State-Space Inference and Learning with Gaussian Processes

Anonymous Authors
Anonymous Institution

Abstract

State-space inference and learning with Gaussian processes (GPs) is an unsolved problem. We propose a new, general methodology for inference and learning in nonlinear state-space models that are described probabilistically by non-parametric GP models. We apply the Expectation Maximization algorithm to iterate between inference in the latent state space and learning the parameters of the underlying GP dynamics model.

Inference (filtering and smoothing) in linear dynamical systems (LDS) and nonlinear dynamical systems (NLDS) is frequently used in many applications including signal processing, state estimation, control, and finance/econometric models. Inference aims to estimate the state of a system from a stream of noisy measurements. Imagine tracking the location of a car based on odometer and a GPS sensors, both of which are noisy. Sequential measurements from both sensors are combined to overcome the noise in the system and to obtain an accurate estimate of the system state. Even when the full state is only partially measured, it can still be inferred; in the car example the engine temperature is unobserved, but can be inferred via the nonlinear relationship from acceleration. To exploit this relationship appropriately, inference techniques in nonlinear models are required; they play an important role in many practical applications.

LDS and NLDS belong to a class of models known as state-space models. A state-space model assumes that there exists a time sequence of latent states $\mathbf{x}_t$ that change with time according to a Markovian process specified by a transition function f. The latent states are observed indirectly in $\mathbf{y}_t$ through a measurement function g. We consider state-space models given by

$$\begin{aligned} \mathbf{x}_t &= f(\mathbf{x}_{t-1}) + \boldsymbol{\epsilon}_{t-1}\,, \quad \mathbf{x}_t \in \mathbb{R}^M\,, \\ \mathbf{y}_t &= g(\mathbf{x}_t) + \boldsymbol{\nu}_t\,, \quad \mathbf{y}_t \in \mathbb{R}^D\,. \end{aligned} \tag{1}$$

Here, the system noise $\boldsymbol{\epsilon}_{t-1} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_\epsilon)$ and the measurement noise $\boldsymbol{\nu}_t \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_\nu)$ are both Gaussian. In the LDS case, f and g are linear functions, whereas the NLDS covers the general nonlinear case.

The goal in *inference* is to guess about the latent states $\mathbf{x}$ using measurements $\mathbf{y}$ only. Bayesian inference of the hidden states (smoothing) in LDS with additive Gaussian noise can be done exactly via Kalman smoothing, see for example [14]. In *learning*, the goal is to infer both f and g from observations $\mathbf{y}_t$.

Linear dynamical systems can only model a limited set of phenomena. As a result there has been increasing interest in studying NLDS for the last few decades. Since exact inference and (parameter) learning in NLDS is generally analytically intractable, approximate methods need to be employed.

Common examples for approximate inference in nonlinear dynamical systems, include the Extended Kalman Filter (EKF) [10], the Unscented Kalman Filter (UKF) [7], and the Assumed Density Filter (ADF) [1, 12]. General forms of the dynamics model for inference and learning were proposed in terms of Radial Basis Functions [5] and neural networks [6]. In the context of modeling human motion, GPs have been used in [18, 9] for inference. Recently, GPs were used for filtering in the context of the UKF, the EKF [8], and the ADF [2].

For nonlinear systems these methods encounter problems: The local linearizations of the EKF and the UKF can lead to overfitting. Furthermore, learning using the UKF and the EKF typically requires a parametric form of the dynamics and measurement functions to be specified in advance. Neural network [6] and RBF [5] approaches have a constant level of uncertainty in the dynamics and measurement functions, which means they do not appropriately quantify uncertainty in f and g. Although probabilistic GPs are used in [18, 9], the MAP estimation (point estimate)

Preliminary work. Under review by AISTATS 2010. Do not distribute.

of the latent states can lead to overconfident predictions because the uncertainty in the latent states is not accounted for. Other GP approaches proposed solely for filtering [2, 8] do take the state uncertainty into account, but require ground-truth observations of the latent states during training, typically a strong assumption in many applications.

In this paper, we address the shortcomings of the methods above by proposing the GPIL algorithm for inference and learning in NLDS. Our flexible framework uses non-parametric GPs to model both the transition function and the measurement function. The GPs naturally account for three kinds of uncertainties in the real dynamical system: system noise, measurement noise, and model uncertainty. The model uncertainty varies depending on the density of the training data. Our model integrates out the latent states unlike [18, 9], where a MAP approximation to the latent states is used. The non-parametric nature of our model does not require a pre-specified parametric form of the measurement and/or dynamics model. At the same time, it does not require any ground truth observations of the latent states $\mathbf{x}$. We propose to learn parameterized GPs for the dynamics and measurement functions using Expectation Maximization (EM) [3].

The main contributions of this paper are twofold:

- We propose a tractable algorithm for approximate inference (smoothing) in GP state-space models.
- Using GP models for f and g, see eq. (1), we propose learning without the need of ground-truth observations $\mathbf{x}_i$ of the latent states.

1 Model and Algorithmic Setup

We consider an NLDS, where the stochastic Markovian transition dynamics of the hidden states and the corresponding measurement function are given by eq. (1). The transition function f and the measurement function g, eq. (1), are both unknown. In order to make predictions, we have to *learn* the transition function f and measurement function g solely given the information of T sequential observations $\mathbf{Y} = [\mathbf{y}_1, \dots, \mathbf{y}_T]$.

We use GPs to model both the unknown transition function f and the unknown measurement function g, and write $f \sim \mathcal{GP}_f$, $g \sim \mathcal{GP}_g$, respectively. A GP is a distribution over functions and is specified by a mean function and a covariance function, also called a *kernel*, [15]. Throughout this paper, we use the squared exponential kernel and a prior mean function that equals zero everywhere. A separate GP is used for each target dimension of f and g.

