Online Variational Filtering and Parameter Learning

Anonymous Author(s)
Affiliation
Address
email

Abstract

We present a general-purpose variational method for *online* state estimation and parameter learning in state-space models (SSMs), a ubiquitous class of dynamic latent variable models for sequential data. As per standard batch variational techniques, we use stochastic gradients to simultaneously optimize a lower bound on the log evidence with respect to both model parameters and a variational approximation. However, unlike existing approaches, our method is able to operate in an entirely online manner, such that historic observations do not require revisitation after being incorporated and the cost of updates at each time step remains constant, despite the growing dimensionality of the posterior. This is achieved by utilizing a non-standard decomposition of this posterior distribution, and corresponding non-standard factorization of our variational approximation, followed by a novel adaptation of recursive value functions from the reinforcement learning literature. We demonstrate the performance of this methodology across several examples, including high-dimensional SSMs and sequential variational auto-encoders.

1 Introduction

Many tasks in machine learning with time series data—such as video prediction [10, 29], speech enhancement [21] or robot localization [17]—often need to be performed online. Online techniques are also necessary in contexts as diverse as target tracking [2], weather prediction [8] or financial forecasting [26]. A popular class of models for these sequential data are SSMs which, when combined with neural networks ideas, can also be used to define powerful sequential Variational Auto-Encoders (VAEs); see e.g. [5, 9, 10, 19]. However, performing inference in SSMs is a challenging problem and approximate inference techniques for such models remain an active research area.

Formally, a SSM is described by two processes: a latent Markov process and an observation process. Even if the model parameters are assumed known, online inference of the states of the latent Markov process is a complex problem known as filtering. Standard approximation techniques such as the extended Kalman Filter (KF), ensemble KF, and unscented KF can be used, but only provide an ad hoc Gaussian approximation to the filter [8, 22]. These approximate filtering methods can be used, in turn, to develop online parameter learning procedures by either augmenting the state with the static parameters or using gradient-based approaches. However, such approaches are notoriously unreliable. Particle Filtering (PF) methods, on the other hand, provide a more principled approach for online state and parameter estimation with theoretical guarantees [11, 25], but the variance of PF estimates scales typically exponentially with the state dimension [4].

Variational techniques provide an attractive alternative for simultaneous learning and inference that scale to high-dimensional latent states and are not restricted to simple parametric approximations. Many such methods have been proposed for SSMs over recent years, e.g. [1, 6, 12, 14, 15, 18, 20]. However, these methods have generally been developed for *batch inference* where one maximizes the Evidence Lower Bound (ELBO) for a *fixed* dataset. As such, they are ill-suited to *online* learning as, whenever a new observation is collected, one would need to update the entire joint variational

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

states distribution whose dimension increases over time. Though a small number of online variational approaches have been developed [19, 31], these rely on significant restrictions of the variational family, leading to approximations that cannot faithfully approximate the posterior distribution of the latent states.

The main contribution of this paper is a novel, principled, variational approach to perform *online* filtering and parameter learning for SSMs, without requiring severe restrictions on the variational family. As per standard batch variational inference, we simultaneously maximize a lower bound on the log evidence with respect to both model parameters and a variational approximation of the joint states posterior. However, our method operates in an entirely online manner and the cost of updates at each time step remains constant. Key to our approach is a non-standard backward decomposition of the variational approximation of the states posterior, combined with a novel representation of the ELBO and its gradients as expectations of recursive values functions akin to those used in Reinforcement Learning (RL) [23]. We experimentally demonstrate our method on a variety of applications.

2 Background

State-Space Models. SSMs are defined by a latent $\mathbb{R}^{d_x}$-valued Markov process $(x_t)_{t\geq 1}$ and $\mathbb{R}^{d_y}$-valued observations $(y_t)_{t\geq 1}$, which are conditionally independent given the latent process. They thus correspond to the generative model

$$x_1 \sim \mu_\theta(x_1), \qquad x_{t+1}|x_t \sim f_\theta(x_{t+1}|x_t), \qquad y_t|x_t \sim g_\theta(y_t|x_t),$$

where $\theta \in \mathbb{R}^{d_\theta}$ is a parameter of interest so that for $y^t := y_{1:t} = (y_1, ..., y_t)$ one has

$$p_\theta(x_{1:t}, y^t) = \mu_\theta(x_1) g_\theta(y_1|x_1) \prod_{k=2}^{t} f_\theta(x_k|x_{k-1}) g_\theta(y_k|x_k).$$

Assume for the time being that θ is known. Given observations $(y_t)_{t\geq 1}$ and parameter values θ, one can perform online state inference by computing the posterior of x_t given y^t which satisfies

$$p_\theta(x_t|y^t) = \frac{g_\theta(y_t|x_t) p_\theta(x_t|y^{t-1})}{p_\theta(y_t|y^{t-1})}, \quad p_\theta(x_t|y^{t-1}) = \int f_\theta(x_t|x_{t-1}) p_\theta(x_{t-1}|y^{t-1}) \mathrm{d}x_{t-1}, \tag{1}$$

with $p_\theta(x_1|y^0) := \mu_\theta(x_1)$. The log evidence $\ell_t(\theta) := \log p_\theta(y^t)$ is then given by

$$\ell_t(\theta) = \textstyle\sum_{k=1}^{t} \log p_\theta(y_k|y^{k-1}), \quad \text{where} \quad p_\theta(y_k|y^{k-1}) = \int g_\theta(y_k|x_k) p_\theta(x_k|y^{k-1}) \mathrm{d}x_k. \tag{2}$$

Here $p_\theta(x_k|y^k)$ is known as the filtering distribution. While the recursion (1) and the sequential decomposition (2) are at the core of most existing online state and parameter inference techniques [11, 22, 31], we will focus here on the joint posterior distribution, also known as the smoothing distribution, of the states $x_{1:t}$ given y^t and the corresponding representation of the log evidence

$$p_\theta(x_{1:t}|y^t) = p_\theta(x_{1:t}, y^t)/p_\theta(y^t), \qquad \ell_t(\theta) = \log p_\theta(y^t) = \log\left(\textstyle\int p_\theta(x_{1:t}, y^t) \mathrm{d}x_{1:t}\right).$$

Variational Inference. For non-linear SSMs, the filtering and smoothing distributions as well as the log evidence are not available analytically and need to be approximated. For data y^t, a standard variational approach uses stochastic gradient techniques to maximize the following ELBO

$$\mathcal{L}_t(\theta, \phi) := \mathbb{E}_{q^\phi(x_{1:t}|y^t)}\left[\log\left(p_\theta(x_{1:t}, y^t)/q^\phi(x_{1:t}|y^t)\right)\right] \leq \ell_t(\theta).$$

Maximizing this ELBO w.r.t. ϕ corresponds to doing variational smoothing while maximizing it w.r.t. θ corresponds to doing parameter learning.

