Bayesian Analysis (2018) **13**, Number 4, pp. 1137–1161

Sequential Monte Carlo Smoothing with Parameter Estimation

Biao Yang*, Jonathan R. Stroud†, and Gabriel Huerta‡

Abstract. We propose two new sequential Monte Carlo (SMC) smoothing methods for general state-space models with unknown parameters. The first is a modification of the particle learning and smoothing (PLS) algorithm of Carvalho, Johannes, Lopes, and Polson (2010), with an adjustment in the backward resampling weights. The second, called Refiltering, is a two-stage method that combines sequential parameter learning and particle smoothing algorithms. We illustrate the methods on three benchmark models using simulated data, and apply them to a stochastic volatility model for daily S&P 500 index returns during the financial crisis. We show that both new methods outperform existing SMC approaches, and that Refiltering is competitive with smoothing approaches based on Markov chain Monte Carlo (MCMC) and Particle MCMC.

Keywords: Bayesian smoothing, particle filtering, particle learning, particle smoothing, state-space models, stochastic volatility.

1 Introduction

State-space models are an extremely general class of models for time series. These models allow for nonlinear and non-Gaussian features that occur in many fields, including finance, ecology, biology and engineering. Over the past few decades, sequential Monte Carlo (SMC) methods have become extremely popular for online state and parameter estimation in state-space models. However, these methods have been largely ignored for *Bayesian smoothing*, which involves retrospective estimation of states and parameters. A paper by Carvalho, Johannes, Lopes, and Polson (2010) has brought renewed attention to this topic. Here, following on that paper, we propose two new SMC smoothing algorithms.

SMC methods, also known as particle filters, were developed for online state estimation in state-space models. The idea was introduced by Gordon, Salmond, and Smith (1993) with the name *Bootstrap Filter*. Pitt and Shephard (1999) then proposed an improved approach called the *Auxiliary Particle Filter*. However, the problem of estimating unknown parameters with SMC methods has generally received less attention. Kitagawa (1998) proposed augmenting the state vector to include the unknown parameters, and applying a particle filter on the augmented state. To avoid over-dispersion

*Senior Associate, Analytics, Discover Financial Services, Riverwoods, IL, 60015, yangbiao@gwmail.gwu.com. Biao Yang was a Ph.D. student in the Department of Statistics, George Washington University when this work was done

†Associate Professor, McDonough School of Business, Georgetown University, Washington, DC, 20057, jrs390@georgetown.edu

‡Professor, Department of Mathematics and Statistics, University of New Mexico, Albuquerque, NM, 87131, ghuerta@stat.unm.edu

 https://doi.org/10.1214/17-BA1088

problems in the estimation of states and parameters, Liu and West (2001) used kernel density estimation for the static parameters. This algorithm remains the most general method for sequential Bayesian state and parameter estimation. Storvik (2002) and Fearnhead (2002) discussed sequential Bayesian inference for parameters in situations where sufficient statistics for the parameters are available. Lopes and Tsay (2011) and Kantas, Doucet, Singh, Maciejowski, and Chopin (2015) provide excellent reviews of existing parameter learning approaches.

There is a large literature on SMC smoothing when parameters are known. Smoothing involves estimation of states conditional on observations up to the final time period. Kitagawa (1996) introduced the idea of particle smoothing by storing the state vector. This is a forward-only algorithm where the smoothed state trajectories are generated recursively by reweighting and resampling the filtered particles within a smoothing (time) window. But as time evolves and the size of the smoothing window increases, the smoothed samples at the start of the time series will degenerate to a single path. Godsill, Doucet, and West (2004) proposed the forward-backward smoother, in which a backward recursion is included and the forward filter particles are reweighted. Other smoothing algorithms include the two-filter smoother of Kitagawa (1996), the generalized two-filter smoother of Briers, Doucet, and Maskell (2010), and the $O(N^2)$ and $O(N)$ smoothing algorithms of Fearnhead, Wyncoll, and Tawn (2010). These methods are collectively known as particle smoothing.

The literature on particle smoothing with unknown parameters is quite limited. To our knowledge, there are only two existing SMC approaches for smoothing with parameter estimation. The first is the self-organizing state-space models of Kitagawa (1998), which uses a fixed-lagged smoother with state augmentation. The second is the particle learning and smoothing (PLS) algorithm of Carvalho, Johannes, Lopes, and Polson (2010). PLS is a two-stage algorithm with a particle filtering and learning step, followed by separate backward reweighting step for each sample of the parameters. However, the PLS smoothing algorithm does not account for the dependence between states and parameters. In this paper, we propose an adjusted PLS algorithm that takes into account this dependence in defining the backwards resampling weights. In addition, we propose a new smoothing algorithm, in which we apply a forward-backward smoother on each parameter drawn from the last filter step and get a smoothed sample of the states, while accounting for parameter uncertainty.

