Rejection-sampling-based ancestor sampling for particle Gibbs

Roland Hostettler and Simo Särkkä

DOI:

10.1109/MLSP.2019.8918852

2019 IEEE INTERNATIONAL WORKSHOP ON MACHINE LEARNING FOR SIGNAL PROCESSING, OCT. 13–16, 2019, PITTSBURGH, PA, USA

REJECTION-SAMPLING-BASED ANCESTOR SAMPLING FOR PARTICLE GIBBS

Roland Hostettler[⋆] *and Simo Särkkä*[†]

[⋆] Department of Engineering Sciences, Uppsala University, Sweden
E-mail: roland.hostettler@angstrom.uu.se
[†]Department of Electrical Engineering and Automation, Aalto University, Finland
E-mail: simo.sarkka@aalto.fi

ABSTRACT

Particle Gibbs with ancestor sampling is an efficient and statistically principled algorithm for learning of dynamic systems. However, the ancestor sampling step used to improve mixing of the Markov chain requires the possibly expensive calculation of a set of ancestor weights for the complete particle system. In this paper, we propose a rejection-sampling-based algorithm for ancestor sampling in particle Gibbs that mitigates this problem. Additionally, performance guarantees and a fallback strategy to prevent suffering from high rejection rates are discussed. The performance of the method is illustrated in two numerical examples.

***Index Terms*—** Particle Markov chain Monte Carlo, particle filters, statistical learning

1. INTRODUCTION

Particle Markov chain Monte Carlo (PMCMC) methods are important tools for (Bayesian) learning of nonlinear, non-Gaussian dynamic systems [1]. One particular class of PMCMC methods are particle Gibbs methods in general and particle Gibbs with ancestor sampling (PGAS) in particular [1, 2]. These have proven to be useful in several applications such as learning of Gaussian process state-space models [3, 4] or learning of latent spatio-temporal Gaussian processes [5], with applications including finance [6] and target tracking [7].

PGAS is based on Gibbs sampling and alternates between sampling the state trajectory and the static parameters of the dynamic system, conditioned on the other parameters and the data. To sample the state trajectory, PGAS uses a Markov kernel based on a particle filter that also incorporates the last trajectory sample as a reference trajectory. To improve mixing of the kernel and speed up convergence of the Markov chain, the reference trajectory is mixed into the new state trajectories using ancestor sampling [8, 9], which yields the conditional particle filter with ancestor sampling (CPF-AS), see [2]. This approach is similar to the auxiliary particle filter [10] or backward simulation smoothing [11] and requires calculating the probability of two partial state trajectories being connected, referred to as ancestor weights, a step that is not required in regular particle filtering. Calculating the ancestor weights can incur significant computational cost, since the ancestor weights have to be calculated for all the particles of the conditional particle filter. For Markovian models, this approximately doubles the computations per time step and for non-Markovian models, where complete past and/or future state trajectories have to be evaluated (e.g., in Gaussian process state-space models, see [3]), the cost can be much higher.

Financial support by the Academy of Finland is gratefully acknowledged.

In this paper, we develop an ancestor sampling step that targets this bottleneck and improves computational scaling of the ancestor sampling step. In particular, we propose an ancestor sampling step based on rejection sampling (RS) rather than sampling from the categorical distribution of the weights directly. Furthermore, we provide a lower bound on the RS acceptance probability and develop a fallback strategy to avoid high rejection rates. In the best case, the proposed method reduces the computational complexity to sampling a single ancestor index and in the worst case, the method provides exact guarantees about the additional computational cost. The proposed method is evaluated in two numerical examples.

The remainder of this paper is organized as follows. Section 2 introduces the problem formally. The proposed method is developed in Section 3 and evaluated in Section 4. Finally, concluding remarks follow in Section 5.

2. PROBLEM FORMULATION

In this paper, we consider Markovian state-space models, that is, models of the form

$$x_0 \sim p(x_0 \mid \theta), \tag{1a}$$
$$x_n \sim p(x_n \mid x_{n-1}, \theta), \tag{1b}$$
$$y_n \sim p(y_n \mid x_n, \theta), \tag{1c}$$

where $x_n \in \mathbb{R}^{d_x}$ and $y_n \in \mathbb{R}^{d_y}$ denote the state and measurement vectors at time step n, respectively, and $\theta \in \mathbb{R}^{d_\theta}$ denotes a vector of static parameters. Furthermore, $p(x_0 \mid \theta)$ denotes the probability density function (pdf) of the initial state, $p(x_n \mid x_{n-1}, \theta)$ is the pdf of the dynamic model, and $p(y_n \mid x_n, \theta)$ denotes the likelihood. Finally, we also assume that the dynamic model is bounded, that is, that

$$p(x_n \mid x_{n-1}) \leq \kappa_n \tag{2}$$

for some positive, possibly time-varying constant κ_n. Note that PGAS is not restricted to models of the form (1)–(2) but can be applied to arbitrary inference problems where a sequence of states $x_{0:N} = \{x_0, x_1, \ldots, x_N\}$ is of interest. However, for simplicity of the theoretical results, we restrict ourselves to this kind of models.

