Neurocomputing 000 (2017) 1–14

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Bayesian inference for time-varying applications: Particle-based Gaussian process approaches

Yali Wang[a,*], Brahim Chaib-draa[b]

[a] Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, China
[b] Department of Computer Science and Software Engineering, Laval University, Canada

ARTICLE INFO

Article history:
Received 24 June 2016
Revised 1 December 2016
Accepted 31 January 2017
Available online xxx

Communicated by Shiliang Sun

Keywords:
Gaussian process
Time-varying applications
Bayesian inference
Sequential Monte Carlo
Non-stationarity
Heteroscedasticity

ABSTRACT

Gaussian process (GP) is a popular non-parametric model for Bayesian inference. However, the performance of GP is often limited in temporal applications, where the input–output pairs are sequentially-ordered, and often exhibit time-varying non-stationarity and heteroscedasticity. In this work, we propose two particle-based GP approaches to capture these distinct temporal characteristics. Firstly, we make use of GP to design two novel state space models which take the temporal order of input–output pairs into account. Secondly, we develop two sequential-Monte-Carlo-inspired particle mechanisms to learn the latent function values and model parameters in a recursive Bayesian framework. Since the model parameters are time-varying, our approaches can model non-stationarity and heteroscedasticity of temporal data. Finally, we evaluate our proposed approaches on a number of challenging time-varying data sets to show effectiveness. By comparing with several related GP approaches, we show that our particle-based GP approaches can efficiently and accurately capture temporal characteristics in time-varying applications.

1. Introduction

Gaussian process (GP) is a popular Bayesian nonparametric model due to its elegant inference framework [1]. However, the performance of GP is often limited in temporal applications [2–4], mainly because of two following reasons. *First*, GP is a batch modeling approach which may be not efficient to make online prediction for the sequentially-ordered temporal data sets [2,4]. *Second*, it is difficult for GP to capture distinct characteristics such as non-stationarity and heteroscedasticity which often exist in the temporal applications [1,5].

To model temporal input–output data pairs sequentially, several online variants of GP have been investigated by designing autoregressive models [6]; local online GP approaches [2,7]; Bayesian online learning with sparsification [4,8–10]; GP-based state space models [11–14] with different Bayesian approximation techniques such as Kalman filter [15,16], assumed density filter [17], Monte Carlo sampling [18–20]. However, the model parameters in these GP approaches are often assumed to be time-invariant. As a result, it may be restricted for these approaches to model time-varying non-stationarity and heteroscedasticity.

In general, non-stationarity refers to the input-dependent smoothness, where the correlation between any two latent function values does not only depend on the similarity between two corresponding input vectors, but it is also related to these two input vectors themselves [1]. Additionally, heteroscedasticity refers to the input-dependent noise, where the output noise is changed along with the location of the corresponding input vector [1]. To capture these distinct data characteristics, a number of GP extensions have been investigated by designing non-stationary covariance functions in GP [1,21,22], adding another GP on the output noise [22–25], warping GP with different nonlinear functions [26–29], developing mixtures of GP experts [30–32]. However, these batch GP approaches are often inefficient to make online prediction for time-varying applications.

To address the difficulties above, we propose two novel particle-based GP approaches in this paper, where one can make online prediction as well as model time-varying non-stationarity and heteroscedasticity in two efficient and accurate recursive Bayesian frameworks. *Firstly*, we take advantage of GP to develop two novel state space models (SSMs) in which the sequential order of temporal data pairs is modeled to make efficient online prediction. *Furthermore*, the parameters in our two SSMs are time-varying to capture non-stationarity and heteroscedasticity in the temporal data sets. Note that, the differences between our two SSMs are the different time-varying assumptions of these model parameters. This mainly accounts the trade-off between efficiency and accuracy

* Corresponding author.
E-mail addresses: yl.wang@siat.ac.cn (Y. Wang), chaib@ift.ulaval.ca (B. Chaib-draa).

http://dx.doi.org/10.1016/j.neucom.2017.01.072
0925-2312/

when performing online inference. *Finally*, based on our two time-varying SSMs, we respectively design two effective particle mechanisms to infer the latent function values and model parameters over time, in order to learn the distinct temporal characteristics in time-varying applications.

On one hand, our approaches are different from those online GP variants, since our approaches can model non-stationarity and heteroscedasticity in the temporal applications. This is mainly because we learn the model parameters over time. On the other hand, our approaches are different from those non-stationary/heteroscedastic GP variants, since our approaches can make efficient online prediction for temporal data sets. This is mainly credited to our novel GP-constructed SSMs with effective particle inference mechanisms.

The rest of this paper is organized as follows. In Section 2, we review the basics of GP. In Section 3, we introduce our particle-based GP approaches in detail. In Section 4, we evaluate our proposed approaches on five challenging time-varying applications, by comparing them with several relevant GP approaches. Finally, we conclude our paper in Section 5.

2. Background

In this section, we first introduce the definition of Gaussian process (GP). Then we review the standard GP regression approach. Finally, we briefly discuss a conventional way to model temporal data with GP regression.

2.1. Definition of Gaussian process

Gaussian process (GP) is *a collection of random variables, any finite number of which have a joint Gaussian distribution* [1]. It has been widely used as a Bayesian prior over the latent function, where the function values at any finite number of inputs are Gaussian-distributed random variables [1,33–35]. Specifically, a GP prior over the latent function $f(\mathbf{x})$ is denoted by

$$f(\mathbf{x}) \sim \mathcal{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')), \tag{1}$$

with a mean function $m(\mathbf{x})$ and a covariance function $k(\mathbf{x}, \mathbf{x}')$ [1],

$$m(\mathbf{x}) = E[f(\mathbf{x})], \tag{2}$$

$$k(\mathbf{x}, \mathbf{x}') = E[(f(\mathbf{x}) - m(\mathbf{x}))(f(\mathbf{x}') - m(\mathbf{x}'))], \tag{3}$$

where $\mathbf{x}, \mathbf{x}' \in \mathbb{R}^{d_x}$ are any two d_x-dimension input vectors.

According to the definition of GP, one can obtain that the prior over the latent function values $f(X) = [f(\mathbf{x}_1), \ldots, f(\mathbf{x}_T)]^T$ at any T input vectors $X = [\mathbf{x}_1, \ldots, \mathbf{x}_T]^T$ is a jointly Gaussian distribution, i.e.,

$$p(f(X)) = \mathcal{N}(\mathbf{m}(X), K(X, X)), \tag{4}$$

where the mean vector $\mathbf{m}(X)$ is computed from the mean function $m(\mathbf{x})$,

$$\mathbf{m}(X) = \begin{bmatrix} m(\mathbf{x}_1) \\ \cdots \\ m(\mathbf{x}_T) \end{bmatrix}, \tag{5}$$

and the covariance matrix $K(X, X)$ is computed from the covariance function $k(\mathbf{x}, \mathbf{x}')$,

$$K(X, X) = \begin{bmatrix} k(\mathbf{x}_1, \mathbf{x}_1) & \cdots & k(\mathbf{x}_1, \mathbf{x}_T) \\ \cdots & \cdots & \cdots \\ k(\mathbf{x}_T, \mathbf{x}_1) & \cdots & k(\mathbf{x}_T, \mathbf{x}_T) \end{bmatrix}. \tag{6}$$

In this work, we follow [1,36] to choose a widely-used GP prior,

$$f(\mathbf{x}) \sim \mathcal{GP}(m(\mathbf{x}) = 0, k(\mathbf{x}, \mathbf{x}')), \tag{7}$$

where the mean function is zero for simplicity.[1] Furthermore, the covariance function is a popular squared exponential (SE) kernel [1,36], i.e.,

$$k(\mathbf{x}, \mathbf{x}') = \sigma_f^2 e^{-0.5(\mathbf{x}-\mathbf{x}')^T L^{-1}(\mathbf{x}-\mathbf{x}')} = \sigma_f^2 k_\ell(\mathbf{x}, \mathbf{x}'), \tag{8}$$

where σ_f^2 is the amplitude parameter, $k_\ell(\mathbf{x}, \mathbf{x}') = e^{-0.5(\mathbf{x}-\mathbf{x}')^T L^{-1}(\mathbf{x}-\mathbf{x}')}$ is the unscaled covariance function in which L is diagonal with the length-scale parameter vector $\ell = [l_1, \ldots, l_{d_x}]^T$. In the following, we illustrate how to use GP to address the standard nonlinear regression task from a Bayesian view.

2.2. Gaussian process regression

Suppose that there is a training set $D = (X, \mathbf{y}) = \{(\mathbf{x}_t, y_t)\}_{t=1}^T$ with T input–output data pairs, where $\mathbf{x}_t \in \mathbb{R}^{d_x}$, $y_t \in \mathbb{R}$, $X = [\mathbf{x}_1, \ldots, \mathbf{x}_T]^T$, $\mathbf{y} = [y_1, \ldots, y_T]^T$. Each output is assumed to be generated from

$$y = f(\mathbf{x}) + \epsilon_y, \tag{9}$$

where the Gaussian noise is $\epsilon_y \sim \mathcal{N}(0, \sigma_y^2)$ with variance σ_y^2. Furthermore, the GP prior over the latent function is assumed to be $f(\mathbf{x}) \sim \mathcal{GP}(0, k(\mathbf{x}, \mathbf{x}'))$ with the SE covariance function $k(\mathbf{x}, \mathbf{x}') = \sigma_f^2 k_\ell(\mathbf{x}, \mathbf{x}')$ in Eq. (8). For convenience, we collect the parameters of the SE covariance function (σ_f^2, ℓ) and the noise variance σ_y^2 into a parameter vector $\Theta = [\sigma_f^2, \ell, \sigma_y^2]^T$.

Given the training set $D = (X, \mathbf{y})$ and M test inputs $X_\star = [\mathbf{x}_\star^1, \ldots, \mathbf{x}_\star^M]^T$, the regression task can be addressed by using GP in the following Bayesian manner. Specifically, predicting the test output vector $\mathbf{y}_\star$ and the model parameters Θ can be interpreted to learn the predictive distribution over $\mathbf{y}_\star$ and Θ,

$$p(\mathbf{y}_\star, \Theta | X_\star, X, \mathbf{y}) = p(\mathbf{y}_\star | X_\star, X, \mathbf{y}, \Theta) p(\Theta | X, \mathbf{y}). \tag{10}$$

Parameter learning by $p(\Theta|X, \mathbf{y})$: As $p(\Theta|X, \mathbf{y}) \propto p(\mathbf{y}|X, \Theta)$, a popular approach to infer Θ is to minimize the negative log likelihood with gradient optimization [1]

$$\begin{aligned} &-\log p(\mathbf{y}|X, \Theta) \\ &= \frac{1}{2}\mathbf{y}^T[K(X, X) + \sigma_y^2 I]^{-1}\mathbf{y} + \frac{1}{2}\log|K(X, X) + \sigma_y^2 I| + \frac{n}{2}\log 2\pi, \end{aligned} \tag{11}$$

where $K(X, X)$ is constructed by using Eq. (6). In practice, this approach works well [1] and thus we apply it for the standard GP regression in this paper.

Output inference by $p(\mathbf{y}_\star|X_\star, X, \mathbf{y}, \Theta)$: After Θ is learned, one can make prediction at test inputs $X_\star$ by using $p(\mathbf{y}_\star|X_\star, X, \mathbf{y}, \Theta)$. *Firstly*, due to the fact that $f(\mathbf{x}) \sim \mathcal{GP}(0, k(\mathbf{x}, \mathbf{x}'))$ and the noise in Eq. (9) is Gaussian, the joint distribution over the training outputs $\mathbf{y}$ and latent function values at test inputs $f(X_\star) = [f(\mathbf{x}_\star^1), \ldots, f(\mathbf{x}_\star^M)]^T$ is Gaussian [1],

$$p(\mathbf{y}, f(X_\star)|X, X_\star, \Theta) = \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K(X, X) + \sigma_y^2 I & K(X, X_\star) \\ K(X_\star, X) & K(X_\star, X_\star) \end{bmatrix}\right), \tag{12}$$

where $K(X_\star, X)$ and $K(X_\star, X_\star)$ are constructed by using $X_\star$ and X in Eq. (6). *Secondly*, based on the conditional property of a joint multivariate Gaussian distribution (Appendix A of [1]), one can obtain that the conditional distribution $p(f(X_\star)|X_\star, X, \mathbf{y}, \Theta)$ is Gaussian,

$$p(f(X_\star)|X_\star, X, \mathbf{y}, \Theta) = \mathcal{N}(\mu_\star, \Sigma_\star), \tag{13}$$

with the following mean vector $\mu_\star$ and covariance matrix $\Sigma_\star$ which are computed from the corresponding joint distribution $p(\mathbf{y}, f(X_\star)|X, X_\star, \Theta)$ in Eq. (12) [1],

$$\mu_\star = K(X_\star, X)[K(X, X) + \sigma_y^2 I]^{-1}\mathbf{y}, \tag{14}$$

[1] Note that it is straightforward to choose other mean functions to do mathematical derivations without difficulties.

