Replica Conditional Sequential Monte Carlo

Alexander Y. Shestopaloff [1 2] Arnaud Doucet [3 2]

Abstract

We propose a Markov chain Monte Carlo (MCMC) scheme to perform state inference in non-linear non-Gaussian state-space models. Current state-of-the-art methods to address this problem rely on particle MCMC techniques and its variants, such as the iterated conditional Sequential Monte Carlo (cSMC) scheme, which uses a Sequential Monte Carlo (SMC) type proposal within MCMC. A deficiency of standard SMC proposals is that they only use observations up to time t to propose states at time t when an entire observation sequence is available. More sophisticated SMC based on lookahead techniques could be used but they can be difficult to put in practice. We propose here replica cSMC where we build SMC proposals for one replica using information from the entire observation sequence by conditioning on the states of the other replicas. This approach is easily parallelizable and we demonstrate its excellent empirical performance when compared to the standard iterated cSMC scheme at fixed computational complexity.

1. Introduction

We consider discrete-time state-space models. They can be described by a latent Markov process $(X_t)_{t\geq 1}$ and an observation process $(Y_t)_{t\geq 1}$, (X_t, Y_t) being $\mathcal{X} \times \mathcal{Y}$-valued, which satisfy $X_1 \sim \mu(\cdot)$ and

$$X_{t+1}|\{X_t = x\} \sim f(\cdot|x) \qquad Y_t|\{X_t = x\} \sim g(\cdot|x) \tag{1}$$

for $t \geq 1$. Our goal is to sample from posterior distribution of the latent states $X_{1:T} := (X_1, ..., X_T)$ given a realization of the observations $Y_{1:T} = y_{1:T}$. This distribution admits a density given by

$$p(x_{1:T}|y_{1:T}) \propto \mu(x_1)g(y_1|x_1)\prod_{t=2}^{T} f(x_t|x_{t-1})g(y_t|x_t). \tag{2}$$

This sampling problem is now commonly addressed using an MCMC scheme known as the iterated cSMC sampler (Andrieu et al., 2010) and extensions of it; see, e.g., (Shestopaloff & Neal, 2018). This algorithm relies on a SMC-type proposal mechanism. A limitation of these algorithms is that they typically use data only up to time t to propose candidate states at time t, whereas the entire sequence $y_{1:T}$ is observed in the context we are interested in. To address these issues, various lookahead techniques have been proposed in the SMC literature; see (Chen et al., 2013) for a review. Alternative approaches relying on a parametric approximation of the backward information filter used for smoothing in state-space models (Briers et al., 2010) have also been recently proposed in (Scharth & Kohn, 2016; Guarniero et al., 2017; Ruiz & Kappen, 2017; Heng et al., 2017). When applicable, these iterative methods have demonstrated good performance. However, it is unclear how these ideas could be adapted to the MCMC framework investigated here. Additionally these methods are difficult to put in practice for multimodal posterior distributions.

In this paper, we propose a novel approach which allows us to build proposals for cSMC that allows considering all observed data in a proposal, based on conditioning on replicas of the state variables. Our approach is based purely on Monte Carlo sampling, bypassing any need for approximating functions in the estimate of the backward information filter.

The rest of this paper is organized as follows. In Section 2, we review the iterated cSMC algorithm and outline its limitations. Section 3 introduces the replica iterated cSMC methodology. In Section 4, we demonstrate the methodology on a linear Gaussian model, two non-Gaussian state space models from (Shestopaloff & Neal, 2018) as well as the Lorenz-96 model from (Heng et al., 2017).

2. Iterated cSMC

The iterated cSMC sampler is an MCMC method for sampling from a target distribution of density $\pi(x_{1:T}) :=$

[1]School of Mathematics, University of Edinburgh, Edinburgh, UK [2]The Alan Turing Institute, London, UK [3]Department of Statistics, University of Oxford, Oxford, UK. Correspondence to: Alexander Y. Shestopaloff <ashestopaloff@turing.ac.uk>.

Proceedings of the 36th International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019. Copyright 2019 by the author(s).

Algorithm 1 Iterated cSMC kernel $K\left(x_{1:T}, x'_{1:T}\right)$

```
cSMC step.
  1. At time $t = 1$
     (a) Sample $b_1$ uniformly on $[N]$ and set $x_1^{b_1} = x_1$.
     (b) For $i \in [N] \setminus \{b_1\}$, sample $x_1^i \sim q_1(\cdot)$.
     (c) Compute $w_1(x_0, x_1^i)$ for $i \in [N]$.
  2. At times $t = 2, \dots, T$
     (a) Sample $b_t$ uniformly on $[N]$ and set $x_t^{b_t} = x_t$.
     (b) For $i \in [N] \setminus \{b_t\}$, sample
         $a_{t-1}^i \sim \mathrm{Cat}\{w_{t-1}(x_{t-2}^{a_{t-2}^j}, x_{t-1}^j); j \in [N]\}$.
     (c) For $i \in [N] \setminus \{b_t\}$, sample $x_t^i \sim q_t(\cdot \mid x_{t-1}^{a_{t-1}^i})$.
     (d) Compute $w_t(x_{t-1}^{a_{t-1}^i}, x_t^i)$ for $i \in [N]$.
Backward sampling step.
  1. At times $t = T$
     (a) Sample $b_T \sim \mathrm{Cat}\{w_T(x_{T-1}^{a_{T-1}^j}, x_T^j); j \in [N]\}$.
  2. At times $t = T - 1, ..., 1$
     (a) Sample $b_t \sim$
         $\mathrm{Cat}\{\beta_{t+1}(x_t^j, x_{t+1}^{b_{t+1}}) w_t(x_{t-1}^{a_{t-1}^j}, x_t^j); j \in [N]\}$.
Output $x'_{1:T} = x_{1:T}^{b_{1:T}} := \left(x_1^{b_1}, \dots, x_T^{b_T}\right)$.
```

$\pi_T\left(x_{1:T}\right)$. It relies on a modified SMC scheme targeting a sequence of auxiliary target probability densities $\{\pi_t\left(x_{1:t}\right)\}_{t=1,...,T-1}$ and a sequence of proposal densities $q_1\left(x_1\right)$ and $q_t(x_t|x_{t-1})$ for $t \in \{2, ..., T\}$. These target densities are such that $\pi_t(x_{1:t})/\pi_{t-1}(x_{1:t-1}) \propto \beta_t(x_{t-1}, x_t)$.

