Inference with Deep Gaussian Process State Space Models

Yuhao Liu,[†] Marzieh Ajirak,[◇] and Petar M. Djurić[◇]
[†]Department of Applied Mathematics and Statistics
[◇] Department of Electrical and Computer Engineering
Stony Brook University, NY 11794

***Abstract*—In this paper, we address the problem of sequential processing of observations modeled by deep Gaussian process state space models. First, we introduce the model where the Gaussian processes are based on random features and where both the transition and observation functions of the models are unknown. Then we propose a method that can estimate the unknowns of the model. The method allows for incremental learning of the system without requiring all the historical information. We also propose an ensemble version of the method, where each member of the ensemble has its own set of features. We show with computer simulations that the method can track the latent states up to scale and rotation.**

I. Introduction

Non-linear state-space models (SSMs) are of importance in science and engineering [3, 13]. In the past decade or so, they have been increasingly studied by using data-driven methods [15]. These methods can learn patterns from data and based on them predict future observations. To avoid overfitting by non-linear SSMs [7], especially for high dimensional systems with a modest amount of observations, Gaussian process state-space models (GPSSMs) have been proposed [1, 21]. In comparison to many modern machine learning methods, their ability to process uncertainties can improve the robustness during the learning processes. For their inference, the variational inference has been put forth [7], including the expectation propagation approach [4]. The expectation-maximization algorithm has also been considered [9], primarily to avoid inaccuracies that result from variational approximation. The authors of [8] have developed various Monte Carlo methods to improve the estimation accuracy. Besides function-space representation of Gaussian processes (GPs), feature-space representations of GPs have also been documented [11]. However, these inference methods all operate in an offline mode.

One way of broadening the function space of a GP is by introducing an ensemble of GPs [17]. Each GP in the ensemble relies on all or a subset of training samples and uses a unique kernel to make predictions. Ensembles of GPs have also been used for combining global approximants with local GPs [20]. In [16], an ensemble of GPs was used for online interactive learning.

There are two types of online GPSSMs in the existing literature. One relies on variational inference [18], and it requires updating all the historical information at each time instant. This in turn incurs large complexity and computational burden. Another approach is based on feature spaces [2], and in its implementation, the number of parameters becomes very large and is the exponential power of the latent dimensions. Further, the approach assumes known hyper-parameters of the kernels.

In this paper, our interest is in deep state-space models. They were proposed for both linear systems [19] and non-linear systems. The inference of these models was based on variational inference [12]. A deep GPSSM was documented in [22] with stochastic differential equation settings and only discussed outputs with one dimension.

In this paper, we propose to overcome the problems of the online setting and the inaccuracies due to variational inference. We introduce novel models, namely online ensemble deep GPSSMs (OED-GPSSMs) based on random features. For these models, we propose a method that allows for incremental learning of the system without requiring all the historical information. We use random features to address the exponential order of parameters, and we take advantage of an ensemble approach to learn the unknown hyper-parameters of the kernels. Our main contribution is the sequential inference with deep GPSSMs, where both the transition and observation functions are unknown. We propose a method that can track the latent states up to scale and rotation.

II. Background

In this section, we provide a very brief background on GPs and sparsity related to GPs.

A. Gaussian processes

A Gaussian Process, written as $\mathcal{GP}\left(m(\cdot), k(\cdot,\cdot|\boldsymbol{\theta})\right)$, is in essence a distribution over functions, where $m(\cdot)$ is a mean function, $k(\cdot,\cdot)$ is a kernel or covariance function, and $\boldsymbol{\theta}$ is the hyper-parameter vector parameterizing the kernel. To simplify the notation, we express a GP as $\mathcal{GP}(m,k)$ or as $\mathcal{GP}(m,k(\boldsymbol{\theta}))$, if $\boldsymbol{\theta}$ is emphasized. For any set of inputs $\mathbf{X} = [\mathbf{x}_j]_{j=1}^J := [\mathbf{x}_1,\ldots,\mathbf{x}_J]^\top$ in the domain of a real-valued function $f \sim \mathcal{GP}(m,k)$, the function values $\mathbf{f} = [f(\mathbf{x}_j)]_{j=1}^J$ are Gaussian distributed, i.e.,

$$p(\mathbf{f}|\mathbf{X}) = \mathcal{N}(\mathbf{f}|\mathbf{m}_\mathbf{X}, \mathbf{K}_{\mathbf{XX}}),$$

The authors thank the support of NSF under Award 2021002.