Please cite this article as: Y. Wang, B. Chaib-draa, Bayesian inference for time-varying applications: Particle-based Gaussian process approaches, Neurocomputing (2017), http://dx.doi.org/10.1016/j.neucom.2017.01.072

$$\Sigma_\star = K(X_\star, X_\star) - K(X_\star, X)[K(X, X) + \sigma_y^2 I]^{-1} K(X_\star, X)^T. \quad (15)$$

Finally, because $p(f(X_\star)|X_\star, X, \mathbf{y}, \Theta)$ in Eq. (13) and the noise in Eq. (9) are Gaussian, the target distribution $p(\mathbf{y}_\star|X_\star, X, \mathbf{y}, \Theta)$ is also Gaussian,

$$p(\mathbf{y}_\star|X_\star, X, \mathbf{y}, \Theta) = \mathcal{N}(\mu_\star, \Sigma_\star + \sigma_y^2 I). \quad (16)$$

Note that, the computation complexity of this GP regression is mainly governed by the covariance matrix inversion of T training pairs in Eqs. (11), (14), and (15). This computation results in $\mathcal{O}(T^3)$ which is often expensive when T is beyond a few thousand [1]. Next, we briefly discuss a conventional way to model temporal data with the standard GP regression.

2.3. Temporal data modeling with GP regression

For a temporal data set, the input–output pairs are *sequentially-ordered*, i.e., the tth input–output pair $D_t = (\mathbf{x}_t, y_t)$ is made available at time t. Hence, in this work we mainly focus on *online prediction*, which is one of the most important tasks in temporal data modeling. Specifically, our goal is to predict the current output y_t given the current input $\mathbf{x}_t$ and the past input–output pairs $(\mathbf{x}_{1:t-1}, y_{1:t-1})$. In the following, we discuss a conventional way to address this task with the standard GP regression.

Suppose that the output at each time step is generated from $y = f(\mathbf{x}) + \epsilon_y$ in Eq. (9), the online prediction task can be interpreted as a predictive distribution within the setting of GP regression,

$$p(y_t|\mathbf{x}_t, \mathbf{x}_{1:t-1}, y_{1:t-1}, \Theta), \quad (17)$$

which can be addressed by Eq. (16) when $X = \mathbf{x}_{1:t-1}$, $\mathbf{y} = y_{1:t-1}$, $X_\star = \mathbf{x}_t$, $\mathbf{y}_\star = y_t$. However, there are mainly two limitations in this solution. *Firstly*, the standard GP regression addresses the online prediction task in an inefficiently-offline way. In this case, the computational complexity would be cubic in t and keep growing when t increases. *Secondly*, the model parameter Θ in GP is time-invariant. As a result, it is difficult for this GP solution to model non-stationarity and heteroscedasticity in time-varying applications.

To make online prediction with high efficiency and accuracy, we propose two novel particle-based GP approaches in the next section to learn the distinct characteristics of temporal data sets over time.

3. Our particle-based Gaussian process approaches

In this section, we describe our particle-based GP approaches in detail. *Firstly*, we design two novel state space models by using GP, to take the sequential order of temporal data into account. *Secondly*, we derive two sequential-Monte-Carlo-inspired particle approaches on our state space models, to perform online prediction efficiently and capture time-varying characteristics (such as non-stationarity and heteroscedasticity) accurately. *Finally*, we take advantage of backward smoothing to effectively infer the history of latent function values given the history of temporal input–output pairs.

3.1. State space models

To take the sequential order of temporal input–output pairs into account, we make use of GP to develop two novel state space models (SSMs) in which the latent state at t is the function value $f(\mathbf{x}_t) = f_t$. Furthermore, to reflect temporal characteristics such as non-stationarity and heteroscedasticity, the model parameters in our SSMs are time-varying.

3.1.1. Time-varying observation model

Based on Eq. (9), the observation model of our two SSMs is

$$y_t = f(\mathbf{x}_t) + \epsilon_{y,t}, \quad \epsilon_{y,t} \sim \mathcal{N}(0, \sigma_{y,t}^2). \quad (18)$$

One can see that, different from Eq. (9), the Gaussian noise $\epsilon_{y,t}$ in our observation model is assumed to be time-varying in order to capture heteroscedasticity of temporal data.

3.1.2. GP-based dynamical transition

To make efficient online prediction, we propose to construct a dynamical transition from $f(\mathbf{x}_{t-1}) = f_{t-1}$ to $f(\mathbf{x}_t) = f_t$ by taking advantage of GP. Since $f(\mathbf{x}) \sim \mathcal{GP}(0, k(\mathbf{x}, \mathbf{x}') = \sigma_f^2 k_\ell(\mathbf{x}, \mathbf{x}'))$, one can obtain that the joint distribution $p(f_t, f_{t-1}|\mathbf{x}_t, \mathbf{x}_{t-1}, \sigma_f^2, \ell)$ is Gaussian,

$$p(f_t, f_{t-1}|\mathbf{x}_t, \mathbf{x}_{t-1}, \sigma_f^2, \ell) = \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} \sigma_f^2 k_\ell(\mathbf{x}_t, \mathbf{x}_t) & \sigma_f^2 k_\ell(\mathbf{x}_t, \mathbf{x}_{t-1}) \\ \sigma_f^2 k_\ell(\mathbf{x}_{t-1}, \mathbf{x}_t) & \sigma_f^2 k_\ell(\mathbf{x}_{t-1}, \mathbf{x}_{t-1}) \end{bmatrix}\right). \quad (19)$$

In this case, the conditional distribution $p(f_t|f_{t-1}, \mathbf{x}_t, \mathbf{x}_{t-1}, \sigma_f^2, \ell)$ is also Gaussian (according to Appendix A of [1]),

$$p(f_t|f_{t-1}, \mathbf{x}_t, \mathbf{x}_{t-1}, \sigma_f^2, \ell) = \mathcal{N}(g(\ell) f_{t-1}, \sigma_f^2 q(\ell)), \quad (20)$$

where

$$g(\ell) = \frac{k_\ell(\mathbf{x}_t, \mathbf{x}_{t-1})}{k_\ell(\mathbf{x}_{t-1}, \mathbf{x}_{t-1})}, \quad (21)$$

$$q(\ell) = k_\ell(\mathbf{x}_t, \mathbf{x}_t) - \frac{k_\ell^2(\mathbf{x}_t, \mathbf{x}_{t-1})}{k_\ell(\mathbf{x}_{t-1}, \mathbf{x}_{t-1})}. \quad (22)$$

From the view of Bayesian filtering [37], $p(f_t|f_{t-1}, \mathbf{x}_t, \mathbf{x}_{t-1}, \sigma_f^2, \ell)$ in Eq. (20) is equivalent to the following dynamical transition from f_{t-1} to f_t, with an additive Gaussian noise ϵ_f:

$$f_t = g(\ell) f_{t-1} + \epsilon_f, \quad \epsilon_f \sim \mathcal{N}(0, \sigma_f^2 q(\ell)). \quad (23)$$

This dynamical transition takes the temporal order into account, hence it is suitable for efficient online prediction. However, the parameters σ_f^2 and ℓ in this transition are time-invariant. As a result, it would not effectively model the time-varying non-stationarity (such as sudden function shifts). Hence, we next propose two time-varying transition models, based on Eq. (23). The parameters in our two transition models are with different time-varying assumptions, in order to balance the trade-off between efficiency and accuracy when modeling the temporal non-stationarity.

3.1.3. Time-varying transition model (I)

We propose our time-varying transition model (I) as follows:

$$f_t = g(\ell) f_{t-1} + \epsilon_{f,t}^{(I)}, \quad \epsilon_{f,t}^{(I)} \sim \mathcal{N}(0, \sigma_{f,t}^2 q(\ell)). \quad (24)$$

Different from Eq. (23), our transition model (I) is time-varying due to the fact that $\sigma_{f,t}^2$ in the transition noise $\epsilon_{f,t}^{(I)}$ is time-varying.

There are two important reasons for this design. Firstly, by assuming a time-varying $\sigma_{f,t}^2$, our time-varying transition model (I) can adaptively adjust the transition noise $\epsilon_{f,t}^{(I)}$ to model the non-stationary transition from f_{t-1} to f_t. Secondly, $\sigma_{f,t}^2$ is linearly-separable with $q(\ell)$ in the transition noise $\epsilon_{f,t}^{(I)}$. In this case, an efficient sequential inference framework, which is inspired by particle learning [38], can be developed to learn $\sigma_{f,t}^2$ over time.

Please cite this article as: Y. Wang, B. Chaib-draa, Bayesian inference for time-varying applications: Particle-based Gaussian process approaches, Neurocomputing (2017), http://dx.doi.org/10.1016/j.neucom.2017.01.072

Table 1
Model parameter comparison. (−) Time-invariant, (+) Time-varying.

Model parameters	Standard GP	Our SSM (I)	Our SSM (II)
Amplitude in SE kernel: σ_f^2	(−): σ_f^2	(+): $\sigma_{f,t}^2$	(+): $\sigma_{f,t}^2$
Lengthscale in SE kernel: ℓ	(−): ℓ	(−): ℓ	(+): ℓ_t
Output noise variance: σ_y^2	(−): σ_y^2	(+): $\sigma_{y,t}^2$	(+): $\sigma_{y,t}^2$

3.1.4. Time-varying transition model (II)

Note that, $\sigma_{f,t}^2$ in our time-varying transition model (I) can represent non-stationarity to some degree, but the flexibility of Eq. (24) may be still limited due to the fact that ℓ in $g(\ell)$ and $q(\ell)$ is time-invariant. In fact, ℓ is the length-scale parameter vector of the SE kernel (Eq. (8)), which weights the importance of different input dimensions. In this case, the time-invariant ℓ may not capture the time-varying importance of different input dimensions in temporal data.

Hence, we propose our time-varying transition model (II) as follows:

$$f_t = g(\ell_t) f_{t-1} + \epsilon_{f,t}^{(II)}, \quad \epsilon_{f,t}^{(II)} \sim \mathcal{N}(0, \sigma_{f,t}^2 q(\ell_t)), \tag{25}$$

where ℓ_t is also assumed to be time-varying for non-stationarity. Furthermore, inspired by Rao-Blackwellized particle filtering [39], we propose to design a small Gaussian random walk on the log of model parameters,[2] i.e., $\phi = \log(\Theta) = \log([\sigma_f^2, \ell, \sigma_y^2]^T)$ to develop efficient online inference mechanism later on,

$$\phi_t = \phi_{t-1} + \mathbf{v}_t^{\phi}, \quad \mathbf{v}_t^{\phi} \sim \mathcal{N}(\mathbf{0}, \tau^2 \mathbf{I}), \tag{26}$$

where τ^2 is the noise variance of this random walk, and it is a predefined hyper-parameter in our experiments.

To sum up, we design two GP-constructed state space models (SSMs) to take the distinct time-varying characteristics of temporal data into account. These two SSMs share the same time-varying observation model (Eq. (18)) to learn the temporal heteroscedasticity, while they have different time-varying transition models (Eq. (24) or Eqs. (25) and (26)) to learn the temporal non-stationarity, depending on whether all the model parameters are time-varying. The parameter assumptions of different models are summarized in Table 1. Compared to the standard GP, our GP-constructed SSM (I) and (II) are more suitable to capture the temporal characteristics, with different time-varying assumptions on the model parameters. In the following, we take advantage of our proposed SSMs to develop efficient sequential inference frameworks for online prediction.

3.2. Sequential Bayesian inference

To make efficient online prediction, we propose to design sequential-Monte-Carlo-inspired mechanisms to infer the latent function values and model parameters over time. Specifically, we make use of the conditionally-linear structures in our two GP-constructed SSMs to improve efficiency and accuracy of sequential Bayesian inference, which is motivated by particle learning [38] and Rao-Blackwellized particle filtering [39] respectively.

3.2.1. Our PL-GP approach: particle learning for GP-constructed SSM (I)

As mentioned before, our GP-constructed SSM (I) consists of the time-varying observation model in Eq. (18) and the time-varying transition model (I) in Eq. (24), where $\sigma_{f,t}^2$ and $\sigma_{y,t}^2$ are time-varying but ℓ is time-invariant. Given the initialized ℓ, our SSM (I) becomes a conditionally-linear structure with respect to $f(\mathbf{x}_t) = f_t$, where the time-varying model parameters $\sigma_{f,t}^2$ and $\sigma_{y,t}^2$ are unknown but linearly-separable. According to this fact, we propose a novel sequential-Monte-Carlo-inspired particle approach for our SSM (I), which is based on particle learning (PL) [38], to efficiently update *sufficient statistics* of f_t, $\sigma_{f,t}^2$ and $\sigma_{y,t}^2$ within a *resampling-propagating* framework.

Specifically, we approximate the intractable posterior $p(f_t, \sigma_{f,t}^2, \sigma_{y,t}^2 | \ell, D_{1:t})$ which can be factorized as follows:

$$p(f_t, \sigma_{f,t}^2, \sigma_{y,t}^2 | \ell, D_{1:t}) = p(\sigma_{f,t}^2, \sigma_{y,t}^2 | f_t, \ell, D_{1:t}) p(f_t | \ell, D_{1:t}), \tag{27}$$

where $D_{1:t} = (\mathbf{x}_{1:t}, y_{1:t})$.