2.1. Algorithm

We define the 'incremental importance weights' for $t \geq 2$ as

$$w_t(x_{t-1}, x_t) := \frac{\pi_t\left(x_{1:t}\right)}{\pi_{t-1}\left(x_{1:t-1}\right) q_t(x_t|x_{t-1})} \propto \frac{\beta_t(x_{t-1}, x_t)}{q_t(x_t|x_{t-1})} \tag{3}$$

and for $t = 1$ as

$$w_1(x_0, x_1) := \frac{\pi_1(x_1)}{q_1(x_1)}. \tag{4}$$

We introduce a dummy variable x_0 to simplify notation. We let $N \geq 2$ be the number of particles used by the algorithm and $[N] := \{1, ..., N\}$. We introduce the notation $\mathbf{x}_t = \left(x_t^1, \dots, x_t^N\right) \in \mathcal{X}^N$, $\mathbf{a}_t = \left(a_t^1, \dots, a_t^N\right) \in \{1, \dots, N\}^N$, $\mathbf{x}_{1:T} = (\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_T)$, $\mathbf{a}_{1:T-1} = (\mathbf{a}_1, \mathbf{a}_2, ..., \mathbf{a}_{T-1})$ and $\mathbf{x}_t^{-b_t} = \mathbf{x}_t \backslash x_t^{b_t}$, $\mathbf{x}_{1:T}^{-b_{1:T}} = \left\{\mathbf{x}_1^{-b_1}, \dots, \mathbf{x}_T^{-b_T}\right\}$, $\mathbf{a}_{t-1}^{-b_t} = \mathbf{a}_{t-1} \backslash a_{t-1}^{b_t}$, $\mathbf{a}_{1:T-1}^{-b_{2:T}} = \left\{\mathbf{a}_1^{-b_2}, \dots, \mathbf{a}_{T-1}^{-b_T}\right\}$ and set $b_t = a_t^{b_{t+1}}$ for $t = 1, ..., T - 1$.

It can be shown that the iterated cSMC kernel, described in Algorithm 1, is invariant w.r.t. $\pi(x_{1:T})$. Given the current state $x_{1:T}$, the cSMC step introduced in (Andrieu et al., 2010) samples from the following distribution

$$\Phi(\mathbf{x}_{1:T}^{-b_{1:T}}, \mathbf{a}_{1:T-1}^{-b_{2:T}} \big| x_{1:T}^{b_{1:T}}, b_{1:T}) = \delta_{x_{1:T}}\left(x_{1:T}^{b_{1:T}}\right) \times \prod_{i=1, i \neq b_1}^{N} q_1\left(x_1^i\right) \prod_{t=2}^{T} \prod_{i=1, i \neq b_t}^{N} \lambda(a_{t-1}^i, x_t^i \big| \mathbf{x}_{t-1}), \tag{5}$$

where

$$\lambda\left(a_{t-1}^i = k, x_t^i \big| \mathbf{x}_{t-1}\right) = \frac{w_{t-1}(x_{t-2}^{a_{t-1}^k}, x_{t-1}^k)}{\sum_{j=1}^N w_{t-1}(x_{t-2}^{a_{t-1}^j}, x_{t-1}^j)} \times q_t(x_t^i \big| x_{t-1}^k). \tag{6}$$

This can be combined to a backward sampling step introduced in (Whiteley, 2010); see (Finke et al., 2016; Shestopaloff & Neal, 2018) for a detailed derivation. It can be shown that the combination of these two steps defined a Markov kernel that preserves the following extended target distribution

$$\gamma(\mathbf{x}_{1:T}, \mathbf{a}_{1:T-1}^{-b_{2:T}}, b_{1:T}) := \frac{\pi(x_{1:T}^{b_{1:T}})}{N^T} \times \Phi(\mathbf{x}_{1:T}^{-b_{1:T}}, \mathbf{a}_{1:T-1}^{-b_{2:T}} \big| x_{1:T}^{b_{1:T}}, b_{1:T}) \tag{7}$$

as invariant distribution. In particular, it follows that if $x_{1:T} \sim \pi$ then $x'_{1:T} \sim \pi$. The algorithm is described in Algorithm 1 where we use the notation $\mathrm{Cat}\{c_i; i \in [N]\}$ to denote the categorical distribution of probabilities $p_i \propto c_i$.

Iterated cSMC has been widely adopted for state space models, i.e. when the target is $\pi(x_{1:T}) = p(x_{1:T}|y_{1:T})$. The default sequence of auxiliary targets one uses is $\pi_t(x_{1:t}) = p(x_{1:t}|y_{1:t})$ for $t = 1, ..., T - 1$ resulting in the incremental importance weights

$$w_t(x_{t-1}, x_t) \propto \frac{f(x_t|x_{t-1}) g(y_t|x_t)}{q_t(x_t|x_{t-1})} \tag{8}$$

for $t \geq 2$ and

$$w_1(x_0, x_1) \propto \frac{\mu(x_1) g(y_1|x_1)}{q_1(x_1)} \tag{9}$$

for $t = 1$. Typically we will attempt to select a proposal which minimizes the variance of the incremental weight, which at time $t \geq 2$ is $q_t^{\mathrm{opt}}(x_t|x_{t-1}) = p(x_t|x_{t-1}, y_t) \propto g(y_t|x_t) f(x_t|x_{t-1})$ or an approximation of it.

2.2. Limitations of Iterated cSMC

When using the default sequence of auxiliary targets for state space models, iterated cSMC does not exploit a key feature of the problem at hand. The cSMC step typically uses a proposal at time t that only relies on the observation y_t, i.e. $q_t(x_t|x_{t-1}) = p\,(x_t|x_{t-1}, y_t)$, as it targets at time t the posterior density $p\,(x_{1:t}|y_{1:t})$. In high-dimensions and/or in the presence of highly informative observations, the discrepancy between successive posterior densities $\{p\,(x_{1:t}|y_{1:t})\}_{t\geq 1}$ will be high. Consequently the resulting importance weights $\{w_t(x_{t-1}^{a_{t-1}^i}, x_t^i); i \in [N]\}$ will have high variance and the resulting procedure will be inefficient.

