VARIATIONAL PARAMETER LEARNING IN SEQUENTIAL STATE-SPACE MODEL VIA
PARTICLE FILTERING

Chenhao Li  
University of Cambridge  
Department of Engineering  
Cambridge, UK

Simon J. Godsill  
University of Cambridge  
Department of Engineering  
Cambridge, UK

ABSTRACT

Parameter learning of the state-space model (SSM) plays a significant role in the modelling of time-series data and dynamical systems. However, the closed-form inference of the parameter posterior is often limited by sequential construction and non-linearity of the SSMs, which has led to the development of sampling-based algorithms such as particle Markov chain Monte Carlo (PMCMC). We present a novel algorithm, the particle filter variational inference (PF-VI) algorithm, which achieves closed-form learning of SSM parameters while tractably inferring the non-linear sequential states. We apply the algorithm to a popular non-linear SSM example and compare its performance against two competing PMCMC algorithms.

Index Terms— Bayesian inference, state-space model, variational Bayes, particle filtering.

1. INTRODUCTION

The state-space model (SSM) provides a powerful probabilistic modelling framework for real-world dynamical systems and time-series data. Originated from control engineering [1], the SSM has been developed extensive over the past few decades with its applications in various areas and disciplines including system control [2], object tracking [3], image processing, [4], economics [5], finance [6, 7] and neuroscience [8]. The dynamics-controlling parameters of the SSM are crucial in determining the diffusion behaviours of the stochastic process and the learning of these parameters not only provides a better model fit and more accurate predictions but also allows useful insights of the system to be extracted from the learned parameters.

Many parameter-learning algorithms have been developed under both Bayesian and non-Bayesian settings. The classic frequentist approaches such as the maximum likelihood estimation (MLE) and the expectation maximisation (EM) algorithm [9] provide point estimates of the parameters by optimising an objective (e.g. likelihood). On the other hand, Bayesian inference focuses on finding posterior distributions of the parameters in the model given priors, which provides better uncertainty quantification but can only be applied in certain conjugate cases. Both variational Bayes (VB) algorithm [10, 11] and expectation propagation (EP) algorithm [12] aim to find approximations to the true posterior, each by minimising a different form of the Kullback-Leibler (KL) divergence [13, 14]. However, the inherent intractability of the general SSM and the large number of sequential latent variables seriously challenge both applicability and computational complexity of these algorithms.

The particle filter (PF) was first proposed in [15] to perform Bayesian inference on sequential states in non-linear/non-Gaussian SSMs. Based on the PF, the authors of [16] propose an iterative parameter-learning framework called PMCMC which combines the PF with Markov chain Monte Carlo samplers to obtain empirical estimations for the joint posterior of both states and parameters. The SMC2 algorithm proposed in [17] can be considered as an online version of the PMCMC algorithm where an additional set of particles is used to explore the online posterior of the parameters during the course of the PF. Despite the online construction, the SMC2 algorithm is too computationally heavy to be used in real online scenarios without generous computation budget [18]. Algorithms that combine the variational inference and the PF were studied in [19] and [20]. These two papers focus more on the inference of sequential states by optimising a PF-estimated objective rather than Bayesian parameters learning.

In this paper, we propose a novel algorithm, termed the PF-VI algorithm, which achieves closed-form approximate inference for the parameter posterior under the VB framework. Our algorithm exploits the strength of the PF and can thus be applied to a range of non-linear/non-Gaussian SSMs. In Section 4, we test the PF-VI algorithm on datasets simulated from a popular non-linear SSM and compare its performance against two competing PMCMC algorithms.

2. BACKGROUND

Here, we briefly review the generic algorithms of VB and PF.

2.1. The variational Bayes algorithm

Variational Bayes (VB) or variational inference (VI) refers to an algorithmic framework for approximating the (joint) posterior distribution of a model. Denoting all random variables of a model as \( \theta = \{ \theta_j \}_{j=1}^J \) and observations as \( Y = \{ y_n \}_{n=1}^N \), the VB algorithm postulate a family/class of distributions \( q(\theta) \) to approximate the true joint posterior \( p(\theta | Y) \) in a closed form. The algorithm then minimises a divergence, often the KL divergence, between the true posterior and the postulated family:

\[
\text{KL}(q || p) = - \mathbb{E}_q \left( \log \frac{p(\theta | Y)}{q(\theta)} \right)
\]

\[
= \int \log \frac{p(\theta | Y)}{q(\theta)} q(\theta) d\theta
\]  