Markov chain Monte Carlo (MCMC) methods have also been used for Bayesian smoothing. Carlin, Polson, and Stoffer (1992) introduced the first MCMC smoothing algorithms for non-normal and nonlinear state-space models. Carter and Kohn (1994) and Frühwirth-Schnatter (1994) proposed the forward-filtering, backward-sampling (FFBS) algorithm and de Jong and Shephard (1995) developed the related simulation smoother for conditionally Gaussian models. The FFBS is a highly efficient block sampler that draws the states jointly given the parameters for linear, Gaussian state-space models. Shephard and Pitt (1997) and Gamerman (1998) provided block sampling algorithms for dynamic exponential family and generalized linear models, respectively. Other MCMC smoothers include Scipione and Berliner (1992), Geweke and Tanizaki (2001), Stroud, Müller, and Polson (2003) and Niemi and West (2010).

Andrieu, Doucet, and Holenstein (2010) introduced another approach, called *Particle Markov chain Monte Carlo* (PMCMC), for off-line state smoothing and parameter estimation. This method combines MCMC and SMC approaches. A popular PMCMC algorithm is the *Particle Marginal Metropolis-Hastings* (PMMH) sampler. At each Markov chain iteration, this method simulates a parameter from a proposal density, and uses it within a SMC algorithm to create a high-dimensional proposal distribution for the smoothed states. One uses the approximate likelihood from the SMC and the prior density for the parameters, to accept or reject the proposed states and parameters. The advantage of PMMH is that it integrates out the states when updating the parameters and generally reduces autocorrelation in the sampler. The disadvantage is that this method is computationally intensive, since one must run a new SMC algorithm at each MCMC iteration.

Our new approaches have a number of advantages relative to existing methods. Refiltering provides samples from the correct posterior distribution as the number of particles goes to infinity. Unlike MCMC methods, our smoothing methods are not iterative and do not require convergence of a Markov chain and can be easily parallelized. Finally, in contrast to MCMC approaches, our methods provide accurate estimates of the marginal likelihood, which is used for Bayesian model evaluation and selection.

The paper is organized as follows. In Section 2, we give a short review of particle filtering and smoothing algorithms. Two new smoothing algorithms are proposed in Section 3. In Section 4, we compare the two new algorithms to PLS, MCMC and PMMH on three simulated examples: an autoregressive plus noise model, a nonlinear growth model, and a chaotic model. In Section 4.4, we compare the algorithms using a stochastic volatility model for daily S&P 500 returns. Finally, Section 5 concludes.

2 Filtering and Smoothing with SMC

Consider a general state-space model defined at discrete times $t = 1, \ldots, T$,

$$\text{Observation}: \quad y_t \sim p(y_t|x_t, \theta), \tag{1}$$

$$\text{Evolution}: \quad x_t \sim p(x_t|x_{t-1}, \theta), \tag{2}$$

$$\text{Initial}: \quad x_0 \sim p(x_0|\theta), \tag{3}$$

where y_t are the observations, x_t are the unobserved states, θ are the static parameters, and $p(\cdot|\cdot)$ denotes a generic conditional distribution. The measurement distribution (1) relates the states to the observations; the transition distribution (2) describes the evolution of the states over time; and the initial distribution (3) summarizes beliefs about the initial state x_0. The Bayesian model is completed with a prior distribution on the unknown parameters, $\theta \sim p(\theta)$. The state-space model (1)–(3) is quite general, making two main assumptions: x_t follows a first-order Markov process, and y_t's are conditionally independent given the states.

Bayesian inference in state-space models requires calculation of the joint posterior distribution of the states and parameters. This can either be done sequentially or retrospectively. Let $x^s = (x_0, \ldots, x_s)$ and $y^s = (y_1, \ldots, y_s)$ denote the states and observations

up to time s. There are two main types of inference in state-space models. The *filtering problem* involves sequential calculation of $p(x_t, \theta|y^t)$, for $t = 1, \ldots, T$. The *smoothing problem* requires calculation of $p(x^T, \theta|y^T)$, retrospectively. Here, the objective is to solve the joint smoothing problem, which requires computing the *joint posterior distribution* for the states and parameters:

$$p(x^T, \theta|y^T) \propto p(\theta) \left[p(x_0|\theta) \prod_{t=1}^{T} p(x_t|x_{t-1}, \theta) p(y_t|x_t, \theta) \right].$$

For most models, this distribution is unavailable in closed form, so Monte Carlo methods are required to sample from it. In particular, we introduce two novel algorithms that draw samples from the joint posterior smoothing distribution $p(x^T, \theta|y^T)$. Furthermore, we compare the performance of these algorithms to the other existing smoothing methods, such as PLS, MCMC and PMMH, using simulation studies and real data examples.

Traditionally, SMC methods assume that θ is known and are designed to approximate $p(x_t|y^t, \theta)$ with a set of weighted samples or particles. The subsections below give a brief overview of sampling importance resampling (SIR) particle filters, particle filters with unknown parameters, and the particle learning and smoothing algorithm.