Since the latent states $\mathbf{x}$ are unobserved, we cannot learn $\mathcal{GP}_f$ and $\mathcal{GP}_g$ directly; instead, we apply the EM algorithm to learn their free parameters. Let us have a look at the "parameters" of a GP. In a standard GP setup, the GP can be considered effectively parameterized by the hyper-parameters, the training inputs, and the training targets. In the considered state-space model, eq. (1), the training inputs can never be observed directly. Therefore, in this paper, we explicitly specify these parameters and parameterize a GP by a *pseudo training set*, which is considered a set of free parameters for learning. These parameters are related to the pseudo training inputs used for sparse GP approximations [16]. The pseudo inputs that parameterize the transition function f are denoted by $\boldsymbol{\alpha} = \{\boldsymbol{\alpha}_i \in \mathbb{R}^M\}_{i=1}^N$ and the corresponding pseudo targets are denoted by $\boldsymbol{\beta} = \{\boldsymbol{\beta}_i \in \mathbb{R}^M\}_{i=1}^N$. Intuitively, the pseudo inputs $\boldsymbol{\alpha}_i$ can be understood as the locations of the means of the Gaussian basis functions (SE kernel), whereas the pseudo targets $\boldsymbol{\beta}_i$ are related to the function value at this location. We interpret the pseudo training set as N pairs of independent observations of transitions from $\mathbf{x}_{t-1}$ to $\mathbf{x}_t$. Note that the pseudo training set does *not* form a time series. To parameterize the measurement function g, we follow the same approach and use the pseudo inputs $\boldsymbol{\xi} = \{\boldsymbol{\xi}_i \in \mathbb{R}^M\}_{i=1}^N$ and pseudo outputs $\boldsymbol{\upsilon} = \{\boldsymbol{\upsilon}_i \in \mathbb{R}^D\}_{i=1}^N$.

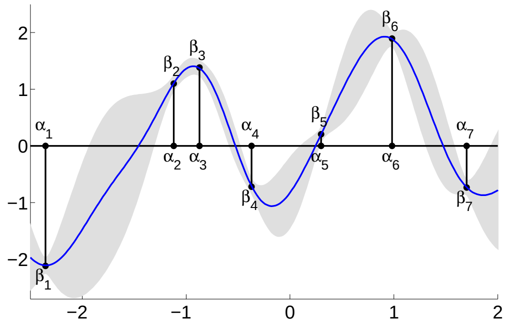

Figure 1: An example of a function predicted by a set of support points. The $\boldsymbol{\alpha}_i$ are the pseudo training inputs, while the $\boldsymbol{\beta}_i$ are the pseudo training targets. The shaded area represents the 95% confidence region around the expected function value (blue, solid).

The pseudo training sets are learned jointly with the kernel hyper-parameters. Unlike the sparse GP model in [16], we do not integrate the pseudo training targets out, but optimize them instead since integration is analytically intractable in our model. An example of a pseudo training set and the corresponding predictive function is shown in Fig. 1. The corresponding graphical model of our model is shown in Fig. 2.

Once the pseudo training set, $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, is deter-

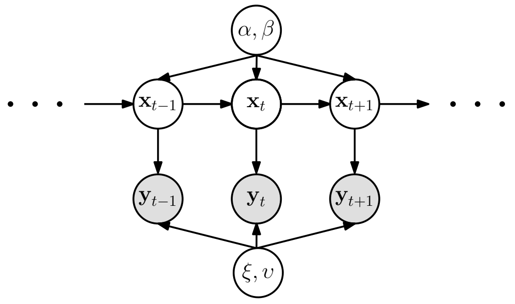

Figure 2: The model employed. The parameter sets $\boldsymbol{\alpha}, \boldsymbol{\beta}$ and $\boldsymbol{\xi}, \boldsymbol{\upsilon}$ serve as a pseudo training set for $\mathcal{GP}_f$ and $\mathcal{GP}_g$, respectively. $\mathcal{GP}_f$ and $\mathcal{GP}_g$ are *not* full GPs, but rather sparse GPs that impose the condition $\mathbf{x}_{t+1} \perp \mathbf{x}_{t-1} | \mathbf{x}_t, \boldsymbol{\alpha}, \boldsymbol{\beta}$.

mined, *predicting* $\mathbf{x}_t$ from $\mathbf{x}_{t-1}$ is a GP prediction using $\mathcal{GP}_f$ [2, 13]. Here, $\mathbf{x}_{t-1}$ serves as the (uncertain) test input, while $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are used as a standard GP training set. Likewise, predicting $\mathbf{y}_t$ from $\mathbf{x}_t$ corresponds to a GP prediction (with uncertain inputs) with $\mathbf{x}_t$ as the test input and a training set defined through $\boldsymbol{\xi}$ and $\boldsymbol{\upsilon}$. The model setup for predictions can be stated as

$$x_{ti} \sim \mathcal{GP}_f(\mathbf{x}_{t-1} | \boldsymbol{\alpha}, \boldsymbol{\beta}_i)\,, \quad y_{tj} \sim \mathcal{GP}_g(\mathbf{x}_t | \boldsymbol{\xi}, \boldsymbol{\upsilon}_j)\,,$$

where x_{ti} is the i^{th} dimension of $\mathbf{x}_t$ and y_{tj} is the j^{th} dimension of $\mathbf{y}_t$. Note that $\mathbf{x}_{t+1} \perp \mathbf{x}_{t-1} | \mathbf{x}_t, \boldsymbol{\alpha}, \boldsymbol{\beta}$, which preserves the Markovian property in eq. (1). Without using a pseudo training set, the conditional independence property would be $\mathbf{x}_{t+1} \perp \mathbf{x}_{t-1} | \mathbf{x}_t, f$, which requires conditioning on the infinite dimensional object f. This makes it difficult to design practical inference algorithms that exploit the Markovian property.

