Variational Marginal Particle Filters

Jinlin Lai **Justin Domke** **Daniel Sheldon**
Manning College of Information and Computer Sciences, University of Massachusetts Amherst
{jinlinlai,domke,sheldon}@cs.umass.edu

Abstract

Variational inference for state space models (SSMs) is known to be hard in general. Recent works focus on deriving variational objectives for SSMs from unbiased sequential Monte Carlo estimators. We reveal that the marginal particle filter is obtained from sequential Monte Carlo by applying Rao-Blackwellization operations, which sacrifices the trajectory information for reduced variance and differentiability. We propose the variational marginal particle filter (VMPF), which is a differentiable and reparameterizable variational filtering objective for SSMs based on an unbiased estimator. We find that VMPF with biased gradients gives tighter bounds than previous objectives, and the unbiased reparameterization gradients are sometimes beneficial.

1 Introduction

Sequential data are often described by state space models (SSMs), where the latent variables x and the observed variables y vary over time. An SSM is defined as

$$p(x_{1:T}, y_{1:T}) = f(x_1) \prod_{t=2}^{T} f(x_t|x_{t-1}) \prod_{t=1}^{T} g(y_t|x_t). \quad (1)$$

With this model, given one or more observed sequences $y_{1:T}$, two tasks are of interest:

- What is the closest distribution $q(x_{1:T}; \varphi)$ to the posterior distribution $p(x_{1:T}|y_{1:T})$?
- What are the best parameters θ for $p(x_{1:T}, y_{1:T})$ to model the observed data?

These two tasks can be simultaneously solved by variational inference (VI) (Blei et al., 2016). Recent advances (Naesseth et al., 2018; Maddison et al., 2017a; Le et al., 2018) improve the performance of VI on SSMs by using sequential Monte Carlo (SMC) or the particle filter (PF) (Naesseth et al., 2019; Doucet and Johansen, 2009) to define a variational objective, which gives *variational* sequential Monte Carlo (VSMC). However, the SMC resampling step is problematic: sampling of discrete ancestor variables is non-differentiable, and thus reparameterization gradient estimators (Kingma and Welling, 2014; Rezende et al., 2014) cannot be used. Most practical implementations use a biased gradient estimator instead. Recently, Corenflos et al. (2021) proposed a fully differentiable particle filter, which gives up the guarantee of being a lower bound but approximates VSMC well.

This paper makes three contributions. First, we give a novel proof of the correctness of SMC in terms of transformations of estimator-coupling pairs (Domke and Sheldon, 2019). This provides a high-level view of SMC that complements existing perspectives. Second, we marginalize (Rao-Blackwellize) over the choice of ancestor variables in each step to get the marginal particle filter (MPF) (Klaas et al., 2005), which gives a novel proof of correctness of MPF and reveals the direct relationship between MPF and SMC. Third, we propose to optimize a variational bound based on MPF, which we call the variational marginal particle filter (VMPF). Since Rao-Blackwellization reduces the variance of estimators, we expect that the VMPF bound is tighter than VSMC and leads to better inference and learning. Furthermore, we observe that MPF can be rephrased without discrete ancestor variables as sampling from mixtures. As it is often possible to efficiently differentiate through mixtures (currently, this is possible for Gaussian or product distributions) (Graves, 2016; Figurnov et al., 2018), *unbiased* reparameterization gradients can be computed for the VMPF when suitable proposal distributions are used.

We conduct experiments using the VMPF on linear Gaussian SSMs, stochastic volatility models and deep

Proceedings of the 25th International Conference on Artificial Intelligence and Statistics (AISTATS) 2022, Valencia, Spain. PMLR: Volume 151.

Markov models (Krishnan et al., 2017). Our first aim is to understand the significance of unbiased gradients. We confirm that these do indeed lead to tighter bounds given sufficient optimization, especially with higher numbers of particles, but that a biased gradient estimator typically performs better on complex models due to lower gradient variance. Our second aim is to understand the significance of the lower variance of the MPF vs SMC for estimating the log-likelihood. We find that, as expected, this manifests as the VMPF bound being tighter than the VSMC bound, which implies better posterior distributions and better models.

We note that, while marginalizing ancestor variables is often beneficial, it prevents applying VMPF to some models, such as VRNNs (Chung et al., 2015), which require access to those variables (see Section 7). The time complexity of VMPF with N particles is $O(N^2)$ (compared to $O(N)$ for VSMC), but in many practical situations the $O(N^2)$ component is dominated by an $O(N)$ component with higher constants.

2 Background

Variational Inference and Couplings. Variational inference seeks to produce an approximation $q(x_{1:T}; \varphi) \approx p(x_{1:T}|y_{1:T})$ given a joint distribution $p(x_{1:T}, y_{1:T})$ such as the SSM in Equation (1) and fixed observations $y_{1:T}$. It is based on the fact that minimizing the KL divergence between the two distributions is equivalent to maximizing the evidence lower bound (ELBO) (Jordan et al., 1999) $L_{\text{ELBO}}(\varphi) = \mathrm{E}_{q(x_{1:T};\varphi)}[\log p(x_{1:T}, y_{1:T}) - \log q(x_{1:T}; \varphi)]$. The ELBO also lower bounds $\log p(y_{1:T})$, which is the basis of variational expectation maximization (VEM) (Beal and Ghahramani, 2003) for learning model parameters θ. This paper will use a generalization of the VI framework based on estimator-coupling pairs (Domke and Sheldon, 2019) , which define a family of algorithms that have the above properties automatically. This is easier to describe with a more general notation: let $\gamma(x)$ be an unnormalized distribution with normalizer $Z = \int \gamma(x)dx$, so the goal is to approximate $\pi(x) = \gamma(x)/Z$. (For Bayesian inference, $\gamma(x) = p(x, y)$, $\pi(x) = p(x|y)$ and $Z = p(y)$, where the evidence y is fixed.)

Definition 1. *An estimator $R(\omega) > 0$ and a distribution $a(x|\omega)$ are a valid estimator-coupling pair for $\gamma(x)$ under distribution $Q(\omega)$ if, for all x,*

$$\mathrm{E}_{Q(\omega)}[R(\omega)a(x|\omega)] = \gamma(x).$$

This implies several things. First, $\mathrm{E}_{Q(\omega)}[R(\omega)] = Z$, meaning R is an unbiased estimator of the normalizer Z. Second, more generally, $\mathrm{E}_{Q(\omega)a(x|\omega)}[R(\omega)f(x)] = \int f(x)\gamma(x)dx$ for any (integrable) test function f. Third, $\pi(\omega, x) = \frac{1}{Z}Q(\omega)R(\omega)a(x|\omega)$ is a "coupling": a distribution that has both $\frac{1}{Z}Q(\omega)R(\omega)$ and $\pi(x)$ marginals. From the fact that $\mathrm{E}[R] = Z$, it is easy to see that $\mathrm{E}[\log R] \le \log Z$. The looseness of this bound can be decomposed as

$$\log Z = \mathrm{E}_{Q(\omega)}[\log R(\omega)] + \mathrm{KL}(Q(x)\|\pi(x)) + \mathrm{KL}(Q(\omega|x)\|\pi(\omega|x)), \quad (2)$$

where $Q(x)$ is the marginal distribution of $Q(\omega)a(x|\omega)$. This implies that maximizing $\mathrm{E}[\log R]$ tends to reduce $\mathrm{KL}(Q(x)\|\pi(x))$. Taking "vanilla" VI for example, given fixed y, a sample $x \sim q$ plays the role of ω, so $Q(x) = q(x)$, $R(x) = p(x, y)/q(x)$ and $a(x|x) = \delta_x(x)$ specify an estimator-coupling pair for $p(x, y)$. Then Equation 2 reduces to the usual decomposition of VI that $\log p(x) = L_{\text{ELBO}} + \mathrm{KL}(q(x)\|p(x|y))$ (Blei et al., 2016). For a more general VI, Q is the sampling algorithm, $\mathrm{E}[\log R]$ gives the lower bound, and $Q \cdot a$ gives the augmented approximate posterior. A key observation is that operations on estimator-coupling pairs can be used to derive new variational objectives. An example is the Rao-Blackwellization operation (Domke and Sheldon, 2019):

Lemma 1. *Suppose that $R_0(\omega, \nu)$ and $a_0(x|\omega, \nu)$ are a valid estimator-coupling pair for $\gamma(x)$ under $Q_0(\omega, \nu)$. Then*

$$R(\omega) = \mathrm{E}_{Q_0(\nu|\omega)} R_0(\omega, \nu),$$
$$a(x|\omega) = \frac{1}{R(\omega)}\mathrm{E}_{Q_0(\nu|\omega)}[R_0(\omega, \nu)a_0(x|\omega, \nu)]$$

are a valid estimator-coupling pair for $\gamma(x)$ under $Q(\omega) = \int Q_0(\omega, \nu)dv$. Furthermore, R has lower variance and gives a tighter bound than R_0, i.e., $\mathrm{Var}\, R(\omega) \le \mathrm{Var}\, R_0(\omega, \nu)$ and $\mathrm{E}[\log R_0(\omega, \nu)] \le \mathrm{E}[\log R(\omega)] \le \log Z$. Denote this operation by $(Q, R, a) = \mathrm{Marginalize}(Q_0, R_0, a_0; \nu)$.

The Marginalize operation reduces the variance of an estimator by marginalizing some variables. It is an example of what is often called Rao-Blackwellization, where an estimator is replaced by its conditional expectation to reduce variance. This operation is the biggest difference between VMPF and VSMC.

Variational Sequential Monte Carlo. VSMC (Naesseth et al., 2019) is an algorithm that constructs approximations for the sequence of target distributions $p(x_{1:t}|y_{1:t})$ using weighted samples. In particular, $p(x_{1:t}|y_{1:t})$ is approximated by a weighted set of particles $(w_t^i, x_{1:t}^{t,i})_{i=1}^N$ as

$$\pi_t(x_{1:t}) = \sum_{i=1}^N \bar{w}_t^i \delta_{x_{1:t}^{t,i}}(x_{1:t}), \text{ where } \bar{w}_t^i = w_t^i / \sum_{j=1}^N w_t^j.$$

We use a notation that explicitly distinguishes particles at different times, so $x_s^{t,i}$ is the sth entry of the ith

particle in iteration t of SMC, and $x_{1:t}^{t,i}$ is the entire particle. Later, we will use $x_{1:t}^{t,1:N}$ to denote the entire collection of N particles. The SMC procedure for sampling particles and computing weights is shown in Algorithm 1. For each time step t and particle i, an ancestor index j is sampled (Line 6) and the corresponding particle is extended using the proposal distribution r_t (Line 7), then weighted (Line 8) and assigned to become the new ith particle (Line 9).

From SMC, we get that $p_{\text{SMC}}(y_{1:T}) = \prod_{t=1}^{T} \frac{1}{N} \sum_{i=1}^{N} w_t^i$ is an unbiased estimator for $p(y_{1:T})$. Variational SMC uses $L_{\text{VSMC}} = \mathrm{E}[\log p_{\text{SMC}}(y_{1:T})]$ as a variational objective (Naesseth et al., 2018; Maddison et al., 2017a; Le et al., 2018). Naesseth et al. (2018) show that

$$L_{\text{VSMC}} \le L_{\text{ELBO}}^{\text{SMC}} \le \log p(y_{1:T}), \tag{3}$$

where $L_{\text{ELBO}}^{\text{SMC}}$ is the ELBO between the proposal $Q(x_{1:T})$ implied by SMC and the target $p(x_{1:T}|y_{1:T})$. VEM is used to simultaneously learn parameters of θ and adjust the proposal distributions r_t by maximizing L_{VSMC}. In practice, biased gradient estimates are most commonly used: the categorical ancestor variables (Line 6) cannot be reparameterized, and other estimators give problematically high variance (Naesseth et al., 2018; Maddison et al., 2017a; Le et al., 2018). A recent paper uses ensemble particle transformations in place of resampling to obtain a fully differentiable filter (DPF) (Corenflos et al., 2021). While this introduces some bias into the likelihood estimator and so does not give a provable lower bound on the log-likelihood, unbiased gradients can be computed using reparameterization methods. We will compare to DPF in Section 6.