Given the data $y_{1:N} = \{y_1, y_2, \ldots, y_N\}$, PGAS can be used to find a sample approximation of the joint posterior pdf of the parameters θ and state trajectory $x_{0:N}$ given by

$$p(x_{0:N}, \theta \mid y_{1:N}) \approx \sum_{k=1}^{K} \delta(\theta - \theta^k)\delta(x_{0:N} - x_{0:N}^k),$$

978-1-7281-0824-7/19/$31.00

where $\delta(\cdot)$ denotes the Kronecker delta function and the superscript k denotes the kth sample. This is achieved by iteratively sampling state trajectories according to $x_{0:N}^k \sim p(x_{0:N} \mid \theta^{k-1}, y_{1:N})$ and parameters according to $\theta^k \sim p(\theta \mid x_{0:N}^k, y_{1:N})$.

Sampling new trajectories is achieved by using CPF-AS, a particle filter that sequentially estimates a particle approximation of the sequence $x_{0:N}$

$$p(x_{0:N} \mid y_{1:N}) \approx \sum_{j=1}^{J} w_N^j \delta(x_{0:N} - x_{0:N}^j) \tag{3}$$

where w_N^j denotes the trajectory weight (note that for brevity, conditioning on the parameters θ is omitted from here on). To achieve this, $J-1$ trajectories are sampled using sequential importance sampling with resampling as follows [12]. Given an approximation of the form (3) at step $n-1$, the $j \in \{1, \ldots, J-1\}$ trajectories are resampled and updated by first sampling an auxiliary index

$$a_n^j \sim q(a_n \mid x_{0:n-1}, y_{1:n}),$$

according to a proposal distribution $q(a_n \mid x_{0:n-1}, y_{1:n})$, followed by sampling the new state

$$x_n^j \sim q(x_n \mid x_{0:n-1}^{a_n^j}, y_{1:n})$$

from the importance distribution $q(x_n \mid x_{0:n-1}^{a_n^j}, y_{1:n})$. Then, the Jth trajectory is updated by using the state $\tilde{x}_n$ of a reference trajectory, which is the state trajectory sampled in the previous iteration of the Gibbs sampler, that is, we set $x_n^J = \tilde{x}_n$.

In order to improve mixing of the Markov chain over trajectories, the sample x_n^J is then connected to the trajectories up to time $n-1$. This is achieved by calculating the probabilities $\{\tilde{w}_{n-1}^j\}_{j=1}^J$ of the reference trajectory sample $\tilde{x}_n$ being a continuation of each of the $j \in \{1, \ldots, J\}$ trajectories $x_{0:n-1}^j$ [2]. Then, the ancestor index a_n^J is sampled according to the probabilities $\{\tilde{w}_{n-1}^j\}_{j=1}^J$ and all the trajectories are updated according to

$$x_{0:n}^j = \{x_{0:n-1}^{a_n^j}, x_n^j\}.$$

The ancestor weights used in this step are given by [2]

$$\tilde{w}_{n-1}^j = \frac{\tilde{v}_{n-1}^j}{\tilde{\eta}_{n-1}},$$

where

$$\begin{aligned} \tilde{v}_{n-1}^j &= w_{n-1}^j \frac{p(y_{1:N}, x_{1:n-1}^j, \tilde{x}_{n:N})}{p(y_{1:n-1}, x_{1:n-1}^j)} \\ &= w_{n-1}^j p(y_{n:N} \mid x_{1:n-1}^j, \tilde{x}_{n:N}) p(\tilde{x}_{n:N} \mid x_{1:n-1}^j) \end{aligned} \tag{4}$$

are the non-normalized ancestor weights, and the normalization constant is given by

$$\tilde{\eta}_{n-1} = \sum_{j=1}^{J} \tilde{v}_{n-1}^j. \tag{5}$$

For Markovian state-space models of the form (1), the non-normalized ancestor weights simplify to [2]

$$\tilde{v}_{n-1}^j = w_{n-1}^j p(\tilde{x}_n \mid x_{n-1}^j). \tag{6}$$

Algorithm 1 CPF-AS

Input: Reference trajectory $\tilde{x}_{0:N}$
Output: Trajectory sample $x_{0:N}^k$