where $\mathbf{m}_\mathbf{X} = [m(\mathbf{x}_j)]_{j=1}^J$ is the mean and $\mathbf{K}_{\mathbf{XX}} := k(\mathbf{X}, \mathbf{X}|\boldsymbol{\theta}) = [k(\mathbf{x}_i, \mathbf{x}_j)]_{i,j}$. Given the observation $\mathbf{f}$ on $\mathbf{X}$, the predictive distribution of a realization $\mathbf{f}^*$ at new inputs $\mathbf{X}^*$ is given by

$$p(\mathbf{f}^*|\mathbf{X}^*, \mathbf{f}, \mathbf{X}) = \mathcal{N}(\mathbf{f}^*|\boldsymbol{\mu}^*, \boldsymbol{\Sigma}^*),$$

with predictive mean and covariance given by

$$\begin{aligned} \boldsymbol{\mu}^* &= \mathbf{m}_{\mathbf{X}^*} + \mathbf{K}_{\mathbf{X}^*\mathbf{X}}\mathbf{K}_{\mathbf{XX}}^{-1}(\mathbf{f} - \mathbf{m}_\mathbf{X}), \\ \boldsymbol{\Sigma}^* &= \mathbf{K}_{\mathbf{X}^*\mathbf{X}^*} - \mathbf{K}_{\mathbf{X}^*\mathbf{X}}\mathbf{K}_{\mathbf{XX}}^{-1}\mathbf{K}_{\mathbf{X}\mathbf{X}^*}. \end{aligned} \tag{1}$$

B. Sparsity and Gaussian processes

Gaussian processes do not scale up well with N, the number of input-output pairs. We observe that in (1), one has to invert the $N \times N$ matrix $\mathbf{K}_{\mathbf{XX}}$, which for large values of N becomes an issue. To ameliorate the problem, we resort to approximations by exploiting the concept of sparsity. One approach to such approximation is based on constructing GPs with features that come from a feature space [14].

Compared with an approximation in function space, a GP with a shift-invariant kernel has another way of approximation, which focuses on a feature space. The vector of basis functions, also known as random features, are comprised of trigonometric functions that are defined by

$$\boldsymbol{\phi}_\mathbf{v}(\mathbf{x}) = \frac{1}{\sqrt{J}}[\sin(\mathbf{x}^\top\mathbf{v}^1), \cos(\mathbf{x}^\top\mathbf{v}^1), ..., \sin(\mathbf{x}^\top\mathbf{v}^J), \cos(\mathbf{x}^\top\mathbf{v}^J)]^\top,$$

where $\mathbf{v}^{(1:J)} = \{\mathbf{v}^j\}_{j=1}^J$ are vectors sampled from the power spectral density of the kernel. Then the kernel function $k(\mathbf{x}, \mathbf{x}')$ can be approximated by $\boldsymbol{\phi}_\mathbf{v}(\mathbf{x})^\top\boldsymbol{\phi}_\mathbf{v}(\mathbf{x}')$ if the kernel is shift-invariant. It brings a type of GP approximation according to

$$f \approx \boldsymbol{\phi}_\mathbf{v}(\mathbf{x})\boldsymbol{\theta}.$$

III. Deep Gaussian Process State Space Models

In this section, we first describe deep GPSSMs and then propose how to make inferences about all the unknowns of these models.

A standard state space model is formed by

$$\mathbf{x}_t = f(\mathbf{x}_{t-1}) + \boldsymbol{\epsilon}_t, \tag{2}$$

$$\mathbf{y}_t = g(\mathbf{x}_t) + \mathbf{e}_t, \tag{3}$$

where (2) represents the state equation with the latent state $\mathbf{x}_t \in \mathbb{R}^{d_x}$ being the state vector at time t, and (3) the observation equation with the observations $\mathbf{y}_t \in \mathbb{R}^{d_y}$. The symbols $\boldsymbol{\epsilon}_t \sim \mathcal{N}(\mathbf{0}, \sigma_\epsilon^2\mathbf{I})$ and $\mathbf{e}_t \sim \mathcal{N}(\mathbf{0}, \sigma_e^2\mathbf{I})$ are Gaussian distributed errors (noises). If $g(\cdot)$ in (3) has a deep structure as in Fig. 1, we write the SSM using the form of random features as

$$\begin{aligned} \text{Transition:}\ \ \mathbf{x}_{0,t} &= \mathbf{H}^\top\boldsymbol{\phi}_\mathbf{v}(\mathbf{x}_{0,t-1}) + \boldsymbol{\epsilon}_t, \\ \text{Deep:}\ \ \mathbf{x}_{l,t} &= \boldsymbol{\Theta}_{l-1}^\top\boldsymbol{\phi}_\mathbf{v}(\mathbf{x}_{l-1,t}) + \mathbf{e}_{l-1,t}, \\ \text{Observation:}\ \ \mathbf{y}_t &= \boldsymbol{\Theta}_L^\top\boldsymbol{\phi}_\mathbf{v}(\mathbf{x}_{L,t}) + \mathbf{e}_{L,t}, \end{aligned}$$

where $l = 1, \ldots, L$ indexes the layers, $\boldsymbol{\phi}_\mathbf{v}$ represents random features with $\mathbf{v} = \{\mathbf{v}_l^{(1:J)}\}_{l=0}^L$, $\mathbf{x}_{l,t} \in \mathbb{R}^{D_l}$ are the hidden states, $\mathbf{e}_{l,t} \sim \mathcal{N}(\mathbf{0}, \sigma_l^2\mathbf{I})$, $\mathbf{H} = [\boldsymbol{\eta}^1, \boldsymbol{\eta}^2, \ldots, \boldsymbol{\eta}^{d_x}]$, and $\boldsymbol{\Theta}_l = [\boldsymbol{\theta}_l^1, \boldsymbol{\theta}_l^2, \ldots, \boldsymbol{\theta}_l^{D_{l+1}}]$ are parameter variables. Specifically,