Additional parameters, such as the initial state distribution, are also learned during training. The hyperparameters for the dynamics GP and the measurement GP are denoted by θ_f and θ_g, respectively. Just as with the LDS, the prior on the initial state is a Gaussian with mean $\boldsymbol{\mu}_0$ and covariance $\boldsymbol{\Sigma}_0$. The entire parameter space can be summarized as $\Theta = \{\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\xi}, \boldsymbol{\upsilon}, \theta_f, \theta_g, \boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0\}$.

We apply the EM algorithm in the NLDS for inference and learning. EM iterates between two steps, the E-step and the M-step. In the E-step (or inference step), we find a posterior distribution $p(\mathbf{X}|\mathbf{Y}, \Theta)$ on the hidden states for a fixed parameter setting Θ. In the M-step, we find the parameters of the dynamical system Θ that maximizes the expected log-likelihood $Q = \mathbb{E}_{\mathbf{X}}[\log p(\mathbf{X}, \mathbf{Y}|\Theta)]$, where the expectation is taken with respect to the E-step distribution on the hidden states $\mathbf{X}$. Optimizing Q converges to a maximum likelihood solution in $p(\mathbf{Y}|\Theta)$. Both the E-step and the M-step require approximations when we model the transition dynamics f and the measurement function g with GPs.

2 Inference (E-step)

The E-step infers a posterior distribution $p(\mathbf{x}_{1:T}|\mathbf{y}_{1:T}, \Theta)$ of the sequence of latent states $\mathbf{X}$ given the observation sequence $\mathbf{Y}$. In the following, we omit the explicit conditioning on Θ for notational brevity. Due to the Markov assumption, the joint distribution of the hidden states is given by

$$p(\mathbf{x}_{1:T}|\mathbf{y}_{1:T}) = p(\mathbf{x}_1|\mathbf{y}_{1:T}) \prod_{t=2}^{T} p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{y}_{1:T})\,. \quad (2)$$

To determine the marginal posterior distributions $p(\mathbf{x}_t|\mathbf{y}_{1:T})$, we use the forward-backward algorithm [14]. The forward-backward algorithm requires solving three sub-problems: time update (Section 2.1.1), measurement update (Section 2.1.2), and the backward sweep (Section 2.2) to complete smoothing. Our use of forward-backward explicitly incorporates the uncertainties and the nonlinearities in the transition dynamics and measurements through GP models $\mathcal{GP}_f$ and $\mathcal{GP}_g$, respectively.

2.1 Forward Sweep (Filtering)

The forward sweep comprises time update and measurement update. They typically alternate in a predictor-corrector setup: First, the time update predicts the hidden state at time t given past observations from time 1 to $t-1$. Second, the measurement update refines the prediction by incorporating new evidence from the current observation at time t.

2.1.1 Time Update

The time update corresponds to computing the one-step-ahead predictive distribution of the hidden state $p(\mathbf{x}_t|\mathbf{y}_{1:t-1})$ using $p(\mathbf{x}_{t-1}|\mathbf{y}_{1:t-1})$ as a (Gaussian) prior on $\mathbf{x}_{t-1}$. Propagating a density on $\mathbf{x}_{t-1}$ to a density on $\mathbf{x}_t$ corresponds to GP prediction (under model $\mathcal{GP}_f$) with uncertain inputs, [13]. The exact mean $\boldsymbol{\mu}_t^p$ and covariance $\mathbf{C}_t^p$ of the predictive distribution can be computed analytically.[1] The predictive distribution on $\mathbf{x}_t$ can therefore be approximated by a Gaussian $\mathcal{N}(\boldsymbol{\mu}_t^p, \mathbf{C}_t^p)$ using exact moment matching.

Analogously, we can approximate the predictive distribution in *observed space*, $p(\mathbf{y}_t|\mathbf{y}_{1:t-1})$, by a Gaussian

[1]We use the notation $\boldsymbol{\mu}_t^p$ and $\mathbf{C}_t^p$ to indicate a one-step-ahead prediction within latent space (with uncertain inputs) from time step $t-1$ to t using the dynamics GP, $\mathcal{GP}_f$, given $\mathbf{y}_{1:t-1}$.

$\mathcal{N}\left(\boldsymbol{\mu}_y, \mathbf{C}_{yy}\right)$ with the exact mean and the exact covariance using the above prediction $p(\mathbf{x}_t|\mathbf{y}_{1:t-1})$ as a prior on $\mathbf{x}_t$. Detailed expressions are given in [2].

2.1.2 Measurement Update

The measurement update computes a posterior distribution $p(\mathbf{x}_t|\mathbf{y}_{1:t})$ by refining the predictive distribution $p(\mathbf{x}_t|\mathbf{y}_{1:t-1})$ by incorporating the most recent measurement $\mathbf{y}_t$. Reporting the results from [2], the updated state distribution (filter distribution) is determined as

$$p(\mathbf{x}_t|\mathbf{y}_{1:t}) = \mathcal{N}(\boldsymbol{\mu}_t^e, \mathbf{C}_t^e)\,, \tag{3}$$

$$\boldsymbol{\mu}_t^e = \boldsymbol{\mu}_t^p + \mathbf{C}_{xy}\mathbf{C}_{yy}^{-1}(\mathbf{y}_t - \boldsymbol{\mu}_y)\,, \tag{4}$$

$$\mathbf{C}_t^e = \mathbf{C}_t^p - \mathbf{C}_{xy}\mathbf{C}_{yy}^{-1}\mathbf{C}_{yx}\,. \tag{5}$$

New evidence from the observation $\mathbf{y}_t$ is incorporated in eq. (4). The cross-covariance $\mathbf{C}_{xy} = \mathrm{Cov}\left[\mathbf{x}_t, \mathbf{y}_t|\mathbf{y}_{1:t-1}\right]$ is determined exactly. The matrix $\mathbf{C}_{xy}\mathbf{C}_{yy}^{-1}$ plays the role of the Kalman gain for linear dynamical systems. However, $\mathbf{C}_{xy}$ and $\mathbf{C}_{yy}$ are based upon predictions with nonlinear GP models for a) the transition dynamics and b) the measurement function in eq. (1).