Ideally one would like to use the sequence of marginal smoothing densities as auxiliary densities, that is $\pi_t(x_{1:t}) = p\,(x_{1:t}|y_{1:T})$ for $t = 1, ..., T-1$. Unfortunately, this is not possible as $p\,(x_{1:t}|y_{1:T}) \propto p\,(x_{1:t}|y_{1:t-1})\,p\,(y_{t:T}|x_t)$ cannot be evaluated pointwise up to a normalizing constant. To address this problem in a standard SMC framework, recent contributions (Scharth & Kohn, 2016; Guarniero et al., 2017; Ruiz & Kappen, 2017; Heng et al., 2017) perform an analytical approximation $\hat{p}\,(y_{t:T}|x_t)$ of the backward information filter $p\,(y_{t:T}|x_t)$ based on an iterative particle mechanism and target instead $\{\hat{p}\,(x_{1:t}|y_{1:T})\}_{t\geq 1}$ where $\hat{p}\,(x_{1:t}|y_{1:T}) \propto p\,(x_{1:t}|y_{1:t-1})\,\hat{p}\,(y_{t:T}|x_t)$ using proposals of the form $q_t\,(x_t|x_{t-1}) \propto f\,(x_t|x_{t-1})\,\hat{p}\,(y_{t:T}|x_t)$. These methods can perform well but it requires a careful design of the analytical approximation and is difficult to put in practice for multimodal posteriors. Additionally, it is unclear how these could be adapted in an iterated cSMC framework without introducing any bias.

Versions of iterated cSMC using an independent approximation to the backward information filter based on Particle Efficient Importance Sampling (Scharth & Kohn, 2016) have been proposed (Grothe et al., 2016) though they still require a choice of analytical approximation and use an approximation to the backward information filter which is global. This can become inefficient in high dimensional state scenarios.

3. Replica Iterated cSMC

We introduce a way to directly use the iterated cSMC algorithm to target a sequence of approximations $\{\hat{p}\,(x_{1:t}|y_{1:T})\}_{t\geq 1}$ to the marginal smoothing densities of a state space model. Our proposed method is based on sampling from a target over multiple copies of the space as done in, for instance, the Parallel Tempering or Ensemble MCMC (Neal, 2010) approaches. However, unlike in these techniques, we use copies of the space to define a sequence of intermediate distributions in the cSMC step informed by the whole dataset. This enables us to draw samples of $X_{1:T}$ that incorporate information about all of the observed data. Related recent work includes (Leimkuhler et al., 2018), where information sharing amongst an ensemble of replicas is used to improve MCMC proposals.

3.1. Algorithm

We start by defining the replica target for some $K \geq 2$ by

$$\bar{\pi}(x_{1:T}^{(1:K)}) = \prod_{k=1}^{K} p(x_{1:T}^{(k)}|y_{1:T}). \tag{10}$$

Each of the replicas $x_{1:T}^{(k)}$ is updated in turn by running Algorithm 1 with a different sequence of intermediate targets which we describe here. Consider updating replica k and let $\hat{p}^{(k)}(y_{t+1:T}|x_t)$ be an estimator of the backward information filter, built using replicas other than the k-th one, $x_{t+1}^{(-k)} = (x_{t+1}^{(1)}, \dots, x_{t+1}^{(k-1)}, x_{t+1}^{(k+1)}, \dots, x_{t+1}^{(K)})$. For convenience of notation, we take $\hat{p}^{(k)}(y_{T+1:T}|x_T) := 1$. At time t, the cSMC does target approximation of the marginal smoothing distribution $p\,(x_{1:t}|y_{1:T})$ as in (Scharth & Kohn, 2016; Guarniero et al., 2017; Ruiz & Kappen, 2017; Heng et al., 2017). This is of the form $\hat{p}^{(k)}\,(x_{1:t}|y_{1:T}) \propto p\,(x_{1:t}|y_{1:t})\,\hat{p}^{(k)}\,(y_{t+1:T}|x_t)$. This means that the cSMC for replica k uses the novel incremental weights at time $t \geq 2$

$$\begin{aligned} w_t^{(k)}(x_{t-1}, x_t) &:= \frac{\hat{p}^{(k)}\,(x_{1:t}|y_{1:T})}{\hat{p}^{(k)}\,(x_{1:t-1}|y_{1:T})\,q_t(x_t|x_{t-1})} \\ &\propto \frac{g(y_t|x_t)f(x_t|x_{t-1})\hat{p}^{(k)}\,(y_{t+1:T}|x_t)}{\hat{p}^{(k)}\,(y_{t:T}|x_{t-1})\,q_t(x_t|x_{t-1})} \end{aligned} \tag{11}$$

and $w_1^{(k)}(x_0, x_1) \propto g(y_1|x_1)\mu(x_1)\hat{p}^{(k)}(y_{t+1:T}|x_1)/q_1(x_1)$. We would like to use the proposal minimizing the variance of the incremental weight, which at time $t \geq 2$ is $q_t^{\text{opt}}(x_t|x_{t-1}) \propto g(y_t|x_t)f(x_t|x_{t-1})\hat{p}^{(k)}\,(y_{t+1:T}|x_t)$ or an approximation of it.

The full replica cSMC update for $\bar{\pi}$ is described in Algorithm 2 and is simply an application of Algorithm 1 to a sequence of target densities for each replica. A proof of the validity of the algorithm is provided in the Supplementary Material.

Algorithm 2 Replica cSMC update

```
For $k = 1, \dots, K$
    1. Build an approximation $\hat{p}^{(k)}\,(y_{t+1:T}|x_t)$ of $p\,(y_{t+1:T}|x_t)$ using the replicas
       $(x_{t+1}^{(1)'}, \dots, x_{t+1}^{(k-1)'}, x_{t+1}^{(k+1)}, \dots, x_{t+1}^{(K)})$ for $t = 1, ..., T-1$.
    2. Run Algorithm 1 with target $\pi(x_{1:T}) = p(x_{1:T}|y_{1:T})$ and auxiliary targets
       $\pi_t(x_{1:t}) = \hat{p}^{(k)}\,(x_{1:t}|y_{1:T})$ for $t = 1, \dots, T-1$ with initial state $x_{1:T}^{(k)}$
       to return $x_{1:T}^{(k')}$.
Output $x_{1:T}^{(1:K)'}$.
```

One sensible way to initialize the replicas is to set them to sequences sampled from standard independent SMC passes. This will start the Markov chain not too far from equilibrium. For multimodal distributions, initialization is particularly crucial, since we need to ensure that different replicas are well-distributed amongst the various modes at the start of the run.