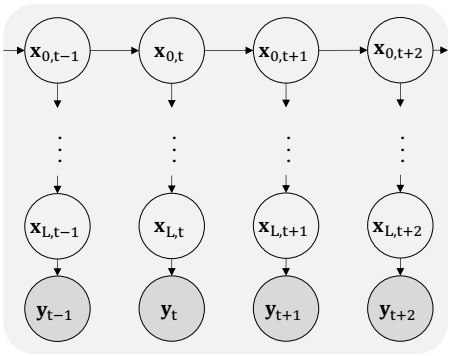

Fig. 1. A generic diagram of a DSS model with L layers.

$d_x = D_0$, $d_y = D_{L+1}$. Also, we denote $\mathbf{y}_t = \mathbf{x}_{L+1,t}$ for convenience. We assume that the parameter variables are all independent, i.e., the columns of $\mathbf{H}$ and $\boldsymbol{\Theta} = \{\boldsymbol{\Theta}_l\}_{l=0}^L$ are independent from the other columns. The independence assumption about the parameter variables implies that the dimensions of $\mathbf{x}_{l,t}$ and $\mathbf{y}_t$ are independent. To do the sequential inference on the distribution of $\mathbf{H}$, $\boldsymbol{\Theta}$, and $\mathbf{x}_{l,t}$, we assign prior distributions $p(\mathbf{H}), p(\boldsymbol{\Theta})$, and $p(\mathbf{x}_{0,0})$ to them and adopt the Bayesian paradigm.

A. Updating the States

Suppose we have obtained the posterior of $\mathbf{x}_{0,t-1}$, $\mathbf{H}$, $\boldsymbol{\Theta}$ at time $t-1$, i.e., $q(\mathbf{x}_{0,t-1}|\mathbf{y}_{1:t-1})$, $q(\mathbf{H}|\mathbf{y}_{1:t-1})$, and $q(\boldsymbol{\Theta}|\mathbf{y}_{1:t-1})$. Then we first define the predictive distribution of $\{\mathbf{x}_{l,t}\}_{l=0}^{L+1}$ from layer 0 to layer $L+1$, We write

$$\begin{aligned} q(\mathbf{x}_{0,t}|\mathbf{y}_{1:t-1}) &= \int p(\mathbf{x}_{0,t}|\mathbf{x}_{0,t-1})q(\mathbf{x}_{0,t-1}|\mathbf{y}_{1:t-1})\mathrm{d}\mathbf{x}_{0,t-1}, \\ q(\mathbf{x}_{l,t}|\mathbf{y}_{1:t-1}) &= \int p(\mathbf{x}_{l,t}|\mathbf{x}_{l-1,t})q(\mathbf{x}_{l-1,t}|\mathbf{y}_{1:t-1})\mathrm{d}\mathbf{x}_{l,t}, \end{aligned} \tag{4}$$

where, we recall, $\mathbf{x}_{L+1,t} = \mathbf{y}_t$, $l = 1, \ldots, L$, and

$$\begin{aligned} p(\mathbf{x}_{0,t}|\mathbf{x}_{0,t-1}, \mathbf{y}_{1:t-1}) &= \int p(\mathbf{x}_{0,t}|\mathbf{x}_{0,t-1}, \mathbf{H}) \\ &\quad \times q(\mathbf{H}|\mathbf{x}_{0,t-1}, \mathbf{y}_{1:t-1})d\mathbf{H}, \end{aligned} \tag{5}$$

$$\begin{aligned} p(\mathbf{x}_{l,t}|\mathbf{x}_{l-1,t}, \mathbf{y}_{1:t-1}) &= \int p(\mathbf{x}_{l,t}|\mathbf{x}_{l-1,t}, \boldsymbol{\Theta}_{l-1}) \\ &\quad \times q(\boldsymbol{\Theta}_{l-1}|\mathbf{x}_{l-1,t}, \mathbf{y}_{1:t-1})d\boldsymbol{\Theta}_{l-1}. \end{aligned} \tag{6}$$

Then we update the posterior of $\{\mathbf{x}_{l,t}\}_{l=0}^L$ from layer L to layer 0 by

$$\begin{aligned} q(\mathbf{x}_{l,t}|\mathbf{y}_{1:t}) &\propto \int p(\mathbf{x}_{l+1,t}|\mathbf{x}_{l,t})q(\mathbf{x}_{l,t}|\mathbf{y}_{1:t-1})q(\mathbf{x}_{l+1,t}|\mathbf{y}_{1:t})\mathrm{d}\mathbf{x}_{l+1,t}, \\ &\approx p(\widehat{\mathbf{x}}_{l+1,t}|\mathbf{x}_{l,t})q(\mathbf{x}_{l,t}|\mathbf{y}_{1:t-1}), \end{aligned} \tag{7}$$