As the true smoothing distribution satisfies $p_\theta(x_{1:t}|y^t) = p_\theta(x_1|y^t) \prod_{k=2}^{t} p_\theta(x_k|y^t, x_{k-1})$, one usually uses the form $q^\phi(x_{1:t}|y^t) = q^\phi(x_1|y^t) \prod_{k=2}^{t} q^\phi(x_k|y^t, x_{k-1})$ for the variational smooth distribution; see e.g. [1, 12, 14, 27]. However this approach is not suitable for *online* variational filtering and parameter learning. When the new observation y_{t+1} is collected, this approach would require recomputing an entirely new variational smoothing distribution with a dimension that increases with time. One can attempt to bypass this problem by restricting ourselves to $q^\phi(x_{1:t}|y^t) = q^\phi(x_1|y^1) \prod_{k=2}^{t} q^\phi(x_k|y^k, x_{k-1})$ as per [19]. However, the switch from conditioning on y^t to y^k prohibits learning an accurate variational approximation of the true smoothing distribution as this formulation does not condition on all relevant data. Moreover, the marginal distribution $q^\phi(x_t|y^t)$ approximating the filtering distribution $p_\theta(x_t|y^t)$ is typically not available analytically.

3 Online Variational Filtering and Parameter Learning

To provide a tractable and practical means of getting around these issues, we now introduce our online variational filtering and parameter estimation approach. Namely, we will show how we can exploit a non-standard representation of the variational smoothing distribution $q^\phi(x_{1:t}|y^t)$ to compute the ELBO and its gradients w.r.t. θ and ϕ in an *online* manner as t increases, with a computational time that remains constant at each time step. This is achieved by using a combination of ideas from dynamic programming and RL. To simplify notation, henceforth we will write $q_t^\phi(x_{1:t}) = q^\phi(x_{1:t}|y^t)$.

3.1 Backward Decomposition of the Variational Smoothing Distribution

The key property that we will be exploiting is that the true smoothing distribution $p_\theta(x_{1:t}|y^t)$ satisfies the following backward decomposition

$$p_\theta(x_{1:t}|y^t) = p_\theta(x_t|y^t)\prod_{k=1}^{t-1} p_\theta(x_k|y^k, x_{k+1}), \quad p_\theta(x_k|y^k, x_{k+1}) = \frac{f_\theta(x_{k+1}|x_k)p_\theta(x_k|y^k)}{p_\theta(x_{k+1}|y^k)}. \quad (3)$$

Equation (3) shows that, conditional upon y^t, $(x_k)_{k=1}^t$ is a reverse-time Markov chain of initial distribution $p_\theta(x_t|y^t)$ and backward Markov transition kernels $p_\theta(x_k|y^k, x_{k+1})$; see e.g. [7, 11]. Crucially the backward transition kernel at time k depends only on the observations until time k. To exploit this, we consider a variational smoothing distribution of the form

$$q_t^\phi(x_{1:t}) = q_t^\phi(x_t)\prod_{k=1}^{t-1} q_{k+1}^\phi(x_k|x_{k+1}), \quad (4)$$

where $q_t^\phi(x_t)$ and $q_{k+1}^\phi(x_k|x_{k+1})$ are variational approximations of the filtering distribution $p_\theta(x_t|y^t)$ and the backward kernel $p_\theta(x_k|y^k, x_{k+1})$ respectively. Using (4), one now has

$$\begin{aligned}\mathcal{L}_t(\theta,\phi) = &\ell_t(\theta) - \mathrm{KL}(q_t^\phi(x_t)||p_\theta(x_t|y^t)) \\ &- \sum_{k=1}^{t-1} \mathbb{E}_{q_t^\phi(x_t)\prod_{\tau=k}^{t-1} q_{\tau+1}^\phi(x_\tau|x_{\tau+1})}\left[\mathrm{KL}(q_{k+1}^\phi(x_k|x_{k+1})||p_\theta(x_k|y^k, x_{k+1}))\right],\end{aligned}$$

where KL is the Kullback–Leibler divergence. Considering the backward decomposition (4) thus makes it possible to learn an arbitrarily accurate variational approximation of the true smoothing distribution whilst still only needing to condition on y^k at time k and not on future observations. Additionally, it follows directly from (4) that we can easily update $q_{t+1}^\phi(x_{1:t+1})$ from $q_t^\phi(x_{1:t})$ using

$$q_{t+1}^\phi(x_{1:t+1}) = q_t^\phi(x_{1:t})m_{t+1}^\phi(x_{t+1}|x_t), \text{ for } m_{t+1}^\phi(x_{t+1}|x_t) := \frac{q_{t+1}^\phi(x_t|x_{t+1})q_{t+1}^\phi(x_{t+1})}{q_t^\phi(x_t)}. \quad (5)$$

Here $m_{t+1}^\phi(x_{t+1}|x_t)$ can be viewed as an approximation of the Markov transition density $q_{t+1}^\phi(x_{t+1}|x_t) \propto q_{t+1}^\phi(x_t|x_{t+1})q_{t+1}^\phi(x_{t+1})$ but it is typically not a proper Markov transition density; i.e. $\int m_{t+1}^\phi(x_{t+1}|x_t)\mathrm{d}x_{t+1} \neq 1$ as $\int q_{t+1}^\phi(x_t|x_{t+1})q_{t+1}^\phi(x_{t+1})\mathrm{d}x_{t+1} \neq q_t^\phi(x_t)$.

Let us assume that $q_{k+1}^\phi(x_k|x_{k+1}) = q_{k+1}^{\phi_k}(x_k|x_{k+1})$ and $q_k^\phi(x_k) = q_k^{\phi_k}(x_k)$, then (4) and (5) suggest that we only need to estimate ϕ_t at time t as y_t does not impact the backward Markov kernels prior to time t. However, we also have to be able to compute estimates of $\nabla_\phi\mathcal{L}_t(\theta,\phi)$ and $\nabla_\theta\mathcal{L}_t(\theta,\phi)$ to optimize parameters in a computational time constant at each time step, without having to consider the entire history of observations y^t. This is detailed in the next subsections where we show that the sequence of ELBOs $\{\mathcal{L}_t(\theta,\phi)\}_{t\geq 1}$ and its gradients $\{\nabla_\theta\mathcal{L}_t(\theta,\phi)\}_{t\geq 1}$ and $\{\nabla_\phi\mathcal{L}_t(\theta,\phi)\}_{t\geq 1}$ can all be computed online when using the variational distributions $\{q_t^\phi(x_{1:t})\}_{t\geq 1}$ defined in (4).

3.2 Forward recursion for the ELBO

We start by presenting a forward-only recursion for the computation of $\{\mathcal{L}_t(\theta,\phi)\}_{t\geq 1}$. This recursion illustrates the parallels between variational inference and RL and is introduced to build intuition.