(1) *How to infer f_t by $p(f_t|\ell, D_{1:t})$ in Eq. (27)*: Since our SSM (I) is conditionally linear with regard to f_t, the sufficient statistics of f_t are its posterior mean $\mu_{t|t}$ and variance $\Sigma_{t|t}$ [38]. For simplicity, we denote the sufficient statistics of f_t as $S_t^f = (\mu_{t|t}, \Sigma_{t|t})$ and $p(f_t|S_t^f) = \mathcal{N}(\mu_{t|t}, \Sigma_{t|t})$. According to the fact that

$$p(f_t | \ell, D_{1:t}) = \int p(f_t|S_t^f) p(S_t^f | \ell, D_{1:t}) dS_t^f = \mathbb{E}[p(f_t|S_t^f) | \ell, D_{1:t}], \tag{28}$$

we thus are more interested in how to infer $p(S_t^f|\ell, D_{1:t})$ than $p(f_t|\ell, D_{1:t})$. This is mainly because we can directly obtain the estimation of f_t after obtaining the approximation of $p(S_t^f|\ell, D_{1:t})$. Hence we use the Bayes rule to factorize $p(S_t^f|\ell, D_{1:t})$ as follows:

$$p(S_t^f | \ell, D_{1:t}) = \frac{p(S_t^f, \ell, D_{1:t})}{p(\ell, D_{1:t})} \propto p(S_t^f, \ell, D_{1:t}), \tag{29}$$

where $p(S_t^f, \ell, D_{1:t})$ can be computed by integrating S_{t-1}^f, $\sigma_{f,t-1}^2$, $\sigma_{y,t-1}^2$ out of the joint distribution $p(S_t^f, S_{t-1}^f, \sigma_{f,t-1}^2, \sigma_{y,t-1}^2, \ell, D_{1:t})$,

$$\begin{aligned}
&p(S_t^f, \ell, D_{1:t}) \\
&= \iiint p(S_t^f, S_{t-1}^f, \sigma_{f,t-1}^2, \sigma_{y,t-1}^2, \ell, D_{1:t}) dS_{t-1}^f d\sigma_{f,t-1}^2 d\sigma_{y,t-1}^2 \\
&\propto \iiint p(S_t^f | S_{t-1}^f, \sigma_{f,t-1}^2, \sigma_{y,t-1}^2, \ell, D_{1:t}) \\
&\quad p(y_t | \mathbf{x}_t, S_{t-1}^f, \sigma_{f,t-1}^2, \sigma_{y,t-1}^2, \ell, D_{1:t-1}) \\
&\quad \times p(S_{t-1}^f, \sigma_{f,t-1}^2, \sigma_{y,t-1}^2 | \ell, D_{1:t-1}) dS_{t-1}^f d\sigma_{f,t-1}^2 d\sigma_{y,t-1}^2.
\end{aligned} \tag{30}$$

Suppose that, at the $(t-1)$th step, $p(S_{t-1}^f, \sigma_{f,t-1}^2, \sigma_{y,t-1}^2 | \ell, D_{1:t-1})$ in Eq. (30) is approximated by N particles $\{S_{t-1}^f[n], \sigma_{f,t-1}^2[n], \sigma_{y,t-1}^2[n]\}_{n=1}^N$. Then, the posterior $p(S_t^f|\ell, D_{1:t})$ in Eq. (29) can be approximated in the following way, based on the integral in Eq. (30).

First, we compute the weights of those particles at $t-1$ by using the prediction distribution $p(y_t|\mathbf{x}_t, S_{t-1}^f, \sigma_{f,t-1}^2, \sigma_{y,t-1}^2, \ell, D_{1:t-1})$ in Eq. (30),

$$\begin{aligned}
w_{t-1}[n] &= p(y_t | \mathbf{x}_t, S_{t-1}^f[n], \sigma_{f,t-1}^2[n], \sigma_{y,t-1}^2[n], \ell, D_{1:t-1}) \\
&\quad (n = 1, \ldots, N),
\end{aligned} \tag{31}$$

where $p(y_t|\mathbf{x}_t, S_{t-1}^f, \sigma_{f,t-1}^2, \sigma_{y,t-1}^2, \ell, D_{1:t-1})$ can be computed by

$$\begin{aligned}
&p(y_t | \mathbf{x}_t, S_{t-1}^f, \sigma_{f,t-1}^2, \sigma_{y,t-1}^2, \ell, D_{1:t-1}) \\
&= \iint p(y_t|f_t, \sigma_{y,t-1}^2) p(f_t | f_{t-1}, \sigma_{f,t-1}^2, \ell, \mathbf{x}_{t-1:t}) p(f_{t-1}|S_{t-1}^f) df_{t-1} df_t \\
&= \mathcal{N}(g(\ell)\mu_{t-1|t-1}, \quad g^2(\ell)\Sigma_{t-1|t-1} + \sigma_{f,t-1}^2 q(\ell) + \sigma_{y,t-1}^2).
\end{aligned} \tag{32}$$

Then, the weights are normalized as

$$\hat{w}_{t-1}[n] = \frac{w_{t-1}[n]}{\sum_{n=1}^N w_{t-1}[n]} \quad (n = 1, \ldots, N). \tag{33}$$

[2] Note that the random walk is used on $\log(\Theta)$ because of $\Theta > 0$. Directly adding a Gaussian random walk on Θ may lead to the unreasonable $\Theta < 0$.

Please cite this article as: Y. Wang, B. Chaib-draa, Bayesian inference for time-varying applications: Particle-based Gaussian process approaches, Neurocomputing (2017), http://dx.doi.org/10.1016/j.neucom.2017.01.072

With the probabilities proportional to the corresponding normalized weights $\{\hat{w}_{t-1}[n]\}_{n=1}^N$, we draw N new particles $\{S_{t-1}^f[k], \sigma_{f,t-1}^2[k], \sigma_{y,t-1}^2[k]\}_{k=1}^N$ from N old particles $\{S_{t-1}^f[n], \sigma_{f,t-1}^2[n], \sigma_{y,t-1}^2[n]\}_{n=1}^N$. This procedure is called *resampling* which can increase importance of particles to approximate the underlying distribution [37].

Second, after *resampling*, the new particles $\{S_{t-1}^f[k], \sigma_{f,t-1}^2[k], \sigma_{y,t-1}^2[k]\}_{k=1}^N$ at $(t-1)$ are *propagated* to $p(S_t^f|S_{t-1}^f, \sigma_{f,t-1}^2, \sigma_{y,t-1}^2, \ell, D_{1:t})$ in Eq. (30). Due to the conditionally-linear structure in our SSM (I), $p(S_t^f|S_{t-1}^f, \sigma_{f,t-1}^2, \sigma_{y,t-1}^2, \ell, D_{1:t})$ is a Kalman filter [38],

$$\mu_{t|t-1} = g(\ell)\mu_{t-1|t-1}, \tag{34}$$

$$\Sigma_{t|t-1} = g^2(\ell)\Sigma_{t-1|t-1} + \sigma_{f,t-1}^2 q(\ell), \tag{35}$$

$$\mu_{t|t} = \mu_{t|t-1} + (y_t - \mu_{t|t-1})\Sigma_{t|t-1}/(\Sigma_{t|t-1} + \sigma_{y,t-1}^2), \tag{36}$$

$$\Sigma_{t|t} = \Sigma_{t|t-1} - \Sigma_{t|t-1}^2/(\Sigma_{t|t-1} + \sigma_{y,t-1}^2). \tag{37}$$

In this case, we feed $\{S_{t-1}^f[k], \sigma_{f,t-1}^2[k], \sigma_{y,t-1}^2[k]\}_{k=1}^N$ at $(t-1)$ into Eqs. (34)–(37) to obtain the particles $\{S_t^f[k]\}_{k=1}^N$ of the sufficient statistics $S_t^f = (\mu_{t|t}, \Sigma_{t|t})$ at t. Note that, $\{S_t^f[k]\}_{k=1}^N$ is the particle approximation of $p(S_t^f|\ell, D_{1:t})$ in Eq. (29). Consequentially, the posterior $p(f_t|\ell, D_{1:t})$ in Eq. (28) can be approximated as a Gaussian mixture by using $\{S_t^f[k]\}_{k=1}^N$,

$$p(f_t|\ell, D_{1:t}) = \int p(f_t|S_t^f)p(S_t^f|\ell, D_{1:t})dS_t^f \approx \frac{1}{N}\sum_{k=1}^N \mathcal{N}(\mu_{t|t}[k], \Sigma_{t|t}[k]), \tag{38}$$

where N particles $\{f_t[k]\}_{k=1}^N$ of the latent function value f_t can be sampled from this Gaussian mixture, in order to infer $\sigma_{f,t}^2$ and $\sigma_{y,t}^2$ in the following.

(2) *How to infer* $\sigma_{f,t}^2$ *and* $\sigma_{y,t}^2$ *by* $p(\sigma_{f,t}^2, \sigma_{y,t}^2|f_t, \ell, D_{1:t})$ *in* Eq. (27): Inspired by [38,40], we propose to take advantage of the conjugate prior to update the sufficient statistics of $\sigma_{f,t}^2$ and $\sigma_{y,t}^2$ recursively. Suppose that the distributions over $\sigma_{f,t}^2$ and $\sigma_{y,t}^2$ are

$$p(\sigma_{f,t}^2|f_t, \ell, D_{1:t}) = \mathcal{IG}(0.5\alpha_t^f, 0.5\beta_t^f), \tag{39}$$

$$p(\sigma_{y,t}^2|f_t, \ell, D_{1:t}) = \mathcal{IG}(0.5\alpha_t^y, 0.5\beta_t^y), \tag{40}$$

where $\mathcal{IG}$ represents the inverse Gamma distribution. In this case, the sufficient statistics of $\sigma_{f,t}^2$ and $\sigma_{y,t}^2$ are (α_t^f, β_t^f) and (α_t^y, β_t^y). For simplicity, we denote sufficient statistics of all these parameters as $S_t^\theta = (\alpha_t^f, \beta_t^f, \alpha_t^y, \beta_t^y)$. Due to the conditionally-linear structure in our SSM (I), one can obtain the following updates by making use of the conjugate property between a Gaussian distribution and an inverse Gamma distribution [38,40],

$$\alpha_t^f = \alpha_{t-1}^f + 1, \tag{41}$$

$$\beta_t^f = \beta_{t-1}^f + (f_t - g(\ell)f_{t-1})^2/q(\ell), \tag{42}$$

$$\alpha_t^y = \alpha_{t-1}^y + 1, \tag{43}$$

$$\beta_t^y = \beta_{t-1}^y + (y_t - f_t)^2. \tag{44}$$

Then, we can obtain $\{S_t^\theta[k]\}_{k=1}^N$ (the particles of sufficient statistics of model parameters at t), by feeding $\{S_{t-1}^\theta[k]\}_{k=1}^N$, $\{f_{t-1}[k]\}_{k=1}^N$ and $\{f_t[k]\}_{k=1}^N$ into Eqs. (41)–(44). Finally, we use $\{S_t^\theta[k]\}_{k=1}^N$ in Eqs. (39) and (40) to sample the particles of model parameters at t, i.e., $\{\sigma_{f,t}^2[k], \sigma_{y,t}^2[k]\}_{k=1}^N$.

(3) *Summary of our PL-GP approach*: The above-mentioned particle mechanism of our PL-GP approach is illustrated in Alg. 1. Based on our PL-GP approach, one can efficiently approximate $p(f_t, \sigma_{f,t}^2, \sigma_{y,t}^2|\ell, D_{1:t})$ in Eq. (27) via a *resampling-propagating* mechanism. It is worth mentioning that, *online prediction* in Section 2.3 is addressed by Eq. (32) in our PL-GP. Since online prediction is based on sequential Monte Carlo framework, the computation complexity at the tth step of our PL-GP is $\mathcal{O}(tN + td_x)$, where N is the number of the particles, d_x is the dimension of the input vector, tN is the main complexity of particle sampling, and td_x is the main complexity of input vector multiplication in Eqs. (21) and (22). One can see that the computation of our PL-GP is linear to t, while the computation of GP is cubic to t in Section 2.3. This indicates that our PL-GP is more efficient for sequential data modeling. Moreover, unlike Eq. (17) in GP, Eq. (32) in our PL-GP contains the time-varying model parameters $\sigma_{f,t}^2$ and $\sigma_{y,t}^2$. As a result, our PL-GP is more suitable to capture non-stationarity and heteroscedasticity in temporal data than GP. However, ℓ in our PL-GP is time-invariant, which may still restrict the performance of our PL-GP. As mentioned before, our GP-constructed SSM (II) is proposed to relax this limitation, by assuming that ℓ_t is time-varying in our transition model (II) (Eqs. (25) and (26)). Hence, we next design an online inference mechanism for our GP-constructed SSM (II) to make effective prediction.

Algorithm 1 Our PL-GP approach.