3 Couplings and Sequential Monte Carlo

In this section we show how SMC can be derived with operations on estimator-coupling pairs. This gives a straightforward and novel proof of unbiasedness, which is the key property that guarantees a lower bound of $\log p(y_{1:T})$ when used with VI. It will also be the basis of our MPF analysis.

We first give Lemma 2, which replicates an estimator-coupling pair N times.

Lemma 2. *Suppose that $R_0(\omega, v)$ and $a_0(x|\omega, v)$ are a valid estimator-coupling pair for $\gamma(x)$ under $Q_0(\omega, v) = Q_0(\omega)Q_0(v|\omega)$. Then*

$$R(\omega, v_1, \ldots, v_N) = \frac{1}{N} \sum_{i=1}^{N} R_0(\omega, v_i),$$

$$a(x|\omega, v_1, \ldots, v_N) = \frac{\sum_{i=1}^{N} R_0(\omega, v_i) a_0(x|\omega, v_i)}{\sum_{i=1}^{N} R_0(\omega, v_i)}$$

are a valid estimator-coupling pair for $\gamma(x)$ under $Q(\omega, v_1, \ldots, v_N) = Q_0(\omega) \prod_{i=1}^{N} Q_0(v_i|\omega)$. Denote this operation by $(Q, R, a) = \text{Replicate}(Q_0, R_0, a_0; v, N)$.

Lemma 2 is used at each step of SMC to get N independent particles. It is a slight generalization of the "IID Mean" method in (Domke and Sheldon, 2019).

All operations so far only work for a fixed target distribution. In SMC, we also need some operation to change the target distribution. For instance, at time t, the target distribution should be extended from $p(x_{1:t-1}, y_{1:t-1})$ to $p(x_{1:t}, y_{1:t})$. Lemma 3 describes how to extend the target distribution.

Lemma 3. *Suppose that $R_0(\omega)$ and $a_0(x|\omega)$ are a valid estimator-coupling pair for $\gamma(x)$ under $Q_0(\omega)$, and $\gamma^0(x, x^0)$ is an unnormalized distribution on the augmented space of (x, x^0). Also suppose that we have a proposal distribution $r(x^0|x)$ such that if $r(x^0|x) = 0$ then $\gamma^0(x, x^0)/\gamma(x) = 0$. Then*

$$R(\omega, \hat{x}, \hat{x}^0) = R_0(\omega) \frac{\gamma^0(\hat{x}, \hat{x}^0)/\gamma(\hat{x})}{r(\hat{x}^0|\hat{x})},$$

$$a(x, x^0|\omega, \hat{x}, \hat{x}^0) = \delta_{(\hat{x}, \hat{x}^0)}(x, x^0)$$

are a valid estimator-coupling pair for $\gamma^0(x, x^0)$ under $Q(\omega, \hat{x}, \hat{x}^0) = Q_0(\omega)a_0(\hat{x}|\omega)r(\hat{x}^0|\hat{x})$. Denote this operation by $(Q, R, a) = \text{ExtendTarget}(Q_0, R_0, a_0; \gamma, \gamma^0, r)$.

If instead $a(x^0|\omega, \hat{x}, \hat{x}^0) = \delta_{\hat{x}^0}(x^0)$, then (R, a) is still a valid estimator-coupling pair for $\gamma^0(x^0) = \int \gamma^0(x, x^0)dx$ under Q. Denote this by $(Q, R, a) = \text{ChangeTarget}(Q_0, R_0, a_0; \gamma, \gamma^0, r)$.

We now use these results to derive SMC with estimator-coupling pairs.

Theorem 1. *For the SSM in Equation (1), given fixed $y_{1:T}$*

$$R\left(x_{1:1}^{1,1:N}, \ldots, x_{1:T}^{T,1:N}\right) = \prod_{t=1}^{T} \frac{1}{N} \sum_{i=1}^{N} w_t^i, \tag{4}$$

$$a\left(x_{1:T} \mid x_{1:1}^{1,1:N}, \ldots, x_{1:T}^{T,1:N}\right) = \sum_{i=1}^{N} w_T^i \delta_{x_{1:T}^{T,i}}(x_{1:T}) \tag{5}$$

form an estimator-coupling pair for $p(x_{1:T}, y_{1:T})$ under the sampling distribution of Algorithm 1 with weights w_t^i as given in Lines 2 and 8 and $w_t^i = w_t^i / \sum_{j=1}^{N} w_t^j$. Thus, for any test function h,

$$\mathrm{E}\left[\prod_{t=1}^{T} \frac{1}{N} \sum_{i=1}^{N} w_t^i \cdot \sum_{i=1}^{N} w_T^i h\left(x_{1:T}^{T,i}\right)\right] = p(y_{1:T}) \cdot \mathrm{E}_{p(x_{1:T}|y_{1:T})}\, h(x_{1:T}). \tag{6}$$

While the unbiasedness conclusion of Equation (6) is well-known (Naesseth et al., 2019; Maddison et al.,

Algorithm 1 Sequential Monte Carlo

Require: $p(x_{1:T}, y_{1:T})$, $y_{1:T}$, $\{r_t(x_t|x_{t-1})\}$, N

1: Sample $x_1^{1,i} \sim r_1(x_1)$ for all i
2: Set $w_1^i = \frac{f_1(x_1^{1,i}) g(y_1|x_1^{1,i})}{r_1(x_1^{1,i})}$ for all i
3: **for** $t = 2, \ldots, T$ **do**
4: **for** $i = 1, \ldots, N$ **do**
5: Set $\overline{w}_{t-1}^j = w_{t-1}^j / \sum_{k=1}^N w_{t-1}^k$ for all j
6: Sample $j \sim$ Categorical $\overline{w}_{t-1}^{1:N}$
7: Sample $x_t^{t,i} \sim r_t(x_t|x_{t-1}^{t-1,j})$
8: Set $w_t^i = \frac{f(x_t^{t,i}|x_{t-1}^{t-1,j}) g(y_t|x_t^{t,i})}{r_t(x_t^{t,i}|x_{t-1}^{t-1,j})}$
9: Set $x_{1:t}^{t,i} = (x_{1:t-1}^{t-1,j}, x_t^{t,i})$

Algorithm 2 Marginal Particle Filter

Require: $p(x_{1:T}, y_{1:T})$, $y_{1:T}$, $\{r_t(x_t|x_{t-1})\}$, N

1: Sample $x_1^i \sim r_1(x_1)$ for all i
2: Set $v_1^i = \frac{f_1(x_1^i) g(y_1|x_1^i)}{r_1(x_1^i)}$ for all i
3: **for** $t = 2, \ldots, T$ **do**
4: **for** $i = 1, \ldots, N$ **do**
5: Set $\overline{v}_{t-1}^j = v_{t-1}^j / \sum_{k=1}^N v_{t-1}^k$ for all j
6: Sample $j \sim$ Categorical $\overline{v}_{t-1}^{1:N}$
7: Sample $x_t^i \sim r_t(x_t|x_{t-1}^j)$
8: Set $v_t^i = \frac{\sum_{j=1}^N v_{t-1}^j f(x_t^i|x_{t-1}^j) g(y_t|x_t^i)}{\sum_{j=1}^N v_{t-1}^j r_t(x_t^i|x_{t-1}^j)}$

2017a, e.g.,), we give a novel proof that breaks SMC into small operations. This proof strategy will form the basis for understanding MPF as applying Rao-Blackwellization operations within SMC.

(Proof sketch). To start, observe that the estimator $R_1(x_1) = \frac{f(x_1)g(y_1|x_1)}{r_1(x_1)} = \frac{p(x_1,y_1)}{r_1(x_1)}$ and the coupling $a_1(x_1|x_1) = \delta_{x_1}(x_1)$ are valid for $p(x_1, y_1)$ under $Q_1(x_1) = r_1(x_1)$. Now, for $t > 1$, define

$$(Q_t, R_t, a_t) = \text{ExtendTarget}\left(Q_{t-1}^N, R_{t-1}^N, a_{t-1}^N; p(x_{1:t-1}, y_{1:t-1}), p(x_{1:t}, y_{1:t}), r_t(x_t|x_{t-1})\right), \quad (7)$$

where for all t,

$$(Q_t^N, R_t^N, a_t^N) = \text{Replicate}\left(Q_t, R_t, a_t; x_{1:t}^t, N\right).$$

Mechanically applying these transformations, Lemmas 3 and 2 yield that the functions R_T^N and a_T^N match Equations (4) and (5) and

$$Q_T^N\left(x_{1:1}^{1,1:N}, \ldots, x_{1:T}^{T,1:N}\right) = \prod_{i=1}^N r_1(x_1^{1,i}) \prod_{t=2}^T \sum_{j=1}^N \overline{w}_{t-1}^j \delta_{x_{1:t-1}^{t-1,j}}(x_{1:t-1}^{t,i}) r_t(x_t^{t,i}|x_{t-1}^{t,i}), \quad (8)$$

which matches the SMC sampling distribution. The claimed result then follows immediately from the fact that R_T^N and a_T^N are a valid estimator-coupling pair for $p(x_{1:T}, y_{1:T})$ under Q_T^N. □

Details appear in the supplement. This proof uses induction on a sequence of estimators with operations that match the steps of the algorithm, and may be easier to understand than proofs that reason about the full expectation and require manipulating complex expressions.In addition, using operations on estimator-coupling pairs, it is possible to *implement* estimators by transforming simple estimators in a way that exactly matches their derivation, which is of interest in probabilistic programming (van de Meent et al., 2018). Douc and Moulines (2008) give a related framework to show consistency and asymptotic normality for SMC using operations on weighted particle systems that preserve those properties. Stites et al. (2021) also derive SMC by operations on proper weighting (Liu and Liu, 2001; Naesseth et al., 2015).

Since SMC can be derived by estimator-coupling pairs, we directly have that $L_{\text{VSMC}} \leq L_{\text{ELBO}}^{\text{SMC}} \leq \log p(y_{1:T})$ as in Equation (3). This result also quantifies the gap $L_{\text{ELBO}}^{\text{SMC}} - L_{\text{VSMC}}$ as a conditional KL divergence.

4 Variational Marginal Particle Filter

We now show that the marginal particle filter (MPF) of Klaas et al. (2005) can also be derived with estimator-coupling pairs, which proves it is unbiased and suitable for use within VI. It uses Marginalize operations not present in SMC, which reduce variance and make VI bounds tighter "locally". Unlike SMC, it uses mixture proposals that can be reparameterized.

In Theorem 1, we see that SMC is not fully reparameterizable because of the Dirac distributions in Q_T^N (Equation (8)), which correspond to the sampling and copying operations in Lines 6 and 9 in Algorithm 1. The non-reparameterizable variables $x_{t,1:t-1}^{t,i}$ are exactly the first $t-1$ entries of each particle. Our general idea is to marginalize these variables using Lemma 1 to get MPF.