3.2. Setup and Tuning

The replica cSMC sampler requires an estimator $\hat{p}^{(k)}\left(y_{t+1:T}|x_t\right)$ of the backward information filter based on $x_{t+1}^{(-k)}$. For our algorithm, we propose an estimator $\hat{p}^{(k)}\left(y_{t+1:T}|x_t\right)$ that is not based on any analytical approximation of $p\left(y_{t+1:T}|x_t\right)$ but simply on a Monte Carlo approximation built using the other replicas,

$$\hat{p}^{(k)}\left(y_{t+1:T}|x_t\right) \propto \sum_{j\neq k} \frac{f(x_{t+1}^{(j)}|x_t)}{p(x_{t+1}^{(j)}|y_{1:t})}, \tag{12}$$

where $p\left(x_{t+1}|y_{1:t}\right)$ denotes the predictive density of x_{t+1}. The rationale for this approach is that at equilibrium the components of $x_{t+1}^{(-k)}$ are an iid sample from a product of $K-1$ copies of the smoothing density, $p\left(x_{t+1}|y_{1:T}\right)$. Therefore, as K increases, (12) converges to

$$\begin{aligned}
&\int \frac{f\left(x_{t+1}|x_t\right)}{p\left(x_{t+1}|y_{1:t}\right)} p\left(x_{t+1}|y_{1:T}\right) dx_{t+1} \\
&\propto \int f\left(x_{t+1}|x_t\right) p\left(y_{t+1:T}|x_{t+1}\right) dx_{t+1} \\
&= p\left(y_{t+1:T}|x_t\right).
\end{aligned} \tag{13}$$

In practice, the predictive density is also unknown and we need to use an approximation of it. Whatever being the approximation $\hat{p}\left(x_{t+1}|y_{1:t}\right)$ of $p\left(x_{t+1}|y_{1:t}\right)$ we use, the algorithm is valid. We note that for $K=2$, any approximation of the predictive density results in the same incremental importance weights.

We propose to approximate the predictive density in (13) by a constant over the entire latent space, i.e. $\hat{p}(x_{t+1}|y_{1:t}) = 1$. We justify this choice as follows. If we assume that we have informative observations, which is typical in many state space modelling scenarios, then $p(x_{t+1}|y_{1:T})$ will tend to be much more concentrated than $p(x_{t+1}|y_{1:t})$. Thus, over the region where the posterior has high density, the predictive density will be approximately constant relative to the posterior density. This suggests approximating the predictive density in (13) by its mean with respect to the posterior density,

$$\begin{aligned}
&\int \frac{f\left(x_{t+1}|x_t\right)}{p\left(x_{t+1}|y_{1:t}\right)} p\left(x_{t+1}|y_{1:T}\right) dx_{t+1} \\
&\approx \frac{\int f\left(x_{t+1}|x_t\right) p\left(x_{t+1}|y_{1:T}\right) dx_{t+1}}{\int p\left(x_{t+1}|y_{1:t}\right) p\left(x_{t+1}|y_{1:T}\right) dx_{t+1}} \\
&\approx \frac{\frac{1}{K}\sum_{k=1}^{K} f(x_{t+1}^{(k)}|x_t)}{\frac{1}{K}\sum_{k=1}^{K} p(x_{t+1}^{(k)}|y_{1:t})}.
\end{aligned} \tag{14}$$

Since the importance weights in cSMC at each time are defined up to a constant, sampling is not affected by the specific value of $\frac{1}{K}\sum_{k=1}^{K} p(x_{t+1}^{(k)}|y_{1:t})$. Therefore, when doing computation it can simply be set to any value, which is what we do.

We note that while the asymptotic argument doesn't hold for the estimator in (14), when the variance of the predictive density is greater than the variance of the posterior density, we expect the estimators in (12) and (14) to be close for any finite K.

An additional benefit to approximating the predictive density by a constant is reduction in the variance of the mixture weights in (12). To see why this can be the case, consider the following example. Suppose the predictive density of x_{t+1} is $\mathcal{N}(\mu, \sigma_0^2)$ and the posterior density is $\mathcal{N}(0, \sigma_1^2)$, where $\sigma_1^2 < \sigma_0^2$. Computing the variance of the mixture weight, we get

$$\begin{aligned}
&\mathrm{Var}\left(\frac{1}{p(x_{t+1}|y_{1:t})}\right) \\
&= \frac{2\pi\sigma_0^2}{\sqrt{2\sigma_1^2\nu_1}} \exp\left[\mu^2\left(\frac{1}{\sigma_0^2} + \frac{1}{(\sigma_0^2)^2\nu_1}\right)\right] \\
&- \frac{2\pi\sigma_0^2}{\sigma_1^2\nu_2} \exp\left[\mu^2\left(\frac{1}{\sigma_0^2} + \frac{1}{(\sigma_0^2)^2\nu_2}\right)\right].
\end{aligned} \tag{15}$$

where

$$\nu_1 = \left(\frac{1}{2\sigma_1^2} - \frac{1}{\sigma_0^2}\right) \qquad \nu_2 = \left(\frac{1}{\sigma_1^2} - \frac{1}{\sigma_0^2}\right). \tag{16}$$

From this we can see that variance increases exponentially with the squared difference of predictive and posterior means, μ^2. As a result, we can get outliers in the mixture weight distribution. If this happens, many of the replicas will end up having low weights in the mixture. This will reduce the effective number of replicas used. Using a constant approximation will weight all of the replicas uniformly, and allow us to construct better proposals, as illustrated in Section 4.1.

A natural extension of the proposed method is to update some of the replicas with other than replica cSMC updates. Samples from these replicas can then be used in estimates of the backward information filter when doing a replica cSMC update. This makes it possible to parallelize the method, at least to some extent. For instance, one possibility is to do parallel independent cSMC updates on some of the replicas.

Performing other than replica cSMC updates on some of the replicas can be useful in multimodal scenarios. If all replicas are located in an isolated mode, and the replica cSMC updates use an estimate of the backward information filter based on replicas in that mode, then the overall Markov chain will tend not to transition well to other modes. Using samples from other types of updates in the estimate of the backward information filter can help counteract this effect by making transitions to other high-density regions possible.

4. Examples

We consider four models to illustrate the performance of our method. In all examples, we assume that the model parameters are known. The first is a simple linear Gaussian model. We use this model to demonstrate that it is sensible to use a constant approximation to the predictive density in our estimator of the backward information filter. We also use the linear Gaussian model to better understand the accuracy and performance of replica cSMC. The second model, from (Shestopaloff & Neal, 2018), demonstrates that our proposed replica cSMC method is competitive with existing state-of-the-art methods at drawing latent state sequences in a unimodal context. The third model, also from (Shestopaloff & Neal, 2018), demonstrates that by updating some replica coordinates with a standard iterated cSMC kernel, our method is able to efficiently handle multimodal sampling without the use of specialized "flip" updates. The fourth model is the Lorenz-96 model from (Heng et al., 2017), which has very low observation noise, making it a challenging case for standard iterated cSMC.