1: ———————— *Resampling* ————————
2: Computing the weights of particles at $(t-1)$ by Eqs. (31) and (32).
3: Resampling N new particles at $(t-1)$, i.e.,$\{S_{t-1}^f[k], f_{t-1}[k]\}_{k=1}^N$ and $\{S_{t-1}^\theta[k], \sigma_{f,t-1}^2[k], \sigma_{y,t-1}^2[k]\}_{k=1}^N$from N old particles at $(t-1)$, based on the normalized weights in Eq. (33).
4: ———————— *Propagating* ————————
5: Propagating these resampled particles at $(t-1)$ into Eqs. (34)–(37) and (38) to obtain$\{S_t^f[k]\}_{k=1}^N$ and $\{f_t[k]\}_{k=1}^N$ at t.
6: Propagating these resampled particles at $(t-1)$ into Eqs. (41)–(44) and (39)–(40) to obtain$\{S_t^\theta[k]\}_{k=1}^N$ and $\{\sigma_{f,t}^2[k], \sigma_{y,t}^2[k]\}_{k=1}^N$ at t.

3.2.2. Our RBPF-GP approach: Rao-Blackwellized particle filtering for GP-constructed SSM (II)

Different from the time-varying transition model (I) in Eq. (24), the time-varying transition model (II) in Eqs. (25) and (26) is highly coupled due to the time-varying ℓ_t. In this case, it is not feasible to perform particle learning for our GP-constructed SSM (II), by updating the sufficient statistics of model parameters online. However, the transition from f_{t-1} to f_t in Eq. (25) is still conditionally linear, given all the time-varying model parameters. This fact still allows us to update the sufficient statistics of f_t online when performing sequential Monte Carlo. It mainly refers to the idea of Rao-Blackwellized particle filtering [39] which is based on an *importance sampling-resampling* framework. Specifically, we approximate the intractable posterior $p(f_t, \boldsymbol{\phi}_{1:t}|\boldsymbol{\tau}, D_{1:t})$,

$$p(f_t, \phi_{1:t}|\tau, D_{1:t}) = p(f_t|\phi_{1:t}, \tau, D_{1:t})p(\phi_{1:t}|\tau, D_{1:t}), \tag{45}$$

for our GP-constructed SSM (II) consisting of Eqs. (25), (26) and (18).

(1) *How to infer* f_t *by* $p(f_t|\boldsymbol{\phi}_{1:t}, \boldsymbol{\tau}, D_{1:t})$ *in* Eq. (45): Given $\boldsymbol{\phi}_t$, the transition from f_{t-1} to f_t in Eq. (25) is conditionally linear. In this

Please cite this article as: Y. Wang, B. Chaib-draa, Bayesian inference for time-varying applications: Particle-based Gaussian process approaches, Neurocomputing (2017), http://dx.doi.org/10.1016/j.neucom.2017.01.072

case, the sufficient statistics of f_t are the posterior mean $\mu_{t|t}$ and variance $\Sigma_{t|t}$. As before, we denote $S_t^f = (\mu_{t|t}, \Sigma_{t|t})$ and $p(f_t|S_t^f) = \mathcal{N}(\mu_{t|t}, \Sigma_{t|t})$. Similar to Eq. (28) in our PL-GP approach, we express $p(f_t|\boldsymbol{\phi}_{1:t}, \boldsymbol{\tau}, D_{1:t})$ as

$$p(f_t|\phi_{1:t}, \tau, D_{1:t}) = \int p(f_t|S_t^f)p(S_t^f|\phi_{1:t}, \tau, D_{1:t})dS_t^f, \tag{46}$$

and we mainly focus on how to infer $p(S_t^f|\phi_{1:t}, \tau, D_{1:t})$. In particular, this posterior $p(S_t^f|\phi_{1:t}, \tau, D_{1:t})$ can be computed by

$$p(S_t^f|\phi_{1:t}, \tau, D_{1:t}) = \int p(S_t^f|S_{t-1}^f, \phi_{1:t}, \tau, D_{1:t})p(S_{t-1}^f|\phi_{1:t-1}, \tau, D_{1:t-1})dS_{t-1}^f. \tag{47}$$

Due to the conditionally-linear structure of Eq. (25), $p(S_t^f|S_{t-1}^f, \phi_{1:t}, \tau, D_{1:t})$ in the integral of Eq. (47) is a Kalman filter [39],

$$\mu_{t|t-1} = g(\ell_t)\mu_{t-1|t-1}, \tag{48}$$

$$\Sigma_{t|t-1} = g^2(\ell_t)\Sigma_{t-1|t-1} + \sigma_t^2 q(\ell_t), \tag{49}$$

$$\mu_{t|t} = \mu_{t|t-1} + (y_t - \mu_{t|t-1})\Sigma_{t|t-1}/(\Sigma_{t|t-1} + \sigma_{y,t}^2), \tag{50}$$

$$\Sigma_{t|t} = \Sigma_{t|t-1} - \Sigma_{t|t-1}^2/(\Sigma_{t|t-1} + \sigma_{y,t}^2). \tag{51}$$

In this case, one can approximate $p(S_t^f|\phi_{1:t}, \tau, D_{1:t})$ with N particles $\{S_t^f[n]\}_{n=1}^N$ which are obtained by feeding $\{S_{t-1}^f[n]\}_{n=1}^N$ and $\{\phi_t[n]\}_{n=1}^N$ into Eqs. (48)–(51). Consequentially, the posterior $p(f_t|\boldsymbol{\phi}_{1:t}, \boldsymbol{\tau}, D_{1:t})$ in Eq. (46) can be approximated as a Gaussian mixture by using $\{S_t^f[n]\}_{n=1}^N$,

$$p(f_t|\phi_{1:t}, \tau, D_{1:t}) \approx \frac{1}{N}\sum_{n=1}^{N}\mathcal{N}(\mu_{t|t}[n], \Sigma_{t|t}[n]). \tag{52}$$

(2) *How to infer* $\boldsymbol{\phi}_t$ *by* $p(\boldsymbol{\phi}_{1:t}|\boldsymbol{\tau}, D_{1:t})$ *in* Eq. (45): According to the Bayes rule, this posterior can be factorized as

$$p(\phi_{1:t}|\tau, D_{1:t}) = \frac{p(\phi_{1:t}, \tau, D_{1:t})}{p(\tau, D_{1:t})} \propto p(\phi_{1:t}, \tau, D_{1:t}), \tag{53}$$

where the joint distribution can be expressed by

$$p(\phi_{1:t}, \tau, D_{1:t}) \propto p(y_t|y_{1:t-1}, \phi_{1:t}, \tau, \mathbf{x}_{1:t}) p(\phi_t|\phi_{t-1}, \tau)p(\phi_{1:t-1}|\tau, D_{1:t-1}), \tag{54}$$

and $p(\phi_t|\phi_{t-1}, \tau) = \mathcal{N}(\phi_{t-1}, \tau^2\mathbf{I})$ which is obtained from Eq. (26). In the following, we approximate the posterior $p(\boldsymbol{\phi}_{1:t}|\boldsymbol{\tau}, D_{1:t})$ by taking advantage of Bayesian factorization in Eq. (54).

First, suppose that N particles of model parameters at the $(t-1)$th step are $\{\phi_{t-1}[n]\}_{n=1}^N$, which can be obtained from $p(\phi_{1:t-1}|\tau, D_{1:t-1})$ in Eq. (54). One can feed $\{\phi_{t-1}[n]\}_{n=1}^N$ into the conditional distribution $p(\phi_t|\phi_{t-1}, \tau)$ and sample N particles of model parameters at the tth step, $\{\phi_t[n]\}_{n=1}^N$.

Second, we weight these particles $\{\phi_t[n]\}_{n=1}^N$ according to the prediction distribution $p(y_t|y_{1:t-1}, \phi_{1:t}, \tau, \mathbf{x}_{1:t})$ in Eq. (54),

$$w_t[n] = p(y_t|y_{1:t-1}, \phi_{1:t}[n], \tau, \mathbf{x}_{1:t}), \tag{55}$$

where $p(y_t|y_{1:t-1}, \phi_{1:t}, \tau, \mathbf{x}_{1:t})$ can be computed by

$$\begin{aligned} &p(y_t|y_{1:t-1}, \phi_{1:t}, \tau, \mathbf{x}_{1:t}) \\ &= \iint p(y_t|f_t, \phi_t)p(f_t|f_{t-1}, \phi_t, \mathbf{x}_{t-1:t}) \\ &\quad p(f_{t-1}|\phi_{1:t-1}, \tau, D_{1:t-1})df_t df_{t-1} \\ &= \mathcal{N}(\mu_{t|t-1}, \Sigma_{t|t-1} + \sigma_{y,t}^2), \end{aligned} \tag{56}$$

and $\mu_{t|t-1}$, $\Sigma_{t|t-1}$ are computed by Eqs. (48) and (49). This weighting procedure is called *importance sampling*, since the importance of the drawn $\{\phi_t[n]\}_{n=1}^N$ is weighted by using the current y_t.

Finally, we compute the normalized weights of these particles by

$$\hat{w}_t[n] = \frac{w_t[n]}{\sum_{n=1}^N w_t[n]} \quad (n = 1, \ldots, N). \tag{57}$$

With the probabilities proportional to $\{\hat{w}_t[n]\}_{n=1}^N$, we draw N new particles for S_t^f and $\boldsymbol{\phi}_t$ to increase importance of particles for the next step. This procedure is called *resampling* as mentioned before.

(3) *Summary of our RBPF-GP approach*: The above-mentioned particle mechanism of our RBPF-GP approach is illustrated in Algorithm 2. Based on our RBPF-GP approach, one can efficiently approximate $p(f_t, \boldsymbol{\phi}_{1:t}|\boldsymbol{\tau}, D_{1:t})$ in Eq. (45) via an *importance sampling-resampling* mechanism. It is worth mentioning that, *online prediction* in Section 2.3 is addressed by using Eq. (56) in our RBPF-GP. Based on sequential Monte Carlo, the computation complexity at the tth step of our RBPF-GP is $O(tN + tNd_x)$, where N is the number of the particles, d_x is the dimension of the input vector, tN is the main complexity of particle sampling. Note that, the computation difference between our RBPF-GP and PL-GP refers to the input vector multiplication in Eqs. (21) and (22), where RBPF-GP is tNd_x while PL-GP is td_x. This is mainly because that ℓ is time-invariant in our PL-GP but time-varying in our RBPF-GP. Hence, the input vector multiplication is only implemented once at each time step of our PL-GP, while this operation needs to be implemented for N particles of ℓ_t at each time step of our RBPF-GP. However, compared to our PL-GP, our RBPF-GP may achieve a better prediction accuracy by learning ℓ_t over time.

Algorithm 2 Our RBPF-GP approach.

1: ———————— *Importance Sampling* ————————————
2: Propagating $\{\phi_{t-1}[n]\}_{n=1}^N$ to the conditional distribution $p(\phi_t|\phi_{t-1}, \tau)$ in Eq. (54) to obtain $\{\phi_t[n]\}_{n=1}^N$.
3: Propagating $\{\phi_t[n]\}_{n=1}^N$ and $\{S_{t-1}^f[n]\}_{n=1}^N$ to Eqs. (48)–(51) to obtain$\{S_t^f[n]\}_{n=1}^N$.
4: Computing the weights of $\{\phi_t[n], S_t^f[n]\}_{n=1}^N$ by Eq. (55).
5: ———————— *Resampling* ————————————
6: Resampling N new particles at t from $\{\phi_t[n], S_t^f[n]\}_{n=1}^N$, based on the normalized weights in Eq. (57).

3.2.3. Backward smoothing

So far, our PL-GP and RBPF-GP approaches are mainly designed for *online prediction* in Section 2.3. One may be also interested in how to estimate the history of latent function values given the history of observations, i.e., $p(f_{1:T}, \sigma_{f,1:T}^2, \sigma_{y,1:T}^2|\ell, D_{1:T})$ for our SSM (I) and $p(f_{1:T}, \boldsymbol{\phi}_{1:T}|\boldsymbol{\tau}, D_{1:T})$ for our SSM (II), where T is the total number of time steps for a temporal data set. This task refers to a backward smoothing task [37,41]. Fortunately, the standard backward smoothing mechanism [42] can be straightforwardly implemented to our PL-GP and RBPF-GP without any difficulties. In order to avoid unnecessary content redundancy in our paper, we recommend [42] for further reading about backward smoothing.

In the following, we show the effectiveness of our proposed PL-GP and RBPF-GP approaches on a number of challenging temporal data sets, in comparison with several relevant GP approaches.

Please cite this article as: Y. Wang, B. Chaib-draa, Bayesian inference for time-varying applications: Particle-based Gaussian process approaches, Neurocomputing (2017), http://dx.doi.org/10.1016/j.neucom.2017.01.072

4. Experiments

In this section, we evaluate our PL-GP and RBPF-GP approaches on five challenging temporal data sets: *synthetic, motor,*[3] *heart rate,*[4] *S&P,*[5] and *bike sharing.*[6] The *synthetic* data set consists of 1000 (time/output) pairs which are generated from

$$y_t = \begin{cases} -30 + \epsilon_t^1, \quad \epsilon_t^1 \sim \mathcal{N}(0,1), \quad (t = 0.01 : 0.01 : 2); \\ 50\sin(0.5\pi t) + \epsilon_t^2, \quad \epsilon_t^2 \sim \mathcal{N}(0,9), \quad (t = 2.01 : 0.01 : 5); \\ 20\cos(\pi t + 0.5\pi) + \epsilon_t^3, \quad \epsilon_t^3 \sim \mathcal{N}(0,100), \\ \qquad (t = 5.01 : 0.01 : 10). \end{cases} \tag{58}$$

The *motor* data set consists of 94 (time/motor acceleration) pairs. The *heart rate* data set consists of 300 (time/heart rate) pairs. The *S&P* data set consists of 500 (input/output) pairs which are weekly recorded from 2000-01-03 to 2009-08-03. Its input vector is 4-dimension, i.e., time, *S&P 100 index, S&P 400 Midcap index, S&P 500 index*; the output is *volatility S&P 500*. The *bike sharing* data set consists of 731 (input/output) pairs from 2011 to 2012, where its input vector is 4-dimension, i.e., normalized temperature in Celsius, normalized feeling temperature in Celsius, normalized humidity, normalized wind speed; the output is the count of total rental bikes in Capital Bikeshare System.[7] Note that, all these temporal data sets exhibit time-varying non-stationarity and/or heteroscedasticity.

