State-Space Inference and Learning with Gaussian Processes

Ryan Turner[1] Marc Peter Deisenroth[1,2] Carl Edward Rasmussen[1,3]
[1]Department of Engineering, University of Cambridge, Trumpington Street, Cambridge CB2 1PZ, UK
[2]Department of Computer Science & Engineering, University of Washington, Seattle, WA 98195, USA
[3]Max Planck Institute for Biological Cybernetics, Spemannstraße 38, 72076 Tübingen, Germany

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 areas, such as 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 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 evolve over 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

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

Here, the system noise $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_\epsilon)$ and the measurement noise $\boldsymbol{\nu} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_\nu)$ 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 find a posterior distribution on the latent states $\mathbf{x}$ using measurements $\mathbf{y}$ only. Bayesian inference of the hidden states (smoothing) in LDS with Gaussian noise can be done exactly via Kalman smoothing (Rauch et al., 1965). In *learning*, the goal is to infer 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 are required.

Examples of approximate inference in nonlinear dynamical systems include the extended Kalman filter (EKF) (Maybeck, 1979), the unscented Kalman filter (UKF) (Julier and Uhlmann, 1997), and the assumed density filter (ADF) (Opper, 1998). Typically, a parametric form for the transition dynamics is assumed. General forms of the dynamics model for inference and learning were proposed in terms of radial basis functions (RBF) (Ghahramani and Roweis, 1999) and neural networks (Honkela and Valpola, 2005). In the context of modeling human motion, Gaussian processes (GPs) have been used for inference (Wang et al., 2008; Ko and Fox, 2009b). Recently, GPs were used for filtering in the context of the UKF, the EKF (Ko and Fox, 2009a), and the ADF (Deisenroth et al., 2009).

For nonlinear systems these methods encounter problems: The local linearizations of the EKF and the UKF can lead to overfitting. Neural network (Honkela and Valpola, 2005) and RBF (Ghahramani and Roweis, 1999) approaches have a constant level of uncertainty

Appearing in Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS) 2010, Chia Laguna Resort, Sardinia, Italy. Volume 9 of JMLR: W&CP 9. Copyright 2010 by the authors.

in the dynamics and measurement functions, which means they do not appropriately quantify uncertainty in f and g. Although probabilistic GPs are used in Wang et al. (2008); Ko and Fox (2009b), the MAP estimation (point estimate) 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 (Deisenroth et al., 2009; Ko and Fox, 2009a) 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. Our model integrates out the latent states unlike Wang et al. (2008); Ko and Fox (2009b), where a MAP approximation to the distribution of the latent state is used. 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) (Dempster et al., 1977).

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* them 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*, (Rasmussen and Williams, 2006). Throughout this paper, we use the squared exponential kernel and a prior mean of zero. Independent GPs are used for each target dimension of f and g.

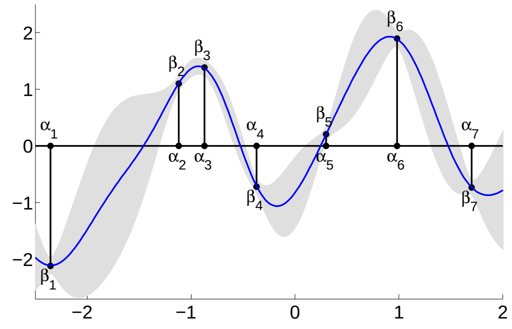

Figure 1: An example of a function distribution inferred from a pseudo training set. The $\boldsymbol{\alpha}_i$ are the pseudo training inputs, while the $\boldsymbol{\beta}_i$ are the pseudo training targets. The shaded area is the 95% confidence region around the mean function (blue, solid).

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 (Snelson and Ghahramani, 2006). 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 $\boldsymbol{\upsilon} = \{\boldsymbol{\upsilon}_i \in \mathbb{R}^D\}_{i=1}^N$. We use $\boldsymbol{\upsilon}$ instead of training to $\mathbf{Y}$ directly since often $N \ll T$.