To do our computations, we used MATLAB on an OS X system, running on an Intel Core i5 1.3 GHz CPU. As a performance metric for the sampler, we used autocorrelation time, which is a measure of approximately how many steps of an MCMC chain are required to obtain the equivalent of one independent sample. The autocorrelation time is estimated based on a set of runs as follows. First, we estimate the overall mean using all of the runs. Then, we use this overall mean to estimate autocovariances for each of the runs. The autocovariance estimates are then averaged and used to estimate the autocorrelations $\hat{\rho}_k$. The autocorrelation time is then estimated as $1 + 2\sum_{m=1}^{M}\hat{\rho}_m$ where M is chosen such that for $m > M$ the autocorrelations are approximately 0. The code to reproduce all the results is publicly available at `https://github.com/ayshestopaloff/replicacsmc`.

4.1. A Linear Gaussian Model

Let $X_t = (X_{1,t}, \dots, X_{d,t})'$ for $t = 1, \dots, T$. The latent process for this model is defined as $X_1 \sim \mathcal{N}(0, \Sigma_1)$, $X_t | \{X_{t-1} = x_{t-1}\} \sim \mathcal{N}(\Phi x_{t-1}, \Sigma)$ for $t = 2, \dots, T$, where

$$\Phi = \begin{pmatrix} \phi_1 & 0 & \cdots & 0 \\ 0 & \phi_2 & \ddots & \vdots \\ \vdots & \ddots & \phi_{d-1} & 0 \\ 0 & \cdots & 0 & \phi_d \end{pmatrix}, \quad \Sigma = \begin{pmatrix} 1 & \rho & \cdots & \rho \\ \rho & 1 & \ddots & \vdots \\ \vdots & \ddots & 1 & \rho \\ \rho & \cdots & \rho & 1 \end{pmatrix},$$

$$\Sigma_1 = \begin{pmatrix} \sigma_{1,1}^2 & \rho\sigma_{1,1}\sigma_{1,2} & \cdots & \rho\sigma_{1,1}\sigma_{1,d} \\ \rho\sigma_{1,2}\sigma_{1,1} & \sigma_{1,2}^2 & \ddots & \vdots \\ \vdots & \ddots & \sigma_{1,d-1}^2 & \rho\sigma_{1,d-1}\sigma_{1,d} \\ \rho\sigma_{1,d}\sigma_{1,1} & \cdots & \rho\sigma_{1,d}\sigma_{1,d-1} & \sigma_{1,d}^2 \end{pmatrix},$$

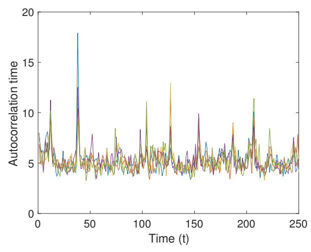
with $\sigma_{1,i}^2 = 1/(1-\phi_i^2)$ for $i = 1, \dots, d$. The observations are $Y_{i,t} | \{X_{i,t} = x_{i,t}\} \sim \mathcal{N}(x_{i,t}, 1)$ for $i = 1, \dots, d$ and $t = 1, \dots, T$. We set $T = 250, d = 5$ and the model's parameters to $\rho = 0.7$ and $\phi_i = 0.9$ for $i = 1, \dots, d$. We generate a sequence from this model to use for our experiments.

Since this is a linear Gaussian model, we are able to compute the predictive density in (12) exactly using a Kalman filter. So for replica k, we can use the following importance densities,

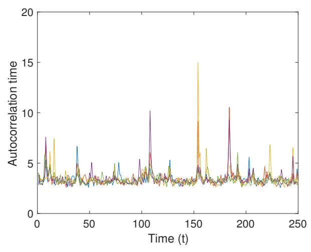
$$\begin{aligned} q_1(x_1) &\propto \mu(x_1) \sum_{j \neq k} \frac{f(x_2^{(j)}|x_1)}{p(x_2^{(j)}|y_1)}, \\ q_t(x_t|x_{t-1}) &\propto f(x_t|x_{t-1}) \sum_{j \neq k} \frac{f(x_{t+1}^{(j)}|x_t)}{p(x_{t+1}^{(j)}|y_{1:t})}, \\ q_T(x_T|x_{T-1}) &\propto f(x_T|x_{T-1}), \end{aligned} \tag{17}$$

where $t = 2, \dots, T-1$. Since these densities are Gaussian mixtures, they can be sampled from exactly. However, as pointed out in the previous section, this approach can be inefficient. We will show experimentally that using a constant approximation to the predictive density in (12) actually improves performance. In all experiments, we intialize all replicas to a sample from an independent SMC pass with the same number of particles as used for cSMC updates. Also, the different runs in our experiments use different random number generator seeds.

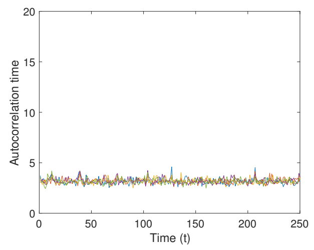
We first check that our replica method produces answers that agree with the posterior mean computed by a Kalman smoother. To do this, we do 10 replica cSMC runs with 100 particles and 2 replicas for 25,000 iterations, updating each replica conditional on the other. We then look at whether the posterior mean of $x_{i,t}$ computed using a Kalman smoother lies within two standard errors of the overall mean of 10



(a) Replica cSMC, 2 replicas.



(b) Replica cSMC, 75 replicas.



(c) Replica cSMC, 75 replicas, constant approximation to predictive.

Figure 1. Estimated autocorrelation times for each latent variable. Different coloured lines correspond to different latent state components. The x-axis corresponds to different times.

replica cSMC runs. We find this happens for about 91.4% of the $x_{i,t}$. This indicates strong agreement between the answers obtained by replica cSMC and the Kalman smoother.

Next, we investigate the effect of using more replicas. To do this, we compare replica cSMC using 2 versus 75 replicas. We do 5 runs of each sampler. Both samplers use 100 particles and we do a total of $5,000$ iterations per run. For the sampler using 75 replicas, we update replica 1 at every iteration and replicas 2 to 75 in sequence at every 20-th iteration. For the sampler using 2 replicas, we update both replicas at every iteration. In both samplers, we update replica 1 with replica cSMC and the remaining replica(s) with iterated cSMC. After discarding 10% of each run as burn-in, we use all runs for a sampler to compute autocorrelation time.