where $l = L, \ldots, 0$, and $\widehat{\mathbf{x}}_{l+1,t}$ is an estimate of $\mathbf{x}_{l+1,t}$ obtained as explained below. In particular, if $q(\mathbf{H}|\mathbf{x}_{0,t-1}, \mathbf{y}_{1:t-1})$ and $q(\boldsymbol{\Theta}_{l-1}|\mathbf{x}_{l-1,t}, \mathbf{y}_{1:t-1})$ are Gaussian distributions, $p(\mathbf{x}_{l,t}|\mathbf{x}_{l-1,t}, \mathbf{y}_{1:t-1})$ is also Gaussian. However, the posterior $q(\mathbf{x}_{l,t}|\mathbf{y}_{1:t-1})$ is not Gaussian because of the nonlinearity of $\boldsymbol{\phi}_\mathbf{v}$. This leads to the use of particle filtering [6].

B. Generation of particles

Suppose we have sampled M particles $\mathbf{x}_{0,t-1}^{(m)}$ from $q(\mathbf{x}_{0,t-1}|\mathbf{y}_{1:t-1})$ at time $t-1$, then we generate particles $\mathbf{x}_{l,t}^{(m)}$ by (4), i.e.,

$$\mathbf{x}_{0,t}^{(m)} \sim p(\mathbf{x}_{0,t}|\mathbf{x}_{0,t-1}^{(m)}), \tag{8}$$

$$\mathbf{x}_{l,t}^{(m)} \sim p(\mathbf{x}_{l,t}|\mathbf{x}_{l-1,t}^{(m)}), \tag{9}$$

where $l = 1, \ldots, L$.

C. Estimation of the predictive PDF

After the transition step, we have received M samples $\mathbf{x}_{L,t}^{(m)}$ at time t. The predictive pdf of $\mathbf{y}_t$ is then given by

$$p(\mathbf{y}_t|\mathbf{y}_{1:t-1}) = \frac{1}{M}\sum_{m=1}^{M} p(\mathbf{y}_t|\mathbf{x}_{L,t}^{(m)}). \tag{10}$$

D. Estimation of the filtering PDFs

Upon receiving $\mathbf{y}_t$, we assign the weights for each particle $\mathbf{x}_{l,t}^{(m)}$ by the likelihood of $\mathbf{x}_{l,t}^{(m)}$ layer by layer, where $l = L, \ldots, 0$ and the weights are given by

$$w_{l,t}^{(m)} \propto p(\widehat{\mathbf{x}}_{l+1,t}|\mathbf{x}_{l,t}^{(m)}). \tag{11}$$

After normalizing the weights, the minimum mean square estimate (MMSE) of $\mathbf{x}_{l,t}$ is obtained by

$$\widehat{\mathbf{x}}_{l,t} = \sum_{m=1}^{M} w_{l,t}^{(m)}\mathbf{x}_{l,t}^{(m)}. \tag{12}$$

The approximation of the posterior of $q(\mathbf{x}_{l,t}|\mathbf{y}_{1:t})$ in (7) is

$$q^M(\mathbf{x}_{l,t}|\mathbf{y}_{1:t}) = \sum_{m=1}^{M} w_{l,t}^{(m)}\delta(\mathbf{x}_{l,t} - \mathbf{x}_{l,t}^{(m)}).$$

E. Updating the parameters

Given the derived posterior and point estimate of $\mathbf{x}_{l,t}$, we proceed with updating the posterior of the parameter variables $\mathbf{H}$ and $\mathbf{\Theta}$. We have,

$$\begin{aligned} q(\mathbf{\Theta}_l|\mathbf{y}_{1:t}) &= \int q(\mathbf{\Theta}_l|\mathbf{x}_{l,t}, \mathbf{x}_{l+1,t})q(\mathbf{x}_{l,t}|\mathbf{y}_{1:t})q(\mathbf{x}_{l+1,t}|\mathbf{y}_{1:t})\mathrm{d}\mathbf{x}_{l:l+1,t} \\ &\approx q(\mathbf{\Theta}|\mathbf{y}_{1:t-1})p(\widehat{\mathbf{x}}_{l+1,t}|\widehat{\mathbf{x}}_{l,t}). \end{aligned} \tag{13}$$

The posterior for $\mathbf{H}$ is similarly given by

$$\begin{aligned} q(\mathbf{H}|\mathbf{y}_{1:t}) &= \int q(\mathbf{H}|\mathbf{x}_{0,t}, \mathbf{x}_{0,t-1})q(\mathbf{x}_{0,t}|\mathbf{y}_{1:t})q(\mathbf{x}_{0,t-1}|\mathbf{y}_{1:t})\mathrm{d}\mathbf{x}_{0,t-1:t} \\ &\approx q(\mathbf{H}|\mathbf{y}_{1:t-1})p(\widehat{\mathbf{x}}_{0,t}|\widehat{\mathbf{x}}_{0,t-1}), \end{aligned} \tag{14}$$