MPF and Couplings. The MPF algorithm is shown in Algorithm 2. Instead of $p(x_{1:t}, y_{1:t})$, it targets the sequence of marginal distributions $p(x_t, y_{1:t})$ for all t. The procedure is very similar to SMC, *but with different weights* for $t > 1$. In MPF, the ith marginal particle at time t is denoted as x_t^i. Using this notation for both

algorithms to facilitate comparison, the weights are:

$$\text{SMC: } w_t^i = \frac{f(x_t^i|x_{t-1}^j)g(y_t|x_t^i)}{r_t(x_t^i|x_{t-1}^j)},\ j \sim \text{Categorical}(\cdot),$$

$$\text{MPF: } v_t^i = \frac{\sum_{j=1}^N v_{t-1}^j f(x_t^i|x_{t-1}^j)g(y_t|x_t^i)}{\sum_{j=1}^N v_{t-1}^j r_t(x_t^i|x_{t-1}^j)}.$$

MPF can be obtained from SMC by two steps: (1) drop the first $t-1$ variables $x_{1:t-1}$ from the target distribution and all particles to target $p(x_t, y_{1:t})$ instead of $p(x_{1:t}, y_{1:t})$, (2) Rao-Blackwellize the ancestor index j from the sampling distribution using the Marginalize operation in each step of SMC. Formally, we have:

Theorem 2. *For the SSM in Equation (1), given fixed* $y_{1:T}$

$$R\left(x_1^{1:N}, \ldots, x_t^{1:N}\right) = \prod_{t=1}^T \frac{1}{N} \sum_{i=1}^N v_t^i, \tag{9}$$

$$a(x_T \mid x_1^{1:N}, \ldots, x_t^{1:N}) = \sum_{i=1}^N v_T^i \delta_{x_T^i}(x_T) \tag{10}$$

form an estimator-coupling pair for $p(x_T, y_{1:T})$ *under the sampling distribution of Algorithm 2 with weights* v_t^i *as specified in Lines 2 and 8 and* $v_t^i = v_t^i / \sum_{j=1}^N v_t^j$. *Thus, for any test function h,*

$$\mathrm{E}\left[\prod_{t=1}^T \frac{1}{N} \sum_{i=1}^N v_t^i \cdot \sum_{i=1}^N v_T^i h(x_T^i)\right] = p(y_{1:T}) \cdot \mathrm{E}_{p(x_T|y_{1:T})}[h(x_T)]. \tag{11}$$

We are not aware of an existing proof of unbiasedness for MPF, though unbiasedness of the normalizing constant estimate (i.e., $h \equiv 1$ in Equation (11)) can be derived from tensor Monte Carlo (TMC) (Aitchison, 2019) (See Supplement 3) or a recent result on auxiliary particle filters (Branchini and Elvira, 2021). Our proof again shows that MPF is obtained by operations on estimator-coupling pairs.

(Proof sketch). The proof is very similar to the proof of Theorem 1, except the ExtendTarget operation in Equation (7) for $t > 1$ is replaced by the following two operations:

$$(Q_t^0, R_t^0, a_t^0) = \text{ChangeTarget}\left(Q_{t-1}^N, R_{t-1}^N, a_{t-1}^N;\right.$$
$$\left.p(x_{t-1}, y_{1:t-1}), p(x_{t-1}, x_t, y_{1:t}), r_t(x_t|x_{t-1})\right),$$

$$(Q_t, R_t, a_t) = \text{Marginalize}\left(Q_t^0, R_t^0, a_t^0; x_{t-1}\right).$$

Recall that ChangeTarget is the same as ExtendTarget (Lemma 3), but drops x_{t-1} from the target distribution. The Marginalize operation then *marginalizes the corresponding variable* from the "internal state" of the estimator. After mechanically applying the transformations of Lemma 3, Lemma 1, and Lemma 2, we get that the functions R_T^N and a_T^N match Equations (9) and (10) and

$$Q_T^N\left(x_1^{1:N}, \ldots, x_t^{1:N}\right) = \prod_{i=1}^N r_1(x_1^i) \prod_{t=2}^T \sum_{j=1}^N v_{t-1}^j r_t(x_t^i|x_{t-1}^j), \tag{12}$$

which matches the MPF sampling distribution. □

In the proof sketch, we can see that the estimator R_t following Marginalize $(\cdot)$ has lower (or the same) variance and gives a tighter (or the same) bound as R_t^0, by Lemma 1. The same reasoning implies that using MPF weights is never worse than using the SMC weights "locally": in iteration t, when targeting $p(x_t, y_{1:t})$, variance is never higher when using the MPF weight calculation in place of the SMC weight calculation, given the weights from iteration $t-1$. This does not necessarily imply the full MPF estimator has lower variance than SMC, but empirical evidence points to it having lower variance (Klaas et al., 2005).

The time complexity of MPF is $O(N^2T)$ compared to $O(NT)$ for SMC. In Line 8 of the MPF algorithm, the density $f(x_t^i|x_{t-1}^j)$ must be computed N^2 times in total vs. N times in total in Line 8 of SMC. But there are only N different conditional distributions: the distributions $f(\cdot|x_{t-1}^j)$ for each particle at the previous time-step. The density calculations can be split into two parts: (1) $O(N)$ pre-processing for each conditional distribution, and (2) evaluating the density N^2 times. For many models, the $O(N)$ pre-processing takes a significant fraction of the time. For example, in our DMM experiments, pre-processing requires neural networks computation and density evaluation only elementary tensor operations, and the times of VMPF and VSMC are indistinguishable for $N \le 16$. Section 2.4 of Aitchison (2019) gives a similar argument.

Variational MPF. Let $p_{\text{MPF}}(y_{1:T}) = \prod_{t=1}^T \frac{1}{N} \sum_{i=1}^N v_t^i$ be the unbiased estimator of $p(y_{1:T})$ from Equation (9). We propose the variational objective

$$L_{\text{VMPF}}(\varphi, \theta) = \mathrm{E}\left[\log p_{\text{MPF}}(y_{1:T}; \varphi, \theta)\right], \tag{13}$$

where φ stands for the parameter of proposal distributions r_t and θ stands for the model parameters. By properties of estimator-coupling pairs, we immediately have that $L_{\text{VMPF}}(\varphi, \theta) \le L_{\text{ELBO}}^{\text{MPF}}(\varphi, \theta) \le \log p(y_{1:T})$, where $L_{\text{ELBO}}^{\text{MPF}}(\varphi, \theta)$ is the ELBO between the marginal proposal $Q(x_T; \varphi)$ implied by the MPF procedure and the target distribution $p(x_T|y_{1:T}; \theta)$, and we can also quantify the gap $L_{\text{ELBO}}^{\text{MPF}} - L_{\text{VMPF}}$ with Equation (2). To compute and optimize this objective, we use Monte

Carlo estimates for the value and gradients. We have two approaches to estimate the gradients.

Biased gradients with categorical sampling. The first approach follows VSMC (Naesseth et al., 2018). We assume that each proposal distribution r_t is reparameterizable and compute gradients of Equation (13) as $\nabla L_{\text{VMPF}}(\varphi, \theta) = \mathrm{E}[\nabla \log p_{\text{MPF}}(y_{1:T}; \varphi, \theta)] + g_{\text{score}}$, where the first term uses the reparameterization trick (Kingma and Welling, 2014; Rezende et al., 2014), but ignores gradient paths for probabilities of categorical variables, which cannot be reparameterized; and g_{score} is a score function term to handle the categorical sampling (Naesseth et al., 2018). As with VSMC, we observe that estimates of g_{score} have very high variance and lead to slow convergence. For this approach, we simply drop g_{score} and estimate the first term, which is a biased gradient estimator. We call this method VMPF with biased gradient (VMPF-BG), which does not have any limitation on the proposals.

Unbiased gradients with implicit reparameterization. A biased gradient estimator can lead to suboptimal inference (Corenflos et al., 2021). A very appealing property of Algorithm 2 is that, as a result of Rao-Blackwellization, the variables j and x_{t-1}^j are not used in weight computations, so Lines 6 and 7 can be combined conceptually into a single draw $x_t^i \sim \sum_{j=1}^N w_{t-1}^j r_t(x_t | x_{t-1}^j)$ from a mixture distribution. This is evident from Equation (12), which includes only the (continuous) mixture density $\sum_{j=1}^N w_{t-1}^j r_t(x_t^i | x_{t-1}^j)$ for x_t^i. It is therefore possible to reparameterize x_t^i by sampling from the mixture and then using implicit reparameterization gradients (Graves, 2016; Figurnov et al., 2018). We then have the fully reparameterized gradient $\nabla L_{\text{VMPF}}(\varphi, \theta) = \mathrm{E}[\nabla \log p_{\text{MPF}}(y_{1:T}; \varphi, \theta)]$, and can form an unbiased estimate by drawing samples and backpropagating through mixtures. This is currently possible for any proposal that is a product distribution, or full-rank Gaussians. We call this method VMPF with unbiased gradients (VMPF-UG). It is a fully reparameterized gradient estimate for a variational filtering objective that lower bounds the log-likelihood for suitable proposal distributions.

5 Related Work

There is significant previous work on improving VI approximations. One direction enriches the variational family directly, for example, with normalizing flows (Papamakarios et al., 2019; Rezende and Mohamed, 2015), copulas (Tran et al., 2015; Han et al., 2016; Hirt et al., 2019), or mixture distributions (Miller et al., 2017). Another direction increases expressiveness by introducing auxiliary variables: this work includes hierarchical variational models (Ranganath et al., 2016), VI with Markov chain Monte Carlo (Salimans et al., 2015; Caterini et al., 2018), variational Gaussian processes (Tran et al., 2016), and importance-weighted VI (IWVI) (Burda et al., 2016; Domke and Sheldon, 2018). Estimator-coupling pairs generalize IWVI and include other variance reduction techniques, such as stratified sampling (Domke and Sheldon, 2019). For SSMs, three papers independently proposed to use SMC as an unbiased estimator to generalize IWVI in another direction (Naesseth et al., 2018; Maddison et al., 2017a; Le et al., 2018). This work builds on the above two ideas.

One limitation of prior SMC variational objectives is that they are not fully differentiable due to resampling steps. Moretti et al. (2019) use the concrete distribution (Maddison et al., 2017b; Jang et al., 2017) to approximate the resampling step, but they focus on the signal-to-noise ratio problem (Rainforth et al., 2018) and do not mention any performance improvement due to differentiability. A series of works employ differentiable neural networks to approximate the resampling function (Karkus et al., 2018; Jonschkowski et al., 2018; Zhu et al., 2020; Ma et al., 2020a,b). However, none produces fully differentiable SMC (Corenflos et al., 2021). Corenflos et al. (2021) use optimal transport to learn an ensemble transform to replace resampling, leading to the first fully differentiable particle filter in the literature. Their likelihood estimator is asymptotically consistent, but biased, so does not give a provable lower bound of the log-likelihood.

Another interesting line of previous work is independent particle filters (IPF) (Lin et al., 2005), which improve SMC with multiple permutations of ancestor variables. With the setting of complete matching, IPF becomes tensor Monte Carlo (TMC) (Aitchison, 2019) for SSMs. We outline an alternate viewpoint of MPF as TMC with specific mixture proposals in Section 3 of the supplement.

Recently, Campbell et al. (2021) propose an online VI which outperforms several online filtering methods on SSM by using a Bellman-type recursion similar to those used in reinforcement learning (Sutton and Barto, 2005).

6 Experiments

We conduct experiments on linear Gaussian SSMs, stochastic volatility models, and deep Markov models (DMMs) (Krishnan et al., 2017) and compare lower bounds obtained by IWVI [*], VSMC with biased gradients (Naesseth et al., 2018), tensor Monte Carlo

[*] This uses $q(x_{1:T}) = r_1(x_1) \prod_{t=2}^T r_t(x_t | x_{t-1})$ as proposal and is equivalent to VSMC without resampling.

(TMC) (Aitchison, 2019) (factorized), differentiable particle filter (DPF) (Corenflos et al., 2021) (evaluated by SMC), VMPF-BG, and VMPF-UG. We implement all algorithms in TensorFlow with TensorFlow Probability (Abadi et al., 2015; Dillon et al., 2017) and train with the Adam optimizer (Kingma and Ba, 2015). The code to replicate the experiments can be found at https://github.com/lll6924/VMPF.

Linear Gaussian State Space Models We first test with linear Gaussian models, for which the exact log-likelihood can be computed by the Kalman filter (KF) (Harvey, 1990). The model is $x_t = Ax_{t-1} + v_t$, $y_t = Cx_t + e_t$, where $v_t \sim N(0, Q)$, $e_t \sim N(0, R)$, and $x_1 \sim N(0, I)$. We follow Naesseth et al. (2018) and set $T = 10$, $(A)_{ij} = \alpha^{|i-j|+1}$ for $\alpha = 0.42$, $Q = I$ and $R = I$. There are two settings for C: "sparse" C has diagonal entries 1 and other entries 0; "dense" C has $C_{ij} \sim N(0, I)$ for all i, j. We vary $d_x = \dim(x_t)$, $d_y = \dim(y_t)$, and whether C is sparse or dense. The same model p is used to generate data and during inference. We choose the proposal distributions $r_t(x_t|x_{t-1}; \varphi) = N(x_t|\mu_t + \mathrm{diag}(\beta_t)Ax_{t-1}, \mathrm{diag}(\sigma_t^2))$ with $\varphi = (\mu_t, \beta_t, \sigma_t^2)_{t=1}^T$ and maximize each objective with respect to φ. In all settings, we first train for 10K iterations with learning rate 0.01, then another 10K iterations with learning rate 0.001. Further lowering the learning rate has little effect.