1: Sample $x_0^j \sim p(x_0)$ for $j \in \{1, \ldots, J-1\}$ and set $x_0^J \leftarrow \tilde{x}_0$
2: Set $w_0^j = 1/J$ for $j \in \{1, \ldots, J\}$
3: **for** $n = 1, \ldots, N$ **do**
4: Sample $a_n^j \sim q(a_n \mid x_{0:n-1}, y_{1:n})$ for $j \in \{1, \ldots, J-1\}$
5: Sample $x_n^j \sim q(x_n \mid x_{0:n-1}^{a_n^j}, y_{1:n})$ for $j \in \{1, \ldots, J-1\}$
6: Set $x_n^J \leftarrow \tilde{x}_n$
7: Sample $a_n^J \sim \mathcal{C}(\{\tilde{w}_{n-1}^{a_n^J}\}_{a_n^J=1}^J)$
8: Set $x_{0:n}^j \leftarrow \{x_{0:n-1}^{a_n^j}, x_n^j\}$ for $j \in \{1, \ldots, J\}$
9: Calculate and normalize the particle weights using (7)
10: **end for**
11: Sample $j \sim \mathcal{C}(\{w_N^j\}_{j=1}^J)$ and set $x_{0:N}^k \leftarrow x_{0:N}^j$

Finally, for all J samples, the non-normalized trajectory weights v_n^j are updated. These are given by [10, 12]

$$v_n^j = w_{n-1}^{a_n^j} \frac{p(y_n \mid x_n^j) p(x_n^j \mid x_{n-1}^{a_n^j})}{q(a_n^j \mid x_{0:n-1}, y_{1:n}) q(x_n^j \mid x_{0:n-1}^{a_n^j}, y_{1:n})} \tag{7}$$

where w_{n-1}^j is the sample weight at $n-1$. From the non-normalized trajectory weights, the updated sample weights then follow to be

$$w_n^j = \frac{v_n^j}{\sum_{i=1}^J v_n^i}.$$

The resulting algorithm is summarized in Algorithm 1 (where $\mathcal{C}(\{w^j\}_{j=1}^J)$ denotes the categorical distribution with weights $\{w^j\}_{j=1}^J$) and it can be shown that this yields an invariant Markov kernel over the state trajectories $x_{0:N}$ [2].

As discussed, sampling of the ancestor index in step 7 connects the reference trajectory $\tilde{x}_{0:N}$ to the other trajectories $x_{0:n-1}$ and greatly improves mixing of the Markov kernel, see [2] for details. Sampling of the ancestor indices is typically achieved by calculating all the ancestor weights $\tilde{w}_{n-1}^j$ for $j \in \{1, \ldots, J\}$ and then sampling a_n^J from the categorical distribution defined by the weights $\tilde{w}_{n-1}^j$. For large numbers of particles J or cases where the ancestor weights are expensive to calculate (e.g., non-Markovian models, see, e.g., [3, 5]), this may incur significant computational cost. In this paper, we develop an algorithm that avoids calculating all the ancestor weights beforehand and thus lowers the computational cost of this step.

3. METHOD

The main idea of the proposed method is based on the fact that only one ancestor index is sampled from the distribution defined by the ancestor weights. Hence, a strategy based on RS that proposes one sample at a time without calculating all the ancestor weights beforehand can be used, similar to the approach used to speed up backward simulation smoothing [13, 14].

In this section, we develop the corresponding ancestor index sampling step. A particular challenge in RS is that the bounding constant required for calculating the acceptance probability is unknown. To circumvent this problem, we derive an explicit lower bound on the acceptance probability (or, equivalently, an upper bound on the RS

bounding constant), making the method practically applicable. We also propose a fallback algorithm for the case when RS suffers from high rejection rates.

3.1. RS-based ancestor sampling

As discussed, the ancestor index a_n^J is typically sampled from the categorical distribution defined by the ancestor weights, that is, from the distribution (step 7 in Algorithm 1)

$$\begin{aligned} p(a_n^J \mid y_{1:N}, x_{1:n-1}, \tilde{x}_{n:N}) &= \mathcal{C}(\{\tilde{w}_{n-1}^{a_n^J}\}_{a_n^J=1}^J) \\ &= \frac{1}{\tilde{\eta}_{n-1}} \sum_{j=1}^J \tilde{v}_{n-1}^j \delta(a_n^J - j). \end{aligned} \tag{8}$$

However, rather than sampling the ancestor index directly from the categorical distribution, RS can be used to sample the ancestor index from a different distribution followed by an accept-reject step to ensure that the sample is from the target distribution (8).