2.2 Backward Sweep

The backward sweep is required for the M-step of the EM algorithm and corresponds to seeking the distribution

$$\gamma(\mathbf{x}_t) \coloneqq p(\mathbf{x}_t|\mathbf{y}_{1:T}) \tag{6}$$

of the hidden state conditioned on all (previous, current, and future) observations. We present a new algorithm for smoothing in NLDS. We initialize $\gamma(\mathbf{x}_T)$ by the last step of the forward sweep, that is, the filter distribution $p(\mathbf{x}_T|\mathbf{y}_{1:T})$. The distributions of the smoothed states $\gamma(\mathbf{x}_{t-1})$ are computed recursively from $t = T$ to 2 according to

$$\gamma(\mathbf{x}_{t-1}) = \int p(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{y}_{1:T})\gamma(\mathbf{x}_t)d\mathbf{x}_t \tag{7}$$

$$= \int p(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{y}_{1:t-1})\gamma(\mathbf{x}_t)d\mathbf{x}_t \tag{8}$$

by integrating out the smoothed hidden state at time step t. Evaluating eq. (8) has two steps. First, we calculate the conditional $p(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{y}_{1:t-1})$ in a "backward inference" step. Second, we solve the integral of this conditional distribution multiplied with $\gamma(\mathbf{x}_t)$ to determine the marginal $p(\mathbf{x}_{t-1}|\mathbf{y}_{1:T})$ Additionally, learning in the M-step requires the computation of the cross-covariances $p(\mathbf{x}_{t-1}, \mathbf{x}_t|\mathbf{y}_{1:T})$. In the following, we discuss these steps in detail.

Backward inference. We compute the conditional distribution $p(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{y}_{1:t-1})$ by first approximating the joint distribution $p(\mathbf{x}_{t-1}, \mathbf{x}_t|\mathbf{y}_{1:t-1})$ with

$$p(\mathbf{x}_{t-1}, \mathbf{x}_t|\mathbf{y}_{1:t-1}) = \mathcal{N}\left(\begin{bmatrix}\boldsymbol{\mu}_{t-1}^e \\ \boldsymbol{\mu}_t^p\end{bmatrix}, \begin{bmatrix}\mathbf{C}_{t-1}^e & \mathbf{C}_{ep} \\ \mathbf{C}_{ep}^{\mathrm{T}} & \mathbf{C}_t^p\end{bmatrix}\right) \tag{9}$$

and then conditioning on $\mathbf{x}_t$. This approximation implicitly linearizes the transition dynamics f (see eq. (1)) globally, in contrast to the local linearization of the EKF. We can compute all of the variables in eq. (9) from three sources: First, $\boldsymbol{\mu}_{t-1}^e$ and $\mathbf{C}_{t-1}^e$ are given by the filter distribution, eq. (4), at time $t-1$. Second, $\boldsymbol{\mu}_t^p$ and $\mathbf{C}_t^p$ are the mean and the covariance of the predictive distribution $p(\mathbf{x}_t|\mathbf{y}_{1:t-1}) = \mathcal{N}\left(\boldsymbol{\mu}_t^p, \mathbf{C}_t^p\right)$ at time t. Third, the cross-covariance $\mathbf{C}_{ep} = \mathrm{Cov}\left[\mathbf{x}_{t-1}, \mathbf{x}_t|\mathbf{y}_{1:t-1}\right]$ can be computed analytically in the forward sweep. We omit the exact equations, but refer to [2], where similar computations are performed to compute the cross-covariance $\mathbf{C}_{xy} = \mathrm{Cov}\left[\mathbf{x}_t, \mathbf{y}_t|\mathbf{y}_{1:t-1}\right]$.

Finally, with $\mathbf{J}_{t-1} \coloneqq \mathbf{C}_{ep}(\mathbf{C}_t^p)^{-1}$, we obtain the desired conditional

$$p(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{y}_{1:t-1}) = \mathcal{N}\big(\mathbf{x}_{t-1}\,|\,\mathbf{m}, \mathbf{S}\big)\,, \tag{10}$$

$$\mathbf{m} = \boldsymbol{\mu}_{t-1}^e + \mathbf{J}_{t-1}(\mathbf{x}_t - \boldsymbol{\mu}_t^p)\,, \quad \mathbf{S} = \mathbf{C}_{t-1}^e - \mathbf{J}_{t-1}\mathbf{C}_{ep}^{\mathrm{T}}\,.$$

Smoothed state distribution. We compute the integral in eq. (8) by exploiting the *mixture property of Gaussians* [4]: Since $p(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{y}_{1:t-1}) = \mathcal{N}\left(\mathbf{J}_{t-1}\mathbf{x}_t + \boldsymbol{\mu}_{t-1}^e - \mathbf{J}_{t-1}\boldsymbol{\mu}_t^p, \mathbf{C}_{t-1}^e - \mathbf{J}_{t-1}\mathbf{C}_{ep}^{\mathrm{T}}\right)$, and $p(\mathbf{x}_t|\mathbf{Y}) = \mathcal{N}\left(\boldsymbol{\mu}_t^s, \mathbf{C}_t^s\right)$, the mixture property leads to the smoothed state distribution

$$p(\mathbf{x}_{t-1}|\mathbf{y}_{1:T}) = \gamma(\mathbf{x}_{t-1}) = \mathcal{N}\big(\mathbf{x}_{t-1}\,|\,\boldsymbol{\mu}_{t-1}^s, \mathbf{C}_{t-1}^s\big)\,,$$
$$\boldsymbol{\mu}_{t-1}^s = \boldsymbol{\mu}_{t-1}^e + \mathbf{J}_{t-1}(\boldsymbol{\mu}_t^s - \boldsymbol{\mu}_t^p)\,,$$
$$\mathbf{C}_{t-1}^s = \mathbf{C}_{t-1}^e + \mathbf{J}_{t-1}(\mathbf{C}_t^s - \mathbf{C}_t^p)\mathbf{J}_{t-1}^{\mathrm{T}}\,. \tag{11}$$