Therefore, equations (13) and (14) exploit the conjugate property of the Gaussian distribution, and the posterior of $\mathbf{\Theta}$ and $\mathbf{H}$ are always Gaussian distributions. The details of implementations are as follows: Suppose we have estimated $\widehat{\mathbf{x}}_t$ and $\widehat{\mathbf{x}}_{t-1}$. Upon receiving $\mathbf{y}_t$, we update $\boldsymbol{\eta}^i$ and $\boldsymbol{\theta}_l^j$ by the Bayesian rule. Let $\boldsymbol{\eta}^i \sim \mathbf{N}(\boldsymbol{\mu}_\eta^i, \mathbf{\Sigma}_\eta^i)$ and $\boldsymbol{\theta}_l^j \sim \mathbf{N}(\boldsymbol{\mu}_{\theta,l}^j, \mathbf{\Sigma}_{\theta,l}^j)$. The Bayesian formula provides a linear update for $\boldsymbol{\mu}_{\eta,t}^i$ and $\mathbf{\Sigma}_{\eta,t}^i$ by

$$\begin{aligned} \boldsymbol{\mu}_{\eta,t}^i &= \boldsymbol{\mu}_{\eta,t-1}^i + \frac{\mathbf{\Sigma}_{\eta,t-1}^i\boldsymbol{\phi}_{\mathbf{v}}(\widehat{\mathbf{x}}_{0,t-1})(\widehat{\mathbf{x}}_{0,t}^i - \boldsymbol{\phi}_{\mathbf{v}}^\top(\widehat{\mathbf{x}}_{0,t-1})\boldsymbol{\mu}_{\eta,t-1}^i)}{\boldsymbol{\phi}_{\mathbf{v}}^\top(\widehat{\mathbf{x}}_{0,t-1})\mathbf{\Sigma}_{\eta,t-1}^i\boldsymbol{\phi}_{\mathbf{v}}(\widehat{\mathbf{x}}_{0,t-1}) + \sigma_\epsilon^2}, \\ \mathbf{\Sigma}_{\eta,t}^i &= \mathbf{\Sigma}_{\eta,t-1}^i - \frac{\mathbf{\Sigma}_{\eta,t-1}^i\boldsymbol{\phi}_{\mathbf{v}}(\widehat{\mathbf{x}}_{0,t-1})\boldsymbol{\phi}_{\mathbf{v}}^\top(\widehat{\mathbf{x}}_{0,t-1})\mathbf{\Sigma}_{\eta,t-1}^i}{\boldsymbol{\phi}_{\mathbf{v}}^\top(\widehat{\mathbf{x}}_{0,t-1})\mathbf{\Sigma}_{\eta,t-1}^i\boldsymbol{\phi}_{\mathbf{v}}(\widehat{\mathbf{x}}_{0,t-1}) + \sigma_\epsilon^2}, \end{aligned} \tag{15}$$

and for $\boldsymbol{\mu}_\theta^j$ and $\mathbf{\Sigma}_\theta^j$ according to

$$\begin{aligned} \boldsymbol{\mu}_{\theta,l,t}^j &= \boldsymbol{\mu}_{\theta,l,t-1}^j + \frac{\mathbf{\Sigma}_{\theta,l,t-1}^j\boldsymbol{\phi}_{\mathbf{v}}(\widehat{\mathbf{x}}_{l,t})(\widehat{\mathbf{x}}_{l+1,t} - \boldsymbol{\phi}_{\mathbf{v}}^\top(\widehat{\mathbf{x}}_{l,t})\boldsymbol{\mu}_{\theta,l,t-1}^j)}{\boldsymbol{\phi}_{\mathbf{v}}^\top(\widehat{\mathbf{x}}_{l,t})\mathbf{\Sigma}_{\theta,l,t-1}^j\boldsymbol{\phi}_{\mathbf{v}}(\widehat{\mathbf{x}}_{l,t}) + \sigma_l^2}, \\ \mathbf{\Sigma}_{\theta,l,t}^j &= \mathbf{\Sigma}_{\theta,l,t-1}^j - \frac{\mathbf{\Sigma}_{\theta,l,t-1}^j\boldsymbol{\phi}_{\mathbf{v}}(\widehat{\mathbf{x}}_{l,t})\boldsymbol{\phi}_{\mathbf{v}}^\top(\widehat{\mathbf{x}}_{l,t})\mathbf{\Sigma}_{\theta,l,t-1}^j}{\boldsymbol{\phi}_{\mathbf{v}}^\top(\widehat{\mathbf{x}}_{l,t})\mathbf{\Sigma}_{\theta,l,t-1}^j\boldsymbol{\phi}_{\mathbf{v}}(\widehat{\mathbf{x}}_{l,t}) + \sigma_l^2}. \end{aligned} \tag{16}$$

F. Smoothing

We adopt the backward smoothing [5], which assigns the weights for particles $\mathbf{x}_{0,t-1}^{(m)}$ at time $t-1$ after updating the states $\widehat{\mathbf{x}}_{0,t}$ and the parameters $\mathbf{H}, \mathbf{\Theta}$ at time t. The weights are proportional to the transition likelihood, i.e.,

$$\begin{aligned} w_{0,t-1}^{(m)} &\propto p(\widehat{\mathbf{x}}_{0,t}|\mathbf{x}_{0,t-1}^{(m)}) \\ &\approx \int p(\widehat{\mathbf{x}}_{0,t}|\mathbf{x}_{0,t-1}^{(m)}, \mathbf{H})q(\mathbf{H}|\mathbf{y}_{1:t})\mathrm{d}\mathbf{H}. \end{aligned}$$

As a consequence, the MMSE of $\mathbf{x}_{0,t-1}$ is

$$\widehat{\mathbf{x}}_{0,t-1} = \sum_{m=1}^{M} w_{0,t-1}^{(m)}\mathbf{x}_{0,t-1}^{(m)}. \tag{17}$$

The procedure is summarized in Algorithm 1.