In this case, we sample $a_n^J \sim \tilde{q}(a_n^J)$ from a proposal distribution $\tilde{q}(a_n^J)$ and accept the sample with probability

$$\begin{aligned} \gamma_n^{a_n^J} &= \frac{p(a_n^J \mid y_{1:N}, x_{1:n-1}, \tilde{x}_{n:N})}{\rho_n \tilde{q}(a_n^J)} \\ &= \frac{\frac{1}{\tilde{\eta}_{n-1}} \tilde{v}_{n-1}^{a_n^J}}{\rho_n \tilde{q}(a_n^J)}. \end{aligned}$$

For the constant ρ_n it must hold that [15]

$$p(a_n^J \mid y_{1:N}, x_{1:n-1}, \tilde{x}_{n:N}) \leq \rho_n \tilde{q}(a_n^J) \quad \forall\, a_n^J \tag{9}$$

to ensure that the proposal distribution encloses the target distribution. However, to maximize the acceptance rate, the bound should also be as tight as possible. Since the normalization constant $\tilde{\eta}_{n-1}$ is unknown (see (4)–(5)) without knowing all the weights, the challenge is to select the constant ρ_n. Thus, Lemma 1 provides an upper bound on the bounding constant, which in turn yields a lower bound on the acceptance probability, for RS ancestor index sampling.

Lemma 1 (Upper bound on bounding constant for ancestor sampling)**.** *Given the non-normalized ancestor weights $\{\tilde{v}_{n-1}^j\}_{j=1}^J$, the acceptance probability for the ancestor index sample a_n^J drawn from the proposal distribution $\tilde{q}(q_n^J)$ is bounded from below by*

$$\gamma_n^{a_n^J} \geq \frac{\tilde{v}_{n-1}^{a_n^J} \min \tilde{q}(a_n^J)}{\tilde{q}(a_n^J) \max_j \{\tilde{v}_{n-1}^j\}_{j=1}^J}.$$

Proof. A conservative choice for ρ_n that fullfils (9) is given by

$$\rho_n = \frac{\max p(a_n^J \mid y_{1:N}, x_{1:n-1}, \tilde{x}_{n:N})}{\min \tilde{q}(a_n^J)}.$$

Furthermore, from (8) it readily follows that

$$\max p(a_n^J \mid y_{1:N}, x_{1:n-1}, \tilde{x}_{n:N}) = \frac{1}{\tilde{\eta}_{n-1}} \max_j \{\tilde{v}_{n-1}^j\}_{j=1}^J,$$

and thus, the upper limit of ρ_n is given by

$$\rho_n = \frac{\max_j \{\tilde{v}_{n-1}^j\}_{j=1}^J}{\tilde{\eta}_{n-1} \min \tilde{q}(a_n^J)}.$$

Then, the lower bound for the acceptance probability becomes

$$\begin{aligned} \gamma_n^{a_n^J} &\geq \frac{\frac{1}{\tilde{\eta}_{n-1}} \tilde{v}_{n-1}^{a_n^J}}{\tilde{q}(a_n^J) \frac{\max_j \{\tilde{v}_{n-1}^j\}_{j=1}^J}{\tilde{\eta}_{n-1} \min \tilde{q}(a_n^J)}} \\ &= \frac{\tilde{v}_{n-1}^{a_n^J} \min \tilde{q}(a_n^J)}{\tilde{q}(a_n^J) \max_j \{\tilde{v}_{n-1}^j\}_{j=1}^J}, \end{aligned}$$

which concludes the proof. □

What remains is to choose the proposal distribution $\tilde{q}(a_n^J)$. To this end, we propose to use a discrete uniform distribution

$$\tilde{q}(a_n^J) = \mathcal{U}\{1, J\},$$

which can readily be sampled from and promotes mixing between the trajectories. Using this proposal distribution, the acceptance probability bound simplifies to

$$\gamma_n^{a_n^J} \geq \frac{\tilde{v}_{n-1}^{a_n^J}}{\max_j \{\tilde{v}_{n-1}^j\}_{j=1}^J}. \tag{10}$$

Note that the lower bound of the acceptance probability (10) is independent of the normalization constant $\tilde{\eta}_{n-1}$ but still requires knowledge of all the weights $\{\tilde{v}_{n-1}^j\}_{j=1}^J$ in the denominator. Hence, we next derive an upper bound on the non-normalized ancestor weights for Markovian state-space models of the form (1).

3.2. Bound for Markovian state-space models

For Markovian state-space models, the non-normalized ancestor weights are given by (6). Furthermore, using (2), an upper bound for the maximum of the non-normalized weights thus follows to be

$$\begin{aligned} \max_j \{\tilde{v}_{n-1}^j\}_{j=1}^J &= \max_j \{w_{n-1}^j p(\tilde{x}_n \mid x_{n-1}^j)\}_{j=1}^J \\ &\leq \kappa_n \max_j \{w_{n-1}^j\}_{j=1}^J. \end{aligned}$$

Conversely, the lower bound of the acceptance probability (with uniform proposal as discussed in Section 3.1) becomes

$$\gamma_n^{a_n^J} \geq \frac{\tilde{v}_{n-1}^{a_n^J}}{\kappa_n \max_j \{w_{n-1}^j\}_{j=1}^J}. \tag{11}$$

The bound (11) is not only independent of the normalization constant $\tilde{\eta}_{n-1}$ but also independent of any (non-normalized) ancestor weights other than the one of the sampled ancestor index a_n^J. Thus, the lower bound of the acceptance probability can be calculated solely based on the currently sampled (proposed) ancestor index, which reduces the required computations.