Cross-covariances for learning. Parameter learning using EM or variational methods requires the full distribution $p(\mathbf{x}_{1:T}|\mathbf{Y})$. Due to the Markov assumption, $\mathbb{E}\left[\mathbf{x}_t|\mathbf{Y}\right]$, $\mathrm{Cov}\left[\mathbf{x}_t|\mathbf{Y}\right]$, and

$$\begin{aligned}&\mathrm{Cov}\left[\mathbf{x}_{t-1}, \mathbf{x}_t|\mathbf{Y}\right]\\ &= \mathbb{E}_{\mathbf{x}_{t-1},\mathbf{x}_t}[\mathbf{x}_{t-1}\mathbf{x}_t^{\mathrm{T}}|\mathbf{Y}] - \mathbb{E}_{\mathbf{x}_{t-1}}[\mathbf{x}_{t-1}|\mathbf{Y}]\,\mathbb{E}_{\mathbf{x}_t}[\mathbf{x}_t|\mathbf{Y}]^{\mathrm{T}}\\ &= \int \mathbb{E}_{\mathbf{x}_{t-1}}[\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{Y}]\mathbf{x}_t^{\mathrm{T}}p(\mathbf{x}_t|\mathbf{Y})d\mathbf{x}_t - \boldsymbol{\mu}_{t-1}^s(\boldsymbol{\mu}_t^s)^{\mathrm{T}}\,.\end{aligned}$$

are sufficient statistics for the entire distribution. Multiplying eq. (9) with $\gamma(\mathbf{x}_t)$ and integrating over $\mathbf{x}_t$, yields the desired cross-covariance

$$\mathrm{Cov}\left[\mathbf{x}_{t-1}, \mathbf{x}_t|\mathbf{Y}\right] = \mathbf{J}_{t-1}\mathbf{C}_t^s\,. \tag{12}$$

Alg. 1 summarizes the E-step.

Unlike [17] the "backward" conditional distribution, eq. (10) is computed without the need of an explicit

Algorithm 1 Forward-backward algorithm for GPIL

```
1: function NLDSsmoother
2: $(\boldsymbol{\mu}_0^e, \mathbf{C}_0^e) \leftarrow (\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0)$   ▷ initialize forward sweep
3: for $t = 1 : T$ do   ▷ forward sweep
4:     compute $p(\mathbf{x}_t|\mathbf{y}_{1:t}) = \mathcal{N}(\boldsymbol{\mu}_t^e, \mathbf{C}_t^e)$   ▷ eq. (4)
5: end for
6: $(\boldsymbol{\mu}_T^s, \mathbf{C}_T^s) \leftarrow (\boldsymbol{\mu}_T^e, \mathbf{C}_T^e)$   ▷ initialize backward sweep
7: for $t = T-1 : 1$ do   ▷ backward sweep
8:     compute $p(\mathbf{x}_t|\mathbf{Y}) = \mathcal{N}(\boldsymbol{\mu}_t^s, \mathbf{C}_t^s)$   ▷ eq. (11)
9:     ▷ cross-covariance between $\mathbf{x}_{t+1}^s, \mathbf{x}_t^s$, eq. (12)
10:    $\mathbf{R}_{t+1}^s \leftarrow \mathbf{C}_{t+1}^s \mathbf{J}_t^{\mathrm{T}}$
11: end for
12: ▷ return sufficient statistics for $p(\mathbf{x}_{1:T}|\mathbf{y}_{1:T}, \Theta)$
13: return $\boldsymbol{\mu}_{1:T}^s,\ \mathbf{C}_{1:T}^s,\ \mathbf{R}_{2:T}^s$
```

backward model. The dynamics GP solely models the forward dynamics of the latent states. Our inference algorithm is robust against numerical problems for high measurement noise, problems which were reported in [19] in the context of inference in NLDS using the UKF and Expectation Propagation [11].

3 Learning (M-Step)

In the following, we derive the M-step for gradient-based optimization of the parameters Θ. In the M-Step, we seek the parameters Θ that maximize the likelihood lower bound $Q = \mathbb{E}_{\mathbf{X}}[\log p(\mathbf{X}, \mathbf{Y}|\Theta)]$ where the expectation is computed under the distribution from the E-Step, meaning $\mathbf{X}$ is treated as random. We decompose Q into

$$Q = \mathbb{E}_{\mathbf{X}}[\log p(\mathbf{X}, \mathbf{Y}|\Theta)] = \mathbb{E}_{\mathbf{X}}[\log p(\mathbf{x}_1|\Theta)] + \mathbb{E}_{\mathbf{X}}\left[\sum_{t=2}^{T} \underbrace{\log p(\mathbf{x}_t|\mathbf{x}_{t-1}, \Theta)}_{\text{Section 3.1}} + \sum_{t=1}^{T} \underbrace{\log p(\mathbf{y}_t|\mathbf{x}_t, \Theta)}_{\text{Section 3.2}}\right] \quad (13)$$

using the factorization properties of the model.

In the following we use the notation $\mu_i(\mathbf{x}) = \mathbb{E}_{f_i}[f_i(\mathbf{x})]$ to refer to the expected value of the i^{th} dimension of f when evaluated at $\mathbf{x}$. Likewise, $\sigma_i^2(\mathbf{x}) = \mathrm{Var}_{f_i}[f_i(\mathbf{x})]$ refers to the variance of the i^{th} dimension of f when evaluated at $\mathbf{x}$.