We can clearly see in Figures 1a and 1b that using more replicas improves performance, before adjusting for computation time. We note that for this simple example, there is no benefit from using replica cSMC with a large number of replicas if we take into account computation time.

To check the performance of using the constant approximation versus the exact predictive density, we run replica cSMC with 75 replicas and the same settings as earlier, except using a constant approximation to the predictive density. Figure 1c shows that using a constant approximation to the predictive density results in peformance better than when using the true predictive density. This is consistent with our discussion in Section 3.2.

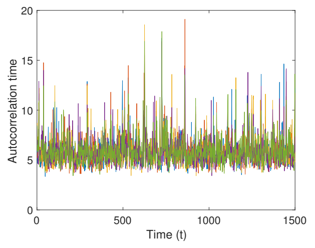


Figure 2. Estimated autocorrelation times for each latent variable. Different coloured lines correspond to different latent state components. The x-axis corresponds to different times.

The linear Gaussian model can also be used to demonstrate that due to looking ahead, a fixed level of precision can be achieved with much fewer particles with replica cSMC than with standard iterated cSMC. In scenarios where the state is high dimensional and the observations are informative, it is difficult to efficiently sample the variables $x_{i,1}$ with standard iterated cSMC using the initial density as the proposal. We do 20 runs of $2,500$ iterations of both iterated cSMC with 700 particles and of replica cSMC with 35 particles and 2 replicas, with each replica updated given the other. We then use the runs to estimate the standard error of the overall mean over 20 runs. For the variable $x_{1,1}$ sampled with iterated cSMC we estimate the standard error to be approximately 0.0111 whereas for replica cSMC the estimated standard error is a similar 0.0081, achieved using only 5% of the particles.

Finally, we verify that the proposed method works well on longer time series by running it on the linear Gaussian model but with the length of the observed sequence set to $T = 1,500$. We use 2 replicas, each updated given the other, and do 5 runs of $5,000$ iterations of the sampler to estimate the autocorrelation time for sampling the latent variables. In Figure 2 we can see that the replica cSMC method does not suffer from a decrease in performance when used on longer time series.

4.2. Two Poisson-Gaussian Models

In this example, we consider the two models from (Shestopaloff & Neal, 2018). Model 1 uses the same latent process as Section 4.1 with $T = 250$, $d = 10$ and $Y_{i,t}|\{X_{i,t} = x_{i,t}\} \sim \text{Poisson}(\exp(c + \sigma x_{i,t}))$ for $i = 1, \ldots, d$ and $t = 1, \ldots, T$ where $c = -0.4$ and $\sigma = 0.6$. For Model 2, we again use the latent process in Section 4.1, with $T = 500, d = 15$ and $Y_{i,t}|\{X_{i,t} = x_{i,t}\} \sim \text{Poisson}(\sigma|x_{i,t}|))$ for $i = 1, \ldots, d$ and $t = 1, \ldots, T$ where $\sigma = 0.8$. We assume the observations are independent given the latent states.

We generate one sequence of observations from each model. A plot of the simulated data along dimension $i = 1$ is shown

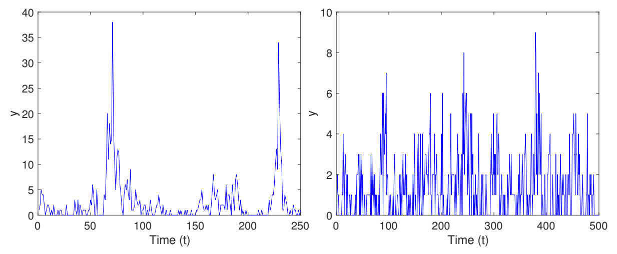

(a) Data for Model 1. (b) Data for Model 2.

Figure 3. Simulated data from the Poisson-Gaussian models.

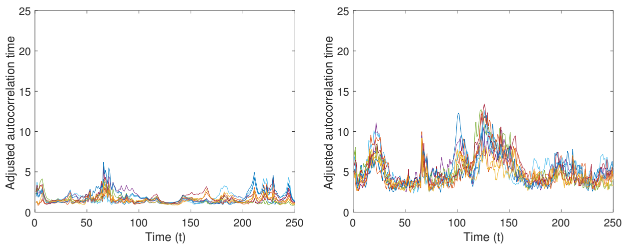

(a) Iterated cSMC+Metropolis. (b) Replica cSMC.

Figure 4. Model 1. Estimated autocorrelation times for each latent variable, adjusted for computation time. Different coloured lines corresponds to different latent state components. The x-axis corresponds to different times.

in Figure 3. We set the importance densities q_t for the replica cSMC sampler to the same ones as in Section 4.1, with a constant approximation to the predictive density.

Model 1

We use replica cSMC with 5 replicas, updating one replica conditional on the other. We start with both sequences initialized to $\mathbf{0}$. We set the number of particles to 200. We do a total of 5 runs of the sampler with $5,000$ iterations, each run with a different random number generator seed. Each iteration of replica cSMC takes approximately 0.80 seconds. We discard 10% of each run as burn-in.

Plots of autocorrelation time comparing replica cSMC to the best method in (Shestopaloff & Neal, 2018) for sampling each of the latent variables are shown in Figure 4. The benchmark method takes approximately 0.21 seconds per iteration. We can see that the proposed replica cSMC method performs relatively well when compared to their best method after adjusting for computation time. The figure for iterated cSMC+Metropolis was reproduced using code available with (Shestopaloff & Neal, 2018).

Model 2

For this model, the challenge is to move between the many different modes of the latent state due to conditioning on $|x_{i,t}|$ in the observation density. The marginal posterior of $x_{i,t}$ has two modes and is symmetric around 0. Additional modes appear due to uncertainty in the signs of state components.

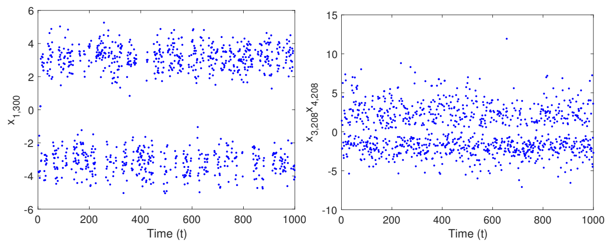

(a) Trace plot for $x_{1,300}$. (b) Trace plot for $x_{3,208}x_{4,208}$.