4.1. Online prediction

We first evaluate our PL-GP and RBPF-GP for *online prediction*, according to Eq. (32) for our PL-GP and Eq. (56) for our RBPF-GP. Unless otherwise stated, the number of particles N in our PL-GP and our RBPF-GP is 200 for all data sets. The pre-defined τ in our RBPF-GP is 0.05/0.1/0.05/0.2/0.05 for *synthetic*/*motor*/*heart rate*/*S&P*/*bike sharing*. The model parameters $\Theta = [\sigma^2, \ell, \sigma_y^2]^T$ in all related approaches are initialized by minimizing the negative log likelihood (Eq. (11)) of the first 300/50/200/200/300 temporal data pairs in *synthetic*/*motor*/*heart rate*/*S&P*/*bike sharing*, with the standard gradient optimization in GPML toolbox.[8] The evaluation of online prediction is based on the rest data pairs of all data sets, where the prediction performance is evaluated by the mean negative log probability (MNLP) [1,17]. Since MNLP penalizes both model uncertainty and inference inconsistency, it is a reliable evaluation criterion for non-stationarity and heteroscedasticity [1,17]. Finally, for all data sets, we run all the methods 20 times and report the average MNLP.

4.1.1. Time-varying properties of our approaches

As mentioned in Section 3, both of σ_f^2 and σ_y^2 are time-varying in our PL-GP, while all of σ_f^2, ℓ and σ_y^2 are time-varying in our RBPF-GP. To evaluate the time-varying properties of our approaches, we implement two baselines for our PL-GP, where either σ_f^2 or σ_y^2 is set to be time-varying. Furthermore, we implement six baselines for our RBPF-GP, where one or two of σ_f^2, ℓ and σ_y^2 are set to be time-varying.

The results are shown in Table 2. First, for both PL-GP and RBPF-GP, the prediction performance is getting better, when more parameters are set to be time-varying. This illustrates that time-varying parameters can help to capture the distinct temporal characteristics in these time-varying data sets. Second, for the same time-varying parameter setting, RBPF-GP generally outperforms PL-GP. It may be credited to the utility of Rao-Blackwellized particle filtering in our RBPF-GP. Finally, RBPF-GP with our proposed setting (σ_f^2, ℓ, σ_y^2 are time-varying) achieves a better performance than PL-GP with our proposed setting (σ_f^2, σ_y^2 are time-varying), with assistance of the time-varying ℓ.

Table 2
The evaluation (MNLP) for time-varying properties of our approaches. Note that, both of σ_f^2 and σ_y^2 are time-varying in our PL-GP, while all of σ_f^2, ℓ and σ_y^2 are time-varying in our RBPF-GP. Hence, we implement two baselines for our PL-GP, where either σ_f^2 or σ_y^2 is set to be time-varying. Furthermore, we implement six baselines for our RBPF-GP, where one or two of σ_f^2, ℓ and σ_y^2 are set to be time-varying.

Methods	σ_f^2 ℓ σ_y^2	*Synthetic*	*Motor*	*Heart rate*	*S&P*	*Bike sharing*
PL-GP$_{(\sigma_f^2)}$	√ × ×	9.31	11.37	3.640	4.33	17.34
PL-GP$_{(\sigma_y^2)}$	× × √	9.83	11.68	4.648	4.39	17.49
Our PL-GP	√ × √	**8.08**	**10.35**	**3.623**	**4.26**	**17.27**
RBPF-GP$_{(\sigma_f^2)}$	√ × ×	7.93	11.22	4.661	4.25	16.88
RBPF-GP$_{(\ell)}$	× √ ×	7.91	10.66	4.828	4.35	16.84
RBPF-GP$_{(\sigma_y^2)}$	× × √	7.83	10.20	4.690	4.55	16.82
RBPF-GP$_{(\sigma_f^2,\ell)}$	√ √ ×	7.86	10.56	3.668	4.25	16.84
RBPF-GP$_{(\sigma_f^2,\sigma_y^2)}$	√ × √	7.71	10.09	3.671	4.21	16.81
RBPF-GP$_{(\ell,\sigma_y^2)}$	× √ √	7.72	10.19	3.780	4.30	16.81
Our RBPF-GP	√ √ √	**7.58**	**9.96**	**3.646**	**4.13**	**16.80**

Table 3
Comparison (MNLP) with related GP approaches for online prediction.

Methods	*Synthetic*	*Motor*	*Heart rate*	*S&P*	*Bike sharing*
ARGP	14.33	11.61	3.757	6.61	17.92
NHARGP	9.14	10.43	3.640	5.38	17.36
LGP	10.27	10.63	3.720	4.54	17.22
SOGP	13.27	10.94	3.751	4.80	18.03
KRLST	8.12	10.38	3.730	4.71	17.38
GP-PF	11.56	11.33	3.795	4.85	18.02
MPGP	12.64	11.07	3.742	4.36	17.38
DGP	8.32	10.80	3.779	4.67	17.87
Our PL-GP	8.08	10.35	**3.623**	4.26	17.27
Our RBPF-GP	**7.58**	**9.96**	3.646	**4.13**	**16.80**

4.1.2. Comparison with related GP approaches

Next, we compare our PL-GP and RBPF-GP with a number of related online GP approaches, including the autoregressive GP (ARGP) which is trained on pairs of (y_t, y_{t+1}) within the standard GP regression framework; the high-order ARGP with a non-stationary neural network kernel in [1] (NHARGP), where the order is 10/10/20/20/20 for *synthetic*/*motor*/*heart rate*/*S&P*/*bike sharing* to balance the tradeoff between computation and accuracy; local GP (LGP) [2]; sparse online GP (SOGP) [8]; kernel recursive least-squares tracker (KRLST) [4]; GP particle filter (GP-PF) [18]; marginalized particle GP (MPGP) [19]; and deep GP (DGP) with sequential inference mechanism [10].

To be fair, for LGP, the prediction at the current step is based on all the local experts generated until this step (due to online construction of GP experts in [2]). For SOGP and KRLST, the prediction for the current step is based on all the data until this step without sparsification. The reason is that we tend to use the best results of these online GPs. For GP-PF and MPGP, the latent state is f_t, similar to our approaches. For DGP, the number of latent layers is one and the dimension of this latent layer is equal to the dimension of input vector. Note that, the sequential inference framework of DGP in [10] is based on sequential Monte Carlo sampling by using particle filter, and hence this DGP can be straightforwardly applied for on-

[3] http://www.stat.cmu.edu/~larry/all-of-statistics/=data/motor.dat.
[4] http://www-psych.stanford.edu/~andreas/Time-Series/SantaFe.html.
[5] http://finance.yahoo.com/stock-center/.
[6] http://archive.ics.uci.edu/ml/datasets/Bike+Sharing+Dataset.
[7] http://www.capitalbikeshare.com/system-data.
[8] http://www.gaussianprocess.org/gpml/code/matlab/doc/.

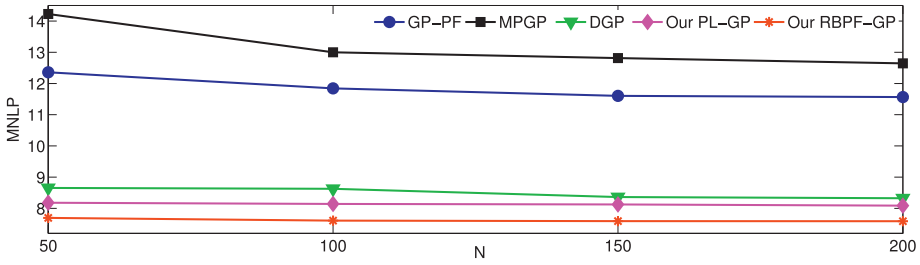


Fig. 1. MNLP (*synthetic*) as a function of the number of particles *N*.

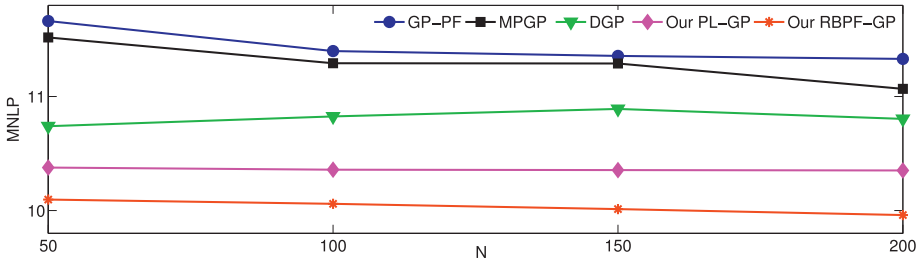


Fig. 2. MNLP (*motor*) as a function of the number of particles *N*.

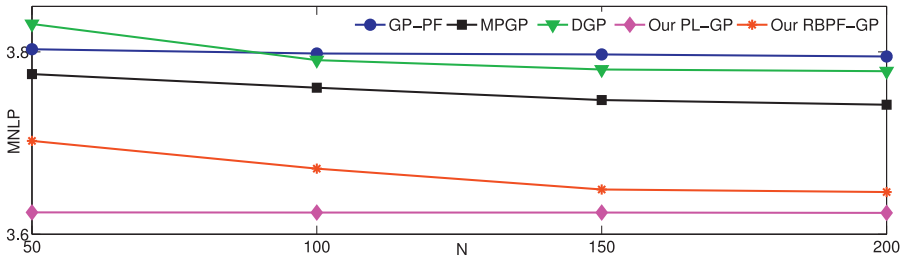


Fig. 3. MNLP (*heart rate*) as a function of the number of particles *N*.

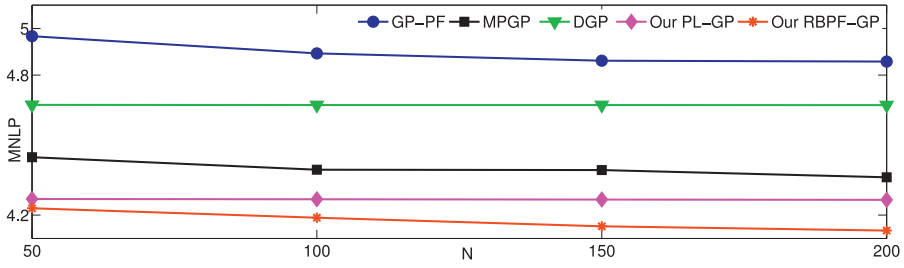


Fig. 4. MNLP (*S&P*) as a function of the number of particles *N*.

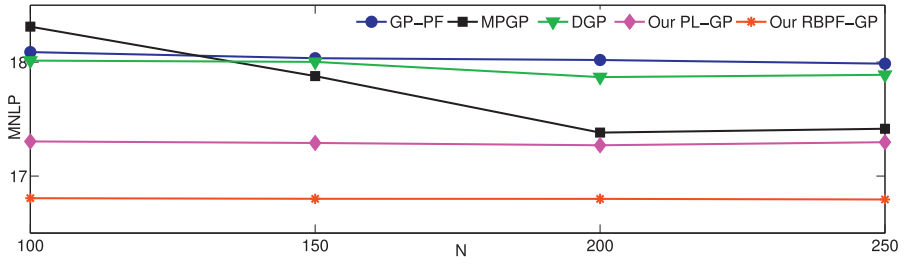
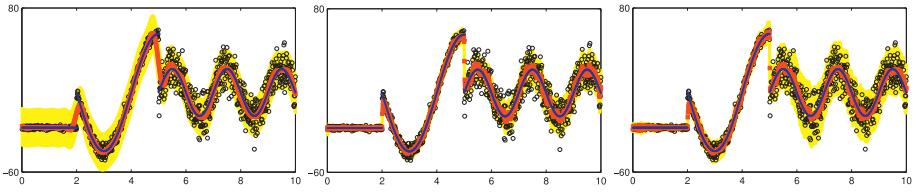


Fig. 5. MNLP (*bike sharing*) as a function of the number of particles *N*.

line prediction without any difficulties. Furthermore, the number of particles *N* in all the particle-based approaches (GP-PF, MPGP, DGP) is 200, which is the same as the one in our PL-GP and RBPF-GP for fairness.