Algorithm 1: Single Sequential GPSSM

```
for m = 1 to M do
    Sample x_{0,0}^{(m)} ~ p(x_{0,0});
    Initialize the weight of x_{0,0}^{(m)} as x_{0,0}^{(m)} = 1/M;
for t = 1 to T do
    for i = 1 to number of iterations do
        Update States:
        Sample x_{l,t}^{(m)} from (8) and (9);
        Predict y_t via (10);
        Assign weights to x_{l,t}^{(m)} by (11);
        Estimate the expectation x̂_{l,t} by (12);
        Update Parameters:
        Update H and Θ via (15), (16);
    Resample x_{l,t}^{(m)} based on their weights;
    Smoothing:
    Update x̂_{0,1:t-1} sequentially by (17).
```

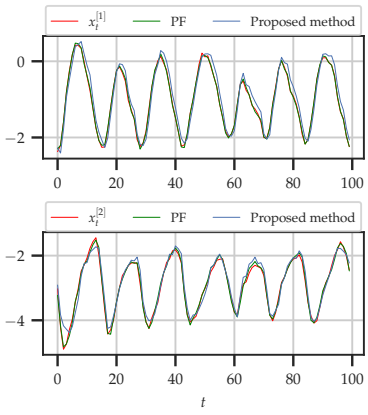

Fig. 2. Results of the state space model described by (20) and obtained by a particle filter and by the proposed method.

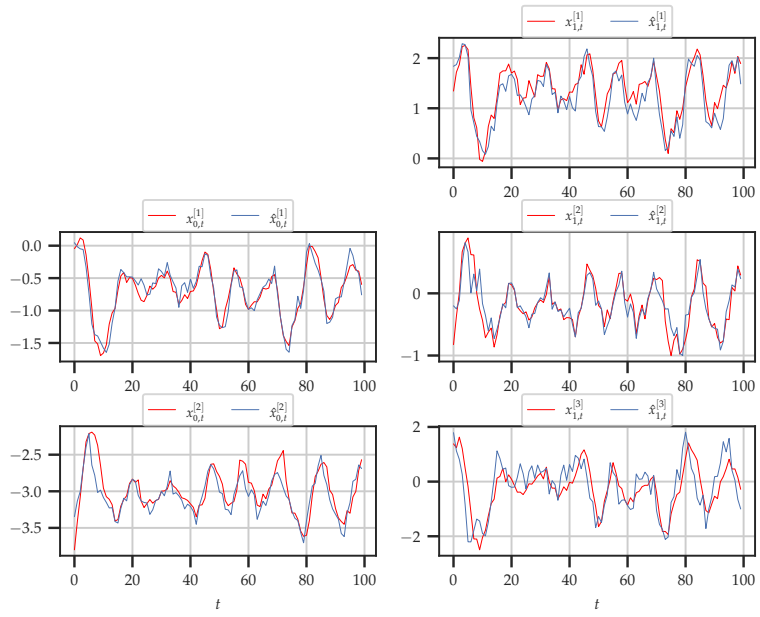

Fig. 3. Results of the DSS model described by (21). On left are the true values and the estimates of the root processes, and on the right, the true values and the estimates of the proposed method.

IV. Ensemble Learning

Use of only a single set of $\mathbf{v} = \{\mathbf{v}_l^{(1:J)}\}_{l=0}^{L}$ might produce significant bias. In order to mitigate this problem, we introduce an ensemble of different sets of $\mathbf{v}$. Denote $\mathbf{v}^s$ as the sth set of pre-selected parameters $\mathbf{v}$ sampled from PSD and its posterior contribution or weight as $w_t^s = p(s|\mathbf{y}_{1:t}, \widehat{\mathbf{x}}_{1:t})$ at time t, where $\widehat{\mathbf{x}}_{1:t} = \widehat{\mathbf{x}}_{0:L,1:t}$. Consequently, the prediction at time t is derived from the total probability theorem, that is,

$$\begin{aligned} & p(\mathbf{y}_t|\mathbf{y}_{1:t-1}, \widehat{\mathbf{x}}_{1:t-1}) \\ &= \sum_{s=1}^{S} p(s|\mathbf{y}_{1:t-1}, \widehat{\mathbf{x}}_{1:t-1}) p(\mathbf{y}_t|s, \mathbf{y}_{1:t-1}, \widehat{\mathbf{x}}_{1:t-1}) \\ &= \sum_{s=1}^{S} w_{t-1}^s p(\mathbf{y}_t|s, \mathbf{y}_{1:t-1}, \widehat{\mathbf{x}}_{1:t-1}), \end{aligned} \tag{18}$$