Figure 5. Trace plots for Model 2.

We use a total of 50 replicas and update 49 of the 50 replicas with iterated cSMC and one replica with replica cSMC. This is done to prevent the Markov chain from being stuck in a single mode while at the same time enabling the replica cSMC update to use an estimate of the backward information filter based on replicas that are distributed across the state space. We initialize all replicas using sequences drawn from independent SMC passes with $1,000$ particles, and run the sampler for a total of $2,000$ iterations. Both replica cSMC and iterated cSMC updates use 100 particles.

In Figure 5 we plot every other sample of the same functions of state as in (Shestopaloff & Neal, 2018) of the replica updated with replica cSMC. This is the the coordinate $x_{1,300}$ with true value -1.99 and $x_{3,208}x_{4,208}$ with true value -4.45. The first has two well-separated modes and the second is ambiguous with respect to sign. We see that the sampler is able to explore different modes, without requiring any specialized “flip” updates or having to use a much larger number of particles, as is the case in (Shestopaloff & Neal, 2018).

We note that the replicas doing iterated cSMC updates tend to get stuck in separate modes for long periods of time, as expected. However, as long as these replicas are well-distributed across the state space and eventually explore it, the bias in the estimate of the backward information filter will be low and vanish asymptotically. The samples from the replica cSMC update will consequently be a good approximation to samples from the target density. Further improvement of the estimate of the backward information filter based on replicas in multimodal scenarios remains an open problem.

4.3. Lorenz-96 Model

Finally, we look at the Lorenz-96 model in a low-noise regime from (Heng et al., 2017). The state function for this

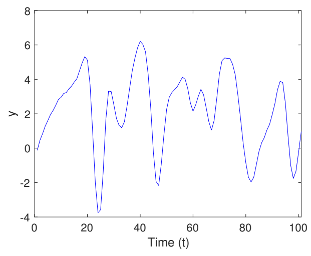


Figure 6. Simulated data from Lorenz-96 model along coordinate $i = 1$.

model is the Itô process $\xi(s) = (\xi_1(s), \ldots, \xi_d(s))$ defined as the weak solution of the stochastic differential equation (SDE)

$$\mathrm{d}\xi_i = (-\xi_{i-1}\xi_{i-2} + \xi_{i-1}\xi_{i+1} - \xi_i + \alpha)\mathrm{d}t + \sigma_f \mathrm{d}B_i \quad (18)$$

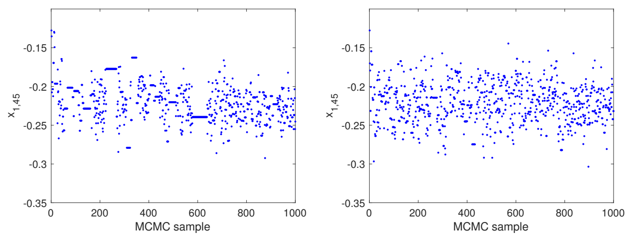
for $i = 1, \ldots, d$, where indices are computed modulo d, α is a forcing parameter, σ_f^2 is a noise parameter and $B(s) = (B_1(s), \ldots, B_d(s))$ is d-dimensional standard Brownian motion. The initial condition for the SDE is $\xi(0) = \mathcal{N}(\mathbf{0}, \sigma_f^2 \mathcal{I}_d)$. We observe the process on a regular grid of size $h > 0$ as $Y_t \sim \mathcal{N}(H\xi(th), R)$, where $t = 0, \ldots, T$. We will assume that the process is only partially observed, with $H_{ii} = 1$ for $i = 1, \ldots, p$ and 0 otherwise, for $p = d - 2$.

We discretize the SDE (18) by numerically integrating the drift using a fourth-order Runge-Kutta scheme and adding Brownian increments. Let u be the mapping obtained by numerically integrating the drift of (18) on $[0, h]$. This discretization produces a state space model with $X_1 \sim \mathcal{N}(\mathbf{0}, \sigma_f^2 \mathcal{I})$, $X_t | \{X_{t-1} = x_{t-1}\} \sim \mathcal{N}(u(x_{t-1}), \sigma_f^2 h \mathcal{I})$ for $t = 2, \ldots, T+1$ and $Y_t | \{X_t = x_t\} \sim \mathcal{N}(Hx_t, R)$ for $t = 1, \ldots, T+1$. We set $d = 16$, $\sigma_f^2 = 10^{-2}$, $R = 10^{-3}\mathcal{I}_p$ and $\alpha = 4.8801$. The process is observed for 10 time units, which corresponds to $h = 0.1$, $T = 100$, and a step size of 10^{-2} for the Runge-Kutta scheme. A plot of data generated from the Lorenz-96 model along one of the coordinates is shown in Figure 6.

We compare the performance of replica cSMC with two replicas, updating each replica conditional on the other, to an iterated cSMC scheme. For iterated cSMC, we use the model's initial density as q_1 and the model's transition density as q_t for $t \geq 2$. For replica cSMC, we use the following importance densities for replica k,

$$\begin{aligned} q_1(x_1) &\propto f(x_1) \sum_{j \neq k} \phi(x_1 | x_2^{(j)}), \\ q_t(x_t | x_{t-1}) &\propto f(x_t | x_{t-1}) \sum_{j \neq k} \phi(x_t | x_{t+1}^{(j)}), \\ q_T(x_T | x_{T-1}) &\propto f(x_T | x_{T-1}), \end{aligned} \quad (19)$$

where $t = 2, \ldots, T-1$ and ϕ is the p-dimensional Gaussian density with mean $Hu^{-1}(x_{t+1}^{(j)})$ and variance $\sigma_f^2 h \mathcal{I}_p$, that is, the mean is computed by running the Runge-Kutta scheme backward in time starting at the replica state $x_{t+1}^{(j)}$. We initialize the iterated cSMC sampler and each replica in the replica cSMC sampler with a sequence drawn from an independent SMC pass with $3,000$ particles. We run replica cSMC with 200 particles for $30,000$ iterations (0.7 seconds per iteration) and compare to standard iterated cSMC with 600 particles, which we also run for $30,000$ iterations (0.7 seconds per iteration), thus making the computational time equal.



(a) Standard cSMC trace, $x_{1,45}$. (b) Replica cSMC trace, $x_{1,45}$.

Figure 7. Lorenz-96 model. Comparison of standard cSMC and replica cSMC.

Figure 7 shows the difference in performance of the two samplers by trace plots of $x_{1,45}$ (true value -0.23), from one of the runs, plotting the samples every 30th iteration. We can see that replica cSMC performs noticeably better when compared to standard iterated cSMC.