Proposition 1. *The ELBO $\mathcal{L}_t(\theta,\phi)$ satisfies for $t \geq 1$*

$$\mathcal{L}_t(\theta,\phi) = \mathbb{E}_{q_t^\phi(x_t)}[V_t^{\theta,\phi}(x_t)] \quad \textit{for} \quad V_t^{\theta,\phi}(x_t) := \mathbb{E}_{q_t^\phi(x_{1:t-1}|x_t)}\left[\log\left(p_\theta(x_{1:t}, y^t)/q_t^\phi(x_{1:t})\right)\right],$$

with the convention $V_1^{\theta,\phi}(x_1) := r_1^{\theta,\phi}(x_0, x_1) := \log(p_\theta(x_1, y_1)/q_1^\phi(x_1))$. *Additionally, we have*

$$V_{t+1}^{\theta,\phi}(x_{t+1}) = \mathbb{E}_{q_{t+1}^\phi(x_t|x_{t+1})}[V_t^{\theta,\phi}(x_t) + r_{t+1}^{\theta,\phi}(x_t, x_{t+1})], \quad \textit{where} \tag{6}$$

$$r_{t+1}^{\theta,\phi}(x_t, x_{t+1}) := \log\left(f_\theta(x_{t+1}|x_t) g_\theta(y_{t+1}|x_{t+1}) / m_{t+1}^\phi(x_{t+1}|x_t)\right). \tag{7}$$

Proposition 1, the proof of which is given in the supplementary material along with all others, shows that we can compute $\mathcal{L}_t(\theta, \phi)$, for $t \geq 1$, online by recursively computing the functions $V_t^{\theta,\phi}$ using (6) and then taking the expectation of $V_t^{\theta,\phi}$ w.r.t. $q_t^\phi(x_t)$ to obtain the ELBO at time t. Thus, given $V_t^{\theta,\phi}$, we can compute $V_{t+1}^{\theta,\phi}$ and $\mathcal{L}_{t+1}(\theta, \phi)$ using only y_{t+1}, with a cost that remains constant in t.

This type of recursion is somewhat similar to those appearing in RL. We can indeed think of the ELBO $\mathcal{L}_t(\theta, \phi)$ as the expectation of a sum of "rewards" $r_k^{\theta,\phi}$ given in (7) from $k = 1$ to $k = t$ which we compute recursively using the "value" function $V_t^{\theta,\phi}(x_t)$. However, while in RL the value function is given by the expectation of the sum of future rewards starting from x_t, the value function defined here is the expectation of the sum of past rewards conditional upon arriving in x_t, yielding the required forwards recursion instead of a backwards recursion.

3.3 Forward recursion for ELBO gradient w.r.t. θ

A similar forward-only recursion can also be obtained to compute $\{\nabla_\theta \mathcal{L}_t(\theta, \phi)\}_{t\geq 1}$. This recursion will be at the core of our online parameter learning algorithm. Henceforth, we will assume that regularity conditions allowing both differentiation and the interchange of integration and differentiation are satisfied.

Proposition 2. *The ELBO gradient* $\nabla_\theta \mathcal{L}_t(\theta, \phi)$ *satisfies, for* $t \geq 1$,

$$\nabla_\theta \mathcal{L}_t(\theta, \phi) = \mathbb{E}_{q_t^\phi(x_t)}[S_t^{\theta,\phi}(x_t)], \quad \textit{where} \quad S_t^{\theta,\phi}(x_t) := \nabla_\theta V_t^{\theta,\phi}(x_t).$$

Additionally, if we define $s_{t+1}^\theta(x_t, x_{t+1}) := \nabla_\theta r_{t+1}^{\theta,\phi}(x_t, x_{t+1}) = \nabla_\theta \log f_\theta(x_{t+1}|x_t) g_\theta(y_{t+1}|x_{t+1})$ *then*

$$S_{t+1}^{\theta,\phi}(x_{t+1}) = \mathbb{E}_{q_{t+1}^\phi(x_t|x_{t+1})}\left[S_t^{\theta,\phi}(x_t) + s_{t+1}^\theta(x_t, x_{t+1})\right]. \tag{8}$$

Proposition 2 shows that we can compute $\{\nabla_\theta \mathcal{L}_t(\theta, \phi)\}_{t\geq 1}$ online by propagating $\{S_t^{\theta,\phi}\}_{t\geq 1}$ using (8) and taking the expectation of the vector $S_t^{\theta,\phi}$ w.r.t. $q_t^\phi(x_t)$ to obtain the gradient at time t. Similar ideas have been previously exploited in the statistics literature to obtain a forward recursion to compute the score vector $\nabla_\theta \ell_t(\theta)$ so as to perform recursive maximum likelihood parameter estimation; see e.g. [11, Section 4]. In this case, one has $\nabla_\theta \ell_t(\theta) = \mathbb{E}_{p_\theta(x_t|y^t)}[S_t^\theta(x_t)]$ where S_t^θ satisfies a recursion similar to (8) with $q_{t+1}^\phi(x_t|x_{t+1})$ replaced by $p_\theta(x_t|y^t, x_{t+1})$.

3.4 Forward recursion for ELBO gradient w.r.t. ϕ

We finally establish forward-only recursions for the gradient of the ELBO w.r.t. ϕ which will allow us to perform online variational filtering. We consider the case where for all k $q_k^\phi(x_k) = q_k^{\phi_k}(x_k)$ and $q_k^\phi(x_{k-1}|x_k) = q_k^{\phi_k}(x_{k-1}|x_k)$ so $\mathcal{L}_t(\theta, \phi) = \mathcal{L}_t(\theta, \phi_{1:t})$ and $V_t^{\theta,\phi}(x_t) = V_t^{\theta,\phi_{1:t}}(x_t)$. At time step t, we will optimize w.r.t. ϕ_t and hold all previous $\phi_{1:t-1}$ constant. Our overall variational posterior (4) is denoted $q_t^{\phi_{1:t}}(x_{1:t})$.

Since the expectation is taken w.r.t. $q_t^{\phi_{1:t}}(x_{1:t})$ in $\mathcal{L}_t$, optimizing ϕ_t is slightly more difficult than for θ. However, we can still derive a forward recursion for the ϕ gradients and we will leverage the reparameterization trick to reduce the variance of the gradient estimates; i.e. we assume that $x_t(\phi_t; \epsilon_t) \sim q_t^{\phi_t}(x_t)$ and $x_{t-1}(\phi_t; \epsilon_{t-1}, x_t) \sim q_t^{\phi_t}(x_{t-1}|x_t)$ when $\epsilon_{t-1} \sim \lambda(\epsilon), \epsilon_t \sim \lambda(\epsilon)$.