The pseudo training sets are learned jointly with the kernel hyper-parameters. Unlike the sparse GP model in Snelson and Ghahramani (2006), 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 GP model is in Fig. 1.

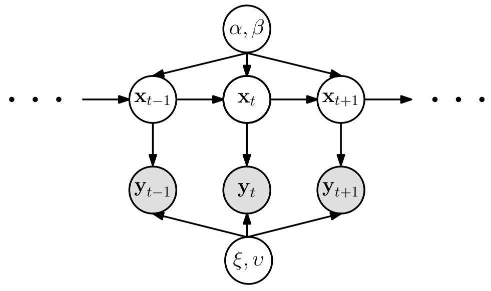

Figure 2: The free parameters $\boldsymbol{\alpha}, \boldsymbol{\beta}$ and $\boldsymbol{\xi}, \boldsymbol{\upsilon}$ serve as a pseudo training set for $\mathcal{GP}_f$ and $\mathcal{GP}_g$, respectively. $\mathcal{GP}_f$ and $\mathcal{GP}_g$ are *not* full GPs, but rather sparse GPs that impose the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

Once the pseudo training set, $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, is determined, *predicting* $\mathbf{x}_t$ from $\mathbf{x}_{t-1}$ is a GP prediction using $\mathcal{GP}_f$ (Deisenroth et al., 2009; Quiñonero-Candela et al., 2003). 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 at 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 (conditioned on the pseudo training set) is

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

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 $\mathbf{x}_{t+1} \perp\!\!\!\perp \mathbf{x}_{t-1} | \mathbf{x}_t, \boldsymbol{\alpha}, \boldsymbol{\beta}$, which preserves the Markovian property in eq. (1). The corresponding graphical model of our model is shown in Fig. 2. Without conditioning on the pseudo training set, the conditional independence property would be $\mathbf{x}_{t+1} \perp\!\!\!\perp \mathbf{x}_{t-1} | \mathbf{x}_t, f$, which requires conditioning on the infinite dimensional object f. This makes it difficult to design practical inference algorithms that exploit the Markovian property.

The initial state distribution is also learned during training. The hyper-parameters 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 $\Theta := \{\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\xi}, \boldsymbol{\upsilon}, \theta_f, \theta_g, \boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0\}$.

GPIL applies 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), GPIL finds a posterior distribution $p(\mathbf{X}|\mathbf{Y}, \Theta)$ on the hidden states for a fixed parameter setting Θ. In the M-step, GPIL finds the parameters of the dynamical system Θ that maximize the expected log-likelihood $Q := \mathbb{E}_{\mathbf{X}}\left[\log p(\mathbf{X}, \mathbf{Y}|\Theta)\right]$, where the expectation is taken with respect to the E-step distribution on the hidden state sequence $\mathbf{X}$. 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 data is given by

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

To determine the marginal posterior distributions $p(\mathbf{x}_t|\mathbf{y}_{1:T})$, we apply the forward-backward algorithm (Rabiner, 1989). 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 models of the transition dynamics and measurements through GP models $\mathcal{GP}_f$ and $\mathcal{GP}_g$.

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, (Quiñonero-Candela et al., 2003). 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.

[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 $\mathbf{y}_{1:t-1}$.

Analogously, we can approximate the predictive distribution in *observed space*, $p(\mathbf{y}_t|\mathbf{y}_{1:t-1})$, by a Gaussian $\mathcal{N}\left(\boldsymbol{\mu}_y, \mathbf{C}_{yy}\right)$ 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 Deisenroth et al. (2009).

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 Deisenroth et al. (2009), the updated state distribution (filter distribution) is determined as

$$p(\mathbf{x}_t|\mathbf{y}_{1:t}) = \mathcal{N}(\boldsymbol{\mu}_t^e, \mathbf{C}_t^e)\,, \tag{2}$$

$$\boldsymbol{\mu}_t^e = \boldsymbol{\mu}_t^p + \mathbf{C}_{xy}\mathbf{C}_{yy}^{-1}(\mathbf{y}_t - \boldsymbol{\mu}_y)\,, \tag{3}$$

$$\mathbf{C}_t^e = \mathbf{C}_t^p - \mathbf{C}_{xy}\mathbf{C}_{yy}^{-1}\mathbf{C}_{yx}\,. \tag{4}$$

New evidence from the observation $\mathbf{y}_t$ is incorporated in eq. (3). 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.