2.1 Particle Filtering

The particle filter was first introduced by Gordon, Salmond, and Smith (1993) to conduct state estimation in nonlinear, non-Gaussian state-space models. Based on importance sampling, the particles $x_{t-1}^{(i)}$ are propagated forward through the system equation, the new particles $\tilde{x}_t^{(i)}$ are resampled with weights $\omega_t^{(i)}$ that are proportional to the likelihood $p(y_t|\tilde{x}_t^{(i)})$ to get filtered particles at time t: $x_t^{(i)}$. The filtering density $p(x_t|y^t)$ can then be approximated by a discrete empirical distribution based on these particles,

$$\begin{aligned} p(x_t|y^t) &\propto p(y_t|x_t) \int p(x_t|x_{t-1}) p(x_{t-1}|y^{t-1}) dx_{t-1} \\ &\approx p(y_t|x_t) \sum_{i=1}^{N} p(x_t|x_{t-1}^{(i)}) \omega_t^{(i)}. \end{aligned} \tag{4}$$

The algorithms to implement the SIR particle filter, along with information about other algorithms, are given in the online supplementary materials (Yang et al., 2017).

2.2 Particle Filtering with Unknown Parameters

To deal with particle filtering with unknown parameters, Kitagawa (1998) introduced the idea of augmenting the state by the parameters as $z_t = (x_t, \theta)'$, then applying a bootstrap filter to the augmented state vector z_t. Kitagawa and Sato (2001) extended the idea by adding artificial noise to the parameters in the evolution equation to avoid particle degeneracy (i.e., collapse of samples to a single value) as time progresses.

Liu and West (2001) proposed an improvement to Kitagawa's method by drawing samples of the parameters from a kernel density of the form

$$p(\theta_{t+1}|y^t) \approx \sum_{j=1}^{N} w_t^{(j)} \mathcal{N}(\theta_{t+1}|m_t^{(j)}, h^2 V_t),$$

at each filter step t. Here $m_t^{(j)} = a\theta_t^{(j)} + (1-a)\bar{\theta}_t$, where $\bar{\theta}_t$ and V_t are the sample mean and variance-covariance matrix of the posterior samples of θ_t at time t, and $a = \sqrt{1-h^2}$ is a smoothing parameter between 0 and 1. Notice that $a = 1$ implies the evolution equation $\theta_{t+1} = \theta_t$, which corresponds to state augmentation with no evolution noise. For the case of $a = 0$, each θ_{t+1} becomes an independent draw from a $N(\bar{\theta}_t, V_t)$. In practice a typical value for a is from 0.95 to 0.99, which implies a strong persistence in the evolution of θ_t.

In situations where $p(\theta|x^t, y^t)$ depends on a low dimensional set of sufficient statistics $s_t = \mathcal{S}(x^t, y^t)$ such that $p(\theta|x^t, y^t) = p(\theta|s_t)$ and s_t can be updated recursively via $s_t = f(s_{t-1}, x_t, y_t)$, the method of Storvik (2002) can be applied to draw samples from the filtering distribution of the parameters. We include the sufficient statistics s_t in the state vector and draw samples of θ based on the sufficient statistics at each time t in the filter. By doing this, the impoverishment problem is mitigated and the exact posterior $p(\theta|y^t)$ can be better and gradually approximated through the filtering process. The approach is based on the decomposition:

$$p(x^t, \theta|y^t) \propto p(x^{t-1}|y^{t-1}) p(\theta|s_{t-1}) p(x_t|x_{t-1}, \theta) p(y_t|x_t, \theta) \tag{5}$$

Storvik's algorithm generates samples from this distribution at each time $t = 1, \ldots, T$, using the following steps.

Storvik's SIR Filter

1. Set $s_0^{(i)} = s_0$ and draw $\theta^{(i)} \sim p(\theta|s_0^{(i)})$ and $x_0^{(i)} \sim p(x_0|\theta^{(i)})$ for $i = 1, \ldots, N$.
2. For each time $t = 1, \ldots, T$:
   (a) Propagate $x_t^{(i)} \sim p(x_t|x_{t-1}^{(i)}, \theta^{(i)})$ for $i = 1, \ldots, N$.
   (b) Compute weights $\omega_t^{(i)} \propto p(y_t|x_t^{(i)}, \theta^{(i)})$ for $i = 1, \ldots, N$.
   (c) Update sufficient statistics $s_t^{(i)} = S(x_t^{(i)}, s_{t-1}^{(i)}, y_t)$ for $i = 1, \ldots, N$.
   (d) Resample N times from $\{(x_t^{(i)}, s_t^{(i)})\}_{i=1}^N$ with weights $\omega_t^{(i)}$, to obtain a sample from $p(x_t, s_t|y^t)$.
   (e) Sample $\theta^{(i)} \sim p(\theta|s_t^{(i)})$ for $i = 1, \ldots, N$.

Steps 2(d)–(e) provide samples from the Bayesian filtering density: $(x_t^{(i)}, \theta^{(i)}) \sim p(x_t, \theta|y^t)$.

2.3 Particle Learning and Smoothing (PLS)

For particle smoothing with unknown parameters, we are interested in estimating the states and parameters conditional on the entire dataset y^T by drawing samples $(x^{T(i)}, \theta^{(i)})$ from the joint posterior $p(x^T, \theta|y^T)$, where T is the total number of time steps.