A standard approach called coordinate ascent variational inference (CAVI) [21] for formulating the approximated posterior \( q(\theta) \) is by...
using the mean-field approximation to independently factorise the joint posterior for each component $\theta_j$, i.e.:

$$q(\theta | \Omega) = \prod_{j=1}^{J} q_j(\theta_j | \Omega_j)$$

where $\Omega = \{\Omega_j\}_{j=1}^{J}$ denotes the hyperparameters of the variational approximation. The VB algorithm thus iteratively cycles through the individual components $\theta_j$ of the approximated posterior and updates the hyperparameters $\Omega_j$ so that:

$$\log q_j(\theta_j | \Omega_j) = \int \log p(\theta | Y) \, q_j(\theta_j | \Omega_j) \, d\theta_j,$$

$$= \mathbb{E}_{q_j} \left\{ \log p(\theta | Y) \right\}$$

where the subscript $(.,j)$ denotes all other components except $j$. It is not trivial to see how such operations of the VB algorithm would minimise the KL divergence in (1). Due to the limited space, we will not show the proof here. Readers may refer to [22] and [21] for detailed derivations. The VB algorithm allows the updates of $\Omega_j$ to be performed by matching the terms associated with $\theta_j$ on both sides of the equation without computing the exact integration in (3).

### 2.2. The generic particle filter

The PF algorithm targets the posterior of the sequential state vectors in a SSM. Unlike the message-passing VB algorithm, the PF uses weighted particles/samples to approximate the sequential posteriors and adopt the concept “survival-of-the-fittest” to filter out unrepresentative (i.e. low-weight) particles during the running of the algorithm.

Denoting a SSM with state vectors $X = \{x_n\}_{n=1}^{N}$ at timestamps $\{t_n\}_{n=1}^{N}$ and corresponding observations/data $Y = \{y_n\}_{n=1}^{N}$, each step of the PF algorithm aims to obtain an empirical estimation of the target posterior or smoothing density [23] at time $t_n$:

$$p(x_{1:n} | y_{1:n}) = \frac{\sum_{p=1}^{N_p} w_{n-1}^p \delta_{x_{1:n-1}}(x_{1:n-1})}{p(y_n | y_{1:n-1})}$$

where $x_{1:n}$ and $y_{1:n}$ are shorthands for state vectors and observations from $t_1$ to $t_n$ respectively. Suppose that the smoothing density at time $t_{n-1}$ can be empirically represented/approximated by a total of $N_p$ weighted particles:

$$p(x_{1:n-1} | y_{1:n}) \approx \sum_{p=1}^{N_p} w_{n-1}^p \delta_{x_{1:n-1}}(x_{1:n-1})$$

where $\delta_x$ is the Dirac measure at $x$. $w_{n-1}^p \geq 0$, $\forall p$ and $\sum_{p=1}^{N_p} w_{n-1}^p = 1$. The recursion in Eq. (4) can thus be expressed as a mixed discrete-continuous distribution:

$$p(x_{1:n} | y_{1:n}) = \sum_{p=1}^{N_p} w_{n-1}^p \delta_{x_{1:n-1}}(x_{1:n-1}) \times \frac{p(x_n | x_{1:n-1}) \, p(y_n | x_n)}{p(y_n | y_{1:n-1})}$$

The above expression of the smoothing density contains the joint masses of the “historic” variables $x_{1:n-1}$, the continuous proposal of the new variable $x_n$ as well as the likelihood for observation $y_n$. Therefore, the propagation of a bootstrap PF at $t_n$ is performed as follows:

1. Sample a past particle trajectory from the collection $\{x_{1:n-1}^p\}_{p=1}^{N_p}$ using the normalised weights $\{w_{n-1}^p\}_{p=1}^{N_p}$.

2. Draw the new variable $x_n^p \sim p(x_n | x_{1:n-1}^p)$; and append it to the sampled trajectory giving $x_{1:n}^p$.

3. Re-weight the new particle trajectory based on the likelihood:

$$\tilde{w}_n^p = w_{n-1}^p \times p(y_n | x_n^p)$$

4. Re-normalise the weights after all $N_p$ particles have been propagated: $w_n^p = \tilde{w}_n^p / \sum_{p=1}^{N_p} \tilde{w}_n^p$.

Running the above pseudo-code sequentially from $t_1$ to $t_N$, we can obtain the particle approximation of the joint posterior $p(x_{1:N} | y_{1:N})$ at the end of the PF algorithm. It can be noted that this approximation asymptotically converges towards the true joint posterior as $N_p \to \infty$.