3.1 Contribution from Transition Model

We focus on finding a lower-bound approximation to the contribution from the transition function, namely,

$$\mathbb{E}_{\mathbf{X}}[\log p(\mathbf{x}_t|\mathbf{x}_{t-1}, \Theta)] = \sum_{i=1}^{M} \mathbb{E}_{\mathbf{X}}\left[\log \mathcal{N}(x_{ti}|\mu_i(\mathbf{x}_{t-1}), \sigma_i^2(\mathbf{x}_{t-1}))\right], \quad (14)$$

which equals

$$-\frac{1}{2}\sum_{i=1}^{M} \underbrace{\mathbb{E}_{\mathbf{X}}\left[\frac{(x_{ti} - \mu_i(\mathbf{x}_{t-1}))^2}{\sigma_i^2(\mathbf{x}_{t-1})}\right]}_{\text{Data Fit Term}} + \underbrace{\mathbb{E}_{\mathbf{X}}\left[\log \sigma_i^2(\mathbf{x}_{t-1})\right]}_{\text{Complexity Term}}.$$

Eq. (14) amounts to an expectation over a nonlinear function of a normally distributed random variable since $\mathbf{X}$ is approximate Gaussian (E-step). The expectation in eq. (14) corresponds to an intractable integral, which is due to the state-dependent variances in our model.

Data fit term. We first consider the data fit term in eq. (14), which is an expectation over the square Mahalanobis distance. For tractability, we approximate the expectation of the ratio

$$\mathbb{E}_{\mathbf{X}}\left[\frac{(x_{ti} - \mu_i(\mathbf{x}_{t-1}))^2}{\sigma_i^2(\mathbf{x}_{t-1})}\right] \approx \frac{\mathbb{E}_{\mathbf{X}}\left[(x_{ti} - \mu_i(\mathbf{x}_{t-1}))^2\right]}{\mathbb{E}_{\mathbf{X}}\left[\sigma_i^2(\mathbf{x}_{t-1})\right]} =: \tilde{M}_f(x_{ti}, \mathbf{x}_{t-1}). \quad (15)$$

Complexity term. We next approximate the complexity penalty in eq. (14), which penalizes uncertainty. The contribution from the logarithm in the expectation can be lower bounded by

$$\mathbb{E}_{\mathbf{X}}\left[\log \sigma_i^2(\mathbf{x}_{t-1})\right] \leq \log \mathbb{E}_{\mathbf{X}}\left[\sigma_i^2(\mathbf{x}_{t-1})\right], \quad (16)$$

where we used Jensen's inequality. Eq. (16) also serves as a Taylor approximation centered at $\mathbb{E}_{\mathbf{X}}\left[\sigma_i^2(\mathbf{x}_{t-1})\right]$, which is accurate to first order for symmetry reasons.

3.2 Contribution from Measurement Model

The measurement function g in eq. (1) is assumed unknown and modeled by $\mathcal{GP}_g$. Our algorithm allows for joint training of the dynamics GP and the measurement GP. An expression $\tilde{M}_g(y_{ti}, \mathbf{x}_t)$ nearly identical to eq. (15) (contribution from the dynamics model) can be computed for the observation model. We can also find a nearly identical measurement model version of eq. (16).

4 Summary of Algorithm

The manifestation of EM in the NLDS is summarized by Alg. 2. The function `NLDSsmoother` implements the E-step. The maximization routine implements the M-step. Following the steps in the M-step (Section 3), we finally approximate the exact objective function Q

Algorithm 2 EM using GPIL

```
1: repeat
2:     ▷ E-step: Section 2, Alg. 1
3:     $\mu^s_{1:T}$, $\mathbf{C}^s_{1:T}$, $\mathbf{R}^s_{2:T}$ ← NLDSsmoother($\mathbf{Y}$, $\Theta$)
4:     ▷ M-step: Section 3, eq. (17)
5:     $\Theta$ ← maximize $\tilde{Q}(\Theta, \mu^s_{1:T}, \mathbf{C}^s_{1:T}, \mathbf{R}^s_{2:T})$ wrt $\Theta$
6: until convergence
7: return $\Theta$, $\mu^s_{1:T}$, $\mathbf{C}^s_{1:T}$, $\mathbf{R}^s_{2:T}$
```

by

$$\begin{aligned}\tilde{Q} = &-\tfrac{1}{2}\sum_{t=2}^{T}\sum_{i=1}^{M}\log\mathbb{E}_{\mathbf{X}}\left[\sigma^2_{f_i}(\mathbf{x}_{t-1})\right] + \tilde{M}_f(x_{ti}, \mathbf{x}_{t-1}) \\ &- \tfrac{1}{2}\sum_{t=1}^{T}\sum_{i=1}^{D}\log\mathbb{E}_{\mathbf{X}}\left[\sigma^2_{g_i}(\mathbf{x}_t)\right] + \tilde{M}_g(y_{ti}, \mathbf{x}_t) \qquad (17) \\ &- \tfrac{1}{2}\log|\boldsymbol{\Sigma}_0| - \mathbb{E}_{\mathbf{X}}\left[\tfrac{1}{2}(\mathbf{x}_1 - \boldsymbol{\mu}_0)^{\mathrm{T}}\boldsymbol{\Sigma}_0^{-1}(\mathbf{x}_1 - \boldsymbol{\mu}_0)\right] .\end{aligned}$$

The partial derivatives of $\tilde{Q}$ with respect to Θ can be computed analytically, which allows for gradient-based parameter optimization.

5 Results

We evaluated our approach on both real and synthetic data sets using one-step-ahead prediction. We compared GPIL predictions to eight other methods, the time independent model (TIM) with $\mathbf{y}_t \sim \mathcal{N}(\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)$, the Kalman filter, the UKF, the EKF, the Autoregressive GP (ARGP) trained on a set of pairs $(\mathbf{y}_i, \mathbf{y}_{i+1})$, and the GP-UKF [8], and the neural network/EKF based NDFA [6].[2] Note that the EKF, the UKF, and the GP-UKF required access to the true functions f and g. For synthetic data, f and g were known. For the real data set, we used functions for f and g that resembled the mean functions of the learned GP models using the GPIL algorithm. For the synthetic data, we also compared to the GPDM [18]. For the real data set, however, GPDM was not tested due to the computational demand when running on the large test data set.

