (25),

$$\hat{\boldsymbol{f}} = \sum_{j=1}^{M} \gamma_j \begin{bmatrix} \frac{\partial \phi_j(\boldsymbol{x})}{\partial x_2} & -\frac{\partial \phi_j(\boldsymbol{x})}{\partial x_1} \end{bmatrix}^\top, \tag{29}$$

where

$$\frac{\partial \phi_j(\boldsymbol{x})}{\partial x_2} = \frac{\pi j}{2L_2\sqrt{L_1}} \sin\left(\frac{\pi j(x_1 + L_1)}{2L_1}\right) \cdot \cos\left(\frac{\pi j(x_2 + L_2)}{2L_2}\right) \tag{30}$$

and similarly for $-\frac{\partial \phi_j(\boldsymbol{x})}{\partial x_1}$. The function (29) satisfies the constraint $\nabla \hat{\boldsymbol{f}} = \boldsymbol{0}$ by construction. Note that the resulting basis-function expansion (29) is vector-valued although the prior (27) is not. We have a set of scalar weights γ_j but the constraint function (23) is two-dimensional and the GP-SSM formulation becomes

$$\boldsymbol{x}_{k+1} = \left(\begin{bmatrix} \gamma_1 & \cdots & \gamma_M \end{bmatrix} \otimes \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}\right) \begin{bmatrix} \frac{\partial \phi_1(\boldsymbol{x}_k)}{\partial x_1} \\ \frac{\partial \phi_1(\boldsymbol{x}_k)}{\partial x_2} \\ \vdots \\ \frac{\partial \phi_M(\boldsymbol{x}_k)}{\partial x_1} \\ \frac{\partial \phi_M(\boldsymbol{x}_k)}{\partial x_2} \end{bmatrix} + \boldsymbol{w}_k. \tag{31}$$

To get (31) in the form (8), we subtract the second equation in (31) from the first, that is,

$$x_{1,k+1} - x_{2,k+1} = \sum_{j=1}^{M} \gamma_j \left(\frac{\partial \phi_j(\boldsymbol{x}_k)}{\partial x_1} + \frac{\partial \phi_j(\boldsymbol{x}_k)}{\partial x_2}\right) + \tilde{w}_k, \tag{32}$$

which reduces the GP-SSM to a one-dimensional system, from which the standard method in Sec. II can be used.

For the results, we set $L_1 = L_2 = 8$, we use the squared exponential kernel with $\sigma = 50$, $\ell = 0.1$, choose $M = 15$ implying 225 basis functions, and we use $N = 300$ particles. We use the root mean-square error (RMSE) of the state $\boldsymbol{x}$ and function $\boldsymbol{f}$, respectively, as the performance metrics. The results are based on $K = 200$ Monte-Carlo runs, where the

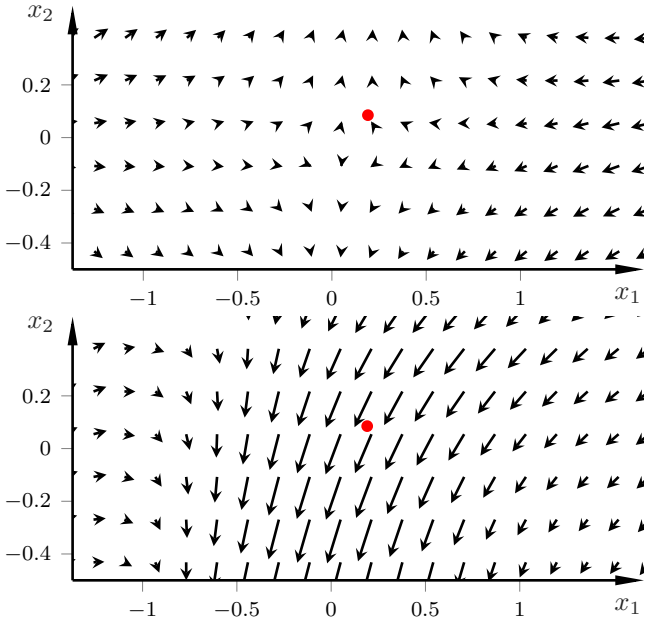

Fig. 5. Reconstructed fields subtracted from the true field for a time step using the basis-function expansion (29) satisfying the constraints (top) and the regular basis-function expansion (3) (bottom). The length of the lines indicate the magnitude of the error. The true state at the current time step is indicated by the red circle.

TABLE I

TME-AVERAGED RMSE FOR 200 MONTE-CARLO RUNS FOR THE DIVERGENCE-FREE EXAMPLE IN SEC. III-D.1.

	ALG. 1	NOMINAL
RMSE(x_1)	0.98	1.77
RMSE(x_2)	0.81	1.72
RMSE(f_1)	0.81	1.87
RMSE(f_2)	0.80	1.86

initial state is randomized over the domain $[-4, 4] \times [-4, 4]$ and each run is $T = 50$ time steps long.