2.2 Backward Sweep

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

$$\gamma(\mathbf{x}_t) \coloneqq p(\mathbf{x}_t|\mathbf{y}_{1:T})\,, \tag{5}$$

that is, the posterior distribution of the hidden state given 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

$$\gamma(\mathbf{x}_{t-1}) = \int p(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{y}_{1:T})\gamma(\mathbf{x}_t)\, d\mathbf{x}_t \tag{6}$$

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

by integrating out the smoothed hidden state at time step t. Evaluating eq. (7) 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

$$p(\mathbf{x}_{t-1}, \mathbf{x}_t|\mathbf{y}_{1:t-1}) = \mathcal{N}\left(\begin{bmatrix}\boldsymbol{\mu}_{t-1}^e \\ \boldsymbol{\mu}_t^p\end{bmatrix}, \begin{bmatrix}\mathbf{C}_{t-1}^e & \mathbf{C}_{ep} \\ \mathbf{C}_{ep}^{\mathrm{T}} & \mathbf{C}_t^p\end{bmatrix}\right) \tag{8}$$

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. (8) from three sources: First, $\boldsymbol{\mu}_{t-1}^e$ and $\mathbf{C}_{t-1}^e$ are given by the filter distribution, eq. (3), 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 Deisenroth et al. (2009), 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

$$p(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{y}_{1:t-1}) = \mathcal{N}\big(\mathbf{x}_{t-1}\,|\,\mathbf{m}, \mathbf{S}\big)\,, \tag{9}$$

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

Smoothed state distribution. We compute the integral in eq. (7) by exploiting the *mixture property of Gaussians* (Gelman et al., 2004): Since

$$p(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{y}_{1:t-1}) \tag{10}$$
$$= \mathcal{N}\left(\mathbf{J}_{t-1}\mathbf{x}_t + \boldsymbol{\mu}_{t-1}^e - \mathbf{J}_{t-1}\boldsymbol{\mu}_t^p, \mathbf{C}_{t-1}^e - \mathbf{J}_{t-1}\mathbf{C}_{ep}^{\mathrm{T}}\right)\,,$$

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

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

are sufficient statistics for the entire distribution. Multiplying eq. (8) 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.

Algorithm 1 Forward-backward algorithm for GPIL

```
1: function NLDSsmoother
2: $p(\mathbf{x}_1) \leftarrow \mathcal{N}(\boldsymbol{\mu}_0, \boldsymbol{\Sigma}_0)$    ▷ init. 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. (3)
5: end for
6: $p(\mathbf{x}_T|\mathbf{Y}) \leftarrow \mathcal{N}(\boldsymbol{\mu}_T^e, \mathbf{C}_T^e)$    ▷ init. 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 $\boldsymbol{\mu}_{1:T}^s,\ \mathbf{C}_{1:T}^s,\ \mathbf{R}_{2:T}^s$
```

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. The factorization properties of the model yield the decomposition Q into

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

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 ith 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 ith dimension of $f(\mathbf{x})$.

3.1 Contribution from Transition Model

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

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

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 unknown and modeled by $\mathcal{GP}_g$. GPIL 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).

In summary, we approximate the exact objective function Q by

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

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

3.3 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.

4 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

Algorithm 2 EM using GPIL

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

time independent model (TIM) with $\mathbf{y}_t \sim \mathcal{N}(\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)$, where c denotes constant values, the Kalman filter, the UKF, the EKF, the autoregressive GP (ARGP) trained on a set of pairs $(\mathbf{y}_i, \mathbf{y}_{i+1})$, the GP-UKF (Ko and Fox, 2009a), and the neural network/EKF based nonlinear dynamical factor analyzer (NDFA) (Honkela and Valpola, 2005).[2] For the synthetic data, we also compared to the Gaussian process dynamical model (GPDM) (Wang et al., 2008). For the real data set, however, GPDM was not tested due to the computational demand when running on the large test data set. 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.