We first examine convergence of each algorithm with $N = 4$. Figure 1 shows the lower bound of each method during training. For sparse C, the VMPF bounds are substantially higher than VSMC, but IWVI is highest. The bound of DPF is between VSMC and VMPF in this case. In contrast, for dense C, IWVI is much worse, while the final bounds of VSMC, DPF and VMPF are similar. Overall, VMPF-BG is never worse than VSMC, TMC or DPF in terms of final bounds or convergence speed, and gives significantly higher bounds for sparse C. The convergence of VMPF-UG is slow,† and the final bound is comparable to, but not higher than, VMPF-BG.

It is likely IWVI performs well for sparse C because the variational family $q(x_{1:T}; \varphi)$ includes the true posterior. Because SMC "greedily" resamples particles with high probability under $p(x_t|y_{1:t})$ (using only the first t observations), it is counterproductive relative to a very accurate model of $p(x_t|y_{1:T})$ (conditioned on *all* observations); see the discussion of sharpness in (Maddison et al., 2017a). The variational family does not include the posterior for dense C, and IWVI is much worse than the SMC-based methods. To further understand this, we reran the sparse C experiment for $d_x = 25$, $d_y = 25$ after fixing $\beta_t = 1$ to impoverish the variational family. The final bounds become -456.20, -453.32, and -451.85 for IWVI, VSMC, and VMPF, respectively, confirming that resampling can be harmful when $q(x_{1:T}; \varphi)$ can already approximate the true posterior very well, but tends to be beneficial otherwise.

The slow convergence of VMPF-UG can be explained by gradient variance (Figure 2, left). Both VSMC and VMPF with biased gradients have low variance and converge quickly. In contrast, VMPF-UG has high gradient variance and slower convergence, especially in early iterations, but variance reduces substantially when close to convergence. This suggests a strategy of using the biased gradient estimator at the beginning of optimization and then switching to the unbiased estimator.

Although we did not observe it for $N = 4$, unbiased gradients can lead to tighter bounds upon convergence, especially for larger numbers of particles (Figure 2, right). For small N, VMPF-BG and VMPF-UG are similar, but as N increases, the gap between the methods increases, which indicates that biased gradients are more of a problem. We conjecture that the bias of gradients for VMPF-BG will not go to zero as N is increased, but the magnitude of the true gradient shrinks as $N \to \infty$, which leads to difficulty in training (Le et al., 2018; Rainforth et al., 2018).

Stochastic Volatility The stochastic volatility model (Chib et al., 2009) is widely used for financial data. It is $x_t = \mu + \Phi(x_{t-1} - \mu) + v_t$, $y_t = \mathrm{diag}(\exp(x_t/2))Be_t$, where $v_t \sim N(0, Q)$, $e_t \sim N(0, I)$, and $x_1 \sim N(\mu, Q)$. The model parameters are $\theta = (\mu, \Phi, Q, B)$, where μ is a vector, Φ and Q are diagonal matrices, and B is either a diagonal or lower triangular matrix (with positive diagonal entries in both cases). We use VEM to learn θ. Following Naesseth et al. (2018), we use the proposal $r_t(x_t|x_{t-1}; \varphi, \theta) \propto f(x_t|x_{t-1}; \theta)N(x_t; \mu_t, \Sigma_t)$ with parameters $\varphi = (\mu_t, \Sigma_t)_{t=1}^T$ and Σ_t diagonal.

We model the exchange rates of 22 international currencies with respect to US dollars for 10 years (monthly from 4/2011 to 3/2021). The data can be downloaded from the US Federal Reserve System‡. Table 1 reports the optimized lower bound for different algorithms for $N \in \{4, 8, 16\}$. VMPF-BG always gives a higher bound than VSMC, and the IWVI bound is always highest. TMC works well for $N = 16$, but is much worse for smaller N. We also notice that DPF works slightly better than VSMC or VMPF on a simpler model with

†For $d_x = 25$, $d_y = 1$, dense C (second panel), 20K iterations were not enough to train VMPF-UG, so we initialiazed it with the parameters from VMPF-BG.

‡https://www.federalreserve.gov/releases/h10/current/

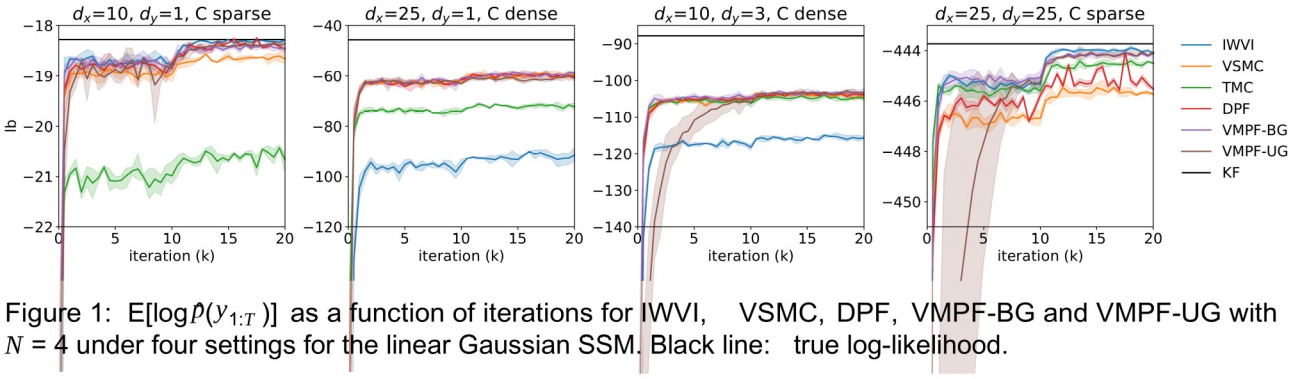

Figure 1: $\mathrm{E}[\log \hat{p}(y_{1:T})]$ as a function of iterations for IWVI, VSMC, DPF, VMPF-BG and VMPF-UG with $N = 4$ under four settings for the linear Gaussian SSM. Black line: true log-likelihood.

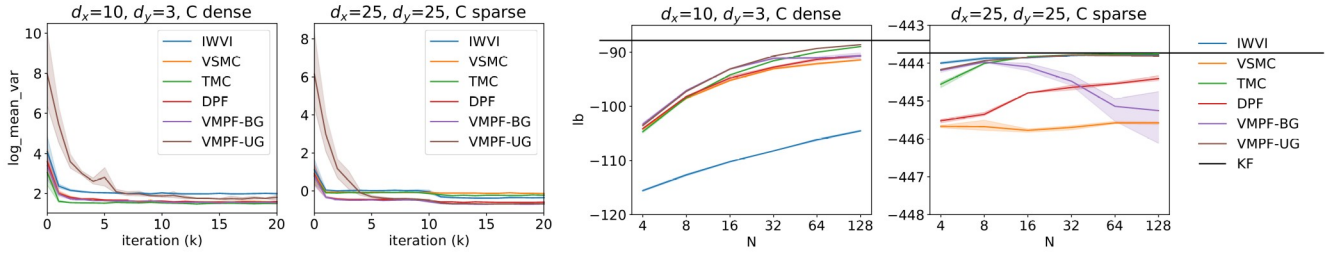

Figure 2: Results for linear Gaussian SSMs.Left: logarithm of mean gradient variance vs. training iteration. Right: final lower bound vs. N. Experimental settings indicated in plots.

diagonal B, but worse than VMPF with triangular B. VMPF-UG is beneficial for diagonal B with $N \in \{4, 8\}$, but becomes increasingly harder to train with large N and a larger model. The IWVI performance is surprising and contrary to similar experiments in (Naesseth et al., 2018), but appears to be another case where the family of proposal distributions can already approximate the posterior very well: we also find the ELBO with "vanilla" VI ($N = 1$) is higher than SMC-based methods with $N > 1$ in both cases.

Deep Markov Models We evaluate the bounds of all methods for deep Markov models (DMMs) on four polyphonic music datasets: Nottingham, JSB, MuseData and Piano-midi.de (Boulanger-Lewandowski et al., 2012). These are typically modeled with VRNNs, but the marginalization of ancestor variables prevents using VMPF for VRNNs (see Section 7 and the supplement). The DMM model is

$$x_t = \mu_\theta(x_{t-1}) + \mathrm{diag}(\exp(\sigma_\theta(x_{t-1})/2))v_t,$$
$$y_t \sim \mathrm{Bernoulli}(\mathrm{sigmoid}(\eta_\theta(x_t))),$$

where $v_t \sim N(0, I)$, $x_0 = 0$, and μ_θ, σ_θ, η_θ are neural networks. To approximate the posterior, we define the proposal distribution

$$r(x_t|x_{t-1}, y_t; \varphi) \propto N(x_t; \mu^x_\varphi(x_{t-1}), \mathrm{diag}(\exp(\sigma^x_\varphi(x_{t-1}))) \cdot N(x_t; \mu^y_\varphi(y_t), \mathrm{diag}(\exp(\sigma^y_\varphi(y_t))))$$

where μ^x_φ, σ^x_φ, μ^y_φ and σ^y_φ are neural networks. Details can be found in supplement. Table 2 shows the results of different methods. We see that in all cases VMPF-BG produces the best results. It is possible to train VMPF-UG in most settings here by initializing with VMPF-BG and/or using gradient clipping, but they lead to a slightly worse results than VMPF-BG.

7 Limitations

We are aware of several limitations or potential limitations. First, for VMPF-UG, implicit reparameterization gradients for mixture distributions require the ability to compute conditional CDFs of each component distribution (Graves, 2016; Figurnov et al., 2018), which is straightforward for Gaussians or product distribution, but may be difficult in general. Current implementations support only product distributions (Dillon et al., 2017). At present, this limits the choice of proposals for VMPF-UG. There is no such limitation for VMPF-BG.

Second, although implicit reparameterization gives unbiased gradients for VMPF, we show that the variance remains high compared to VSMC and VMPF-BG, which restricts applying VMPF-UG in some cases, especially to complex models. Future work can focus on reducing the variance.

Third, unlike VSMC, VMPF only works for the the marginal objective $p(x_t|y_{1:t})$, which restricts some applications, for example, to VRNNs (Chung et al., 2015). See the supplement for discussion.

Table 1: Stochastic volatility model lower bounds for IWVI, VSMC, DPF, VMPF-BG and VMPF-UG (higher is better). Mean and standard deviation of 3 runs are reported.

	Method	$N = 1$	$N = 4$	$N = 8$	$N = 16$
Diagonal B	IWVI	7216.09 (0.42)	**7219.77** (0.17)	**7220.60** (0.51)	**7221.51** (0.10)
	VSMC		7200.57 (0.11)	7198.13 (0.30)	7197.70 (0.09)
	TMC		7193.21 (0.08)	7202.16 (0.07)	7208.67 (0.11)
	DPF		7209.42 (0.99)	7209.05 (1.13)	7210.22 (0.24)
	VMPF-BG		7205.29 (0.17)	7205.04 (0.16)	7205.90 (0.35)
	VMPF-UG		7208.57 (0.27)	7208.95 (0.28)	7206.35 (0.27)
Triangular B	IWVI	8585.16 (1.00)	**8590.87** (0.27)	**8593.19** (0.64)	**8595.78** (0.82)
	VSMC		8573.04 (0.23)	8572.58 (0.19)	8572.01 (0.08)
	TMC		8554.89 (0.18)	8570.56 (0.01)	8582.32 (0.10)
	DPF		8572.74 (2.97)	8574.32 (1.65)	8574.32 (0.32)
	VMPF-BG		8576.57 (0.61)	8578.53 (0.20)	8581.21 (0.39)
	VMPF-UG		8556.64 (3.63)	8543.40 (1.48)	8538.15 (5.36)

Table 2: Test set nats per timestep for DMM trained with IWVI, VSMC, DPF, VMPF-BG (higher is better) on four polyphonic music datasets. Mean and standard deviation of 3 runs are reported.