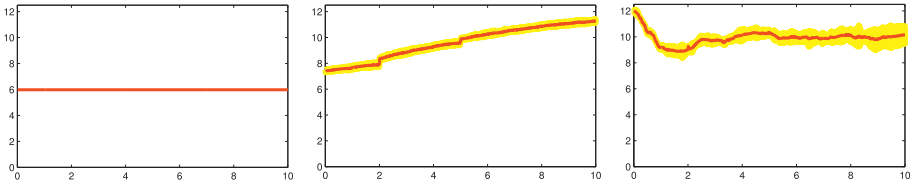
The results are shown in Table 3. One can see that, our PL-GP and RBPF-GP consistently outperform other online GP approaches for all the time-varying data sets. This illustrates that our approaches can successfully capture non-stationarity and/or heteroscedasticity in these time-varying applications, compared to other online GP approaches. Moreover, our RBPF-GP achieves a better accuracy than our PL-GP for most cases, which is mainly due

Please cite this article as: Y. Wang, B. Chaib-draa, Bayesian inference for time-varying applications: Particle-based Gaussian process approaches, Neurocomputing (2017), http://dx.doi.org/10.1016/j.neucom.2017.01.072



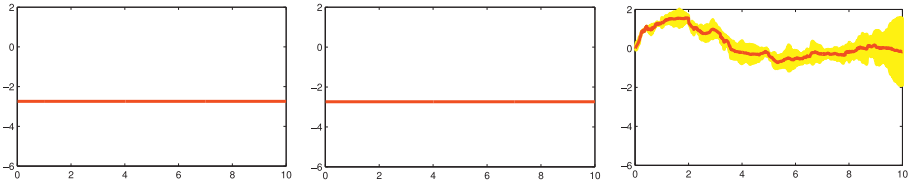
(a) GP (b) our PL-GP-S (c) our RBPF-GP-S

Fig. 6. Posterior of $y_{1:T}$ (*synthetic*). The observations are the black circles and the ground truth of latent function is the blue line. We show the estimated mean $\pm$ 2$\times$ standard deviation (red line with yellow interval) of GP, our PL-GP-S and our RBPF-GP-S. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)



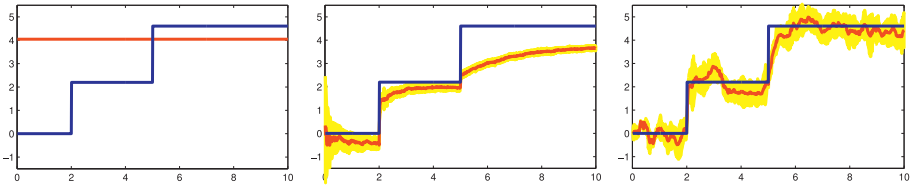
(a) GP (b) our PL-GP-S (c) our RBPF-GP-S

Fig. 7. Posterior of $\log(\sigma_f^2)$ over time (*synthetic*). Note that $\log(\sigma_f^2)$ is time-invariant in GP, hence the estimated $\log(\sigma_f^2)$ in GP is a constant over time (red line in (a)). On the contrary, $\log(\sigma_f^2)$ is time-varying in our approaches. We thus show the estimated mean $\pm$ 2$\times$ standard deviation of $\log(\sigma_f^2)$ in our PL-GP-S and our RBPF-GP-S (red line with yellow interval in (b) and (c)). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)



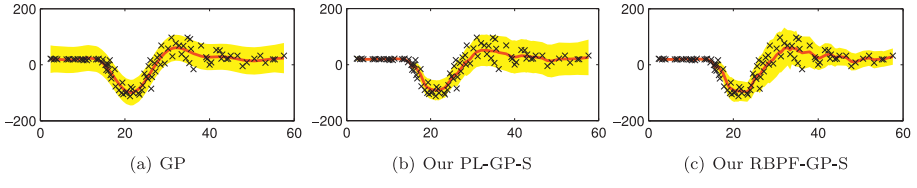
(a) GP (b) our PL-GP-S (c) our RBPF-GP-S

Fig. 8. Posterior of $\log(\ell)$ over time (*synthetic*). Note that $\log(\ell)$ is time-invariant in GP and our PL-GP-S, hence the estimated $\log(\ell)$ in GP and our PL-GP-S is a constant over time (red line in (a) and (b)). On the contrary, $\log(\ell)$ is time-varying in our RBPF-GP-S. We thus show the estimated mean $\pm$ 2$\times$ standard deviation of $\log(\ell)$ in our RBPF-GP-S (red line with yellow interval in (c)). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)



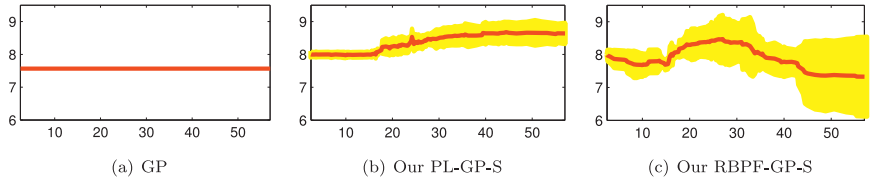
(a) GP (b) our PL-GP-S (c) our RBPF-GP-S

Fig. 9. Posterior of $\log(\sigma_y^2)$ over time (*synthetic*). The ground truth of $\log(\sigma_y^2)$ is the blue line. Note that $\log(\sigma_y^2)$ is time-invariant in GP, hence the estimated $\log(\sigma_y^2)$ in GP is a constant over time (red line in (a)). On the contrary, $\log(\sigma_y^2)$ is time-varying in our approaches. We thus show the estimated mean $\pm$ 2$\times$ standard deviation of $\log(\sigma_y^2)$ in our PL-GP-S and our RBPF-GP-S (red line with yellow interval in (b) and (c)). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)



(a) GP (b) Our PL-GP-S (c) Our RBPF-GP-S

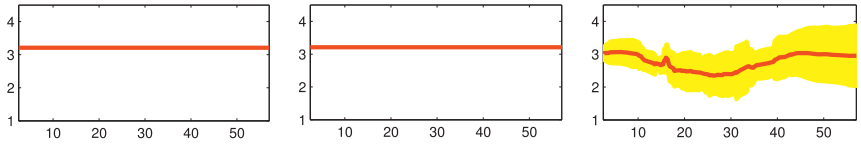
Fig. 10. Posterior of $y_{1:T}$ (*motor*). The observations are the black crosses. We show the estimated mean $\pm$ 2$\times$ standard deviation (red line with yellow interval) of GP, our PL-GP-S and our RBPF-GP-S. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)



(a) GP (b) Our PL-GP-S (c) Our RBPF-GP-S

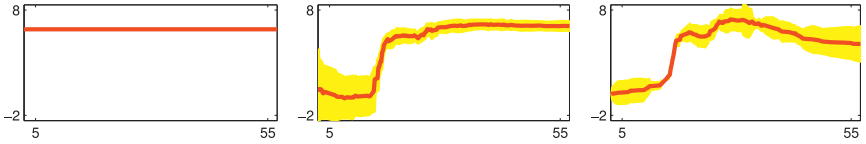
Fig. 11. Posterior of $\log(\sigma_f^2)$ over time (*motor*). The time-invariant estimation by GP is the red line in (a). The time-varying estimations by our PL-GP-S and our RBPF-GP-S are the red lines with yellow intervals in (b) and (c). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Please cite this article as: Y. Wang, B. Chaib-draa, Bayesian inference for time-varying applications: Particle-based Gaussian process approaches, Neurocomputing (2017), http://dx.doi.org/10.1016/j.neucom.2017.01.072



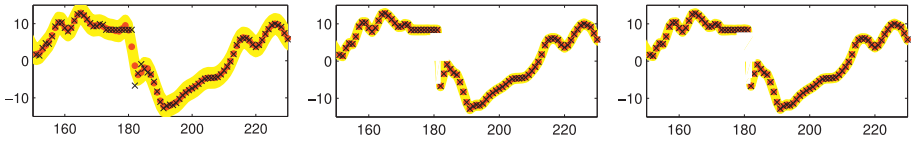
(a) GP (b) Our PL-GP-S (c) Our RBPF-GP-S

Fig. 12. Posterior of $\log(\ell)$ over time (*motor*). Note that $\log(\ell)$ is time-invariant in GP and our PL-GP-S. Hence, the estimations by these two approaches are the red lines in (a) and (b). The time-varying estimation by our RBPF-GP-S is the red line with yellow interval in (c). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)



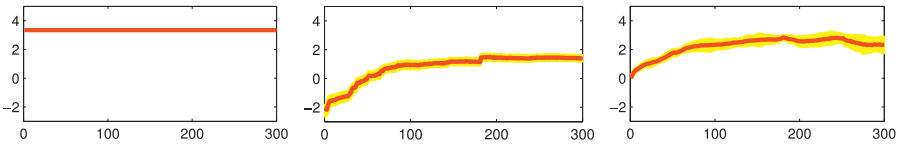
(a) GP (b) Our PL-GP-S (c) Our RBPF-GP-S

Fig. 13. Posterior of $\log(\sigma_y^2)$ over time (*motor*). The time-invariant estimation by GP is the red line in (a). The time-varying estimations by our PL-GP-S and our RBPF-GP-S are the red lines with yellow intervals in (b) and (c). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)



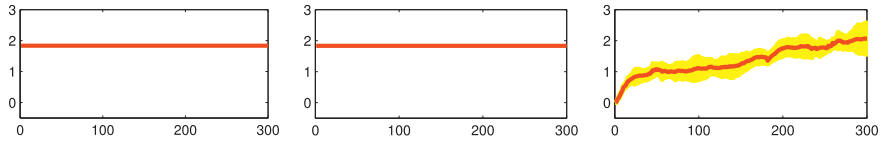
(a) GP (b) Our PL-GP-S (c) Our RBPF-GP-S

Fig. 14. Posterior of $y_{1:T}$ (*heart rate*). The notations are the same as Fig. 10. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)



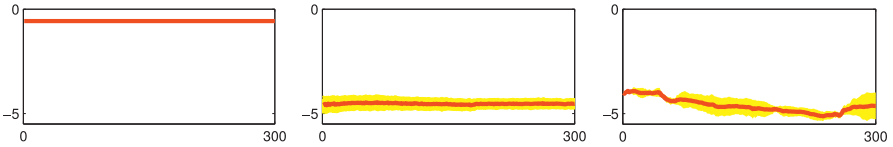
(a) GP (b) Our PL-GP-S (c) Our RBPF-GP-S

Fig. 15. Posterior of $\log(\sigma_f^2)$ over time (*heart rate*). The notations are the same as Fig. 11. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)



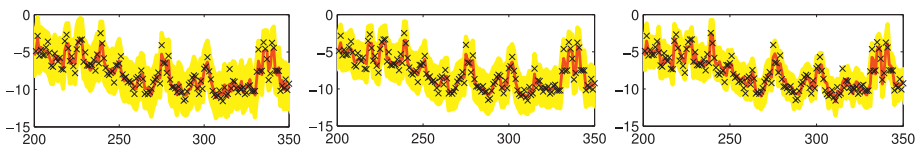
(a) GP (b) Our PL-GP-S (c) Our RBPF-GP-S

Fig. 16. Posterior of $\log(\ell)$ over time (*heart rate*). The notations are the same as Fig. 12. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)



(a) GP (b) Our PL-GP-S (c) Our RBPF-GP-S

Fig. 17. Posterior of $\log(\sigma_y^2)$ over time (*heart rate*). The notations are the same as Fig. 13. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)



(a) GP (b) Our PL-GP-S (c) Our RBPF-GP-S

Fig. 18. Posterior of $y_{1:T}$ (*S&P*). The notations are the same as Fig. 10. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Please cite this article as: Y. Wang, B. Chaib-draa, Bayesian inference for time-varying applications: Particle-based Gaussian process approaches, Neurocomputing (2017), http://dx.doi.org/10.1016/j.neucom.2017.01.072

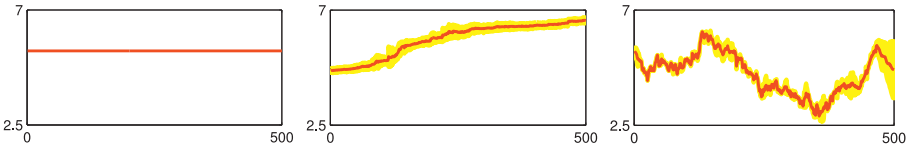


Fig. 19. Posterior of $\log(\sigma_f^2)$ over time (*S&P*). The notations are the same as Fig. 11. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

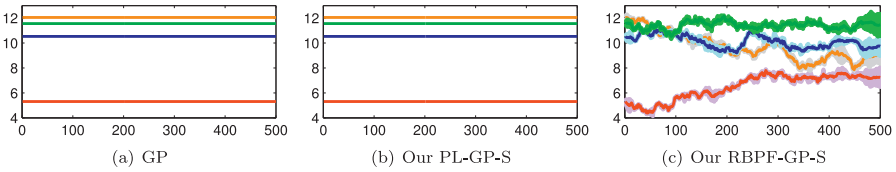


Fig. 20. Posterior of $\log(\ell)$ over time (*S&P*). Since the input of *S&P* is four dimension, the length-scale ℓ is a four dimension vector in all approaches, where different colors represent different dimensions in (a)–(c). Furthermore, ℓ is time-invariant in GP and our PL-GP-S, hence the estimations of these two approaches are lines in (a)–(b). The time-varying estimations of our RBPF-GP-S are lines with their corresponding intervals in (c). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