Fig. 5 shows the reconstructed error field $\Delta \boldsymbol{f}_k = \hat{\boldsymbol{f}}_k - \boldsymbol{f}_k$ at time step $k = 45$ for one realization of the learning method using constrained basis functions in (29) (upper plot) and the standard basis-function expansion (3) (lower plot). The results indicate that the reconstructed error is significantly reduced when using the constrained basis-function formulation. Fig. 5 is for a particular time step of a particular realization and does not necessarily mean that constraint satisfaction improves estimation performance at multiple time steps. To verify the impact the constrained basis-function expansion has on estimation, Table I displays the time-averaged RMSE for the state and function estimates, respectively. The results show about 50% reductions in estimation error when using Algorithm 1 for constructing a function approximation that obeys the constraints.

IV. CONCLUSION

In this paper we analyzed our recently developed recursive Bayesian inference and learning method (see [10]) in terms of its ability to learn systems with a priori known constraints. The method is based on PFs and a reduced-rank formulation of GPs. In the formulation, a finite basis-function expansion of the GP is used. We showed that thanks to the basis-function interpretation of a GP, the method can by construction satisfy various constraints on the unknown part of the system. In particular, we exemplified with symmetry, antisymmetry, boundary conditions, and linear operator constraints.

In terms of learning and subsequent estimation accuracy, the ability to incorporate constraint satisfaction into the GP formulation leads to a much improved learning performance and also heavily improved state-estimation performance. For the divergence-free linear operator constraint in Sec. III-D.1, both the state-estimation error and the function error showed a decreased in RMSE with about, and sometimes more than, 50%, In addition, such intrinsic constraint satisfaction means that the method needs fewer particles and/or basis functions, thus reducing computational complexity.

REFERENCES

[1] C. Rasmussen and C. Williams, *Gaussian Processes for Machine Learning*. Cambridge, MA, USA: MIT Press, 2006.

[2] G. Pillonetto and G. De Nicolao, "A new kernel-based approach for linear system identification," *Automatica*, vol. 46, no. 1, pp. 81–93, 2010.

[3] A. McHutchon, "Nonlinear modelling and control using Gaussian processes," phdthesis, Univ. Cambridge, 2014.

[4] R. Frigola-Alcade, "Bayesian time series learning with Gaussian processes," phdthesis, Univ. Cambridge, 2015.

[5] A. Svensson and T. B. Schön, "A flexible state-space model for learning nonlinear dynamical systems," *Automatica*, vol. 80, pp. 189–199, 2017.

[6] R. Frigola, F. Lindsten, T. B. Schön, and C. E. Rasmussen, "Bayesian inference and learning in Gaussian process state-space models with particle MCMC," in *Adv. Neural Information Processing Systems*, Lake Tahoe, NV, Dec. 2013.

[7] F. Tobar, P. M. Djurić, and D. P. Mandic, "Unsupervised state-space modeling using reproducing kernels," *IEEE Trans. Signal Process.*, vol. 63, no. 19, pp. 5210–5221, 2015.

[8] H. Bijl, T. B. Schön, J.-W. van Wingerden, and M. Verhaegen, "System identification through online sparse Gaussian process regression with input noise," *J. Syst. Control*, vol. 2, pp. 1–11, 2017.

[9] Y. Liu and P. M. Djurić, "Gaussian process state-space models with time-varying parameters and inducing points," in *Eur. Signal Process. Conf.*, Amsterdam, The Netherlands, Jan. 2020.

[10] K. Berntorp, "Online Bayesian inference and learning of Gaussian-process state-space models," *Automatica*, vol. 129, p. 109613, 2021.

[11] A. Solin and S. Särkkä, "Hilbert space methods for reduced-rank Gaussian process regression," *Statistics and Computing*, vol. 30, no. 2, pp. 419–446, 2020.

[12] A. Kullberg, I. Skog, and G. Hendeby, "Online joint state inference and learning of partially unknown state-space models," *arXiv e-prints*, 2021.

[13] C. Jidling, N. Wahlström, A. Wills, and T. B. Schön, "Linearly constrained Gaussian processes," in *Int. Conf. Neural Information Process. Syst.*, Long Beach, CA, Dec. 2017.

[14] A. P. Dawid, "Some matrix-variate distribution theory: notational considerations and a Bayesian application," *Biometrika*, vol. 68, no. 1, pp. 265–274, 1981.

[15] E. Özkan, V. Šmídl, S. Saha, C. Lundquist, and F. Gustafsson, "Marginalized adaptive particle filtering for nonlinear models with unknown time-varying noise parameters," *Automatica*, vol. 49, no. 6, pp. 1566–1575, 2013.

[16] K. Berntorp and S. Di Cairano, "Tire-stiffness and vehicle-state estimation based on noise-adaptive particle filtering," *IEEE Trans. Control Syst. Technol.*, vol. 27, no. 3, pp. 1100–1114, 2018.

[17] K. Berntorp, R. Quirynen, T. Uno, and S. Di Cairano, "Trajectory tracking for autonomous vehicles on varying road surfaces by friction-adaptive nonlinear model predictive control," *Veh. Syst. Dyn.*, vol. 58, no. 5, pp. 705–725, 2019.

[18] H. B. Pacejka, *Tire and Vehicle Dynamics*, 2nd ed. Oxford, United Kingdom: Butterworth-Heinemann, 2006.