N	Method	Nottingham	JSB	MuseData	Piano-midi.de
4	IWVI	-3.86 (0.04)	-7.40 (0.01)	-8.19 (0.04)	-8.78 (0.01)
	VSMC	-3.38 (0.03)	-7.16 (0.01)	-7.68 (0.02)	-8.39 (0.03)
	TMC	-3.65 (0.02)	-7.51 (0.01)	-8.41 (0.02)	-9.00 (0.01)
	DPF	-3.33 (0.02)	-7.14 (0.01)	-7.75 (0.01)	-8.45 (0.01)
	VMPF-BG	**-3.29** (0.03)	**-7.04** (0.01)	**-7.67** (0.00)	**-8.35** (0.01)
8	IWVI	-3.82 (0.03)	-7.38 (0.01)	-8.16 (0.04)	-8.75 (0.01)
	VSMC	-3.20 (0.02)	-6.96 (0.00)	-7.43 (0.01)	-8.20 (0.01)
	TMC	-3.40 (0.02)	-7.24 (0.01)	-8.03 (0.01)	-8.64 (0.01)
	DPF	-3.19 (0.00)	-6.95 (0.01)	-7.40 (0.01)	-8.31 (0.00)
	VMPF-BG	**-3.09** (0.03)	**-6.80** (0.01)	**-7.39** (0.01)	**-8.12** (0.01)
16	IWVI	-3.84 (0.03)	-7.33 (0.02)	-8.16 (0.02)	-8.74 (0.02)
	VSMC	-3.06 (0.02)	-6.81 (0.00)	-7.22 (0.02)	-8.03 (0.00)
	TMC	-3.23 (0.02)	-6.98 (0.02)	-7.72 (0.00)	-8.37 (0.01)
	DPF	-3.08 (0.02)	-6.78 (0.00)	-7.22 (0.01)	-8.18 (0.02)
	VMPF-BG	**-2.96** (0.02)	**-6.64** (0.01)	**-7.14** (0.01)	**-7.92** (0.01)

Acknowledgements

We thank Javier Burroni, Tomas Geffner and the anonymous reviewers for comments that greatly improved the manuscript. This material is based upon work supported by the National Science Foundation under Grant Nos. 1749854, 1908577 and 2045900.

References

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL https://www.tensorflow.org/ . Software available from tensorflow.org.

Laurence Aitchison. Tensor Monte Carlo: Particle methods for the GPU era. In *Advances in Neural Information Processing Systems 32:Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 7146–7155, 2019.

Matthew J. Beal and Zoubin Ghahramani. The variational bayesian EM algorithm for incomplete data: with application to scoring graphical model structures. *Bayesian statistics*, 7(453-464):210, 2003.

David M. Blei, Alp Kucukelbir, and Jon D. McAuliffe. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112: 859 – 877, 2016.

Nicolas Boulanger-Lewandowski, Yoshua Bengio, and Pascal Vincent. Modeling temporal dependencies in high-dimensional sequences:Application to polyphonic music generation and transcription. In *Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK, June 26 - July 1, 2012*. icml.cc / Omnipress, 2012.

Nicola Branchini and Víctor Elvira. Optimized auxiliary particle filters: adapting mixture proposals via convex optimization. In *Proceedings of the 37th Conference in Uncertainty in Artificial Intelligence, UAI 2021, Online, July 27-30, 2021*, 2021.

Yuri Burda, Roger B. Grosse, and Ruslan Salakhutdinov. Importance weighted autoencoders. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.

Andrew Campbell, Yuyang Shi, Thomas Rainforth, and Arnaud Doucet. Online variational filtering and parameter learning. *Advances in Neural Information Processing Systems*, 34, 2021.

Anthony L. Caterini, Arnaud Doucet, and Dino Sejdinovic. Hamiltonian variational auto-encoder. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 8178–8188, 2018.

Siddhartha Chib, Yasuhiro Omori, and Manabu Asai. Multivariate stochastic volatility. In *Handbook of financial time series*, pages 365–400. Springer, 2009.

Junyoung Chung, Kyle Kastner, Laurent Dinh, Kratarth Goel, Aaron C. Courville, and Yoshua Bengio. A recurrent latent variable model for sequential data. In *Advances in Neural Information Processing Systems 28:Annual Conference on NeuralInformation Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 2980–2988, 2015.

Adrien Corenflos, James Thornton, George Deligiannidis, and Arnaud Doucet. Differentiable particle filtering via entropy-regularized optimal transport. In *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 2100–2111. PMLR, 2021.

Joshua V. Dillon, Ian Langmore, Dustin Tran, Eugene Brevdo, Srinivas Vasudevan, Dave Moore, Brian Patton, Alex Alemi, Matthew D. Hoffman, and Rif A. Saurous. Tensorflow distributions. *arXiv preprint arXiv:1711.10604*, 2017.

Justin Domke and Daniel R. Sheldon. Importance weighting and variational inference. In *Advances in Neural Information Processing Systems 31:Annual Conference on NeuralInformation Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 4475–4484, 2018.

Justin Domke and Daniel R. Sheldon. Divide and couple: Using Monte Carlo variational objectives for posterior approximation. In *Advances in Neural Information Processing Systems 32:Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 338–347, 2019.

Randal Douc and Eric Moulines. Limit theorems for weighted samples with applications to sequential

Monte Carlo methods. *The Annals of Statistics*, 36(5):2344 – 2376, 2008.

Arnaud Doucet and Adam M. Johansen. A tutorial on particle filtering and smoothing: Fifteen years later. *Handbook of nonlinear filtering*, 12(656-704):3, 2009.

Mikhail Figurnov, Shakir Mohamed, and Andriy Mnih. Implicit reparameterization gradients. In *Advances in Neural Information Processing Systems 31:Annual Conference on NeuralInformation Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*, pages 439–450, 2018.

Alex Graves. Stochastic backpropagation through mixture density distributions. *arXiv preprint arXiv:1607.05690*, 2016.

Shaobo Han, Xuejun Liao, David B. Dunson, and Lawrence Carin. Variational Gaussian copula inference. In *Proceedings of the 19th International Conference on Artificial Intelligence and Statistics, AISTATS 2016, Cadiz, Spain, May 9-11, 2016*, volume 51, pages 829–838. JMLR.org, 2016.

Andrew C. Harvey. *Forecasting, Structural Time Series Models and the Kalman Filter*. Cambridge University Press, 1990.doi: 10.1017/CBO9781107049994.

Marcel Hirt, Petros Dellaportas, and Alain Durmus. Copula-like variational inference. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 2955–2967, 2019.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017.

Rico Jonschkowski, Divyam Rastogi, and Oliver Brock. Differentiable particle filters: End-to-end learning with algorithmic priors. In *Robotics: Science and Systems XIV, Carnegie Mellon University, Pittsburgh, Pennsylvania, USA, June 26-30, 2018*, 2018.

Michael I. Jordan, Zoubin Ghahramani, Tommi S. Jaakkola, and Lawrence K. Saul. An introduction to variational methods for graphical models. *Mach. Learn.*, 37(2):183–233, 1999.

Péter Karkus, David Hsu, and Wee Sun Lee. Particle filter networks with application to visual localization. In *2nd Annual Conference on Robot Learning, CoRL 2018, Zürich, Switzerland, 29-31 October 2018, Proceedings*, volume 87 of *Proceedings of Machine Learning Research*, pages 169–178. PMLR, 2018.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.

Mike Klaas, Nando de Freitas, and Arnaud Doucet. Toward practical N2 Monte Carlo: the marginal particle filter. In *UAI '05, Proceedings of the 21st Conference in Uncertainty in Artificial Intelligence, Edinburgh, Scotland, July 26-29, 2005*, pages 308–315. AUAI Press, 2005.

Rahul G. Krishnan, Uri Shalit, and David A. Sontag. Structured inference networks for nonlinear state space models. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA*, pages 2101–2109. AAAI Press, 2017.

Tuan Anh Le, Maximilian Igl, Tom Rainforth, Tom Jin, and Frank Wood. Auto-encoding sequential Monte Carlo. In *6th International Conference on Learning Representations,ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*, 2018.

Ming T Lin, Junni L Zhang, Qiansheng Cheng,and Rong Chen. Independent particle filters. *Journal of the American Statistical Association*, 100(472): 1412–1421, 2005.

Jun S Liu and Jun S Liu. *Monte Carlo strategies in scientific computing*, volume 10. Springer, 2001.

Xiao Ma, Péter Karkus, David Hsu, and Wee Sun Lee. Particle filter recurrent neural networks. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 5101–5108. AAAI Press, 2020a.

Xiao Ma, Péter Karkus, David Hsu, Wee Sun Lee, and Nan Ye. Discriminative particle filter reinforcement learning for complex partial observations. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020b.

Chris J. Maddison, Dieterich Lawson, George Tucker, Nicolas Heess,Mohammad Norouzi, Andriy Mnih, Arnaud Doucet, and Yee Whye Teh. Filtering variational objectives. In *Advances in Neural Information Processing Systems 30Annual Conference on Neural Information Processing Systems 2017, December*

4-9, 2017, Long Beach, CA, USA, pages 6573–6583, 2017a.

Chris J. Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*, 2017b.

Andrew C. Miller, Nicholas J. Foti, and Ryan P. Adams. Variational boosting: Iteratively refining posterior approximations. In *International Conference on Machine Learning*, pages 2420–2429. PMLR, 2017.

Antonio Khalil Moretti, Zizhao Wang, Luhuan Wu, Iddo Drori, and Itsik Pe'er. Particle smoothing variational objectives. *arXiv preprint arXiv:1909.09734*, 2019.

Christian A. Naesseth, Fredrik Lindsten, and Thomas B. Schön. Nested sequential Monte Carlo methods. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 1292–1301. JMLR.org, 2015.

Christian A. Naesseth,Scott W. Linderman, Rajesh Ranganath, and David M. Blei. Variational sequential Monte Carlo. In *International Conference on Artificial Intelligence and Statistics, AISTATS 2018, 9-11 April 2018, Playa Blanca, Lanzarote, Canary Islands, Spain*, volume 84 of *Proceedings of Machine Learning Research*, pages 968–977. PMLR, 2018.

Christian A. Naesseth, Fredrik Lindsten, and Thomas B. Schön. Elements of sequential Monte Carlo. *Found. Trends Mach. Learn.*, 12(3):307–392, 2019.

George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. *arXiv preprint arXiv:1912.02762*, 2019.

Tom Rainforth, Adam Kosiorek, Tuan Anh Le, Chris Maddison, Maximilian Igl, Frank Wood, and Yee Whye Teh. Tighter variational bounds are not necessarily better. In *International Conference on Machine Learning*, pages 4277–4285. PMLR, 2018.

Rajesh Ranganath, Dustin Tran, and David M. Blei. Hierarchical variational models. In *Proceedings of the 33nd International Conference on Machine Learning, ICML 2016, New York City, NY, USA, June 19-24, 2016*, volume 48, pages 324–333. JMLR.org, 2016.

Danilo Jimenez Rezende and Shakir Mohamed.Variational inference with normalizing flows. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37, pages 1530–1538. JMLR.org, 2015.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International conference on machine learning*,pages 1278–1286. PMLR, 2014.

Tim Salimans, Diederik P. Kingma, and Max Welling. Markov chain Monte Carlo and variational inference: Bridging the gap. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37, pages 1218–1226. JMLR.org, 2015.

Sam Stites, Heiko Zimmermann, Hao Wu, Eli Sennesh, et al. Learning proposals for probabilistic programs with inference combinators. In *Proceedings of the 37th Conference in Uncertainty in Artificial Intelligence, UAI 2021, Online, July 27-30, 2021*, 2021.

Richard S. Sutton and Andrew G. Barto. Reinforcement learning: An introduction. *IEEE Transactions on Neural Networks*, 16:285–286, 2005.

Dustin Tran, David M. Blei, and Edoardo M. Airoldi. Copula variational inference. In *Advances in Neural Information Processing Systems 28:Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 3564–3572, 2015.

Dustin Tran, Rajesh Ranganath, and David M. Blei. Variational Gaussian process.In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.

Jan-Willem van de Meent, Brooks Paige, Hongseok Yang, and Frank Wood. An introduction to probabilistic programming. *arXiv preprint arXiv:1809.10756*, 2018.