### 3. ALGORITHM

In this section, we introduce the proposed PF-VI algorithm on a general SSM with a set of static parameters. We derive an alternative form of factorisation for the approximating posterior in the VB framework where the PF can be readily applied to tackle the tractability of the SSM and the high-dimensionality of the sequential variables.

#### 3.1. A general SSM with static parameters

Suppose that the general SSM defined in Section 2.2 contains an additional set of static model parameters $\gamma$ that needs to be learned from the observations $Y$. Without loss of generality, we may assume that these parameters are involved in both the transition density $p(x_n | x_{n-1}, \gamma)$ and the observation likelihood $p(y_n | x_n, \gamma)$. The joint probability of all random variables in the model can thus be written as (omitting the hyperparameters):

$$p(X, Y, \gamma) = p(\gamma) \times p(x_1 | \gamma) \times p(y_1 | x_1, \gamma) \times \prod_{n=2}^{N} p(x_n | x_{n-1}, \gamma) \times p(y_n | x_n, \gamma)$$

#### 3.2. The PF-VI algorithm

We propose an alternative way of factorising the approximating posterior $q(\theta)$ where now $\theta = \{x_{1:N}, \gamma\}$ by applying a partial mean-field approximation:

$$p(x_{1:N}, \gamma | Y) \approx q(x_{1:N}, \gamma) = q(x_{1:N}) \, q(\gamma)$$

Such a factorisation naturally partitions the random variables into a set of time-indexed sequential variables and a set of static model parameters. Furthermore, by assuming less independence among variables the VB algorithm can often yield better approximation accuracy to the targeted posterior [24]. While it is relatively easy to postulate the parametric form for $q(\gamma)$, the non-linearity of the SSM usually prohibits any class of distributions for $q(x_{1:N})$ from achieving closed-form tractable inferences. Regardless, the VB routine following Eq. (3) allows us to optimise the KL divergence by setting:

$$\log q(x_{1:N}) \equiv \log p(x_{1:N}, \gamma | Y))q(\gamma)$$

$$= \log p(x_{1:N}, \gamma, Y))q(\gamma) + \text{const.} \quad \text{(Bayes’ rule)}$$

$$= \log p(x_1 | \gamma))q(\gamma) + \log p(y_1 | x_1, \gamma))q(\gamma) + \sum_{n=2}^{N} \left\{ \log p(x_n | x_{n-1}, \gamma))q(\gamma) + \log p(y_n | x_n, \gamma))q(\gamma) \right\}$$

+ const.
where \( \langle \cdot \rangle_q \) is the expectation operator with respect to the density \( q \). The constant term summarizes the terms that are not functions of \( x_{1:N} \) and contributes to the normalisation of \( q(x_{1:N}) \). In order to solve the intractable approximating posterior above, we first construct the following recursion for \( q(x_{1:n}) \) at each timestamp \( t_n \):

\[
\log q(x_1) \equiv \langle \log p(x_1 | \gamma) \rangle_q + \langle \log p(y_1 | x_{1-1}, \gamma) \rangle_q + \langle \log p(x_{1-1} | x_{1-2}, \gamma) \rangle_q + \text{const.}
\]

\[
\log q(x_{1:n}) \equiv \log q(x_{1:n-1}) + \langle \log p(y_n | x_n, \gamma) \rangle_q + \langle \log p(x_{n+1} | x_n, \gamma) \rangle_q + \text{const.}
\]

It can be seen that each approximating posterior above is a function of a subset of the joint state vector \( x_{1:N} \) and thus contains a subset of the terms in Eq. (9). Moreover, we define the intermediate distributions \( \tilde{q}(x_{1:n}) \) for \( n = 1, \ldots, N \) such that:

\[
\log \tilde{q}(x_{1:n}) \equiv \log q(x_{1:n}) - \langle \log p(x_{n+1} | x_n, \gamma) \rangle_q + \text{const.}
\]

This closely resembles the recursion in Eq. (4) except that we are now interested in averages/expectations over \( q(\gamma) \) rather than the conditional densities. Additionally, we can deduce by inspection that the end condition gives \( \log \tilde{q}(x_{1:N}) \equiv \log q(x_{1:N}) \) as the desired approximating posterior for the sequential variables. This allows us to apply the PF reviewed in Section 2.2 and obtain the empirical particle representation of \( q(x_{1:N}) \) by running the PF across all timestamps \( \{t_n\}_{n=1}^N \).