Finally, note that the dynamic model in many state-space models is an (exactly or approximately) discretized version of an underlying stochastic differential equation [16, 17]. In this case, the transition pdf is often a Gaussian pdf of the form

$$p(x_n \mid x_{n-1}) = \mathcal{N}(x_n; f(x_{n-1}), Q_n)$$

with mean $f(x_{n-1})$ and covariance matrix Q_n. In this case, it readily follows that the transition pdf is bounded by

$$\kappa_n = (2\pi)^{-d_x/2} |Q_n|^{-1/2}.$$

Algorithm 2 RS ancestor sampling with early stopping

Input: Trajectories and their weights $\{x^j_{1:n-1}, w^j_{n-1}\}_{j=1}^J$, bounding constant κ_n, reference trajectory sample $\tilde{x}_n$
Output: Ancestor index sample a_n^J

Set $I = \{1, \ldots, J\}$ and $l = 0$
do
  Sample $a_n^\star \sim \mathcal{U}\{1, J\}$
  Calculate $\tilde{v}_{n-1}^{a_n^\star}$ using (6)
  Calculate $\gamma_n^{a_n^\star}$ using (11)
  Sample $u \sim \mathcal{U}(0, 1)$
  if $u \leq \gamma_n^{a_n^\star}$ **then**
    Accept sample and set $a_n^J \leftarrow a_n^\star$
  end if
  Set $l \leftarrow l + 1$ and $I \leftarrow I \setminus \{a_n^\star\}$
while Sample not accepted and $l < L$
if Sample not accepted **then**
  Calculate ancestor weights $\tilde{v}_{n-1}^j$ for $j \in I$
  Normalize ancestor weights $\tilde{w}_{n-1}^j = \frac{\tilde{v}_{n-1}^j}{\sum_{i=1}^J \tilde{v}_{n-1}^i}$
  Sample $a_n^J \sim \mathcal{C}(\{\tilde{w}_{n-1}^j\}_{a_n^J=1}^J)$
end if

3.3. Early stopping and fallback

An important aspect of the proposed method is that the lower bound of the acceptance probability might be too lose (i.e., too far from the actual acceptance probability), which in turn leads to a low acceptance rate. In the worst case, this might lead to a significantly higher computational cost than the original strategy of computing all the ancestor weights and sampling from the resulting categorical distribution. To address this, we propose an early stopping and fallback strategy similar to [14].

In particular, we allow at most L consecutive RS trials. If none of the L proposed samples is accepted, we fall back to exhaustive sampling from the categorical distribution. To make the fallback strategy as efficient as possible, the non-normalized weights of the ancestor indices proposed during RS should be stored such that only the ancestor weights for the remaining indices need to be calculated. This guarantees that at best, the ancestor sampling step scales according to $\mathcal{O}(1)$ and at worst according to $\mathcal{O}(J + L - 1)$ (whereas sampling from the categorical distribution always scales according to $\mathcal{O}(J)$). This yields the ancestor sampling algorithm in Algorithm 2.

4. NUMERICAL ILLUSTRATIONS

In this section, we evaluate the proposed method in a simulation and real data example.

4.1. Univariate nonlinear growth model

In the first example, we consider the univariate nonlinear growth model (UNGM) commonly used for benchmarking sequential Monte Carlo methods given by [18]

$$
\begin{aligned}
p(x_0) &= \mathcal{N}(x_0; 0, 5), \\
p(x_n \mid x_{n-1}) &= \mathcal{N}\left(x_n; \frac{x}{2} + \frac{25x}{1+x^2} + 8\cos(1.2n), 10\right), \\
p(y_n \mid x_n) &= \mathcal{N}\left(y_n; \frac{x^2}{20}, 1\right).
\end{aligned}
$$

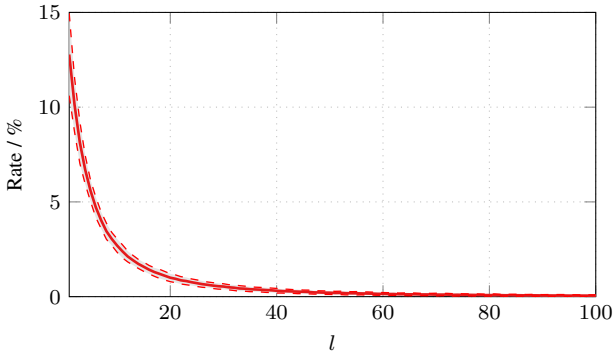

Fig. 1. Mean percentage together with 2σ-bounds of ancestor indices sampled using RS as a function of the iteration number l ($L = 100$).