Proposition 3. *The ELBO gradient* $\nabla_{\phi_t} \mathcal{L}_t(\theta, \phi_{1:t})$ *satisfies for* $t \geq 1$

$$\nabla_{\phi_t} \mathcal{L}_t(\theta, \phi_{1:t}) = \nabla_{\phi_t} \mathbb{E}_{q_t^{\phi_t}(x_t)}[V_t^{\theta,\phi_{1:t}}(x_t)] = \mathbb{E}_{\lambda(\epsilon_t)}[\nabla_{\phi_t} V_t^{\theta,\phi_{1:t}}(x_t(\phi_t; \epsilon_t))].$$

Additionally, one has

$$\nabla_{\phi_{t+1}} V_{t+1}^{\theta,\phi_{1:t+1}}(x_{t+1}(\phi_{t+1};\epsilon_{t+1}))$$
$$= \mathbb{E}_{\lambda(\epsilon_t)}\Bigg[T_t^{\theta,\phi_{1:t}}(x_t(\phi_{t+1};\epsilon_t, x_{t+1}(\phi_{t+1};\epsilon_{t+1})))\frac{\mathrm{d}x_t(\phi_{t+1};\epsilon_t, x_{t+1}(\phi_{t+1};\epsilon_{t+1}))}{\mathrm{d}\phi_{t+1}}$$
$$+ \nabla_{\phi_{t+1}} r_{t+1}^{\theta,\phi_{t:t+1}}(x_t(\phi_{t+1};\epsilon_t, x_{t+1}(\phi_{t+1};\epsilon_{t+1})), x_{t+1}(\phi_{t+1};\epsilon_{t+1}))\Bigg],$$

where $T_t^{\theta,\phi_{1:t}}(x_t) := \frac{\partial}{\partial x_t} V_t^{\theta,\phi_{1:t}}(x_t)$ *satisfies the forward recursion*

$$T_{t+1}^{\theta,\phi_{1:t+1}}(x_{t+1}) = \mathbb{E}_{\lambda(\epsilon_t)}\Bigg[T_t^{\theta,\phi_{1:t}}(x_t(\phi_{t+1};\epsilon_t, x_{t+1}))\frac{\partial x_t(\phi_{t+1};\epsilon_t, x_{t+1})}{\partial x_{t+1}} + \nabla_{x_{t+1}} r_{t+1}^{\theta,\phi_{t:t+1}}(x_t(\phi_{t+1};\epsilon_t, x_{t+1}), x_{t+1})\Bigg]. \quad (9)$$

Here, $\frac{\mathrm{d}x_t(\phi_{t+1};\epsilon_t, x_{t+1}(\phi_{t+1};\epsilon_{t+1}))}{\mathrm{d}\phi_{t+1}}$, $\frac{\partial x_t(\phi_{t+1};\epsilon_t, x_{t+1})}{\partial x_{t+1}}$ *are Jacobians of appropriate dimensions.*

3.5 Estimating the ELBO and its Gradients

Since at time t we optimize ϕ_t and hold $\phi_{1:t-1}$ constant, we have $S_t^{\theta,\phi}(x_t) = S_t^{\theta,\phi_{1:t}}(x_t)$. Practically, we are not able to compute in closed-form the functions $V_t^{\theta,\phi_{1:t}}(x_t)$, $S_t^{\theta,\phi_{1:t}}(x_t)$ and $T_t^{\theta,\phi_{1:t}}(x_t)$ appearing in the forward recursions of $\mathcal{L}_t(\theta,\phi_{1:t})$, $\nabla_\theta \mathcal{L}_t(\theta,\phi_{1:t})$ and $\nabla_\phi \mathcal{L}_t(\theta,\phi_{1:t})$ respectively. However, we can exploit the above recursions to approximate the true functions online using regression as is commonly done in RL. We then show how to use these gradients for online filtering and parameter learning.

We approximate $S_{t+1}^{\theta,\phi_{1:t+1}}$ with $\hat{S}_{t+1}$. Equation (8) shows that $\hat{S}_{t+1}$ can be learned using $\hat{S}_t$ through regression of the simulated dataset[1] $\{x_{t+1}^i, \hat{S}_t(x_t^i) + s_{t+1}^\theta(x_t^i, x_{t+1}^i)\}$ with $(x_t^i, x_{t+1}^i) \overset{\text{i.i.d.}}{\sim} q_{t+1}^{\phi_{t+1}}(x_t, x_{t+1})$ for $i = 1, ..., N$ (see Supplementary Material A.1 for derivation). We can use neural networks to model $\hat{S}_t$ or Kernel Ridge Regression (KRR). The use of KRR to estimate gradients for variational learning has recently been demonstrated by [28].

We similarly approximate $T_{t+1}^{\theta,\phi_{1:t+1}}$ with $\hat{T}_{t+1}$. As before, we can model $\hat{T}_{t+1}$ using neural networks or KRR. We use recursion (9) and $\hat{T}_t$ to create the following dataset for regression[2]

$$\Big\{x_{t+1}^i,\ \ \hat{T}_t(x_t(\phi_{t+1};\epsilon_t^i, x_{t+1}^i))\frac{\partial x_t(\phi_{t+1};\epsilon_t^i, x_{t+1}^i)}{\partial x_{t+1}^i} + \nabla_{x_{t+1}^i} r_{t+1}^{\theta,\phi_{t:t+1}}(x_t(\phi_{t+1};\epsilon_t^i, x_{t+1}^i), x_{t+1}^i)\Big\},$$

where $x_{t+1}^i \sim q_{t+1}^{\phi_{t+1}}(x_{t+1})$ and $\epsilon_t^i \sim \lambda(\epsilon)$ for $i = 1, ..., N$.

Note that if one is interested in computing online an approximation of the ELBO, we can again similarly approximate $V_t^{\theta,\phi_{1:t}}(x_t)$ using regression to obtain $\hat{V}_t(x_t)$ by leveraging (6). We will call the resulting approximate ELBO the Recursive ELBO (RELBO). We could also then differentiate $\hat{V}_t(x_t)$ w.r.t. x_t to obtain an alternative method for estimating $T_t^{\theta,\phi_{1:t}}$. However, we are ultimately interested in accurate gradients and during the regression this approach exploits none of the readily available gradient information. Indeed, we found in early experiments this approach does not work as well as approximating $T_t^{\theta,\phi_{1:t}}$ with $\hat{T}_t$ directly.