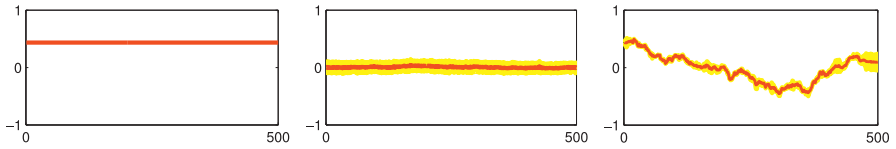


Fig. 21. Posterior of $\log(\sigma_y^2)$ over time (*S&P*). The notations are the same as Fig. 13. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

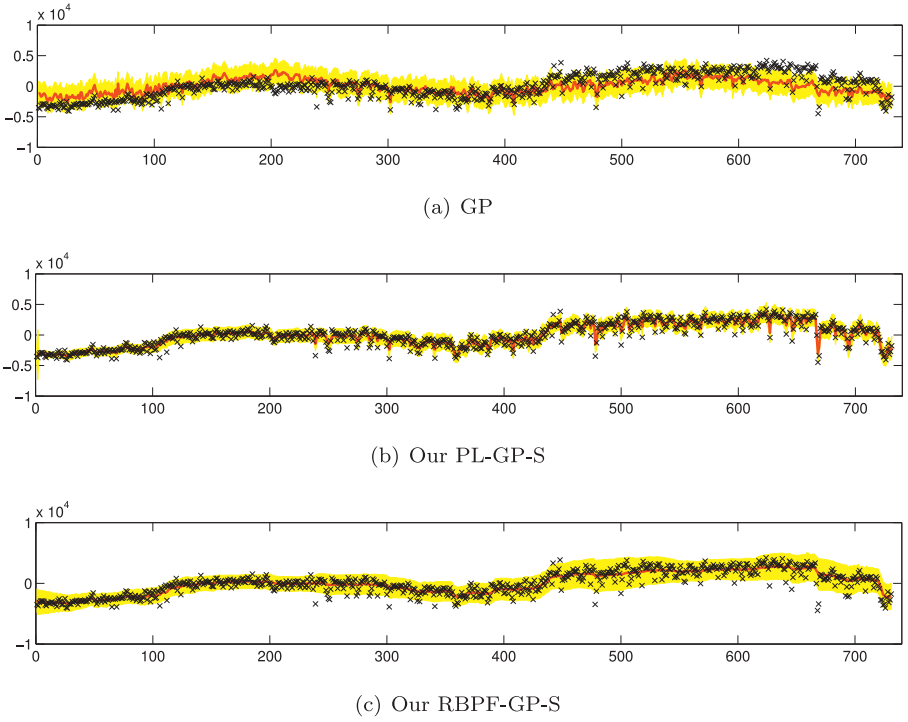
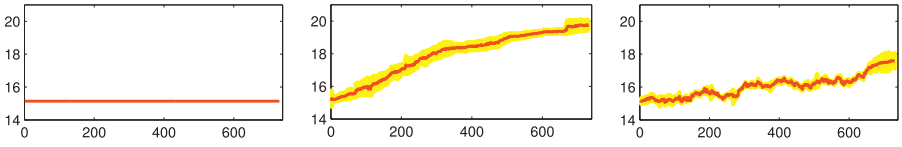


Fig. 22. Posterior of $y_{1:T}$ (*bike sharing*). The notations are the same as Fig. 10. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

to the fact that all the model parameters are time-varying in our RBPF-GP.

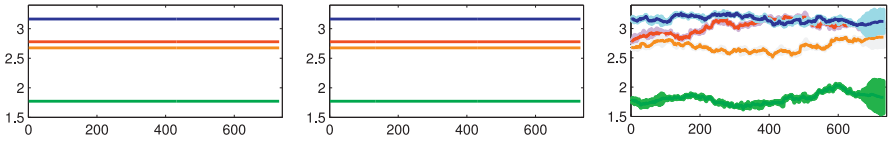
Additionally, since our approaches are based on sequential Monte Carlo, the number of particles N is an important factor for the prediction accuracy of our approaches. We thus compare our PL-GP and RBPF-GP approaches to other particle-based GP approaches, i.e., GP-PF [18], MPGP [19] and DGP [10], by evaluating MNLP as a function of N for all the data sets. The results are shown in Figs. 1–5. As expected, the prediction performance of all the particle-based GP approaches tends to be improved when N

Please cite this article as: Y. Wang, B. Chaib-draa, Bayesian inference for time-varying applications: Particle-based Gaussian process approaches, Neurocomputing (2017), http://dx.doi.org/10.1016/j.neucom.2017.01.072



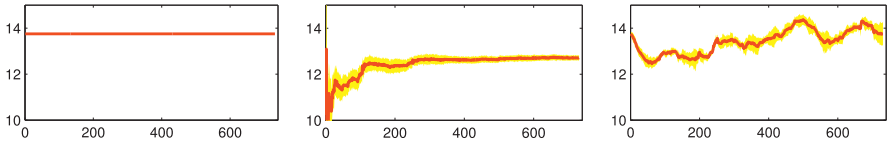
(a) GP (b) Our PL-GP-S (c) Our RBPF-GP-S

Fig. 23. Posterior of $\log(\sigma_f^2)$ over time (*bike sharing*). The notations are the same as Fig. 11. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)



(a) GP (b) Our PL-GP-S (c) Our RBPF-GP-S

Fig. 24. Posterior of $\log(\ell)$ over time (*bike sharing*). Since the input of *bike sharing* is four dimension, the length-scale ℓ is a four dimension vector in all approaches, where different colors represent different dimensions in (a)–(c). Furthermore, ℓ is time-invariant in GP and our PL-GP-S, hence the estimations of these two approaches are lines in (a) and (b). The time-varying estimations of our RBPF-GP-S are lines with their corresponding intervals in (c). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)



(a) GP (b) Our PL-GP-S (c) Our RBPF-GP-S

Fig. 25. Posterior of $\log(\sigma_y^2)$ over time (*bike sharing*). The notations are the same as Fig. 13. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

increases. Furthermore, our PL-GP and RBPF-GP approaches outperform other particle-based GP approaches, since we learn the model parameters over time to capture non-stationarity and/or heteroscedasticity.

4.2. Backward smoothing

After analyzing online prediction, we focus on backward smoothing in Section 3.2.3 to further validate the effectiveness of our approaches. Hence, we perform our PL-GP and RBPF-GP with the standard backward smoothing approach [42], given all the temporal input–output pairs in the history. For clarity, we denote our approaches with backward smoothing as our PL-GP-S and our RBPF-GP-S in the following.

4.2.1. Qualitative evaluation

To illustrate that our approaches can model time-varying non-stationarity and heteroscedasticity correctly, we first implement our PL-GP-S and RBPF-GP-S to visualize the posterior of output and the posterior of log model parameters $\log(\sigma_f^2)$, $\log(\ell)$, $\log(\sigma_y^2)$ over time, in comparison with GP.

The *synthetic* data set exhibits both time-varying non-stationarity (two discontinuities in the latent function) and heteroscedasticity (three levels of observation noise). As shown in Fig. 6, these distinct temporal characteristics are mistakenly interpreted by GP, but successfully modeled by our PL-GP-S and RBPF-GP-S approaches. This is mainly because the model parameters are time-invariant in GP but time-varying in our approaches. The posteriors of all the model parameters in Figs. 7–9 can be a clear proof of this fact.

The *motor* data set exhibits heteroscedasticity that consists of a flat low noise region, a curve region and flat high noise region at different time periods [30]. As shown in Figs. 10 and 13, GP fails to model the low-noise region, while our PL-GP-S and RBPF-GP-S successfully explain all three noise regions by learning the time-varying $\log(\sigma_y^2)$. More interestingly, our RBPF-GP-S in Fig. 13(c) models the decreasing noise in the flat high noise region [24]. Finally, we show the posteriors over other parameters of our approaches in Figs. 11 and 12.

The *heart rate* data set mainly exhibits non-stationarity that consists of sudden signal shifts. We thus show the posterior for the sudden shift region (from the 150th to 230th step) in Fig. 14. To capture this sudden change (around the 180th step), GP interprets it as a large noise which can be seen in Fig. 17. It introduces the unnecessary high uncertainty in other time periods in Fig. 14. On the contrary, both our PL-GP-S and RBPF-GP-S correctly capture this sudden shift with a compact confidence interval, by learning model parameters over time in Figs. 15 and 16.

The *S&P* and *bike sharing* data sets contain both non-stationarity and heteroscedasticity. Moreover, the input for these two data sets is a multi-dimensional vector which is not only related to the time step. All these facts make Bayesian inference quite challenging. We show the posteriors of these two data sets, along with the time axis in Figs. 18 and 22. As expected, the performance of GP is limited due to non-stationarity and heteroscedasticity. Additionally, the power of our PL-GP-S tends to be reduced for these two multi-dimensional-input cases. This may be because the length-scale parameter ℓ is time-invariant for our PL-GP-S, as shown in Figs. 20 and 24. Finally, our RBPF-GP-S achieves the best performance for these two temporal data sets by learning all the parameters over time in Figs. 19–21 and 23–25.

4.2.2. Quantitative evaluation

Since the ground truth of latent function values at all time steps is known in the *synthetic* data set, we further evaluate the backward smoothing accuracy of our approaches on this data set. Specifically, we compare our PL-GP-S and RBPF-GP-S with several widely-used batch GP approaches which can capture non-stationarity and heteroscedasticity, including sparse pseudo-input

Please cite this article as: Y. Wang, B. Chaib-draa, Bayesian inference for time-varying applications: Particle-based Gaussian process approaches, Neurocomputing (2017), http://dx.doi.org/10.1016/j.neucom.2017.01.072

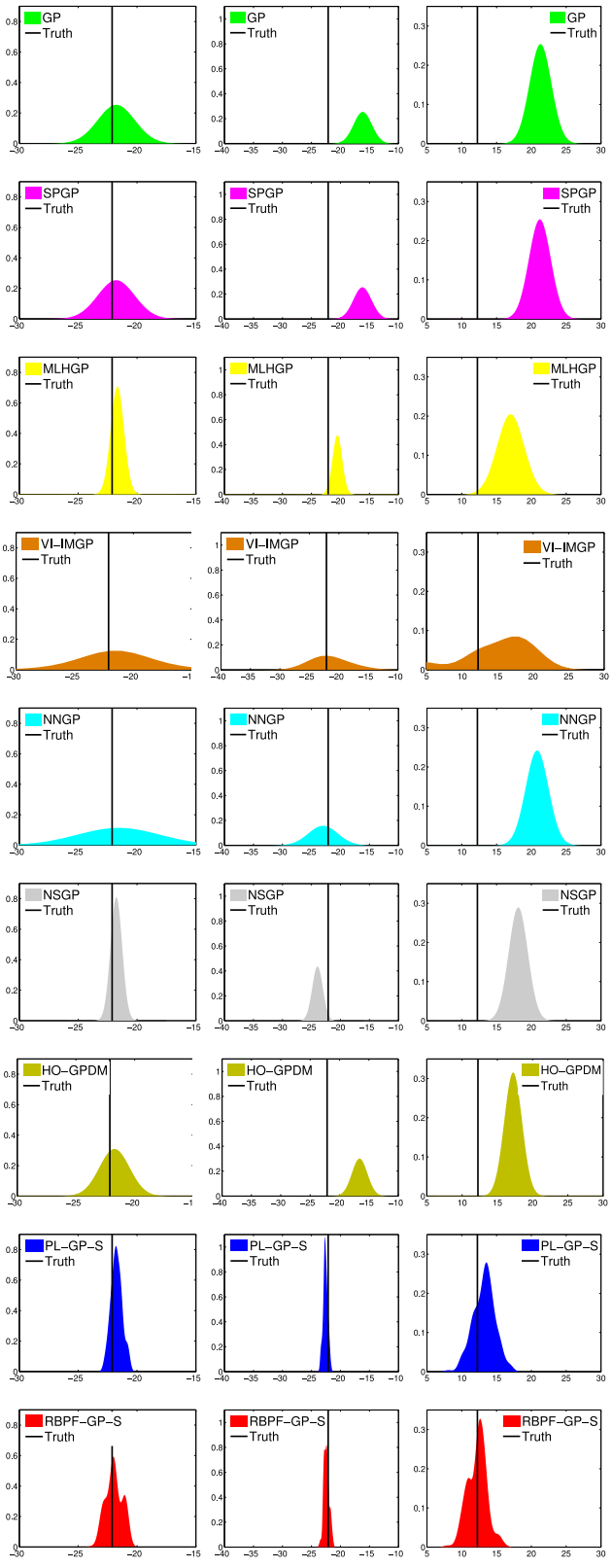


Fig. 26. Posterior of f_t (*synthetic*) at t =0.54, 1.93, 5.07 (from left to right column).

Table 4
MSE between the ground truth and the estimation of latent function values (*synthetic*). RT represents the running time.