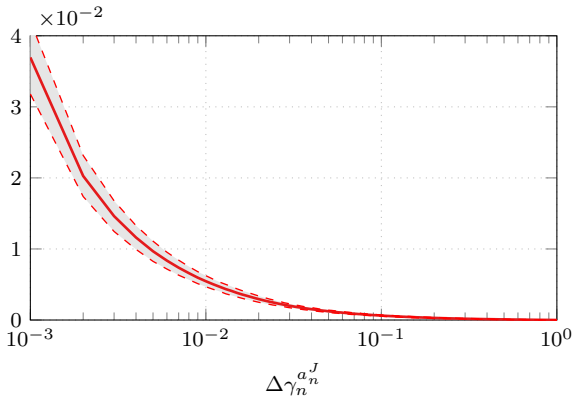

Fig. 2. Empirical distribution of the difference $\Delta\gamma_n^{a_n^J}$ between the true acceptance probability and the lower bound (10) together with its 2σ-bounds.

Since the main interest is in analyzing the behavior of ancestor sampling, all the parameters are assumed known in this case and focus lies on sampling of the state trajectories.

The parameters of PGAS are chosen as follows. In total, 150 trajectories are sampled of which 50 are discarded as burn-in. Furthermore, $J = 100$ particles are used in CPF-AS and $L = 100$ is used to analyze the behavior of RS with increasing l. (In practice, it is preferred to chose $L \ll J$ to avoid excessive computations in the worst-case scenario.) In total, 100 Monte Carlo simulations with $N = 100$ time samples are run.

In this example, the mean root mean squared errors are 1.63 ($\pm$0.45) and 1.62 ($\pm$0.44) for categorical sampling and RS, respectively. More importantly, in total 95.7 % ($\pm$1.4 %) of the ancestor indices are sampled using RS (within the $L = 100$ trials) when using RS. Fig. 1 shows the percentage of RS sampled ancestor indices as a function of the number of trials l. As it can be seen, a large portion of the indices is sampled within the first 20 trials (77.0 % in total). This indicates that RS-based ancestor sampling performs well in this case, significantly lowering the complexity of this step.

Additionally, Fig. 2 shows the empirical distribution of the difference between the exact acceptance probability, which requires knowledge of all the ancestor weights, and its lower bound (10). As it can be seen, the difference between the bound and the actual acceptance probability is largely below 0.1 with a large fraction being

smaller than 0.01. Hence, the lower bound (10) is a good approximation for this model.

4.2. Gaussian process learning

In this example, we consider learning of a temporal Gaussian process modeling stochastic volatility. In particular, the return y_n at time t_n of an asset is modeled as

$$f(t) \sim \mathcal{GP}(0, k(t, t'; \theta)),$$
$$p(y_n \mid f_n) = \mathcal{N}(y_n; 0, \exp(f_n)),$$

where $f(t)$ is a Gaussian process with covariance function $k(t, t'; \theta)$ modeling the log-volatility and $f_n \triangleq f(t_n)$. We use a sum of two basic covariance functions, one capturing the long-term behavior (long length scale), and one capturing the short-term variations (short length scale) given by

$$k(t, t'; \theta) = k_{\text{OU}}(t, t'; \theta_1) + k_{\text{OU}}(t, t'; \theta_2)$$

where $k_{\text{OU}}(\cdot)$ is the Ornstein–Uhlenbeck covariance function [19]

$$k_{\text{OU}}(t, t'; \theta) = \sigma^2 \exp\left(-\frac{|t - t'|}{\ell}\right)$$

with hyperparameters $\theta = \begin{bmatrix}\sigma^2 & \ell\end{bmatrix}^{\mathsf{T}}$. It has been shown that this model has an equivalent state-space representation, and thus, it can efficiently be learned using PMCMC and particle Gibbs, see [5, 20].

In this example, also the hyperparameters $\theta_1 = \begin{bmatrix}\sigma_1^2 & \ell_1\end{bmatrix}^{\mathsf{T}}$ and $\theta_2 = \begin{bmatrix}\sigma_2^2 & \ell_2\end{bmatrix}^{\mathsf{T}}$ need to sampled, in addition to the state trajectories. The variances σ_1^2 and σ_2^2 are sampled directly from their conditional posteriors, whereas Metropolis-within-Gibbs sampling is used for the lengthscales ℓ_1 and ℓ_2, see [5] for details.