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. 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.

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

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

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

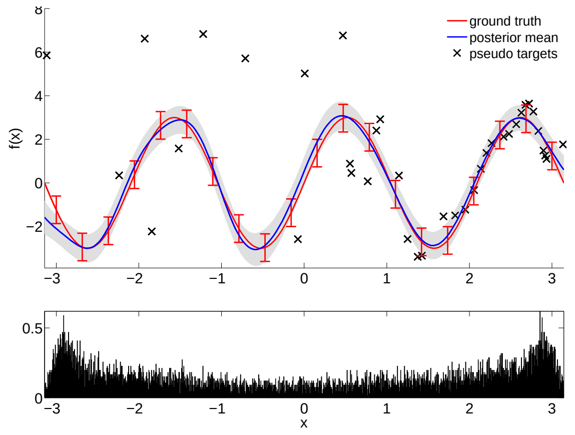

Figure 3: True (red) and learned (blue) transition function with histogram of the inputs x_t in the test set. Both the red error bars and the shaded area represent twice the standard deviation of the system noise. Pseudo targets are represented by black crosses.

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. Note that the histogram does not represent a uniform distribution since the state of the system spends more time around the stable equilibrium points of the sine function, namely $x = \pm 3$. By contrast, not many states are distributed around the unstable equilibrium point $x = 0$.

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, GPIL could still outperform them since both the UKF and GP-UKF had trouble with the high curvature in the dynamics 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 GPIL we speculate that the filtering/prediction (E-step) method is better in GPIL than the GP-UKF confirming the results from Deisenroth et al. (2009). Although the EKF performs better than the UKF, its linearizations of the

[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.

Table 1: Comparison of 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	2.07±0.0103	1.91	1.29±0.0273	0.783
	ARGP	1.01±0.0170	0.663	1.25±0.0298	0.793
	NDFA	2.20±0.00515	2.18	14.6±0.374	1.06
	GPDM	3330±386	2.13	N/A	N/A
	GPIL $\star$	**0.917 ± 0.0185**	**0.654**	**0.684 ± 0.0357**	**0.769**
requires	UKF	4.55±0.133	2.19	1.84±0.0623	0.938
prior	EKF	1.23±0.0306	0.665	1.46±0.0542	0.905
knowledge	GP-UKF	6.15±0.649	2.06	3.03±0.357	0.884

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. A state-space allows models to capture longer-order correlations without increasing the dimensionality of the parameter space. However, ARGP 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 GPIL allowed it to outperform the simpler analytic models despite its approximations. The approximations in our model are at least as good as the approximations used in EKF, UKF, GPDM, etc.

The values in Table 1 indicate that some models either find the signal in the data (ARGP, GPIL, EKF) or they do not. The NLL-values of the GPDM are high since the predicted variances are consistently underestimated.

Real data. We used historical snowfall data in Whistler, BC, Canada[3] to evaluate GPIL and other methods on real data. The models were trained on two years of data; 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.

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 nearly linear. Note that for temperatures above freezing no snow occurs, which resulted in a hinge measurement model. GPIL learned a hinge-like function for the measurement model, Fig. 4, which allowed for predicting no snowfall the next day with high probability.

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

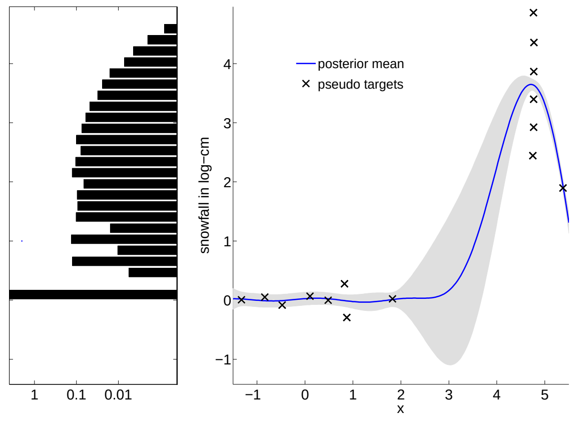

Figure 4: Learned measurement GP (right) and log histogram of the observations (left) during testing, real data set. The gray area is twice the predictive standard deviation (model uncertainty plus measurement noise). Pseudo targets are represented by crosses.