The Kalman filter, the EKF, and the UKF are standard state-space approaches to time series analysis. We compared against the ARGP because it is a powerful non-parametric Bayesian approach to time series prediction; one-step-ahead predictions with ARGP can be done analytically, but multi-step predictions require moment matching approximations. We tested ARGP on orders of 1 to 20 and selected the one with the best performance, order 10 for the real data set and order 1 for the synthetic data set. Note that the ARGP and the TIM have no notion of a latent space and cannot be used for estimation of the latent state, that is, they cannot be applied in a typical tracking and/or control setup.

[2]Implementations of the Kalman filter and the UKF are based on the software available at `http://www.cs.ubc.ca/~murphyk/Software` and `http://www.cs.ubc.ca/~nando/software`, respectively.

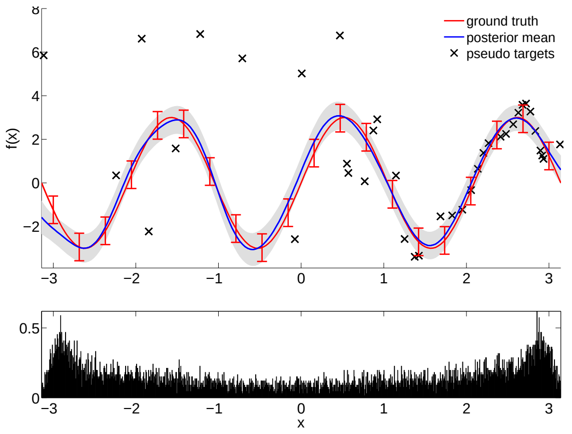

Figure 3: True (red) and learned (blue) transition function with histogram of the inputs x_t in the test set. Both red error bars and the shaded area represent twice the standard deviation of the system noise. Pseudo targets are represented by black crosses.

The evaluation metrics were the negative log predictive likelihood (NLL) and the root mean squared error (RMSE) between the mean of the prediction and the true value. Note that unlike the NLL, the RMSE does not account for uncertainty.

Synthetic data. We considered an illustrative example where the hidden state followed sinusoidal dynamics specified by

$$x_t = 3\sin(3\,x_{t-1}) + \epsilon\,, \quad \epsilon \sim \mathcal{N}\left(0, \sigma^2_\epsilon\right)\,, \quad \sigma^2_\epsilon = 0.1^2\,.$$

Furthermore, we considered a linear measurement model, $y_t = x_t + \nu$, with $\nu \sim \mathcal{N}\left(0, \sigma^2_\nu\right)$, $\sigma^2_\nu = 0.1^2$.

The results were produced using a pseudo training set of size $N = 50$, $T = 100$ training observations and 10,000 test observations. Quantitative results for the sinusoidal dynamics are shown in Table 1. The true dynamics model is compared to the learned dynamics model $\mathcal{GP}_f$ in Fig. 3. The error bars (shaded area) on the learned dynamics model include both system noise *and* model uncertainty.

The UKF and the EKF required access to the *true* generating process. Having the true dynamics model gives the UKF and the EKF a distinct advantage over the competing methods. However, the GPIL could still outperform them since both the UKF and GP-UKF had trouble with the high curvature in the dynam-

Table 1: Comparison of the GPIL with six other methods on the sinusoidal dynamics example and the Whistler snowfall data. We trained on daily snowfall from Jan. 1 1972–Dec. 31 1973 and tested on next day predictions for 1974–2008. We report the NLL per data point and the RMSE as well as the NLL 95% error bars.

	Method	NLL synth.	RMSE synth.	NLL real	RMSE real
general	TIM	2.21±0.0091	2.18	1.47±0.0257	1.01
	Kalman [14]	2.07±0.0103	1.91	1.29±0.0273	0.783
	ARGP [15]	1.01±0.0170	0.663	1.25±0.0298	0.793
	NDFA [6]	2.20±0.00515	2.18	14.6±0.374	1.06
	GPDM [18]	3330±386	2.13	N/A	N/A
	GPIL $\star$	**0.917 ± 0.0185**	**0.654**	**0.684 ± 0.0357**	**0.769**
requires prior knowledge	UKF [7]	4.55±0.133	2.19	1.84±0.0623	0.938
	EKF [10]	1.23±0.0306	0.665	1.46±0.0542	0.905
	GP-UKF [8]	6.15±0.649	2.06	3.03±0.357	0.884

ics model, which caused them to predict future observations with unreasonably high certainty (overconfidence). The GP-UKF used the same pseudo training set during test as the GPIL algorithm; given that the GP-UKF performed worse than the GPIL we speculate that the filtering/prediction (E-step) method is better in the GPIL than the GP-UKF confirming the results from [2]. Although the EKF performs better than the UKF, its linearizations of the dynamics appear to be worse than all approximations made by the GPIL algorithm.

The ARGP was disadvantaged because it had no notion of a latent state-space. However, it was still competitive with the state-space models. The analytic nature of the Kalman filter did not make up for its inappropriate modeling assumptions, that is, the linearity of the dynamics model; it is not able to predict with appropriate variances, resulting in a high NLL. The flexibility of the GPIL allowed it to outperform the simpler analytic models despite its approximations. The approximations in our model are as least as good as the approximations used in EKF, UKF, GPDM, etc.

Real data. We used historical snowfall data in Whistler, BC, Canada[3] to evaluate the GPIL and other methods on real data. The models were trained on two years of data; the GPIL used a pseudo training set of size $N = 15$; we evaluated the models' ability to predict next day snowfall using 35 years of test data. The results are shown in Table 1.