This model is applied to the IBM share data for the period January 1993 to December 2003, which consists of $N = 2772$ datapoints. We use $J = 250$ particles in the particle filter and sample a total of 150 parameter and trajectory samples, of which 50 are discarded as burn-in. Thus, the posterior is approximated with $K = 100$ samples. We compare the results for CPF-AS with (standard) categorical sampling and the proposed method, where a maximum of $L = 20$ rejection sampling trials are used.

Fig. 3 shows the estimated log-volatility (posterior mean of the sampled trajectories), together with the realized log-volatility (based on sampling the asset's price every 5 minutes on the given trade day). As it can be seen, both methods learn the volatility equally well as no significant difference is visible. This is also reflected in the root mean squared errors between the estimated and realized log-volatility that are 0.69 and 0.63 for categorical- and RS-based ancestor sampling, respectively.

Furthermore, in this example, 61.6 % of all the ancestor indices (415 800 in this case) are sampled using RS. Fig. 4 again shows the percentage of ancestor indices sampled as a function of the iteration index l. This follows the same pattern as for the UNGM example, but is cut-off at $L = 20$ in this case. Again, the majority of ancestor indices are sampled within the first few RS trials and thus, increasing L does not increase performance significantly. Finally, Fig. 5 shows the empirical distribution of the difference between the exact RS acceptance probability and the lower bound. Again, the difference in this case is below 0.1 for the vast majority of ancestor indices and the lower bound is again a good approximation of the acceptance probability.

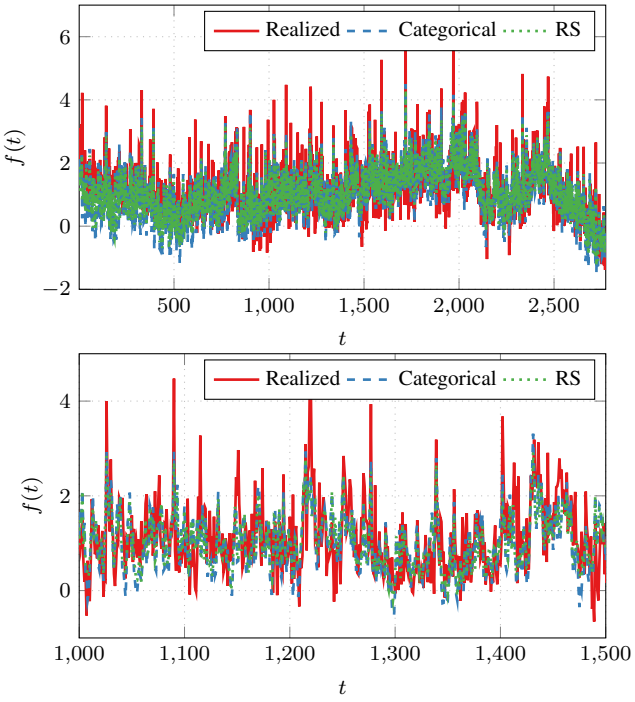

Fig. 3. Realized and estimated log-volatility for the whole data set (top), and detail view between $t = 1000$ and $t = 1500$ (bottom).

5. CONCLUSIONS

In this paper, a method for ancestor sampling in CPF-AS based on RS has been proposed. The method aims at reducing the computational complexity of the ancestor index sampling step by only considering one ancestor index at a time and best- and worst-case guarantees of the computational complexity have been provided.

The method has shown good performance in the considered examples, but naturally, it heavily depends on the model under consideration and its parameters. In particular, the provided bounds might be too loose for some models, which would yield high rejection rates. Future work will investigate this further and also consider non-Markovian models, where higher gains can be expected.

6. REFERENCES

[1] C. Andrieu, A. Doucet, and R. Holenstein, "Particle Markov chain Monte Carlo methods," *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, vol. 72, no. 3, pp. 269–342, 2010.

[2] F. Lindsten, M. I. Jordan, and T. B. Schön, "Particle Gibbs with ancestor sampling," *Journal of Machine Learning Research*, vol. 15, pp. 2145–2184, 2014.

[3] R. Frigola, F. Lindsten, T. B. Schön, and C. E. Rasmussen, "Bayesian inference and learning in Gaussian process state-space models with particle MCMC," in *Advances in Neural Information Processing Systems 26*, 2013, pp. 3156–3164.

[4] A. Svensson and T. B. Schön, "A flexible state-space model for learning nonlinear dynamical systems," *Automatica*, vol. 80, pp. 189–199, 2017.