Carvalho et al. (2010) showed that after running the filtering and learning algorithm, the filtered particles can be resampled to obtain draws from the smoothing distribution. The procedure is based on Bayes' rule and the decomposition of the joint posterior smoothing distribution written as

$$p(x^T, \theta|y^T) = p(x_T, \theta|y^T) \prod_{t=1}^{T-1} p(x_t|x_{t+1}, \theta, y^t), \tag{6}$$

where

$$p(x_t|x_{t+1}, \theta, y^t) \propto p(x_{t+1}|x_t, \theta)p(x_t|\theta, y^t). \tag{7}$$

The steps of this algorithm are now presented:

PLS Algorithm

1. *(Forward Filter)* Run the particle learning algorithm to generate samples $\{(x_t^{(i)}, \theta^{(i)})\}_{i=1}^N$ from $p(x_t, \theta|y^t)$ at each time $t = 1, \ldots, T$.

2. *(Backward Smoother)* Select a pair $(x_T^{(i)}, \theta^{(i)})$ from Step 1, and simulate backwards: For $t = T-1, \ldots, 1$, resample the particles $\{x_t^{(j)}\}_{j=1}^N$ from Step 1 with weights proportional to $\omega_t^{(j)} = p(x_{t+1}^{(i)}|x_t^{(j)}, \theta^{(i)})$ to generate $x_t^{(i)}$ from the smoothing distribution.

Carvalho et al. (2010) claim that PLS is an extension of Godsill, Doucet, and West (2004) to state-space models with unknown parameters. However, note that in the backward smoothing step of PLS, a fixed $\theta^{(i)}$ is selected first and the resampling weights are evaluated proportional to $p(x_{t+1}^{(i)}|x_t^{(j)}, \theta^{(i)})$. Thus, we should use samples drawn from $p(x_t|\theta^{(i)}, y^t)$, i.e., the filter samples with respect to this fixed $\theta^{(i)}$, but this is not the case for PLS.

The particles from the forward pass of PLS are from the marginal density $p(x_t|y^t)$, not from the conditional density $p(x_t|\theta, y^t)$. Reweighting these particles using the transition density ignores the dependence between states and parameters. This causes inaccurate smoothing estimates when such dependency is strong. Figure 1 shows the dependency between the filtered samples of the states and parameters for the AR(1) plus noise model that is presented in Section 4. Correlations between x_t and θ greater than 0.5 can be observed at the beginning of the time series. The simulation studies presented in Section 4 show that PLS gives poor smoothing estimates, particularly at early time periods of the time series.

In the following section we introduce two new smoothing algorithms. The first algorithm relies on a transformation of (4) and an adjustment of the weights in the back-

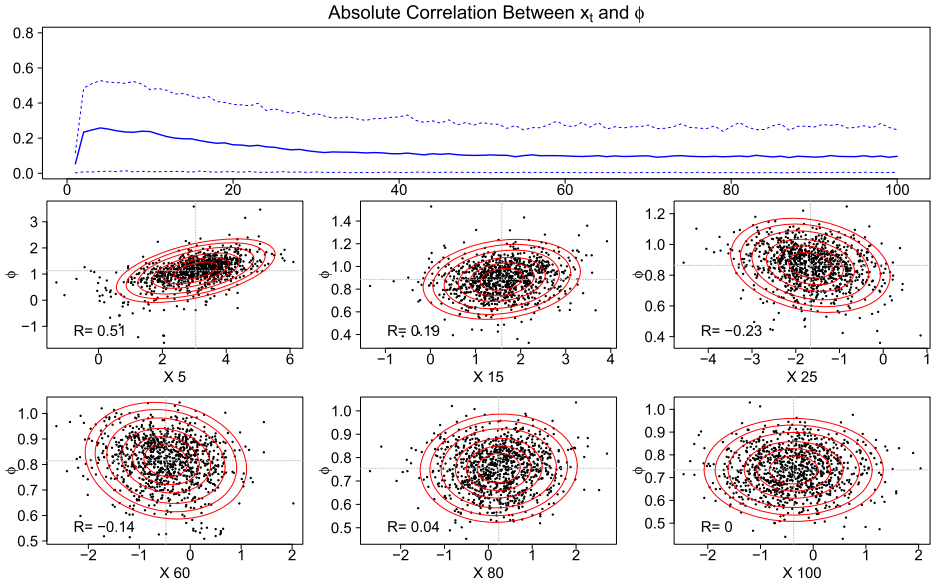

Figure 1: Posterior dependence between x_t and ϕ for the AR plus noise model (see Section 4.1). Top: means and 95% intervals for the absolute correlation between x_t and ϕ at each time t. Results are based on 500 simulated datasets. Middle and Bottom: posterior samples of x_t and ϕ at selected time steps for one dataset. The contours represent the fitted normal densities used in the PLSa algorithm.

ward smoothing pass to refine PLS. The second algorithm involves a separate forward filtering-backward sampling pass for each sampled parameter.