5. Conclusion

We presented a novel sampler for latent sequences of a non-linear state space model. Our proposed method leads to several questions. The first is whether there are other ways to estimate the predictive density that does not result in mixture weights with high variance. Another question is to develop better guidelines on choosing the number of replicas to use in a given scenario. It would also be interesting to look at applications of replica cSMC in non time-series examples. Finally, while the proposed method offers an approach for sampling in models with multimodal state distributions, further improvement is needed.

References

Andrieu, C., Doucet, A., and Holenstein, R. Particle Markov chain Monte Carlo (with discussion). *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72 (4):269–342, 2010.

Briers, M., Doucet, A., and Maskell, S. Smoothing algorithms for state-space models. *Annals of the Institute of Statistical Mathematics*, 62(1):61–89, 2010.

Chen, R., Ming, L., and Liu, J. S. Lookahead strategies for sequential Monte Carlo. *Statistical Science*, 28(1):69–94, 2013.

Finke, A., Doucet, A., and Johansen, A. On embedded hidden Markov models and particle Markov chain Monte Carlo methods. Technical report, arXiv:1610.08962, 2016.

Grothe, O., Kleppe, T., and Liesenfeld, R. Bayesian analysis in non-linear non-Gaussian state-space models using particle Gibbs. Technical report, arXiv:1601.01125, 2016.

Guarniero, P., Johansen, A. M., and Lee, A. The iterated auxiliary particle filter. *Journal of the American Statistical Association*, 112(520):1636–1647, 2017.

Heng, J., Bishop, A., Deligiannidis, G., and Doucet, A. Controlled sequential Monte Carlo. Technical report, arXiv:1708.08396, 2017.

Leimkuhler, B., Matthews, C., and Weare, J. Ensemble preconditioning for Markov chain Monte Carlo simulation. *Statistics and Computing*, 28:277–290, 2018.

Neal, R. MCMC using ensembles of states for problems with fast and slow variables such as Gaussian process regression. Technical report, arXiv:1101.0387, 2010.

Ruiz, H. C. and Kappen, H. J. Particle smoothing for hidden diffusion processes: adaptive path integral smoother. *IEEE Transactions on Signal Processing*, 65(12):3191–3203, 2017.

Scharth, M. and Kohn, R. Particle efficient importance sampling. *J. Econometrics*, 190(1):133–147, 2016.

Shestopaloff, A. and Neal, R. Sampling latent states for high-dimensional non-linear state space models with the embedded HMM method. *Bayesian Analysis*, 13(3):797–822, 2018.

Whiteley, N. Discussion of particle Markov chain Monte Carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 72(4):306–307, 2010.

Supplementary Material for Replica Conditional Sequential Monte Carlo

Alexander Y. Shestopaloff[1 2] **Arnaud Doucet**[3 2]

1. Validity of Replica cSMC

It is easy to see that the proposed update leaves $\bar{\pi}$ invariant. Let $M_{x_{1:T}^{(-k)}}(x_{1:T}^{(k)'}|x_{1:T}^{(k)})$ be the cSMC transition kernel used to update replica $x_{1:T}^{(k)}$, $k = 1, \dots, K$, where $x_{1:T}^{(-k)} := (x_{1:T}^{(1)'}, \dots, x_{1:T}^{(k-1)'}, x_{1:T}^{(k+1)}, \dots, x_{1:T}^{(K)})$. The replica update is a composition of the $M_{x_{1:T}^{(-k)}}$ so we can write the replica cSMC transition kernel M as a product, $M(x_{1:T}^{(1:K)'}|x_{1:T}^{(1:K)}) = \prod_{k=1}^{K} M_{x_{1:T}^{(-k)}}(x_{1:T}^{(k)'}|x_{1:T}^{(k)})$.

The replica cSMC transition kernel M then leaves $\bar{\pi}$ invariant since we have

$$\int \bar{\pi}(x_{1:T}^{(1:K)}) M(x_{1:T}^{(1:K)'}|x_{1:T}^{(1:K)}) dx_{1:T}^{(1:K)}$$

$$= \int \prod_{k=1}^{K} p(x_{1:T}^{(k)}|y_{1:T}) M_{x_{1:T}^{(-k)}}(x_{1:T}^{(k)'}|x_{1:T}^{(k)}) dx_{1:T}^{(1:K)}$$

$$= \int \left[\int p(x_{1:T}^{(1)}|y_{1:T}) M_{x_{1:T}^{(-1)}}(x_{1:T}^{(1)'}|x_{1:T}^{(1)}) dx_{1:T}^{(1)} \right]$$

$$\times \prod_{k=2}^{K} p(x_{1:T}^{(k)}|y_{1:T}) M_{x_{1:T}^{(-k)}}(x_{1:T}^{(k)'}|x_{1:T}^{(k)}) dx_{1:T}^{(2:K)}$$

$$= p(x_{1:T}^{(1)'}|y_{1:T}) \int \left[\int p(x_{1:T}^{(2)}|y_{1:T}) M_{x_{1:T}^{(-2)}}(x_{1:T}^{(2)'}|x_{1:T}^{(2)}) dx_{1:T}^{(2)} \right]$$

$$\times \prod_{k=3}^{K} p(x_{1:T}^{(k)}|y_{1:T}) M_{x_{1:T}^{(-k)}}(x_{1:T}^{(k)'}|x_{1:T}^{(k)}) dx_{1:T}^{(3:K)}$$

$$= p(x_{1:T}^{(1)'}|y_{1:T}) p(x_{1:T}^{(2)'}|y_{1:T})$$

$$\times \int \prod_{k=3}^{K} p(x_{1:T}^{(k)}|y_{1:T}) M_{x_{1:T}^{(-k)}}(x_{1:T}^{(k)'}|x_{1:T}^{(k)}) dx_{1:T}^{(3:K)}$$

$$= \prod_{k=1}^{K} p(x_{1:T}^{(k)'}|y_{1:T}) \quad \text{(by induction)}$$

$$= \bar{\pi}(x_{1:T}^{(1:K)'}).$$

[1]School of Mathematics, University of Edinburgh, Edinburgh, UK [2]The Alan Turing Institute, London, UK [3]Department of Statistics, University of Oxford, Oxford, UK. Correspondence to: Alexander Y. Shestopaloff <ashestopaloff@turing.ac.uk>.

Proceedings of the 36th International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019. Copyright 2019 by the author(s).