By approximating $T_{t+1}^{\theta,\phi_{1:t+1}}$ with $\hat{T}_{t+1}$ and $S_{t+1}^{\theta,\phi_{1:t+1}}$ with $\hat{S}_{t+1}$, we introduce some bias into our gradient estimates. We can trade bias for variance by using modified recursions; e.g.

$$S_{t+1}^{\theta,\phi_{1:t+1}}(x_{t+1}) = \mathbb{E}_{q_t^{\phi_{t-L+2:t+1}}(x_{t-L+1:t}|x_{t+1})}[S_{t-L+1}^{\theta,\phi_{1:t-L+1}}(x_{t-L+1}) + \textstyle\sum_{k=t-L+1}^{t} s_k^\theta(x_k, x_{k+1})].$$

As L increases, we will reduce bias but increase variance. Such ideas are also commonly used in RL but we will limit ourselves here to using $L = 1$.

[1] We define $\hat{S}_0 := 0$, $x_0^i := \emptyset$, $s_1^\theta(x_0, x_1) = \nabla_\theta \log \mu_\theta(x_1) g_\theta(y_1|x_1)$ and $q_1^{\phi_1}(x_0, x_1) := q_1^{\phi_1}(x_1)$.

[2] Similarly, we define $\hat{T}_0 := 0$, $x_0^i := \emptyset$ and $r_1^{\theta,\phi_{0:1}}$ as in Proposition 1.

Algorithm 1: Online Variational Filtering and Parameter Learning

for $t = 1, \ldots, T$ **do**

  /* Update ϕ_t using M stochastic gradient steps */

  for $m = 1, \ldots, M$ **do**

    Sample $x_{t-1}^i, x_t^i \sim q_t^{\phi_t}(x_{t-1}, x_t)$ using reparametrization trick for $i = 1, ..., N$

    $\phi_t \leftarrow \phi_t + \gamma_m \frac{1}{N} \sum_{i=1}^N \{\hat{T}_{t-1}(x_{t-1}^i) \frac{dx_{t-1}^i}{d\phi_t} + \nabla_{\phi_t} r_t(x_{t-1}^i, x_t^i)\}$

  end

  /* Update $\hat{T}_t(x_t)$ and $\hat{S}_t(x_t)$ as in Section 3.5 */

  $\hat{T}_t(x_t) \overset{regression}{\leftarrow} \hat{T}_{t-1}(x_{t-1}) \frac{\partial x_{t-1}}{\partial x_t} + \nabla_{x_t} r_t(x_{t-1}, x_t)$.

  $\hat{S}_t(x_t) \overset{regression}{\leftarrow} \hat{S}_{t-1}(x_{t-1}) + s_t^{\theta_{t-1}}(x_{t-1}, x_t)$.

  /* Update θ using a stochastic gradient step */

  Sample $x_{t-1}^i, x_t^i \sim q_t^{\phi_t}(x_{t-1}, x_t), \quad \tilde{x}_{t-1}^i \sim q_{t-1}^{\phi_{t-1}}(x_{t-1})$ for $i = 1, ..., N$

  $\theta_t \leftarrow \theta_{t-1} + \eta_t \frac{1}{N} \sum_{i=1}^N \{\hat{S}_{t-1}(x_{t-1}^i) + s_t^{\theta_{t-1}}(x_{t-1}^i, x_t^i) - \hat{S}_{t-1}(\tilde{x}_{t-1}^i)\}$

end

3.6 Online Parameter Estimation

Assume, for the sake of argument, that one has access to the log evidence $\ell_t(\theta)$ and that the observations arise from the SSM with parameter $\theta^\star$. Under regularity conditions, the average log-likelihood $\ell_t(\theta)/t$ converges as $t \to \infty$ towards a function $\ell(\theta)$ which is maximized at $\theta^\star$; see e.g. [7, 25]. We can maximize this criterion on-line using Recursive Maximum Likelihood Estimation (RMLE) [11, 16, 24, 25] which consists of updating the parameter estimate θ using

$$\theta_{t+1} = \theta_t + \eta_{t+1} \left(\mathbb{E}_{p_{\theta_{0:t}}(x_t, x_{t+1}|y^{t+1})}[S_t(x_t) + s_{t+1}^{\theta_t}(x_t, x_{t+1})] - \mathbb{E}_{p_{\theta_{0:t-1}}(x_t|y^t)}[S_t(x_t)] \right). \quad (10)$$

The term in brackets on the r.h.s. of (10) is an approximation of the gradient of the log-predictive $p_\theta(y_{t+1}|y^t)$ evaluated at θ_t. It is given by $\nabla \log p_{\theta_{0:t}}(y^{t+1}) - \nabla \log p_{\theta_{0:t-1}}(y^t)$ with the notation $\nabla \log p_{\theta_{0:t}}(y^{t+1})$ corresponding to the expectation of the sum of terms $s_{k+1}^{\theta_k}(x_k, x_{k+1})$ w.r.t. the joint posterior states distribution defined by using the SSM with parameter θ_k at time $k+1$.

We proceed similarly in the variational context and update the parameter using

$$\theta_{t+1} = \theta_t + \eta_{t+1} \left(\mathbb{E}_{q_{t+1}^{\phi_{t+1}}(x_t, x_{t+1})} \left[\hat{S}_t(x_t) + s_{t+1}^{\theta_t}(x_t, x_{t+1}) \right] - \mathbb{E}_{q_t^{\phi_t}(x_t)} \left[\hat{S}_t(x_t) \right] \right).$$

Here $\hat{S}_t(x_t)$ is an approximation of $S_t(x_t)$ satisfying $S_{t+1}(x_{t+1}) := \mathbb{E}_{q^{\phi_{t+1}}(x_t|x_{t+1})}[S_t(x_t) + s_{t+1}^{\theta_t}(x_t, x_{t+1})]$. We compute $\hat{S}_t$ as in Section 3.5 with a dataset using $\hat{S}_{t-1}$ and θ_{t-1}.

Putting everything together, we summarize our method in Algorithm 1 using a simplified notation to help build intuition. We re-iterate the algorithm computation cost does not grow with t. We need only store fixed size $\hat{T}$ and $\hat{S}$ models as well as the most recent ϕ_t and θ_t parameters. When performing backpropagation, $\hat{T}$ and $\hat{S}$ summarize all previous gradients, meaning we do not have to roll all the way back to $t = 1$. Therefore, we only incur standard backpropagation computational cost w.r.t. ϕ and θ. To scale to large d_x, we can use mean field $q_t^{\phi_t}(x_t)$, $q_t^{\phi_t}(x_{t-1}|x_t)$ keeping costs linear in d_x.

4 Related Work

Both [27] and [12] have explored the use of RL ideas in the context of variational inference. [12] approximates the likelihood of future points in a SSM using temporal difference learning [23], but the proposed algorithm is not online. [27] presents a backward-in-time recursion for the ELBO initialized at the time of the last observation for a specific SSM. However, when a new observation is collected at the next time step, the backward recursion has to be re-run which would lead to a computational time increasing linearly at each time step.