3 New Smoothing Algorithms

3.1 PLS with Adjustment (PLSa)

The backward reweighting scheme in the PLS algorithm assumes that the filtering algorithm provides samples from the conditional distribution, $p(x_t|\theta, y^t)$, when in fact the samples come from the joint distribution, $p(x_t, \theta|y^t)$, and hence, from the marginal distribution, $p(x_t|y^t)$. Thus, PLS does not provide samples from the target smoothing distribution. We consider the following rearrangement of (7):

$$\underbrace{p(x_t|x_{t+1}, \theta, y^t)}_{\text{smoother}} \propto \underbrace{p(x_{t+1}|x_t, \theta)\,\frac{p(x_t|\theta, y^t)}{p(x_t|y^t)}}_{\text{weights}}\,\underbrace{p(x_t|y^t)}_{\text{filter}}, \tag{8}$$

so, $\omega_t^{(j)} = p(x_{t+1}|x_t^{(j)}, \theta)p(x_t^{(j)}|\theta, y^t)/p(x_t^{(j)}|y^t)$ are the resampling weights for the backward smoothing pass. These are the weights for the filtered particles in the smoothing

algorithm. Note that in general, we cannot compute these resampling weights exactly, since the joint filtering distribution $p(x_t, \theta|y^t)$ is not available in closed form. Therefore, we propose to use a multivariate normal approximation for $p(x_t, \theta|y^t)$ based on the filtered particles $\{(x_t^{(i)}, \theta^{(i)})\}_{i=1}^N$, in combination with appropriate transformations to the states and/or parameters if necessary.

The particle learning and smoothing algorithm with adjustment (PLSa) proceeds in the same way as PLS, but uses adjusted weights in the backward pass. Based on the simulation results in Section 4, we find that the adjustment of the weights matters: the adjusted version outperformed the original one, especially in the beginning of the time series, where the PLS usually has problems to recover the true states and/or parameters.

PLSa Algorithm

1. *(Forward Filter)* Run a filtering and learning algorithm to obtain samples $\{(x_t^{(i)}, \theta^{(i)})\}_{i=1}^N$ from $p(x_t, \theta|y^t)$ for $t = 1, \ldots, T$. Use the filtered samples to construct a multivariate normal approximation at each time t, where the mean vector and covariance matrix are approximated with sample moments from $\{(x_t^{(i)}, \theta^{(i)})\}_{i=1}^N$.

$$p(x_t, \theta|y^t) \approx \mathcal{N}\left(\begin{pmatrix}\mu_t^x \\ \mu_t^\theta\end{pmatrix}, \begin{pmatrix}\Sigma_t^x & \Sigma_t^{x\theta} \\ \Sigma_t^{\theta x} & \Sigma_t^\theta\end{pmatrix}\right).$$

   This implies that the marginal and conditional distributions are also normal: $p(x_t|y^t) \approx \mathcal{N}(\mu_t^x, \Sigma_t^x)$, and $p(x_t|\theta, y^t) \approx \mathcal{N}(\mu_t^{x|\theta}, \Sigma_t^{x|\theta})$, where the conditional mean and covariance are given by the well-known formulas for normal distributions.

2. *(Backward Smoother)* Select a pair $(x_T^{(i)}, \theta^{(i)})$ from Step 1, and simulate backwards: For $t = T-1, \ldots, 1$, resample the $\{x_t^{(j)}\}_{j=1}^N$ from Step 1 with weights proportional to

$$\begin{aligned}\omega_t^{(j)} &= p(x_{t+1}^{(i)}|x_t^{(j)}, \theta^{(i)}) \left(\frac{p(x_t^{(j)}|\theta^{(i)}, y^t)}{p(x_t^{(j)}|y^t)}\right) \\ &\approx p(x_{t+1}^{(i)}|x_t^{(j)}, \theta^{(i)}) \left(\frac{\mathcal{N}(x_t^{(j)}|\mu_t^{x|\theta^{(i)}}, \Sigma_t^{x|\theta})}{\mathcal{N}(x_t^{(j)}|\mu_t^x, \Sigma_t^{xx})}\right)\end{aligned}$$

   to generate $x_t^{(i)}$.

There are two points to note here. First, although our description above does not explicitly mention it, when Storvik's algorithm or Particle Learning is used, then the sufficient statistics s_t are stored and sampled at each time in the forward filter (Step 1). However, the PLSa algorithm is actually more general, and can be used for other sequential filtering and learning algorithms that do not involve sufficient statistics, such as Liu and West (2001). Second, although we have chosen to approximate the joint posterior using a normal distribution, other density estimators could also be used, for example, a kernel density estimator as in Silverman (1986). The main point here is that we need some analytical approximation to compute the additional density ratio in the weights.

3.2 Refiltering Smoothing Algorithm

In addition to the PLSa modification, we propose the following new smoothing algorithm. The idea behind this algorithm is the following decomposition of the joint posterior distribution:

$$p(x^T, \theta|y^T) = p(x^T|y^T, \theta)\, p(\theta|y^T). \tag{9}$$

Therefore, Storvik's forward filter, particle learning, or a more general filtering method such as Liu and West (2001) can be implemented to get samples of the parameters at the last time step, i.e. $\theta^{(i)} \sim p(\theta|y^T)$. Then for each sample, $\theta^{(i)}$, a forward-backward smoothing algorithm is applied as in Godsill et al. (2004) to get one state trajectory $x^{T(i)}$ from $p(x^T|y^T, \theta^{(i)})$. Repeating this process for each $\theta^{(i)}$ produces samples of the states from the marginal smoothing density $p(x^T|y^T)$.