The Kalman filter was incapable of such predictions since it assumes linear functions f and g.

5 Discussion and Conclusions

In GPIL, the latent states $\mathbf{x}_t$ are *never* observed directly. Solely observations $\mathbf{y}_t$ can be accessed to train the latent dynamics and measurement functions. Contrarily, direct access to ground truth observations of a latent state sequence was required in Deisenroth et al. (2009); Ko and Fox (2009a) for training. Parameterizing a GP using the pseudo training set is one way to *train* a GP with unknown inputs. In principle, we would train the model by integrating the pseudo training set out. However, this approach is analytically intractable.

In contrast to Wan and van der Merwe (2001), the "backward" conditional distribution in eq. (9) requires only a forward model. Our inference method is robust against numerical problems for high measurement noise, problems reported in Ypma and Heskes (2005) in the context of inference in NLDS using the UKF and expectation propagation (Minka, 2001).

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 lower variance estimator than sampling x_{ti} as well. The sampling approach can give a slightly better performance over the deterministic approximations discussed in this paper.

With GPIL 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. GPIL 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, which are jointly learned using a gradient-based optimizer (M-step). We demonstrated that GPIL 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. GPIL MATLAB code will be available at `http://mlg.eng.cam.ac.uk/rdturner/`.

Acknowledgements

We thank Zoubin Ghahramani and Steven Bottone for valuable suggestions and discussions. This work was supported by both Rockwell Collins, Inc. (formerly, DataPath, Inc.) and the German Research Foundation (DFG) through grant RA 1030/1-3.

References

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

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

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

Gelman, A., Carlin, J. B., Stern, H. S., and Rubin, D. B. (2004). *Bayesian Data Analysis*. Second. Chapman & Hall/CRC.

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

Honkela, A. and Valpola, H. (2005). Unsupervised variational Bayesian learning of nonlinear models. In Saul, L. K., Weiss, Y., and Bottou, L., editors, *Advances in Neural Information Processing Systems 17*, pp. 593–600. MIT Press, Cambridge, MA.

Julier, S. J. and Uhlmann, J. K. (1997). 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.

Kalman, R. E. (1960). A New Approach to Linear Filtering and Prediction Problems. *Transactions of the ASME — Journal of Basic Engineering*, 82(Series D):35–45.

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

Ko, J. and Fox, D. (2009b). Learning GP-BayesFilters via Gaussian Process Latent Variable Models. In *Proceedings of Robotics: Science and Systems*, Seattle, WA, USA.

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

Minka, T. P. (2001). *A Family of Algorithms for Approximate Bayesian Inference*. PhD thesis, MIT, Cambridge, MA, USA.

Opper, M. (1998). A Bayesian approach to online learning. In *Online Learning in Neural Networks*, pages 363–378. Cambridge University Press.

Quiñonero-Candela, J., Girard, A., Larsen, J., and Rasmussen, C. E. (2003). Propagation of uncertainty in Bayesian kernel models—application to multiple-step ahead forecasting. In *ICASSP 2003*, vol. 2, pp. 701–704.

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

Rasmussen, C. E. and Williams, C. K. I. (2006). *Gaussian Processes for Machine Learning*. The MIT Press.

Rauch, H. E., Tung, F., and Striebel, C. T. (1965). Maximum Likelihood Estimates of Linear Dynamical Systems. *AIAA Journal*, 3:1445–1450.

Snelson, E. and Ghahramani, Z. (2006). Sparse Gaussian processes using pseudo-inputs. In *Advances in Neural Information Processing Systems 18*, pp. 1257–1264.

Wan, E. A. and van der Merwe, R. (2001). *Kalman Filtering and Neural Networks*, chapter The Unscented Kalman Filter, pp. 221–280. Wiley.

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

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