To perform online variational inference, [31] proposes to use the decomposition (2) of the log evidence $\ell_t(\theta)$ and lower bound each term $\log p_\theta(y_k|y^{k-1})$ appearing in the sum using

$$\mathbb{E}_{q_k^\phi(x_{k-1}, x_k)} \left[\log \frac{f_\theta(x_k|x_{k-1}) g_\theta(y_k|x_k) p_\theta(x_{k-1}|y^{k-1})}{q_k^\phi(x_{k-1}, x_k)} \right] \leq \log p_\theta(y_k|y^{k-1}). \quad (11)$$

Unfortunately, the term on the l.h.s. of (11) cannot be evaluated unbiasedly as it relies on the intractable filter $p_\theta(x_{k-1}|y^{k-1})$ so [31] approximates it by $p_\theta(x_{k-1}|y^{k-1}) \approx q^\phi_{k-1}(x_{k-1})$ to obtain the following *Approximate* ELBO (AELBO) by summing over k:

$$\widetilde{\mathcal{L}}_t(\theta, \phi_{1:t}) = \sum_{k=1}^{t} \mathbb{E}_{q^\phi_k(x_{k-1},x_k)} \left[\log r^{\theta,\phi}_k(x_{k-1}, x_k)\right]. \tag{12}$$

[31] makes the additional assumption $q^\phi_k(x_{k-1}, x_k) = q^\phi_{k-1}(x_{k-1})q^\phi_k(x_k)$ and we will refer to (12) in this case as AELBO-1. While [10] does not consider online learning, their objective is actually a generalization of [31], with $q^\phi_k(x_{k-1}, x_k) = q^\phi_k(x_k)q^\phi_k(x_{k-1}|x_k)$, and we will refer to (12) in this case as AELBO-2. It can be easily shown that AELBO-2 is only equal to the true ELBO given in Proposition 1 in the unrealistic scenario where $q^\phi_k(x_k) = p_\theta(x_k|y^k)$ for all k. Moreover, in both cases the term $p_\theta(x_{k-1}|y^{k-1})$ is replaced by $q^\phi_{k-1}(x_{k-1})$, causing a term involving θ to be ignored in gradient computation. The approach developed here can be thought of as a way to correct the approximate ELBOs computed in [10, 31] in a principled manner, which takes into account the discrepancy between the filtering and approximate filtering distributions, and maintains the correct gradient dependencies in the computation graph. Finally [30] relies on PF to do online variational inference. However the variational approximation of the filtering distribution is only implicit at its expression includes an intractable expectation and, like any other PF techniques, it will scale poorly when the state is high dimensional [4].

5 Experiments

5.1 Linear Gaussian State-Space Models

We first consider a linear Gaussian SSM for which the filtering distributions can be computed using the KF and the analytic RMLE scheme (10) can be implemented. Here the model is defined as

$$f_\theta(x_t|x_{t-1}) = \mathcal{N}(x_t; Fx_{t-1}, U), \quad g_\theta(y_t|x_t) = \mathcal{N}(y_t; Gx_t, V),$$

where $F \in \mathbb{R}^{d_x \times d_x}$, $G \in \mathbb{R}^{d_y \times d_x}$, $U \in \mathbb{R}^{d_x \times d_x}$, $V \in \mathbb{R}^{d_y \times d_y}$, $\theta = \{F, G\}$. We let $q^{\phi_t}_t$ be

$$q^{\phi_t}_t(x_t) = \mathcal{N}\left(x_t; \mu_t, \text{diag}(\sigma^2_t)\right), \quad q^{\phi_t}_t(x_{t-1}|x_t) = \mathcal{N}\left(x_{t-1}; W_t x_t + b_t, \text{diag}(\tilde{\sigma}^2_t)\right),$$

with $\phi_t = \{\mu_t, \log \sigma_t, W_t, b_t, \log \tilde{\sigma}_t\}$. In our experiments, we set the matrices F, G, U, V to be diagonal, so that $p_\theta(x_t|y^t)$ and $p_\theta(x_{t-1}|y^{t-1}, x_t)$ are in the variational family.

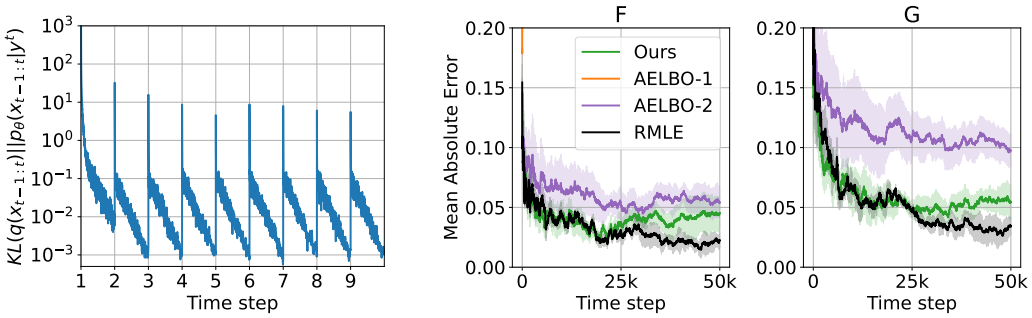
For $d_x = d_y = 10$, we first demonstrate accurate state inference by learning ϕ_t at each time step whilst holding θ fixed at the true value. We represent $\hat{T}_t(x_t)$ non-parametrically using KRR. Full details for all experiments are given in the supplement. Figure 1a illustrates how given extra computation, our variational approximation comes closer and closer to the ground truth, the accuracy being limited by the convergence of each inner stochastic gradient ascent procedure. We then consider online learning of the parameters F and G using Algorithm 1 comparing our result to RMLE and a variation of Algorithm 1 using AELBO-1 and 2 (see Section 4). Our methodology converges much closer to the analytic baseline (RMLE) than AELBO-2 [10] and exhibits less variance, even though the variational family is sufficiently expressive for AELBO-2 to learn the correct backward transition. In addition, we find that AELBO-1 [31] did not produce reliable parameter estimates in this example, as it relies on a crude variational approximation that ignores the dependence between x_{k-1} and x_k. As expected, our method does not perform as well as the analytic RMLE, as inevitably small errors will be introduced during stochastic optimization and regression.