Supposing that the intermediate distribution \( \tilde{q}(x_{1:n-1}) \) can be accurately approximated by a collection of \( N_p \) (weighted) particles, the intermediate distribution at timestamp \( t_n \) targeted by the PF is:

\[
q(x_{1:n}) \approx \sum_{p=1}^{N_p} w_{p,n-1} \delta_{x_{1:n-1}}(x_{1:n-1}) \times \exp\{\langle \log p(y_n | x_n, \gamma) \rangle_q\} \times \exp\{\langle \log p(x_n | x_{n-1}, \gamma) \rangle_q\} \times \text{const.}
\]

where (i) and (ii) are the weight-correcting likelihood and transition density of the state vector respectively, each subject to a multiplicative constant which can be conveniently handled by the PF routine. With an appropriate choice of the parametric form for \( q(\gamma) \), both likelihood and transition density can be deduced analytically from (i) and (ii) by matching the relevant terms (e.g. moment matching) without computing the expectations exactly. Hence, the sequential inference stage of the PF-VI algorithm outputs the particle representation of the approximating posterior:

\[
q(x_{1:N}) \approx \sum_{p=1}^{N_p} w_{N} \delta_{x_{1:N}}(x_{1:N})
\]

The other stage of the PF-VI algorithm involves the closed-form inference of the static model parameters following the generic VB procedure.

**Fig. 1:** Posterior distributions and sample histograms obtained after 100 burn-in iterations. The true values of the parameters are indicated in red dashed lines.

**Cedure:**

\[
\log q(\gamma) \\
\equiv \langle \log p(x_{1:N}, \gamma, Y) \rangle_q + \text{const.}
\]

\[
= \log p(\gamma) + \langle \log p(x_1 | \gamma) \rangle_q + \langle \log p(y_1 | x_{1-1}, \gamma) \rangle_q + \sum_{n=2}^{N} \left\{ \log p(x_n | x_{n-1}, \gamma) + \log p(y_n | x_n, \gamma) \right\} + \text{const.}
\]

\[
= \log p(\gamma) + \sum_{n=1}^{N_p} \log p(x_n | x_{n-1}, \gamma) + \log p(y_n | x_n, \gamma) + \sum_{n=2}^{N} \left\{ \log p(x_n | x_{n-1}, \gamma) + \log p(y_n | x_n, \gamma) \right\} + \text{const.}
\]

where the last line is obtained by substituting in the particle representation of \( q(x_{1:N}) \) in (12). We see from (13) that the learning of \( q(\gamma) \) takes weighted contribution from all particles obtained from the PF stage. This remarkable feature of the PF-VI avoids wasteful information usage as opposed to the popular particle Gibbs (PG) algorithm [16] and improves convergence stability in the meantime.

**4. APPLICATION**

In this section, we apply the PF-VI algorithm on a popular toy nonlinear SSM [15, 23] with unknown innovation noise and observation noise.
Table 1: Posterior MSEs for different algorithms. **BOLD** is the best.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Parameter</th>
<th>Cheat</th>
<th>PF-VI</th>
<th>PMMH</th>
<th>PG</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a)</td>
<td>$\sigma_n^p$</td>
<td>0.035</td>
<td><strong>0.025</strong></td>
<td>0.139</td>
<td>0.273</td>
</tr>
<tr>
<td></td>
<td>$\sigma_n^q$</td>
<td>2.22</td>
<td><strong>3.16</strong></td>
<td>6.35</td>
<td>5.93</td>
</tr>
<tr>
<td>(b)</td>
<td>$\sigma_n^p$</td>
<td>3.50</td>
<td><strong>2.19</strong></td>
<td>22.28</td>
<td>16.86</td>
</tr>
<tr>
<td></td>
<td>$\sigma_n^q$</td>
<td>2.22</td>
<td><strong>10.58</strong></td>
<td>36.99</td>
<td>35.62</td>
</tr>
</tbody>
</table>

and an observation model:

$$y_n = \frac{x_n^2}{2n} + w_n \quad (15)$$

where $x_1 \sim \mathcal{N}(0, 5)$, $v_n \overset{i.i.d.}{\sim} \mathcal{N}(0, \sigma_v^2)$ and $w_n \overset{i.i.d.}{\sim} \mathcal{N}(0, \sigma_w^2)$. The model contains two "priorly" independent static parameters, which we denote $\gamma = \{\sigma_p, \sigma_w\}$. This SSM is often used to test the performance of the PF methods.