The GPIL learned a GP model for a scalar close-to-linear stochastic latent transition function. A possible interpretation of the results is that the daily precipitation is almost linear. Note that for temperatures above freezing no snow occurs, which resulted in a hinge measurement model. The GPIL successfully learned a hinge-like function for the measurement model, Fig. 4,

[3]`http://www.climate.weatheroffice.ec.gc.ca/`

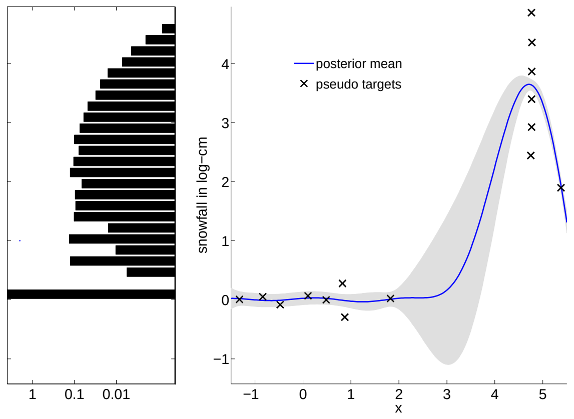

Figure 4: Learned measurement GP (right panel) and log histogram of the observations (left panel) during testing, real data set. The gray area is twice the predictive standard deviation of the model uncertainty plus the measurement noise. Pseudo targets are represented by black crosses.

which allowed for predicting no snowfall the next day with high probability. The Kalman filter was incapable of such predictions since it assumes linear functions f and g, respectively.

6 Discussion and Conclusions

Parameterizing a GP using the pseudo training set is one way to *train* a GP with unknown inputs. In principle, we could train the model by integrating the pseudo training set out. However, this approach is analytically intractable.

It is possible to compute the expectations in eqs. (13) and (14) by sampling $\mathbf{X}$ from the E-step distribution. If we sample $\mathbf{x}_{t-1}$ then x_{ti} can be integrated out analytically providing a much lower variance estimator than sampling x_{ti} as well. The sampling approach can give a small increase in performance over the deterministic approximations discussed in this paper.

In our model, the latent states $\mathbf{x}_t$ are *never* observed directly. We solely have access to observations $\mathbf{y}_t$ to train the latent dynamics and measurement functions. By contrast, direct access to ground truth observations of a latent state sequence was required in [2, 8] to train the dynamics model.

We introduced a general method for inference and learning in nonlinear state-space models using EM. Both the transition function between the hidden states and the measurement function are modeled by GPs allowing for quantifying model uncertainty and flexible modeling. Our approach exploits the properties of GPs and allows for approximate smoothing in closed form (E-step). The free parameters of the GPs are their hyper-parameters and a pseudo training set. We employed a gradient-based optimizer to learn the free parameters (M-step). We showed that our learning approach successfully learned nonlinear (latent) dynamics based on noisy measurements only. Moreover, our algorithm outperformed standard and state-of-the-art approaches for time-series predictions and inference, such as Kalman filtering, the EKF, the UKF, the GP-UKF, the GPDM, the NDFA, and AR models.

References

[1] X. Boyen and D. Koller. Tractable inference for complex stochastic processes. In *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence*, pp. 33–42, San Francisco, CA, USA, 1998. Morgan Kaufmann.

[2] M. P. Deisenroth, M. F. Huber, and U. D. Hanebeck. Analytic moment-based Gaussian process filtering. In *Proceedings of the 26th International Conference on Machine Learning*, pp. 225–232, Montreal, Canada, 2009. Omnipress.

[3] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, series B*, 39(1):1–38, 1977.

[4] A. Gelman, J. B. Carlin, H. S. Stern, and D. B. Rubin. *Bayesian Data Analysis.* 2nd Edition. Chapman & Hall/CRC, 2004.

[5] Z. Ghahramani and S. Roweis. Learning nonlinear dynamical systems using an EM algorithm. In *Advances in Neural Information Processing Systems 11*, pp. 599–605. 1999.

[6] A. Honkela and H. Valpola. Unsupervised variational Bayesian learning of nonlinear models. In *Advances in Neural Information Processing Systems 17*, pp. 593–600. The MIT Press, Cambridge, MA, 2005.

[7] S. J. Julier and J. K. Uhlmann. A new extension of the Kalman filter to nonlinear systems. In *Proceedings of AeroSense: 11th Symposium on Aerospace/Defense Sensing, Simulation and Controls*, pp. 182–193, Orlando, FL, USA, 1997.

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

[9] J. Ko and D. Fox. Learning GP-BayesFilters via Gaussian process latent variable models. In *Proceedings of Robotics: Science and Systems*, Seattle, USA, 2009.

[10] P. S. Maybeck. *Stochastic Models, Estimation, and Control*, vol. 141 of *Mathematics in Science and Engineering.* Academic Press, Inc., 1979.

[11] T. P. Minka. *A Family of Algorithms for Approximate Bayesian Inference.* PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 2001.

[12] M. Opper. A Bayesian approach to online learning. In *Online Learning in Neural Networks*, pp. 363–378. Cambridge University Press, 1998.

[13] J. Quiñonero-Candela, A. Girard, J. Larsen, and C. E. Rasmussen. Propagation of uncertainty in Bayesian kernel models—application to multiple-step ahead forecasting. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 701–704, 2003.

[14] L. Rabiner. A tutorial on HMM and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, February 1989.

[15] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning.* The MIT Press, 2006.

[16] E. Snelson and Z. Ghahramani. Sparse Gaussian processes using pseudo-inputs. In *Advances in Neural Information Processing Systems 18*, pp. 1257–1264. The MIT Press, 2006.

[17] E. A. Wan and R. van der Merwe. *The unscented Kalman filter*, chapter in Kalman Filtering and Neural Networks, pp. 221–280. Wiley, 2001.

[18] J. M. Wang, D. J. Fleet, and A. Hertzmann. Gaussian process dynamical models for human motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2):283–298, 2008.

[19] A. Ypma and T. Heskes. Novel approximations for inference in nonlinear dynamical systems using expectation propagation. *Neurocomputing*, 69:85–99, 2005.