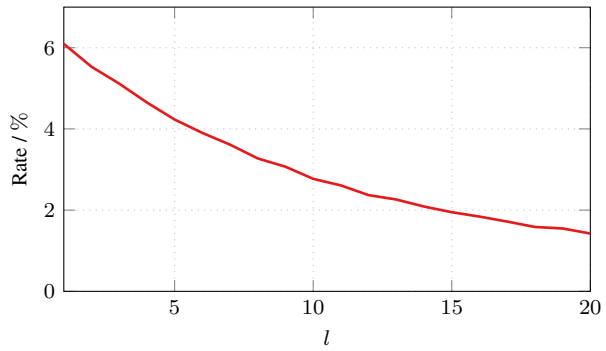

Fig. 4. Percentage of ancestor indices sampled using RS as a function of the iteration number l ($L = 20$).

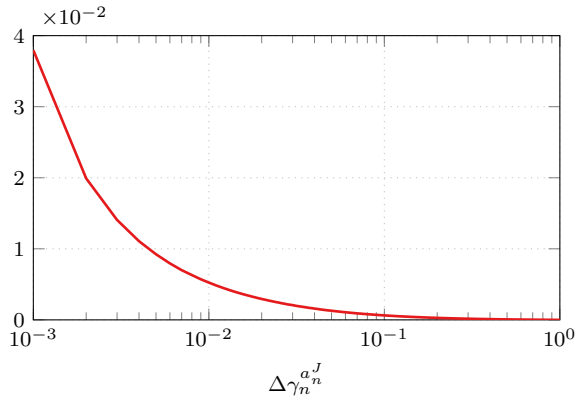

Fig. 5. Empirical distribution of the difference $\Delta\gamma_n^{a_n^J}$ between the true acceptance probability and the lower bound (11).

[5] R. Hostettler, S. Särkkä, and S. J. Godsill, "Rao–Blackwellized particle MCMC for parameter estimation in spatio-temporal Gaussian processes," in *27th IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, Tokyo, Japan, September 2017.

[6] J. Han, X.-P. Zhang, and F. Wang, "Gaussian process regression stochastic volatility model for financial time series," *IEEE Journal of Selected Topics in Signal Processing*, vol. 10, no. 6, pp. 1015–1028, September 2016.

[7] J. Kwon, R. Dragon, and L. Van Gool, "Joint tracking and ground plane estimation," *IEEE Signal Processing Letters*, vol. 23, no. 11, pp. 1514–1517, November 2016.

[8] N. Whiteley, "Discussion on particle Markov chain Monte Carlo methods," *Journal of the Royal Statistical Society: Series B*, vol. 72, pp. 306–307, 2010.

[9] N. Whiteley, C. Andrieu, and A. Doucet, "Efficient Bayesian inference for switching state-space models using discrete particle Markov chain Monte Carlo methods," Bristol Statistics Research, Tech. Rep., 2010.

[10] M. K. Pitt and N. Shephard, "Filtering via simulation: Auxiliary particle filters," *Journal of the American Statistical Association*, vol. 94, no. 446, pp. 590–599, 1999.

[11] S. J. Godsill, A. Doucet, and M. West, "Monte Carlo smoothing for nonlinear time series," *Journal of the American Statistical Association*, vol. 99, no. 465, pp. 156–168, 2004.

[12] A. Doucet and A. M. Johansen, "A tutorial on particle filtering and smoothing: Fifteen years later," in *Handbook of Nonlinear Filtering*, ser. Oxford Handbooks, D. Crisan and B. Rozovskii, Eds. Oxford, UK: Oxford University Press, 2011, vol. 12, pp. 656–704.

[13] R. Douc, A. Garivier, E. Moulines, and J. Olsson, "Sequential Monte Carlo smoothing for general state space hidden Markov models," *The Annals of Applied Probability*, vol. 21, no. 6, pp. 2109–2145, December 2011.

[14] E. Taghavi, F. Lindsten, L. Svensson, and T. B. Schön, "Adaptive stopping for fast particle smoothing," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, Vancouver, BC, Canada, May 2013, pp. 6293–6297.

[15] C. Andrieu, N. de Freitas, A. Doucet, and M. I. Jordan, "An introduction to MCMC for machine learning," *Machine Learning*, vol. 50, pp. 5–43, 2003.

[16] B. Øksendal, *Stochastic Differential Equaitons: An Introduction with Applications*, 6th ed. Springer, 2010.

[17] S. Särkkä, *Bayesian Filtering and Smoothing*. Cambridge, UK: Cambridge University Press, 2013.

[18] G. Kitagawa, "Monte Carlo filter and smoother for non-Gaussian nonlinear state space models," *Journal of Computational and Graphical Statistics*, vol. 5, no. 1, pp. 1–25, 1996.

[19] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. The MIT Press, 2006.

[20] S. Särkkä, A. Solin, and J. Hartikainen, "Spatiotemporal learning via infinite-dimensional Bayesian filtering and smoothing: A look at Gaussian process regression through Kalman filtering," *IEEE Signal Processing Magazine*, vol. 30, no. 4, pp. 51–61, July 2013.