Michael Zhu, Kevin Murphy, and Rico Jonschkowski. Towards differentiable resampling. *arXiv preprint arXiv:2004.11938*, 2020.

Supplementary Material: Variational Marginal Particle Filters

A Proof of the Operations

A.1 Proof of Lemma 1

Proof. We have that

$$\begin{aligned}\int Q(\omega)R(\omega)a(x|\omega)d\omega &= \int Q(\omega)R(\omega)\frac{1}{R(\omega)}\int \frac{Q_0(\omega,\nu)}{Q(\omega)}R_0(\omega,\nu)a_0(x|\omega,\nu)d\nu\, d\omega \\ &= \int\int Q_0(\omega,\nu)R_0(\omega,\nu)a_0(x|\omega,\nu)d\nu d\omega \\ &= y(x).\end{aligned}$$

For the variance, we have

$$\begin{aligned}\mathrm{Var}\, R_0(\omega,\nu) &= \mathrm{Var}\,\mathrm{E}[R_0(\omega,\nu)|\omega] + \mathrm{E}\,\mathrm{Var}[R_0(\omega,\nu)|\omega] \\ &\geq \mathrm{Var}\,\mathrm{E}[R_0(\omega,\nu)|\omega] \\ &= \mathrm{Var}\, R(\omega).\end{aligned}$$

For the lower bounds,

$$\begin{aligned}\mathrm{E}_{\omega,\nu}[\log R_0(\omega,\nu)] &= \mathrm{E}_\omega\ \mathrm{E}_{\nu|\omega}[\log R_0(\omega,\nu)] \\ &\leq \mathrm{E}_\omega\ \log \mathrm{E}_{\nu|\omega}[R_0(\omega,\nu)] \\ &= \mathrm{E}_\omega[\log R(\omega)],\end{aligned}$$

and $\mathrm{E}_\omega[\log R(\omega)] \leq \log Z$ because we have shown that R and a are valid estimator-coupling pair for y. □

A.2 Proof of Lemma 2

Proof.

$$\begin{aligned}&\int\cdots\int Q(\omega,\nu_1,\ldots,\nu_N)R(\omega,\nu_1,\ldots,\nu_N)a(x|\omega,\nu_1,\ldots,\nu_N)d\omega\, d\nu_1\ldots d\nu_N \\ &= \int\cdots\int Q_0(\omega)\prod_{i=1}^N Q_0(\nu_i|\omega)\left(\frac{1}{N}\sum_{i=1}^N R_0(\omega,\nu_i)\right)\left(\frac{\sum_{i=1}^N R_0(\omega,\nu_i)a_0(x|\omega,\nu_i)}{\sum_{i=1}^N R_0(\omega,\nu_i)}\right)d\omega\, d\nu_1\ldots d\nu_N \\ &= \frac{1}{N}\sum_{i=1}^N \int\cdots\int Q_0(\omega)\prod_{j=1}^N Q_0(\nu_j|\omega)R_0(\omega,\nu_i)a_0(x|\omega,\nu_i)d\omega\, d\nu_1\ldots d\nu_N \\ &= \frac{1}{N}\sum_{i=1}^N \int Q_0(\omega)Q_0(\nu_i|\omega)R_0(\omega,\nu_i)a_0(x|\omega,\nu_i)\, d\omega\, d\nu \\ &= y(x).\end{aligned}$$

□

A.3 Proof of Lemma 3

Proof. For ExtendTarget,

$$\begin{aligned}
&\int\int\int Q(\omega, \hat{x}, x^0) R(\omega, \hat{x}, x^0) a(x, x^0 | \omega, \hat{x}, x^0)\, dx^0\, d\hat{x}\, d\omega \\
&= \int\int\int Q_0(\omega) a_0(\hat{x}|\omega) r(x^0|\hat{x}) R_0(\omega) \frac{y^0(\hat{x}, x^0)/y(\hat{x})}{r(x^0|\hat{x})} \delta_{(\hat{x}, x^0)}(x, x^0)\, dx^0\, d\hat{x}\, d\omega \\
&= \int\int Q_0(\omega) a_0(\hat{x}|\omega) R_0(\omega)\, d\omega \int \frac{y^0(\hat{x}, x^0)}{y(\hat{x})} \delta_{(\hat{x}, x^0)}(x, x^0)\, dx^0\, d\hat{x} \\
&= \int\int y^0(\hat{x}, x^0) \delta_{(\hat{x}, x^0)}(x, x^0)\, dx^0\, d\hat{x} \\
&= y^0(x, x^0).
\end{aligned}$$

For ChangeTarget , proceed as above with $\delta_{x^0}(x^0)$ in place of $\delta_{(\hat{x}, x^0)}(x, x^0)$. The steps are the same until the second to last line, which becomes

$$\int\int y^0(\hat{x}, x^0) \delta_{x^0}(x^0)\, dx^0\, d\hat{x} = \int y^0(x, x^0) d\hat{x} = y^0(x^0).$$

□

B Derivation of SMC and MPF

B.1 Proof of Theorem 1

We first repeat relevant definitions from the algorithm and theorem statement. The weights are

$$w_1^i = \frac{f(x_1^{1,i}) g(y_1 | x_1^{1,i})}{r_1(x_1^{1,i})}, \qquad w_t^i = \frac{f(x_t^{t,i} | x_{t-1}^{t,i}) g(y_t | x_t^{t,i})}{r_t(x_t^{t,i} | x_{t-1}^{t,i})} \text{ for } t > 1.$$

The normalized weights are $\overline{w}_t^i = w_t^i / \sum_{j=1}^N w_t^j$.

We wish to show that, for all t,

$$Q_t^N(x_{1:1}^{1,1:N}, \ldots, x_{1:t}^{t,1:N}) = \prod_{i=1}^N \left[r_1(x_{1:1}^{1,i}) \prod_{\tau=2}^t \sum_{j=1}^N \overline{w}_{\tau-1}^j \delta_{x_{1:\tau-1}^{\tau-1,j}}(x_{1:\tau-1}^{\tau,i}) r_\tau(x_\tau^{\tau,i} | x_{\tau-1}^{\tau,i}) \right], \tag{14}$$

$$R_t^N(x_{1:1}^{1,1:N}, \ldots, x_{1:t}^{t,1:N}) = \prod_{\tau=1}^t \frac{1}{N} \sum_{i=1}^N w_\tau^i, \tag{15}$$

$$a_t^N(\cdot | x_{1:1}^{1,1:N}, \ldots, x_{1:t}^{t,1:N}) = \sum_{i=1}^N \overline{w}_t^i \delta_{x_{1:t}^{t,i}}(\cdot) \tag{16}$$

define an estimator-coupling pair for $p(x_{1:t}, y_{1:t})$.

We can check that these match the SMC algorithm at step t: that is, Q_t^N is sampling distribution, R_t^N is the likelihood estimator, and a_t^N is approximation to $p(x_{1:t} | y_{1:t})$. So, after proving this, the conclusion of the theorem follows immediately.

We will show inductively that (Q_t^N, R_t^N, a_t^N) are obtained by applying appropriate operations on estimator-coupling pairs. In particular, the procedure is

$$(Q_t^N, R_t^N, a_t^N) = \text{Replicate } Q_t, R_t, a_t; x_{1:t}^t, N ,$$

where

$$
\begin{aligned}
Q_1(x_{1:1}^1) &= r_1(x_1^1), \\
R_1(x_{1:1}^1) &= \frac{p(x_1^1, y_1)}{r_1(x_1^1)} = \frac{f(x_1^1)g(y_1|x_1^1)}{r_1(x_1^1)}, \\
a_1(\cdot|x_{1:1}^1) &= \delta_{x_{1:1}^1}(\cdot),
\end{aligned}
$$

and, for $t > 1$,

$$
(Q_t, R_t, a_t) = \text{ExtendTarget}\left(Q_{t-1}^N, R_{t-1}^N, a_{t-1}^N; p(x_{1:t-1}, y_{1:t-1}), p(x_{1:t}, y_{1:t}), r_t(x_t|x_{t-1})\right).
$$

The proof is a mechanical application of these operations.

For the base case, it is immediate that (Q_1, R_1, a_1) are an estimator-coupling pair for $p(x_1, y_1)$, and easy to verify that applying Lemma 2 yields (Q_1^N, R_1^N, a_1^N) that match Equations (14)–(16) and give an estimator-coupling pair for $p(x_1, y_1)$.

For the induction step ($t > 1$), we first apply Lemma 3 to obtain (Q_t, R_t, a_t) from $(Q_{t-1}^N, R_{t-1}^N, a_{t-1}^N)$ using the ExtendTarget operation. We get

$$
\begin{aligned}
Q_t(x_{1:1}^{1,1:N}, \ldots, x_{1:t-1}^{t-1,1:N}, x_{1:t}^t) &= Q_{t-1}^N(x_{1:1}^{1,1:N}, \ldots, x_{1:t-1}^{t-1,1:N}) \\
&\quad \cdot a_{t-1}^N(x_{1:t-1}^t | x_{1:1}^{1,1:N}, \ldots, x_{1:t-1}^{t-1,1:N}) \cdot r_t(x_t^t|x_{t-1}^t) \\
&= \left[\prod_{i=1}^N r_1(x_{1:1}^{1,i})\prod_{\tau=2}^{t-1}\sum_{j=1}^N w_{\tau-1}^j \delta_{x_{1:\tau-1}^{\tau-1,j}}(x_{1:\tau-1}^{\tau,i}) r_\tau(x_\tau^{\tau,i}|x_{\tau-1}^{\tau,i})\right] \\
&\quad \cdot \sum_{j=1}^N w_{t-1}^j \delta_{x_{1:t-1}^{t-1,j}}(x_{1:t-1}^t) r_\tau(x_t^t|x_{t-1}^t).
\end{aligned}
$$

$$
\begin{aligned}
R_t(x_{1:1}^{1,1:N}, \ldots, x_{1:t-1}^{t-1,1:N}, x_{1:t}^t) &= R_{t-1}^N(x_{1:1}^{1,1:N}, \ldots, x_{1:t-1}^{t-1,1:N}) \cdot \frac{p(x_{1:t}^t, y_{1:t})/p(x_{1:t-1}^t, y_{1:t})}{r_t(x_t^t|x_{t-1}^t)} \\
&= \prod_{\tau=1}^{t-1}\left[\frac{1}{N}\sum_{i=1}^N w_\tau^j \frac{f(x_t^t|x_{t-1}^t)g(y_t|x_t^t)}{r_t(x_t^t|x_{t-1}^t)}\right].
\end{aligned}
$$

$$
a_t(\cdot|x_{1:1}^{1,1:N}, \ldots, x_{1:t-1}^{t-1,1:N}, x_{1:t}^t) = \delta_{x_{1:t}^t}(\cdot).
$$

By Lemma 3, (Q_t, R_t, a_t) define an estimator-coupling pair for $p(x_{1:t}, y_{1:t})$.

It is now straightforward to verify that applying Lemma 2 give the triple (Q_t^N, R_t^N, a_t^N) defined in Equations (14)–(16), and therefore (Q_t^N, R_t^N, a_t^N) define an estimator-coupling pair for $p(x_{1:t}, y_{1:t})$, and the result is proved.

B.2 Proof of Theorem 2

We again repeat the relevant definitions. The weights used in VMPF are

$$
v_1^j = \frac{f(x_1^i)g(y_1|x_1^i)}{r_1(x_1^i)}, \quad v_t^j = \frac{\sum_{j=1}^N v_{t-1}^j f(x_t^i|x_{t-1}^j)g(y_t|x_t^i)}{\sum_{j=1}^N v_{t-1}^j r_t(x_t^i|x_{t-1}^j)} \text{ for } t > 1.
$$

The normalized weights are $\bar{v}_t^j = v_t^i / \sum_{j=1}^N v_t^j$.