Two sets of data are simulated from the SSM for testing: (a) 100 observations $\{y_n\}_{n=1}^{100}$ with relatively small observation noise $\sigma_n^p = 1$ and $\sigma_n^q = 10$; and (b) 100 observations with large observation noise $\sigma_n^p = \sigma_n^q = 10$. We adopt two identical diffuse inverse-Gamma priors $\mathcal{IG}(\alpha = 0.01, \beta = 0.01)$ for both $\sigma_p$ and $\sigma_w$. The PMMH sampler uses a 2D Gaussian random-walk proposal with diagonal covariance. A good (i.e. close-to-truth) initialisation with $\sigma^p_n(0) = \sigma^q_n(0) = 20$ is used for both PG and PMMH algorithms in this first experiment; while the PF-VI algorithm assumes $q(\sigma_n^p) = q(\sigma_n^q) = \mathcal{IG}(0.01, 0.01)$ (i.e. prior) for the first iteration. Running all three inference algorithms for 500 iterations with a bootstrap PF and $N_p = 2000$ particles, Fig. 1 shows the posterior results as the algorithm shows no convergence within the 500 iterations.

![L2 distance](image)

(i) A fair initialisation with $\sigma^p_n(0) = \sigma^q_n(0) = 100$.

![Log L2 distance](image)

(ii) A poor initialisation randomly drawn from the diffuse prior.

**Fig. 2:** $l_2$-distances and log $l_2$-distances between the posterior samples and the truth for all 500 iterations. (b) omits the PMMH results as the algorithm shows no convergence within the 500 iterations.

![ACFs](image)

**Fig. 3:** ACFs of the parameters obtained on the small observation noise with fair initialisation $\sigma^p_n(0) = \sigma^q_n(0) = 100$ and varying particle numbers $N_p$.

Consider the non-linear SSM with a transition function:

$$x_n = \frac{x_{n-1}}{2} + 25 \frac{x_{n-3}}{1 + x_{n-1}} + 8 \cos(1.2n) + v_n \quad (14)$$

example was used in [16] to evaluate the performance of these two PMCMC algorithms.

We have introduced the PF-VI algorithm, a new variational parameter-learning approach that provides accurate and robust closed-form Bayesian inference for the parameters in non-linear SSMs. By using the PF and an alternative factorisation of the approximating posterior, the algorithm efficiently handles the non-linear sequential state inference that is otherwise intractable using the VB algorithm. Results have shown that the PF-VI algorithm is able to provide more accurate and more robust learning performance compared to the well-established PG and PMMH algorithms. Further improvements may be achieved by adopting various PF techniques such as Rao-Blackwellisation [25, 26] and backward smoothing [27, 28].

5. SUMMARY

We test and compare the algorithms' robustness to different

Fig. 2 compares the convergence speed of the three algorithms by showing the (log) $l_2$-distances between the posterior samples and the truth against iterations under two different initial settings (for PG and PMMH): (i) a fair initialisation $\sigma^p_n(0) = \sigma^q_n(0) = 100$; and (ii) a poor initialisation drawn from the diffuse $\mathcal{IG}$ prior. Dataset (a) is used for this test. From the figure, we can see that both PG and PMMH rely on a good initialisation to ensure fast convergence; whilst the PF-VI algorithm is initialisation-free and can take the diffuse prior as the initial "guess" for $q(\sigma_n^p)$ and $q(\sigma_n^q)$.

The number of particles $N_p$ in the PF directly affects the sequential inference accuracy and largely determines its computational cost. We test and compare the algorithms' robustness to different $N_p$ values in the PF stage. Using dataset (a) and initialisation (i), Fig. 3 shows the normalised auto-correlation functions (ACFs) computed on posterior samples for the PG and the PMMH and posterior means for the PF-VI. Comparing within each algorithm, it is clear that convergence performance improves as $N_p$ increases with faster decay and smaller oscillations of the ACF. The ACF of the PF-VI is overall better and much less sensitive to the change of $N_p$ compared to the other two algorithms, which indicates that the algorithm convergence is more robust against suboptimal PF performance.

5. SUMMARY

We have introduced the PF-VI algorithm, a new variational parameter-learning approach that provides accurate and robust closed-form Bayesian inference for the parameters in non-linear SSMs. By using the PF and an alternative factorisation of the approximating posterior, the algorithm efficiently handles the non-linear sequential state inference that is otherwise intractable using the VB algorithm. Results have shown that the PF-VI algorithm is able to provide more accurate and more robust learning performance compared to the well-established PG and PMMH algorithms. Further improvements may be achieved by adopting various PF techniques such as Rao-Blackwellisation [25, 26] and backward smoothing [27, 28].
6. REFERENCES