and the posterior weight is updated by

$$\begin{aligned} w_t^s &= p(s|\mathbf{y}_{1:t}, \widehat{\mathbf{x}}_{1:t}) \\ &= \frac{p(s|\mathbf{y}_{1:t-1}, \widehat{\mathbf{x}}_{1:t-1}) p(\mathbf{y}_t|s, \mathbf{y}_{1:t-1}, \widehat{\mathbf{x}}_{1:t-1})}{p(\mathbf{y}_t|\mathbf{y}_{1:t-1}, \widehat{\mathbf{x}}_{1:t-1})} \\ &\propto w_{t-1}^s p(\mathbf{y}_t|s, \mathbf{y}_{1:t-1}, \widehat{\mathbf{x}}_{1:t-1}). \end{aligned} \tag{19}$$

We note that the estimation of the latent states by the random feature-based methods are identifiable up to scale, shift, and rotation [10]. To this end, we arbitrarily fix the rotation of X by taking the singular value decomposition of the MMSE estimate, $\widehat{\mathbf{X}} = \mathbf{USV}^\top$, and setting $\mathbf{X}$ to be the columns of the left singular vectors with the largest singular values. The inference of the hyper-parameters σ_l^2 and σ_ϵ^2 is based on the standard stochastic gradient descent.

V. Numerical Results

A. Estimation of a state space model

We tested the proposed methodology on the following state space model:

$$\begin{aligned} x_t^{[1]} &= 0.9x_{t-1}^{[1]} + 0.5\sin(x_{t-1}^{[2]}) + u_t^{[1]}, \\ x_t^{[2]} &= 0.1(x_{t-1}^{[1]})^3 - 0.9x_{t-1}^{[2]} + u_t^{[2]}, \\ y_t^{[1]} &= 1.8\cos(x_t^{[1]}) - 0.7\sin(x_t^{[2]}) + v_t^{[1]}, \\ y_t^{[2]} &= 0.5x_t^{[1]} - 1.3\sin(x_t^{[1]}) + v_t^{[2]}, \\ y_t^{[3]} &= 2.0x_t^{[1]} - 0.4x_{2,t}^{[2]} + v_t^{[3]}, \\ y_t^{[4]} &= 0.05\left(x_t^{[1]}\right)^3 + v_t^{[4]}, \\ y_t^{[5]} &= x_t^{[2]}/(1 + (x_t^{[2]})^2) + v_t^{[5]}, \end{aligned} \tag{20}$$

which is a model with two-dimensional state process, and a five-dimensional observed time signal (thus, $d_x = 2$ and $d_y = 5$). The noises were zero-mean Gaussian with variances equal to 0.01. We sampled the random features from the power spectral density of a kernel with radial basis functions and with all of its hyperparameters set to one. Since the dimension of the state space was 2, we sampled $R = 50$ two-dimensional features. As a benchmark for comparison, we applied a particle filter which was designed for the exact functions in the state and observation equations. In implementing the particle filter, we used $M = 100$ particles.

Figure 2 shows the true values of the state $x_t^{[1]}$ and $x_t^{[2]}$ over time. The figures also provide the obtained estimates by both the particle filter and our method. Remarkably, the performance of our method is very close to that of the particle

filter even though our method did not use knowledge about the functions in the state space model.

B. Estimation of processes of a DSS model

We generated data from a DSS model with two hidden layers, the deepest hidden layer denoted as x_0 with $d_{x_0} = 2$, the middle hidden layer x_1 with $d_{x_1} = 3$, and $d_y = 4$. The model is given by

$$\begin{aligned}
\text{Layer } 0: \quad & x_{0,t}^{[1]} = 0.9x_{0,t-1}^{[1]} + 0.5\sin(x_{0,t-1}^{[1]}) + u_{2,t}^{[1]}, \\
& x_{0,t}^{[2]} = 0.5\sin(x_{0,t-1}^{[1]}) + 0.9x_{0,t-1}^{[2]}] + u_{2,t}^{[1]}, \\
\text{Layer } 1: \quad & x_{1,t}^{[1]} = 1.8\cos(x_{0,t}^{[1]}) - 0.7\sin(x_{0,t}^{[1]}) + u_{1,t}^{[1]}, \\
& x_{1,t}^{[2]} = 0.5x_{0,t}^{[1]} - 1.3\sin(x_{0,t}^{[2]}) + u_{1,t}^{[1]} \\
& x_{1,t}^{[3]} = 2x_{0,t}^{[1]} - 0.4x_{0,t}^{[2]} + u_{1,t}^{[2]} \\
\text{Observations}: \quad & y_t^{[1]} = 0.01(x_{1,t}^{[1]})^2 + 1.2x_{1,t}^{[3]} + v_t^{[1]}, \\
& y_t^{[2]} = 1.2\sin(x_{1,t}^{[1]}) - 0.5x_{1,t}^{[2]} + 0.7x_{1,t}^{[3]} + v_t^{[2]}, \\
& y_t^{[3]} = x_{1,t}^{[1]}x_{1,t}^{[2]} + v_t^{[3]}, \\
& y_t^{[4]} = 5x_{1,t}^{[2]}/(1 + x_{1,t}^{[2]^2}) + v_t^{[4]},
\end{aligned} \tag{21}$$

and otherwise, the same parameters were used as in the first experiment. The results are shown in Fig 3. They clearly show that the proposed method is capable of accurately estimating all the latent processes.