We wish to show that, for all t,

$$Q_t^N(x_1^{1:N}, \ldots, x_t^{1:N}) = \prod_{i=1}^{N} \left[r_1(x_1^i) \prod_{\tau=2}^{t} \sum_{j=1}^{N} v_{\tau-1}^j \, r_\tau(x_\tau^i | x_{\tau-1}^j) \right], \tag{17}$$

$$R_t^N(x_1^{1:N}, \ldots, x_t^{1:N}) = \prod_{\tau=1}^{t} \frac{1}{N} \sum_{i=1}^{N} v_\tau^i, \tag{18}$$

$$a_t^N(\cdot | x_1^{1:N}, \ldots, x_t^{1:N}) = \sum_{i=1}^{N} v_t^i \delta_{x_t^i}(\cdot) \tag{19}$$

define an estimator-coupling pair for $p(x_t, y_{1:t})$.

We can check that these match the MPF algorithm at step t: that is, Q_t^N is sampling distribution, R_t^N is the likelihood estimator, and a_t^N is approximation to $p(x_t | y_{1:t})$. So, after proving this, the conclusion of the theorem follows immediately.

We will show inductively that (Q_t^N, R_t^N, a_t^N) are obtained by applying appropriate operations on estimator-coupling pairs to. In particular, the procedure is

$$(Q_t^N, R_t^N, a_t^N) = \text{Replicate}\,(Q_t, R_t, a_t; x_t, N)$$

for all t, where

$$\begin{aligned} Q_1(x_1) &= r_1(x_1), \\ R_1(x_1) &= \frac{p(x_1, y_1)}{r_1(x_1)} = \frac{f(x_1) g(y_1 | x_1)}{r_1(x_1)}, \\ a_1(\cdot | x_1) &= \delta_{x_1}(\cdot), \end{aligned}$$

and, for $t > 1$,

$$(Q_t^0, R_t^0, a_t^0) = \text{ChangeTarget}\left(Q_{t-1}^N, R_{t-1}^N, a_{t-1}^N; p(x_{t-1}, y_{1:t-1}), p(x_{t-1}, x_t, y_{1:t}), r_t(x_t | x_{t-1})\right),$$

$$(Q_t, R_t, a_t) = \text{Marginalize}\left(Q_t^0, R_t^0, a_t^0; x_{t-1}\right).$$

The proof is again a mechanical application of these operations.

The base case is identical to the base case in the proof of Theorem 1, and yields that (Q_1^N, R_1^N, a_1^N) have the form in Equations (17)–(19) and define an estimator-coupling pair for $p(x_1, y_1)$.

For the induction step ($t > 1$), we first apply Lemma 3 as in the proof of Theorem 1, except using the ChangeTarget operation instead of ExtendTarget, to get

$$\begin{aligned} Q_t^0(x_1^{1:N}, \ldots, x_{t-1}^{1:N}, x_{t-1}, x_t) &= \prod_{i=1}^{N} \left[r_1(x_1^i) \prod_{\tau=2}^{t-1} \sum_{j=1}^{N} v_{\tau-1}^j \, r_\tau(x_\tau^i | x_{\tau-1}^j) \right] \\ &\quad \cdot \sum_{j=1}^{N} v_{t-1}^j \, \delta_{x_{t-1}^j}(x_{t-1}) r_t(x_t | x_{t-1}), \\ R_t^0(x_1^{1:N}, \ldots, x_{t-1}^{1:N}, x_{t-1}, x_t) &= \prod_{\tau=1}^{t-1} \frac{1}{N} \sum_{i=1}^{N} v_\tau^i \, \frac{f(x_t | x_{t-1}) g(y_t | x_t)}{r_t(x_t | x_{t-1})}, \\ a_t^0(\cdot | x_1^{1:N}, \ldots, x_{t-1}^{1:N}, x_{t-1}, x_t) &= \delta_{x_t}(\cdot). \end{aligned}$$

which define an estimator-coupling pair for $p(x_t, y_{1:t})$.

Next, we apply the Marginalize operation using Lemma 1. Marginalizing x_{t-1} from Q_t^0 gives

$$\begin{aligned}
&Q_t(x_1^{1:N}, \ldots, x_{t-1}^{1:N}, x_t) \\
&= \int Q_t^0(x_1^{1:N}, \ldots, x_{t-1}^{1:N}, x_{t-1}, x_t)\, dx_{t-1} \\
&= \int \left[\prod_{i=1}^N r_1(x_1^i) \prod_{\tau=2}^{t-1} \sum_{j=1}^N w_{\tau-1}^j r_\tau(x_\tau^i | x_{\tau-1}^j) \right] \sum_{j=1}^N w_{t-1}^j \delta_{x_{t-1}^j}(x_{t-1}) r_t(x_t | x_{t-1})\, dx_{t-1} \\
&= \prod_{i=1}^N \left[r_1(x_1^i) \prod_{\tau=2}^{t-1} \sum_{j=1}^N w_{\tau-1}^j r_\tau(x_\tau^i | x_{\tau-1}^j) \right] \sum_{j=1}^N \int w_{t-1}^j \delta_{x_{t-1}^j}(x_{t-1}) r_t(x_t | x_{t-1})\, dx_{t-1} \\
&= \prod_{i=1}^N \left[r_1(x_1^i) \prod_{\tau=2}^{t-1} \sum_{j=1}^N w_{\tau-1}^j r_\tau(x_\tau^i | x_{\tau-1}^j) \right] \sum_{j=1}^N w_{t-1}^j r_t(x_t | x_{t-1}^j).
\end{aligned}$$

The conditional distribution is

$$\begin{aligned}
Q_t^0(x_{t-1} \mid x_1^{1:N}, \ldots, x_{t-1}^{1:N}, x_t) &= \frac{Q_t^0(x_1^{1:N}, \ldots, x_{t-1}^{1:N}, x_{t-1}, x_t)}{Q_t^0(x_1^{1:N}, \ldots, x_{t-1}^{1:N}, x_t)} \\
&= \frac{\sum_{j=1}^N w_{t-1}^j \delta_{x_{t-1}^j}(x_{t-1}) r_t(x_t | x_{t-1})}{\sum_{j=1}^N w_{t-1}^j r_t(x_t | x_{t-1}^j)}.
\end{aligned}$$

The new estimator R_t is the conditional expectation

$$\begin{aligned}
&R_t(x_1^{1:N}, \ldots, x_{t-1}^{1:N}, x_t) \\
&= \mathop{\mathbb{E}}_{Q_t^0(x_{t-1} \mid x_1^{1:N}, \ldots, x_{t-1}^{1:N}, x_t)} R_t^0(x_1^{1:N}, \ldots, x_{t-1}^{1:N}, x_{t-1}, x_t) \\
&= \int \frac{\sum_{j=1}^N w_{t-1}^j \delta_{x_{t-1}^j}(x_{t-1}) r_t(x_t | x_{t-1})}{\sum_{j=1}^N w_{t-1}^j r_t(x_t | x_{t-1}^j)} \left[\prod_{\tau=1}^{t-1} \left(\frac{1}{N} \sum_{i=1}^N v_\tau^i \frac{f(x_t | x_{t-1}) g(y_t | x_t)}{r_t(x_t | x_{t-1})} \right) \right] dx_{t-1} \\
&= \frac{1}{\sum_{j=1}^N w_{t-1}^j r_t(x_t^i | x_{t-1}^j)} \left[\prod_{\tau=1}^{t-1} \frac{1}{N} \sum_{i=1}^N v_\tau^i \right] \int \sum_{j=1}^N w_{t-1}^j \delta_{x_{t-1}^j}(x_{t-1}) f(x_t | x_{t-1}) g(y_t | x_t)\, dx_{t-1} \\
&= \left[\prod_{\tau=1}^{t-1} \frac{1}{N} \sum_{i=1}^N v_\tau^i \right] \frac{\sum_{j=1}^N w_{t-1}^j f(x_t | x_{t-1}^j) g(y_t | x_t)}{\sum_{j=1}^N w_{t-1}^j r_t(x_t^i | x_{t-1}^j)}.
\end{aligned}$$

The new coupling a_t is

$$\begin{aligned}
&a_t(x_1^{1:N}, \ldots, x_{t-1}^{1:N}, x_t) \\
&= \frac{1}{R_t(x_1^{1:N}, \ldots, x_{t-1}^{1:N}, x_t)} \\
&\quad \cdot \mathop{\mathbb{E}}_{Q_t^0(x_{t-1} \mid x_1^{1:N}, \ldots, x_{t-1}^{1:N}, x_t)} R_t^0(x_1^{1:N}, \ldots, x_{t-1}^{1:N}, x_{t-1}, x_t) a_t^0(\cdot | x_1^{1:N}, \ldots, x_{t-1}^{1:N}, x_{t-1}, x_t) \\
&= \frac{1}{R_t(x_1^{1:N}, \ldots, x_{t-1}^{1:N}, x_t)} \mathop{\mathbb{E}}_{Q_t^0(x_{t-1} \mid x_1^{1:N}, \ldots, x_{t-1}^{1:N}, x_t)} R_t^0(x_1^{1:N}, \ldots, x_{t-1}^{1:N}, x_{t-1}, x_t)\ \delta_{x_t}(\cdot) \\
&= \delta_{x_t}(\cdot).
\end{aligned}$$

It is now straightforward to verify that applying Lemma 2 give the triple (Q_t^N, R_t^N, a_t^N) defined in Equations (17)–(19), and therefore (Q_t^N, R_t^N, a_t^N) define an estimator-coupling pair for $p(x_t, y_{1:t})$, and the result is proved.

Algorithm 3 Independent Particle Filters

Require: $p(x_{1:T}, y_{1:T})$, $y_{1:T}$, $\{r_t(x_t)\}$, N

1: Sample $x_1^i \sim r_1(x_1)$ for all i
2: Set $u_1^i = \frac{f(x_1^i)g(y_1|x_1^i)}{r_1(x_1^i)}$ for all i
3: **for** $t = 2, \ldots, T$ **do**
4: Generate L permutations $k_t^{(1:L,1:N)}$
5: **for** $i = 1, \ldots, N$ **do**
6: Sample $x_t^i \sim r_t(x_t)$
7: Set $u_t^i = \frac{\sum_{l=1}^{L} u_{t-1}^{k_{l,i}} f(x_t^i|x_{t-1}^{k_{l,i}})g(y_t|x_t^i)}{L \cdot r_t(x_t^i)}$

Algorithm 4 Tensor Monte Carlo for SSM

Require: $p(x_{1:T}, y_{1:T})$, $y_{1:T}$, $\{r_t(x_t)\}$, N

1: Sample $x_1^i \sim r_1(x_1)$ for all i
2: Set $z_1^i = \frac{f(x_1^i)g(y_1|x_1^i)}{r_1(x_1^i)}$ for all i
3: **for** $t = 2, \ldots, T$ **do**
4: **for** $i = 1, \ldots, N$ **do**
5: Sample $x_t^i \sim r_t(x_t)$
6: Set $z_t^i = \frac{\sum_{j=1}^{N} z_{t-1}^j f(x_t^i|x_{t-1}^j)g(y_t|x_t^i)}{N \cdot r_t(x_t^i)}$

C Deriving MPF from IPF

We directly give the detail of IPF in Algorithm 3. In line 4 of Algorithm 3, L distinct permutations ($k_t^{l_1,i} \neq k_t^{l_2,i}$ for any l_1, l_2, i satisfying $l_1 \neq l_2$) of $1, \ldots, N$ should be generated. Proposition 1 of IPF (Lin et al., 2005) implies that for any test function h,

$$\mathrm{E}\left[\frac{1}{N}\sum_{i=1}^{N} u_t^i h(x_t^i)\right] = p(y_{1:t}) \cdot \mathrm{E}_{p(x_t|y_{1:t})}[h(x_t)]. \tag{20}$$

If $h \equiv 1$, we have $\mathrm{E}\left[\frac{1}{N}\sum_{i=1}^{N} u_t^i\right] = p(y_{1:t})$.