Since the run time for the forward filter is negligible compared to the backward smoother, ($O(N)$ vs $O(N^2)$, respectively), in simulation studies, we found that this algorithm almost has the same speed as PLS, but with differences in the accuracy of the state estimates. The algorithm is presented below:

Refiltering Algorithm

1. *(Forward Filter)* Use Storvik, Particle Learning or Liu & West to run a forward filter and learning algorithm and generate $\theta^{(i)} \sim p(\theta|y^T)$.
2. *(Forward-Backward Smoother)* For each $\theta^{(i)}, i = 1, \ldots, N_0$, run a forward-backward smoothing algorithm to get a sample $x^{T(i)} \sim p(x^T|y^T, \theta^{(i)})$.

Note that this algorithm has a complexity of $O(N^2)$, the same as PLS, but it can be made an $O(N)$ algorithm in two ways. The first way is to choose a small number of states $n_0 \ll N$ for the forward-backward smoother in step 2. The second idea is to use a small number of parameter draws of size N_0 in step 2. Our simulation studies show that both ideas provide a substantial speed up with only a minor loss of accuracy.

In cases where the state-space model is linear and the errors are Gaussian, i.e., $x_t|x_{t-1} \sim \mathcal{N}(G_t x_{t-1}, W)$, $y_t|x_t \sim \mathcal{N}(F_t' x_t, V)$, Step 2 of the refiltering algorithm can be implemented using a forward filtering, backward sampling (FFBS) algorithm (Carter and Kohn, 1994; Frühwirth-Schnatter, 1994). A Kalman filter forward pass is first run and then a sample, backwards in time, is generated. Note that $p(x_t|y^{t-1}) \sim \mathcal{N}(a_t, R_t)$ is the prior and $p(x_t|y^t) \sim \mathcal{N}(m_t, C_t)$ is the posterior of the state at each time point t, which depends on the parameters $\theta = (F_t, G_t, V, W)$.

Refiltering with FFBS

1. *(Forward Filter)* Use Storvik, Particle Learning, or Liu & West to run a forward filter and learning algorithm and generate samples $\theta^{(i)} \sim p(\theta|y^T), i = 1, \ldots, N$.
2. *(FFBS)* For each $\theta^{(i)}, i = 1, \ldots, N_0$, run an FFBS algorithm to generate samples $x^{T(i)} \sim p(x^T|y^T, \theta^{(i)})$. This requires two steps. (a) Run a Kalman filter and store

the forecast and filtered moments a_t, R_t, m_t, C_t for $t = 1, \ldots, T$. (b) Sample $x_T^{(i)} \sim \mathcal{N}(m_T, C_T)$. For $t = T-1$ to 1, sample $x_t^{(i)} \sim p(x_t|x_{t+1}^{(i)}, \theta^{(i)}, y^t) = \mathcal{N}(h_t, H_T)$, in which $h_t = m_t + B_t(x_{t+1}^{(i)} - a_{t+1})$, $H_t = C_t - B_t R_{t+1} B_t'$ and $B_t = C_t G_{t+1}' R_{t+1}^{-1}$.

We note that similar simplifications in Step 2 can be made for other models where $p(x^T|\theta, y^T)$ is available in closed form and can be simulated directly, for example, for hidden Markov models using forward-backward samplers (e.g., Scott, 2002).

The refiltering algorithm consists of a filtering and smoothing step to generate from $\theta \sim p^N(\theta|y)$ and $x \sim p^N(x|\theta, y)$, respectively. The combined algorithm provides samples from $p^N(\theta, x|y) = p^N(\theta|y)p^N(x|\theta, y)$. As $N \to \infty$, the cumulative distribution function (CDF) of $p^N(\theta|y)$ converges to the CDF associated with $p(\theta|y)$. In addition the CDF of $p^N(x|\theta, y)$ converges to the CDF of $p(x|\theta, y)$ thus implying that the joint CDF $p^N(\theta, x|y)$ converges to the joint CDF of $p(\theta, x|y)$. The refiltering algorithm can be used with any filtering and smoothing algorithm, and its asymptotic properties depend on the choice of algorithms. In this paper, we consider only filtering algorithms based on sufficient statistics (Storvik, 2002; Carvalho et al., 2010). These methods provide asymptotic draws from the marginal posterior $p(\theta|y)$ (see discussion in Lopes et al., 2011). For the smoothing step, we use either the FFBS algorithm or the forward-backward smoother of Godsill et al. (2004). FFBS provides *exact draws* from $p(x|\theta, y)$ and the Monte Carlo smoother provides *asymptotic draws* from it (see proof in Godsill et al., 2004). Thus, we view the refiltering algorithm as asymptotically exact for the examples considered in the paper.

4 Numerical Illustrations

4.1 AR(1) Plus Noise Model

We first consider the AR(1) plus noise model:

$$\begin{aligned} y_t &= x_t + v_t, \qquad v_t \sim \mathcal{N}(0, V), \\ x_t &= \phi x_{t-1} + w_t, \quad w_t \sim \mathcal{N}(0, W). \end{aligned}$$

This is a common benchmark model for testing state-space algorithms (Storvik, 2002; Polson, Stroud, and Müller, 2008). For this model, FFBS can be implemented and a long MCMC chain with 150,000 iterations is used as a standard for comparisons with other smoothing algorithms. The simulation study under this model is based on $T = 100$ observations with parameter values $V = 1$, $W = 1$, $\phi = 0.75$ and $x_0 = 0$.