5.2 Chaotic Recurrent Neural Network

We next evaluate the performance of our algorithm for state estimation in non-linear, high-dimensional SSMs. We reproduce the Chaotic Recurrent Neural Network (CRNN) example in [30], but with state dimension $d_x = 5, 20$, and 100. The non-linear model is given by an Euler approximation of the continuous-time recurrent neural network dynamics

$$f(x_t|x_{t-1}) = \mathcal{N}\left(x_t; x_{t-1} + \Delta\tau^{-1}\left(-x_{t-1} + \gamma W \tanh(x_{t-1})\right), U\right),$$

and the observation is a linear model with an additive noise from a Student's t-distribution. We compare our algorithm against ensemble KF, bootstrap PF, as well as variational methods



(a) (b)

Figure 1: **(a)** $\mathrm{KL}(q_t^{\phi_t}(x_{t-1}, x_t)||p_\theta(x_{t-1}, x_t|y^t))$ vs time step of the SSM. Between each time step, we plot the progress of the KL over 5000 iterations of inner loop ϕ_t optimization. **(b)** Mean Absolute Error for model parameters F (left) and G (right) vs time step (AELBO-1 off the scale).

using AELBO-1 and AELBO-2. We let $q_t^{\phi_t}(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \mathrm{MLP}_t^{\phi_t}(x_t), \mathrm{diag}(\tilde{\sigma}_t^2))$ and $q_t^{\phi_t}(x_t) = \mathcal{N}\left(x_t; \mu_t, \mathrm{diag}(\sigma_t^2)\right)$ where we use a 1-layer Multi-Layer Perceptron (MLP) with 100 neurons for each $q_t^{\phi_t}(x_{t-1}|x_t)$. We generate a dataset of length 100 using the same settings as [30], and each algorithm is run 10 times to report the mean and standard deviation. We also match approximately the computational complexity for all methods. From Table 1, we observe that the EnKF performs poorly on this non-linear model, while the PF performance degrades significantly with d_x in line with theoretical results. Among variational methods, AELBO-1 does not give as accurate state estimation, while AELBO-2 and our method achieve the lowest error in terms of RMSE. However, our method achieves the highest ELBO i.e. lowest KL between the variational approximation and the true posterior since θ is fixed - an effect not represented using just the RMSE. We confirm this is the case in the supplement by comparing our variational filter means μ_t against the 'ground truth' posterior mean for $d_x = 5$ computed using PF with 10 million particles. Furthermore, our method is also able to accurately estimate the true ELBO online. Figure 2 shows that our online estimate of the ELBO, RELBO (Section 3.5), is very close to the true ELBO, whereas AELBO-2 is biased and consistently overestimates it. Further, AELBO-1 is extremely loose meaning its posterior approximation is very poor.

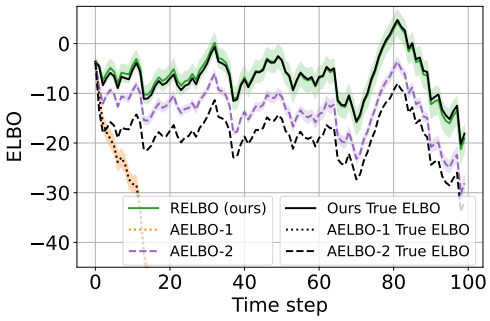


Figure 2: Estimates and true values of the ELBO on the Chaotic RNN task. RELBO uses KRR for $\hat{V}_t$ whilst for the other methods we use eq. (12).

5.3 Sequential Variational Auto-Encoder

We demonstrate the scalability of our method on a sequential VAE application. In this problem, an agent observes a long sequence of frames that could, for example, come from a robot traversing a new environment. The frames are encoded into a latent representation using a pre-trained decoder. The agent must then learn online the transition dynamics within this latent space using the single stream of input images. The SSM is summarized as

$$f_\theta(x_t|x_{t-1}) = \mathcal{N}(x_t; \mathrm{NN}_\theta^f(x_{t-1}), U), \qquad g(y_t|x_t) = \mathcal{N}(y_t; \mathrm{NN}^g(x_t), V),$$

where $d_x = 32$, NN_θ^f is a residual MLP and NN^g a convolutional neural network. NN_θ^f is learned online whilst NN^g is fixed and is pre-trained on unordered images from similar environments using the standard VAE objective [13]. We perform this experiment on a video sequence from a DeepMind Lab environment [3] (GNU GPL license). We use the same $q_t^{\phi_t}$ parameterization as for the CRNN but with a 2 hidden layer MLP with 64 neurons. KRR is used to learn $\hat{T}_t$ whereas we use an MLP for learning $\hat{S}_t$. We found that MLPs scale better than KRR as d_θ is high. Our online algorithm is run

Table 1: Root Mean Squared Error between filtering mean and true state and the average true ELBO for the 5 methods in varying dimensions on the Chaotic RNN task.

d_x		EnKF	BPF	AELBO-1	AELBO-2	Ours
5	Filter RMSE	0.1450±0.0026	**0.1026±0.0001**	0.1284±0.0035	0.1035±0.0012	0.1032±0.0005
	ELBO (nats)	-	-	-220.52±6.2768	-30.944±2.2928	**-15.845±1.7385**
	Time per step	1.0998	0.9268	1.5067	2.2270	2.6899
20	Filter RMSE	0.1541±0.0016	0.1092±0.0014	0.1355±0.0012	0.1086±0.0004	**0.1082±0.0003**
	ELBO (nats)	-	-	-928.80±10.463	-393.68±3.9053	**-340.36±3.9730**
	Time per step	5.1879	3.8932	2.3587	2.7000	3.5935
100	Filter RMSE	0.1571±0.0017	0.2493±0.0122	0.1239±0.0006	0.1070±0.0001	**0.1068±0.0001**
	ELBO (nats)	-	-	-4247.9±20.905	-2069.7±11.814	**-1794.7±5.4173**
	Time per step	6.4546	4.6184	3.2697	4.5539	5.9263

(a) Before training

(b) After training

Figure 3: Frames predicted by rolling out NN_θ^f from two different starting points, before and after training. Between each frame, 3 transition steps are taken.

on a sequence of 4000 images after which we probe the quality of the learned NN_θ^f. The results are shown in Figure 3. Before training, NN_θ^f predicts no meaningful change but after training it predicts movements the agent could realistically take. We quantify this further in the supplement by showing that the approximate average log likelihood $\ell_t(\theta)/t$ computed using Monte Carlo increases through training, thereby confirming our method can successfully learn high-dimensional model parameters of the agent movement in a fully online fashion.

6 Discussion

Limitations. As with any stochastic variational inference technique, we need to ensure our variational family is expressive enough and our stochastic gradient method finds a reasonable minimum to obtain an accurate approximate posterior. We should also use function approximators that are flexible enough to keep the bias in our gradient estimates small. In essence, we have re-framed the problem of accurate online inference as a supervised learning task using flexible function approximators, something which the deep learning community is very adept at. Further applications could borrow more RL and deep learning ideas such as variance reduction techniques for gradient estimates, and meta-learning to improve regression accuracy and reduce the constant time cost per iteration.