VI. Summary

In this paper, we addressed the estimation of the unknowns of a deep SSM using GPs modeled by random features. We presented an algorithm that relies on a two-stage procedure where at each time instant we first estimate the hidden states and then we update the parameters of the GPs. We presented two examples, and they demonstrate that the proposed method can track the estimated processes accurately.

References

[1] T. Beckers and S. Hirche. Stability of Gaussian process state space models. In *2016 European Control Conference (ECC)*, pages 2275–2281. IEEE, 2016.

[2] K. Berntorp. Online Bayesian inference and learning of Gaussian-process state–space models. *Automatica*, 129:109613, 2021.

[3] S. A. Billings. *Nonlinear System Identification: NARMAX Methods in the Time, Frequency, and Spatio-Temporal Domains*. John Wiley & Sons, 2013.

[4] T. D. Bui, J. Yan, and R. E. Turner. A unifying framework for Gaussian process pseudo-point approximations using power expectation propagation. *The Journal of Machine Learning Research*, 18(1):3649–3720, 2017.

[5] C. M. Carvalho, M. S. Johannes, H. F. Lopes, and N. G. Polson. Particle learning and smoothing. *Statistical Science*, 25(1):88–106, 2010.

[6] P. M. Djurić, J. H. Kotecha, J. Zhang, Y. Huang, T. Ghirmai, M. F. Bugallo, and J. Miguez. Particle filtering. *IEEE Signal Processing Magazine*, 20(5):19–38, 2003.

[7] R. Frigola, Y. Chen, and C. E. Rasmussen. Variational Gaussian process state-space models. *Advances in Neural Information Processing Systems*, 27, 2014.

[8] R. Frigola, F. Lindsten, T. B. Schön, and C. E. Rasmussen. Bayesian inference and learning in Gaussian process state-space models with particle mcmc. *Advances in Neural Information Processing Systems*, 26, 2013.

[9] R. Frigola, F. Lindsten, T. B. Schön, and C. E. Rasmussen. Identification of Gaussian process state-space models with particle stochastic approximation EM. *IFAC Proceedings Volumes*, 47(3):4097–4102, 2014.

[10] G. Gundersen, M. Zhang, and B. Engelhardt. Latent variable modeling with random features. In *International Conference on Artificial Intelligence and Statistics*, pages 1333–1341. PMLR, 2021.

[11] R. Herbrich, N. Lawrence, and M. Seeger. Fast sparse Gaussian process methods: The informative vector machine. *Advances in Neural Information Processing Systems*, 15, 2002.

[12] M. Karl, M. Soelch, J. Bayer, and P. Van der Smagt. Deep variational Bayes filters: Unsupervised learning of state space models from raw data. *arXiv preprint arXiv:1605.06432*, 2016.

[13] J. Kocijan. *Modelling and Control of Dynamic Systems Using Gaussian Process Models*. Springer, 2016.

[14] M. Lázaro-Gredilla, J. Quinonero-Candela, C. E. Rasmussen, and A. R. Figueiras-Vidal. Sparse spectrum Gaussian process regression. *The Journal of Machine Learning Research*, 11:1865–1881, 2010.

[15] L. Ljung. System identification: Theory for the user. *PTR Prentice Hall, Upper Saddle River, NJ*, 1999.

[16] Q. Lu, G. Karanikolas, Y. Shen, and G. B. Giannakis. Ensemble Gaussian processes with spectral features for online interactive learning with scalability. In *International Conference on Artificial Intelligence and Statistics*, pages 1910–1920, 2020.

[17] D. Nguyen-Tuong, M. Seeger, and J. Peters. Model learning with local Gaussian process regression. *Advanced Robotics*, 23(15):2015–2034, 2009.

[18] S.-S. Park, Y.-J. Park, Y. Min, and H.-L. Choi. Online Gaussian process state-space model: Learning and planning for partially observable dynamical systems. *arXiv preprint arXiv:1903.08643*, 2019.

[19] S. S. Rangapuram, M. W. Seeger, J. Gasthaus, L. Stella, Y. Wang, and T. Januschowski. Deep state space models for time series forecasting. *Advances in Neural Information Processing Systems*, 31, 2018.

[20] E. Snelson and Z. Ghahramani. Local and global sparse Gaussian process approximations. In *Artificial Intelligence and Statistics*, pages 524–531. PMLR, 2007.

[21] C. K. Williams and C. E. Rasmussen. *Gaussian Processes for Machine Learning*, volume 2. MIT Press Cambridge, MA, 2006.

[22] Z. Zhao, M. Emzir, and S. Särkkä. Deep state-space Gaussian processes. *Statistics and Computing*, 31(6):1–26, 2021.