The model includes three unknown parameters $\theta = (\phi, W, V)$. We assume conjugate priors, with $\phi|W \sim \mathcal{N}(b_0, B_0^{-1}W)$, $W \sim \mathcal{IG}(n_0, d_0)$, $V \sim \mathcal{IG}(\nu_0, \delta_0)$, where $\mathcal{IG}(a, b)$ denotes the inverse-gamma distribution with shape and rate parameters a and b, respectively. We choose hyperparameter values $n_0 = 2$, $\nu_0 = 2$, $d_0 = 2$, $\delta_0 = 2$, $b_0 = 0.5$ and $B_0 = 1$, implying fairly diffuse prior distributions. The conjugate priors for the parameters allow us to apply Storvik's algorithm based on the sufficient statistics $s_t = (b_t, B_t, n_t, d_t, \nu_t, \delta_t)$, and the following updating recursions:

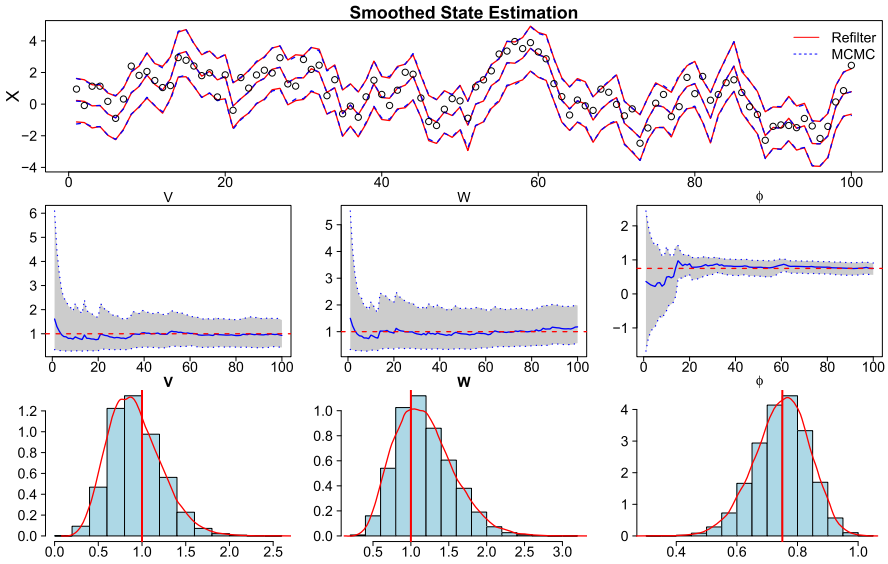

Figure 2: State and parameter estimation for the AR(1) plus noise model (see Section 4.1). Top row: smoothed means and 95% intervals for the states using the refiltering algorithm (solid lines) and long MCMC (dashed lines). Middle row: filtered means and 95% intervals for the parameters using Storvik (2002)'s algorithm. Bottom row: posterior distributions for the parameters at the last time step for the Storvik algorithm (histograms) and the long MCMC (density curves). True parameter values are indicated by horizontal lines in Row 2 and vertical lines in Row 3.

$$B_t = B_{t-1} + x_{t-1}^2, \qquad b_t = B_t^{-1}(B_{t-1}b_{t-1} + x_{t-1}x_t),$$
$$n_t = n_{t-1} + 1/2, \qquad d_t = d_{t-1} + (b_{t-1}^2 B_{t-1} + x_t^2 - b_t^2 B_t)/2,$$
$$\nu_t = \nu_{t-1} + 1/2, \qquad \delta_t = \delta_{t-1} + (y_t - x_t)^2/2.$$

Figure 2 shows the parameter learning plots and the posterior distribution at the last time period $T = 100$, corresponding to Storvik's filtering algorithm with 50,000 particles and MCMC with 150,000 iterations, respectively. From the figure, we notice the true parameters values are learned properly and the samples of the parameters at the last time step are well concentrated around the true parameter values. Also, the samples from Storvik's filter agree well with samples from a long MCMC. State smoothing by refiltering and the result of a long MCMC are also presented in Figure 2. We notice that the mean, 2.5^{th} and 97.5^{th} quantiles of the smoothing samples almost coincide for the two methods at each time step t.