Conclusion. In this paper, we presented a fully online approach for performing variational state estimation and parameter learning. We use the backward decomposition of the variational smoothing distribution along with forward recursions for gradients to obtain a method with constant time computational cost. This obviates the need for the additional assumptions of prior work that preclude accurate variational approximations. In our experiments, we demonstrated that the methodology outperforms standard approaches for non-linear filtering such as the Ensemble Kalman Filter and Particle Filter when the state is high dimensional. We also validated its ability to learn high dimensional model parameters by training neural networks online in a sequential VAE model.

References

[1] Archer, E., Park, I. M., Buesing, L., Cunningham, J., and Paninski, L. (2015). Black box variational inference for state space models. *arXiv preprint arXiv:1511.07367*.

[2] Beard, M., Vo, B. T., and Vo, B.-N. (2020). A solution for large-scale multi-object tracking. *IEEE Transactions on Signal Processing*, 68:2754–2769.

[3] Beattie, C., Leibo, J. Z., Teplyashin, D., Ward, T., Wainwright, M., Küttler, H., Lefrancq, A., Green, S., Valdés, V., Sadik, A., et al. (2016). Deepmind lab. *arXiv preprint arXiv:1612.03801*.

[4] Bengtsson, T., Bickel, P., Li, B., et al. (2008). Curse-of-dimensionality revisited: Collapse of the particle filter in very large scale systems. In *Probability and Statistics: Essays in Honor of David A. Freedman*, pages 316–334. Institute of Mathematical Statistics.

[5] Chung, J., Kastner, K., Dinh, L., Goel, K., Courville, A., and Bengio, Y. (2015). A recurrent latent variable model for sequential data. In *Advances in Neural Information Processing Systems*.

[6] Courts, J., Hendriks, J., Wills, A., Schön, T., and Ninness, B. (2020). Variational state and parameter estimation. *arXiv preprint arXiv:2012.07269*.

[7] Douc, R., Moulines, E., and Stoffer, D. (2014). *Nonlinear Time Series: Theory, Methods and Applications with R Examples*. CRC press.

[8] Evensen, G. (2009). *Data Assimilation: The Ensemble Kalman Filter*. Springer Science & Business Media.

[9] Fraccaro, M., Sønderby, S. K., Paquet, U., and Winther, O. (2016). Sequential neural models with stochastic layers. In *Advances in Neural Information Processing Systems*.

[10] Gregor, K., Papamakarios, G., Besse, F., Buesing, L., and Weber, T. (2019). Temporal difference variational auto-encoder. In *International Conference on Learning Representations*.

[11] Kantas, N., Doucet, A., Singh, S. S., Maciejowski, J., and Chopin, N. (2015). On particle methods for parameter estimation in state-space models. *Statistical Science*, 30(3):328–351.

[12] Kim, G.-H., Jang, Y., Yang, H., and Kim, K.-E. (2020). Variational inference for sequential data with future likelihood estimates. In *International Conference on Machine Learning*.

[13] Kingma, D. and Welling, M. (2014). Auto-encoding variational Bayes. In *International Conference on Learning Representations*.

[14] Krishnan, R. G., Shalit, U., and Sontag, D. (2017). Structured inference networks for nonlinear state space models. In *AAAI Conference on Artificial Intelligence*, pages 2101–2109.

[15] Le, T. A., Igl, M., Rainforth, T., Jin, T., and Wood, F. (2018). Auto-encoding sequential Monte Carlo. In *International Conference on Learning Representations*.

[16] LeGland, F. and Mevel, L. (1997). Recursive estimation in hidden Markov models. In *Proceedings of the 36th IEEE Conference on Decision and Control*, volume 4, pages 3468–3473.

[17] Ma, X., Karkus, P., Hsu, D., Lee, W. S., and Ye, N. (2020). Discriminative particle filter reinforcement learning for complex partial observations. *International Conference on Learning Representations*.

[18] Maddison, C. J., Lawson, J., Tucker, G., Heess, N., Norouzi, M., Mnih, A., Doucet, A., and Teh, Y. (2017). Filtering variational objectives. *Advances in Neural Information Processing Systems*.

[19] Marino, J., Cvitkovic, M., and Yue, Y. (2018). A general method for amortizing variational filtering. *Advances in Neural Information Processing Systems*.

[20] Naesseth, C. A., Linderman, S. W., Ranganath, R., and Blei, D. M. (2018). Variational sequential Monte Carlo. In *International Conference on Artificial Intelligence and Statistics*.

[21] Richter, J., Carbajal, G., and Gerkmann, T. (2020). Speech enhancement with stochastic temporal convolutional networks. In *Interspeech*.

[22] Särkkä, S. (2013). *Bayesian Filtering and Smoothing*. Cambridge University Press.

[23] Sutton, R. S. and Barto, A. G. (2018). *Reinforcement Learning: An Introduction*. MIT press.

[24] Tadić, V. B. (2010). Analyticity, convergence, and convergence rate of recursive maximum-likelihood estimation in hidden Markov models. *IEEE Transactions on Information Theory*, 56(12):6406–6432.

[25] Tadić, V. Z. and Doucet, A. (2021). Asymptotic properties of recursive particle maximum likelihood estimation. *IEEE Transactions on Information Theory*, 67(3):1825–1848.

[26] Tsay, R. S. (2005). *Analysis of Financial Time Series*, volume 543. John Wiley & Sons.

[27] Weber, T., Heess, N., Eslami, A., Schulman, J., Wingate, D., and Silver, D. (2015). Reinforced variational inference. In *Advances in Neural Information Processing Systems Workshop*.

[28] Wenliang, L. K., Moskovitz, T., Kanagawa, H., and Sahani, M. (2020). Amortised learning by wake-sleep. In *International Conference on Machine Learning*.

[29] Yingzhen, L. and Mandt, S. (2018). Disentangled sequential autoencoder. In *International Conference on Machine Learning*.

[30] Zhao, Y., Nassar, J., Jordan, I., Bugallo, M., and Park, I. M. (2019). Streaming variational Monte Carlo. *arXiv preprint arXiv:1906.01549*.

[31] Zhao, Y. and Park, I. M. (2020). Variational online learning of neural dynamics. *Frontiers in Computational Neuroscience*, 14.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] Our novel method is described in Section 3 and the promised performance is demonstrated in Section 5
   (b) Did you describe the limitations of your work? [Yes] See Section 6
   (c) Did you discuss any potential negative societal impacts of your work? [Yes] See Supplementary Material C
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes] We state the set up for our results in Section 2
   (b) Did you include complete proofs of all theoretical results? [Yes] We include all the proofs in Supplementary Material B
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] See Supplementary Material A.2 A.3, A.4
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] See Figure 1b, Figure 2 and Table 1
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] See Supplementary Material A.2 A.3, A.4
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes] See Section 5.3 and Supplementary Material A.4

(b) Did you mention the license of the assets? [Yes] See Section 5.3 and Supplementary Material A.4

(c) Did you include any new assets either in the supplemental material or as a URL? [N/A]

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]

5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]