The idea of TMC is to draw many copies of each individual variable from independent proposal distributions, and then average over all (exponentially many) combinations of joint samples to get an unbiased estimator. We assume for each latent variable x_t, we have N samples $x_t^i \sim r_t(x_t)$, $i = 1, \ldots, N$. So

$$\begin{aligned} p_{TMC} &= \frac{1}{N^T}\sum_{i_1,i_2,\ldots,i_T} \frac{p(x_1^{i_1}, \ldots, x_T^{i_T}, y_{1:T})}{r_1(x_1^{i_1}) \ldots r_T(x_T^{i_T})} \\ &= \frac{1}{N^T}\sum_{i_1,i_2,\ldots,i_T} \frac{f(x_1^{i_1})g(y_1|x_1^{i_1})}{r_1(x_1^{i_1})} \cdot \frac{f(x_2^{i_2}|x_1^{i_1})g(y_2|x_2^{i_2})}{r_2(x_2^{i_2})} \cdot \ldots \cdot \frac{f(x_T^{i_T}|x_{T-1}^{i_{T-1}})g(y_T|x_T^{i_T})}{r_T(x_T^{i_T})} \end{aligned} \tag{21}$$

will be an unbiased estimator for $p(y_{1:T})$. Computing the summation in Equation (21) can be accelerated by summing over $i_1, i_2, \ldots, i_T$ in order. If we define $z_1^i = \frac{f_1(x_1^i)g(y_1|x_1^i)}{r_1(x_1^i)}$, then summing over i_1 gives

$$\begin{aligned} p_{TMC} &= \frac{1}{N^T}\sum_{i_1,i_2,\ldots,i_T} z_1^{i_1} \cdot \frac{f(x_2^{i_2}|x_1^{i_1})g(y_2|x_2^{i_2})}{r_2(x_2^{i_2})} \cdot \ldots \cdot \frac{f(x_T^{i_T}|x_{T-1}^{i_{T-1}})g(y_T|x_T^{i_T})}{r_T(x_T^{i_T})} \\ &= \frac{1}{N^{T-1}}\sum_{i_2,\ldots,i_T} \boxed{\sum_{i_1=1}^{N} \frac{z_1^{i_1} f(x_2^{i_2}|x_1^{i_1})g(y_2|x_2^{i_2})}{N \cdot r_2(x_2^{i_2})}} \cdot \ldots \cdot \frac{f(x_T^{i_T}|x_{T-1}^{i_{T-1}})g(y_T|x_T^{i_T})}{r_T(x_T^{i_T})}. \end{aligned} \tag{22}$$

We can further define the boxed variable in Equation (22) by $z_2^{i_2}$ and continue the summation. This gives a filtering style framework as in Algorithm 4. In the end, $\frac{1}{N}\sum_{i=1}^{N} z_t^i$ will also be an unbiased estimator for $p(y_{1:t})$.

C.1 Deriving TMC for SSM from IPF

Now we show that TMC is IPF with complete matching ($L = N$). Under this circumstance, $k_t^{1,i}, \ldots, k_t^{N,i}$ will also be a permutation of $1, \ldots, N$. So Line 7 of Algorithm 3 will become

$$u_t^i = \frac{\sum_{j=1}^{N} u_{t-1}^j f(x_t^i|x_{t-1}^j)g(y_t|x_t^i)}{N \cdot r_t(x_t^i)},$$

which exactly matches Line 6 of Algorithm 4.

C.2 Deriving MPF from TMC for SSM

The proposal distributions of IPF and TMC can be extended to condition on all past particles, i.e. $r_t = r_t(x_t | x_{1:t-1}^{1:N})$ (See 7.2 of Aitchison (2019) about non-factorized TMC), which includes the case of MPF. However, no formal theory supports any specific form of proposal distribution, such as the mixture distribution used by MPF. Additionally, full details of how to construct the non-factorised approximate posterior and ensure differentiability are not given. Finally, by using couplings to derive MPF and VMPF, we obtain the interpretation as auxiliary variable VI as described in the main text, which is not obvious from the unbiasedness result of Equation (20) alone.

This paper fills these gaps by deriving MPF from SMC directly and revealing the KL decomposition for VMPF. Now we verify that MPF is TMC with a specific mixture distribution. If we define $v_t^j = z_t^j / \prod_{\tau=1}^{t-1} \frac{1}{N} \sum_{j=1}^N v_\tau^j$ and replace the proposal distributions for $t > 1$ by $\sum_{j=1}^N v_{t-1}^j r_t(x_t^i | x_{t-1}^j)$, we will have $v_1^i = z_1^i$ for all i and

$$
\begin{aligned}
z_t^i &= \frac{\sum_{j=1}^N z_{t-1}^j f(x_t^i | x_{t-1}^j) g(y_t | x_t^i)}{N \cdot \sum_{j=1}^N v_{t-1}^j r_t(x_t^i | x_{t-1}^j)} \\
\Leftrightarrow \left(\prod_{\tau=1}^{t-1} \frac{1}{N} \sum_{j=1}^N v_\tau^j \right) v_t^i &= \left(\prod_{\tau=1}^{t-2} \frac{1}{N} \sum_{j=1}^N v_\tau^j \right) \frac{\sum_{j=1}^N v_{t-1}^j f(x_t^i | x_{t-1}^j) g(y_t | x_t^i)}{N \cdot \sum_{j=1}^N v_{t-1}^j r_t(x_t^i | x_{t-1}^j)} \\
\Leftrightarrow \left(\prod_{\tau=1}^{t-1} \frac{1}{N} \sum_{j=1}^N v_\tau^j \right) v_t^i &= \left(\prod_{\tau=1}^{t-1} \frac{1}{N} \sum_{j=1}^N v_\tau^j \right) \frac{\sum_{j=1}^N v_{t-1}^j f(x_t^i | x_{t-1}^j) g(y_t | x_t^i)}{\sum_{j=1}^N v_{t-1}^j r_t(x_t^i | x_{t-1}^j)} \\
\Leftrightarrow v_t^i &= \frac{\sum_{j=1}^N v_{t-1}^j f(x_t^i | x_{t-1}^j) g(y_t | x_t^i)}{\sum_{j=1}^N v_{t-1}^j r_t(x_t^i | x_{t-1}^j)}.
\end{aligned} \tag{23}
$$

Therefore, the weight computation of MPF is equivalent to Line 6 of Algorithm 4 if the specific mixture distribution is chosen. And we directly have the unbiased estimator

$$
\frac{1}{N} \sum_{i=1}^N z_t^i = \prod_{\tau=1}^t \frac{1}{N} \sum_{i=1}^N v_\tau^i \tag{24}
$$

to be in the same form as MPF.

D Limitation on VRNN

A VRNN is a sequential latent variable model

$$
p(x_{1:T}, y_{1:T}) = \prod_{t=1}^T f(x_t | h_t) g(y_t | x_t, h_t)
$$

where h_1 is constant and $h_t = h(x_{t-1}, y_{t-1}, h_{t-1})$ for $t > 1$. The variational posterior distribution is factorized as

$$
q(x_{1:T} | y_{1:T}) = \prod_{t=1}^T q_t(x_t | h_t, y_t).
$$

We interpret a VRNN as a SSM that has both deterministic latent variables h_t and stochastic latent variables x_t. The full model is

$$
p(x_{1:T}, h_{2:T}, y_{1:T}) = \prod_{t=1}^T f(x_t | h_t) g(y_t | x_t, h_t) \prod_{t=2}^T \delta_{h(x_{t-1}, y_{t-1}, h_{t-1})}(h_t).
$$

To be clearer, we define $X_t = (x_t, h_t)$ to be the full latent variable, so the model becomes $p(X_{1:T}, y_{1:T})$. VSMC can be used to learn the parameters of a VRNN. However, VMPF is not applicable to learning VRNN parameters. If we set the proposal distribution to be

$$
r_t(X_t | X_{t-1}, y_{1:t}) = \delta_{h(x_{t-1}, y_{t-1}, h_{t-1})}(h_t) q_t(x_t | h_t, y_t),
$$

Table 3: Training schedule on stochastic volatility model.

	Diagonal B		Triangular B	
Phase	Learning Rate	Epochs	Learning Rate	Epochs
1	0.01	50000	0.003	100000
2	0.001	50000	0.0003	100000
3	0.0001	50000	0.00003	100000
4	0.00001	50000	0.000003	100000

the weighting function for VMPF for $t > 1$ would become

$$v_t^i = \frac{\sum_{j=1}^N v_{t-1}^j \delta_{h(x_{t-1}^j, y_{t-1}, h_{t-1}^j)}(h_t^i) f(x_t^i | h_t^i) g(y_t | x_t^i, h_t^i)}{\sum_{j=1}^N v_{t-1}^j \delta_{h(x_{t-1}^j, y_{t-1}, h_{t-1}^j)}(h_t^i) q_t(x_t^i | h_t, y_t)}.$$

We know that, in general, for all j, the values $h(x_{t-1}^j, y_{t-1}, h_{t-1}^j)$ will be different from each other. Due to the presence of Dirac function, the summation in the numerator as well as the denominator will only have one non-zero term, so VMPF collapses to VSMC. This failure can be understood in terms of marginalization: if we are able to access h_t^i with MPF at t, we can reconstruct the corresponding (x_{t-1}^j, h_{t-1}^j) that generates (x_t^i, h_t^i), but (x_{t-1}^j, h_{t-1}^j) should be marginalized.

E Experiment Details

We run all experiments on CPU. For models with multiple data, the batch size is 1. The experiment details can be found below.

E.1 Details of experiments on stochastic volatility models

Suppose the raw data of the exchange rate is $d_{0:T} = (d_0, d_1, \ldots, d_T)$, we get $y_{1:T}$ by

$$y_t = \log d_t - \log d_{t-1}$$

for $1 \le t \le T$. For both diagonal B and triangular B case, we restrict elements of Φ to be in $[0, 1]$. For all algorithms, we train with the same learning rate scheduler (See Table 3), except for VMPF-UG for diagonal B with $N = 16$, where we train with 0.3x of the listed learning rate. Further reducing the learning rate has little effect on the results. To stabilize the training of VMPF-UG, we use gradient clipping with threshold 100. With that, VMPF-UG for diagonal B with $N = 8, 16$ are not stable in early iterations, and we only keep stable runs.

E.2 Details of experiments on deep Markov models

The four polyphonic music datasets are sequences of 88-dimensional binary vectors. Recall that the DMM is

$$x_t = \mu_\theta(x_{t-1}) + \mathrm{diag}(\exp(\sigma_\theta(x_{t-1})/2))v_t,$$
$$y_t \sim \mathrm{Bernoulli}(\mathrm{sigmoid}(\eta_\theta(x_t))),$$

where $v_t \sim N(0, I)$, $x_0 = 0$, and $\mu_\theta, \sigma_\theta, \eta_\theta$ are neural networks. The architectures are

$$\mu_\theta, \sigma_\theta = \mathrm{Linear}(d_h \to 176) \circ \mathrm{LeakyReLU} \circ \mathrm{Linear}(88 \to d_h),$$
$$\eta_\theta = \mathrm{Linear}(d_h \to 88) \circ \mathrm{LeakyReLU} \circ \mathrm{Linear}(88 \to d_h)$$

where d_h varies on different datasets. And we use the proposal distribution

$$r(x_t | x_{t-1}, y_t; \varphi) \propto N(x_t; \mu_\varphi^x(x_{t-1}), \mathrm{diag}(\exp(\sigma_\varphi^x(x_{t-1})))) \cdot N(x_t; \mu_\varphi^y(y_t), \mathrm{diag}(\exp(\sigma_\varphi^y(y_t))))$$

where $\mu_\varphi^x, \sigma_\varphi^x, \mu_\varphi^y$ and σ_φ^y are neural networks. The architectures are

$$\mu_\varphi^x, \sigma_\varphi^x = \mathrm{Linear}(d_h \to 176) \circ \mathrm{LeakyReLU} \circ \mathrm{Linear}(88 \to d_h),$$
$$\mu_\varphi^y, \sigma_\varphi^y = \mathrm{Linear}(d_h \to 176) \circ \mathrm{LeakyReLU} \circ \mathrm{Linear}(88 \to d_h).$$

Table 4: Training details of DMMs.

	Nottingham	JSB	MuseData	Piano-midi.de
Learning Rate	0.001,0.0001	0.001,0.0001	0.001,0.0001	0.001,0.0001
Epochs	500,100	1000,200	350,100	600,150
Hidden Units d_h	128	64	128	128

For training, we use the default train, validation, and test set split. For all datasets, the learning rate, training epochs and hidden units h_t can be found in Table 4. The comma separates different phases of training:we train with the larger learning rates for some epochs and reduce the learning rate for some additional epochs.We report the performance on test set with the best parameters during training determined by the validation set.