Methods	MSE	RT (s)
GP	9.43	150
SPGP	9.40	50
MLHGP	7.48	350
VI-IMGP	7.39	1010
NNGP	7.12	80
NSGP	6.80	1800
HO-GPDM	6.38	736
Our PL-GP-S	6.10	**30**
Our RBPF-GP-S	**5.80**	55

GP (SPGP) which is originally used for computation reduction but can model non-stationarity and heteroscedasticity [43], most likely heteroscedastic GP (MLHGP) [24], GP with a non-stationary Neural Network covariance function (NNGP) [1], non-stationary GP (NSGP) [22], infinite mixtures of Gaussian processes with variational inference (VI-IMGP in which T is 5, C is 200 and S is 200) [32], and high-order Gaussian process dynamical model (HO-GPDM in which state space model is four-order and the latent state is one-dimension) [14]. Additionally, we use the mean squared error (MSE) between the ground truth and the estimation of latent function values to evaluate accuracy, and use the running time (RT) of different approaches to evaluate efficiency.

The results in Table 4 clearly indicate that our PL-GP-S and RBPF-GP-S outperform other non-stationary and/or heteroscedastic GP approaches, in terms of both accuracy and efficiency. Note that, the accuracy of our RBPF-GP-S is higher than our PL-GP-S but the efficiency is lower than our PL-GP-S, as expected. The main reason is that all the time-varying parameters are learned in our RBPF-GP-S but ℓ is time-invariant in our PL-GP-S. But still, our RBPF-GP-S is faster than most of related GP approaches. Hence, both our PL-GP-S and RBPF-GP-S are efficient and accurate for time-varying data modeling.

Finally, we show the posterior of f_t at $t = 0.54, 1.93, 5.07$ by using different GP approaches above. The choice of these three time steps is based on the fact that, $t = 0.54$ is within the time period which is strongly influenced by heteroscedasticity; $t = 1.93$ and 5.07 are within the time periods which are strongly influenced by non-stationarity. In Fig. 26, the posteriors of GP, SPGP, VI-IMGP, NNGP, HO-GPDM at $t = 0.54$ are relatively flat-shaped. This illustrates that these approaches may be limited with strong heteroscedasticity. As a result, these approaches may make a relatively-uncertain estimation of latent function value, even though $t = 0.54$ is at the small-noise-level region of this data set. On the contrary, MLHGP, NSGP, our PL-GP-S and RBPF-GP-S can take heteroscedasticity into account for uncertainty reduction. Hence, the posteriors of these approaches are peak-shaped at $t = 0.54$. Furthermore, at $t = 1.93$ and 5.07, our PL-GP-S and RBPF-GP-S can successfully model non-stationarity (sudden function shifts) with a low uncertainty, while the rest GP approaches tend to estimate the latent function values mistakenly.

5. Conclusion

To model the challenging temporal characteristics (such as sequential-order, non-stationarity and heteroscedasticity) in the time-varying applications, we proposed two particle-based GP approaches in this paper. First, we take advantage of GP to design two novel state-space models in order to model the sequential order of temporal data. Then, we develop two particle mechanisms to infer the latent function values and the model parameters in a

Please cite this article as: Y. Wang, B. Chaib-draa, Bayesian inference for time-varying applications: Particle-based Gaussian process approaches, Neurocomputing (2017), http://dx.doi.org/10.1016/j.neucom.2017.01.072

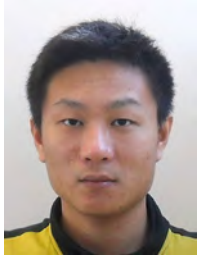
recursive Bayesian framework. Because the parameters of our models are time-varying, we can capture the temporal non-stationarity and heteroscedasticity. Finally, we show the effectiveness of our approaches on a number of challenging time-varying applications, in comparison with several relevant GP approaches. In the future, it would be interesting to use other particle smoothing methods [44] or design the non-stationary covariance function [22] for our particle-based GP approaches to further improve computation efficiency and model flexibility.

Acknowledgments

This work has been supported by Natural Sciences and Engineering Research Council of Canada (NSERC) (CG029213), National Natural Science Foundation of China (61502470), Shenzhen Research Program (JCYJ20160229193541167).

References

[1] C.E. Rasmussen, C.K.I. Williams, Gaussian Process for Machine learning, MIT Press, 2006.
[2] D. Nguyen-Tuong, J. Peters, M. Seeger, Local Gaussian process regression for real time online model learning and control, in: Proceedings of International Conference on Neural Information Processing Systems (NIPS), 2008.
[3] S. Särkkä, J. Hartikainen, Infinite-dimensional Kalman filtering approach to spatio-temporal Gaussian process regression, in: Proceedings of International Conference on Artificial Intelligence and Statistics (AISTATS), 2012.
[4] S.V. Vaerenbergh, M. Lázaro-Gredilla, I. Santamaría, Kernel recursive least-squares tracker for time-varying regression, IEEE Trans. Neural Netw. Learn. Syst. 23 (8) (2012) 1313–1326.
[5] C. Plagemann, Gaussian processes for flexible robot learning, Albert-Ludwigs-Universität Freiburg, 2008 (Ph.D. thesis).
[6] A. Girard, C.E. Rasmussen, J. Quinonero Candela, R. Murray-Smith, Gaussian process priors with uncertain inputs – application to multiple-step ahead time series forecasting, in: Proceedings of Neural Information Processing Systems (NIPS), 2002.
[7] V. Tresp, A Bayesian committee machine, Neural Comput. 12 (2000) 2719–2741.
[8] L. Csató, M. Opper, Sparse online Gaussian processes, Neural Comput. 14 (3) (2002) 641–668.
[9] H. Bijl, J. van Wingerden, T.B. Schön, M. Verhaegen, Online sparse Gaussian process regression using FITC and PITC approximations, in: Proceedings of IFAC Symposium on System Identification, 2015.
[10] Y. Wang, M. Brubaker, B. Chaib-Draa, R. Urtasun, Sequential inference for deep Gaussian process, in: Proceedings of International Conference on Artificial Intelligence and Statistics (AISTATS), 2016.
[11] J. Wang, D. Fleet, A. Hertzmann, Gaussian process dynamical models for human motion, IEEE Trans. Pattern Anal. Mach. Intell. 30 (2) (2008) 283–298.
[12] R. Frigola-Alcalde, Bayesian time series learning with Gaussian processes, University of Cambridge, 2015 (Ph.D. thesis).
[13] J. Zhao, S. Sun, Revisiting Gaussian process dynamical models, in: Proceedings of International Joint Conference on Artificial Intelligence (IJCAI), 2015.
[14] J. Zhao, S. Sun, High-order gaussian process dynamical models for traffic flow prediction, IEEE Trans. Intell. Transp. Syst. 17 (7) (2016) 2014–2019.
[15] S. Reece, S. Roberts, An Introduction to Gaussian processes for the Kalman filter expert, in: Proceedings of International Conference on Information Fusion (FUSION), 2010.
[16] J. Hartikainen, S. Särkkä, Kalman filtering and smoothing solutions to temporal Gaussian process regression models, in: Proceedings of IEEE International Workshop on Machine Learning for Signal Processing (MLSP), 2010.
[17] M.P. Deisenroth, M.F. Huber, U.D. Hanebeck, Analytic moment-based Gaussian process filtering, in: Proceedings of International Conference on Machine Learning (ICML), 2009.
[18] J. Ko, D. Fox, GP-BayesFilters: Bayesian Filtering Using Gaussian Process Prediction and Observation Models, in: Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2008.
[19] Y. Wang, B. Chaib-draa, A marginalized particle Gaussian process regression, in: Proceedings of International Conference on Neural Information Processing Systems (NIPS), 2012, pp. 1196–1204.
[20] Y. Wang, M.A. Brubaker, B. Chaib-draa, R. Urtasun, Bayesian filtering with online Gaussian process latent variable models, in: Proceedings of International Conference on Uncertainty in Artificial Intelligence (UAI), 2014, pp. 849–857.
[21] C.J. Paciorek, M.J. Schervish, Nonstationary covariance functions for Gaussian process regression, in: Proceedings of International Conference on Neural Information Processing Systems (NIPS), 2004.
[22] M. Heinonen, H. Mannerström, J. Rousu, S. Kaski, H. Lähdesmäki, Non-stationary Gaussian process regression with Hamiltonian Monte Carlo, in: Proceedings of International Conference on Artificial Intelligence and Statistics (AISTATS), 2016.
[23] P.W. Goldberg, C.K.I. Williams, C.M. Bishop, Regression with input-dependent noise: a Gaussian process treatment, in: Proceedings of International Conference on Neural Information Processing Systems (NIPS), 1998, pp. 493–499.
[24] K. Kersting, C. Plagemann, P. Pfaff, W. Burgard, Most likely heteroscedastic Gaussian process regression, in: Proceedings of International Conference on Machine Learning (ICML), 2007.
[25] M. Lázaro-Gredilla, M.K. Titsias, Variational heteroscedastic Gaussian process regression, in: Proceedings of International Conference on Machine Learning (ICML), 2011.
[26] E. Snelson, C.E. Rasmussen, Z. Ghahramani, Warped Gaussian processes, in: Proceedings of International Conference on Neural Information Processing Systems (NIPS), 2003, pp. 337–344.
[27] M. Lázaro-Gredilla, Bayesian warped Gaussian processes, in: Proceedings of International Conference on Neural Information Processing Systems (NIPS), 2012, pp. 1619–1627.
[28] A.G. Wilson, D.A. Knowles, Z. Ghahramani, Gaussian process regression networks, in: Proceedings of International Conference on Machine Learning (ICML), 2012, pp. 599–606.
[29] A.C. Damianou, N.D. Lawrence, Deep Gaussian processes, in: Proceedings of International Conference on Artificial Intelligence and Statistics (AISTATS), 2013, pp. 207–215.
[30] C.E. Rasmussen, Z. Ghahramani, Infinite mixtures of Gaussian process experts, in: Proceedings of International Conference on Neural Information Processing Systems (NIPS), 2002.
[31] E. Meeds, S. Osindero, An alternative infinite mixture of Gaussian process experts, in: Proceedings of International Conference on Neural Information Processing Systems (NIPS), 2006.
[32] S. Sun, X. Xu, Variational inference for infinite mixtures of gaussian processes with applications to traffic flow prediction, IEEE Trans. Intell. Transp. Syst. 12 (2) (2011) 466–475.
[33] D.J.C. MacKay, Introduction to Gaussian processes, in: Proceedings of International Conference on Neural Networks and Machine Learning, 1998, pp. 133–165.
[34] C.M. Bishop, Pattern Recognition and Machine Learning, Springer, 2006.
[35] D. Barber, Bayesian Reasoning and Machine Learning, Cambridge University Press.
[36] N. Lawrence, Probabilistic non-linear principal component analysis with Gaussian process latent variable models, J. Mach. Learn. Res. 6 (2005) 1783–1816.
[37] O. Cappé, S.J. Godsill, E. Moulines, An overview of existing methods and recent advances in sequential Monte Carlo, Proc. IEEE 95 (5) (2007) 899–924.
[38] C.M. Carvalho, M.S. Johannes, H.F. Lopes, N.G. Polson, Particle learning and smoothing, Stat. Sci. 25 (1) (2010) 88–106.
[39] A. Doucet, N. de Freitas, K. Murphy, S. Russell, Rao-Blackwellised particle filtering for dynamic Bayesian networks, in: Proceedings of International Conference on Uncertainty in Artificial Intelligence (UAI), 2000.
[40] G. Storvik, Particle filters for state-space models with the presence of unknown static parameters, IEEE Trans. Signal Process. 50 (2) (2002) 281–289.
[41] A. Doucet, N.d. Freitas, N. Gordon, Sequential Monte Carlo Methods in Practice, Springer, 2001.
[42] S.J. Godsill, A. Doucet, M. West, Monte Carlo smoothing for nonlinear time series, J. Am. Stat. Assoc. 99 (465) (2004) 156–168.
[43] E. Snelson, Z. Ghahramani, Sparse Gaussian processes using pseudo-inputs, in: Proceedings of International Conference on Neural Information Processing Systems (NIPS), 2006.
[44] P. Fearnhead, D. Wyncoll, J. Tawn, A sequential smoothing algorithm with linear computational cost, Biometrika 97 (2) (2010) 447–464.



Yali Wang received his Ph.D. in Computer Science from Laval University in 2014. He is currently an Assistant Professor at Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences. His research interests are machine learning and statistics, computer vision and deep learning.

Brahim Chaib-draa received a Diploma in Computer Engineering from École Supérieure d'Électricité (SUPELEC), Paris, France, in, 1978 and a Ph.D. degree in Computer Science from the Université du Hainaut-Cambrésis, Valenciennes, France, in 1990. In 1990, he joined the Department of Computer Science and Software Engineering at Laval University, Québec, QC, Canada, where he is a Professor and Group Leader of the Data Analytics and Mobile-Autonomous Systems (DAMAS) Group. His research interests include Artificial Intelligence, Machine Learning and Complex Decision Making. He is the author of several technical publications in these areas. He is on the Editorial Boards of different journals among them IEEE Transactions on SMC, and Computational Intelligence. Dr. Chaib-draa is a member of ACM and AAAI and senior member of the IEEE Computer Society.

Please cite this article as: Y. Wang, B. Chaib-draa, Bayesian inference for time-varying applications: Particle-based Gaussian process approaches, Neurocomputing (2017), http://dx.doi.org/10.1016/j.neucom.2017.01.072