To show that PLSa outperforms PLS based on the same computation time, we ran 500 simulations for each of these two methods and compared the standardized absolute errors over time, i.e., $\hat{e}_t^* = |\hat{x}_t - \hat{x}_t^{true}|/\sigma(x_t|y^T)$ for $t = 1, \ldots, T$, where $\hat{x}_t^{true}$ and $\sigma(x_t|y^T)$ are the smoothed mean and standard deviation for x_t computed from

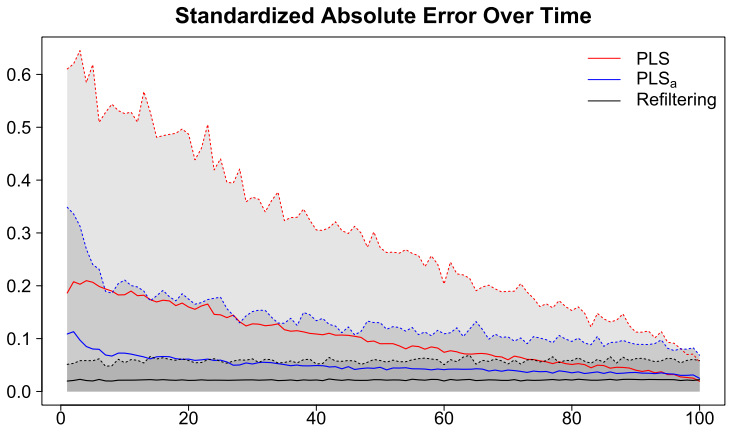

Figure 3: AR(1) Plus Noise Model. Standardized absolute errors over time for three smoothing algorithms compared to a long MCMC. The results are based on 500 simulations. The long MCMC is treated as the truth. The solid lines are the means of standardized absolute errors at time t among 500 simulations, while the dashed lines represent the 95^{th} quantile.

the long MCMC with 150,000 iterations, and $\hat{x}_t$ is the smoothed mean from the other algorithms. The results are shown in Figure 3. From this plot, we can see the main difference between the two smoothing algorithms appears at the beginning of the series, in which the dependence of states and parameters is strong, so the adjustment matters. As time progresses, the dependency of states and parameters decreases, and the results from the PLSa coincide with PLS. In addition, the results from the refiltering smoothing algorithm are also shown in this figure. For this AR(1) model, refiltering substantially outperforms the other two methods, and its accuracy in terms of absolute standardized errors is fairly constant over time. Note that the number of particles for the three smoothing algorithms was adjusted to assure similar computation time for all methods.

To compare the performance of all of the smoothing algorithms, we implemented long and short MCMC runs, pseudo-marginal Metropolis-Hastings (PMMH), PLS, PLSa, refiltering, $O(N)$ refiltering, and refiltering with FFBS using 500 simulated datasets. To allow PMMH to provide the best possible results, we consider the following. The last draw and the standard deviations of the parameter samples from a very long MCMC are given as the initial parameter values in PMMH and used to choose the covariance of the random walk proposal respectively. For each iteration, a SMC algorithm is run with 500 particles. All SMC based smoothing algorithms are run in parallel on 16 cores on a single node. Based on a similar run time, the mean standardized absolute errors over time (MAE* $= \sum_{t=1}^{T} |\hat{e}_t^*|/T$) are listed in Table 1.

From the table, we notice that the MAE* values for PLSa are about half as large as for PLS. For the refiltering algorithms, the MAE* magnitude is only about one fifth of that for PLS. Hence, both of the new smoothing algorithms outperform the PLS smoothing algorithm of Carvalho et al. (2010). If RF is compared with MCMC

and PMMH in state estimation, RF-FFBS gives almost the same results as MCMC. Both Refiltering and MCMC dominate PMMH in state estimation. The column labeled MAE* represents the mean standardized absolute error between the posterior mean of the parameters at the last time step for a long MCMC in relation to the other algorithms. From the table, we notice that both Refiltering and MCMC provide a much better parameter estimation than PMMH.

Method	Time (s)	Iterations M	Particles N	MAE* States	MAE* Params
Long MCMC	290	150000	–	–	–
MCMC	15	7035	–	0.016	0.034
PMMH	15	170	500	0.287	0.407
PLS	16	–	1200	0.138	0.048
PLSa	16	–	500	0.076	0.048
Refiltering	15	–	10000/150	0.024	0.048
RF-FFBS	**16**	–	**14000**	**0.017**	**0.048**

Table 1: Comparison of smoothing algorithms for the AR(1) plus noise model (see Section 4.1). MAE* denotes the standardized mean absolute error. Results are based on 500 simulated datasets.

4.2 Non-stationary Growth Model

We next consider the non-stationary growth model:

$$\begin{aligned} y_t &= \frac{x_t^2}{20} + v_t, \\ x_t &= \alpha x_{t-1} + \beta \frac{x_{t-1}}{1 + x_{t-1}^2} + \gamma \cos(1.2(t-1)) + w_t, \end{aligned}$$

where $w_t \sim \mathcal{N}(0, W)$ and $v_t \sim \mathcal{N}(0, V)$. This benchmark nonlinear time series model has been used by Carlin et al. (1992) to test MCMC smoothing, by Gordon et al. (1993) to test the bootstrap filter, and by Briers et al. (2010) to test the forward-backward smoothing with known parameters. The new smoothing methods are tested on this model where the parameters $\theta = (\alpha, \beta, \gamma, W, V)$ are treated as unknown.

We generated $T = 100$ observations using parameter values $\alpha = 0.5$, $\beta = 25$, $\gamma = 8$, $V = 5$ and $W = 1$. Conjugate priors for the parameters are adopted and are similar to those given in Carlin et al. (1992), i.e. $(\alpha, \beta, \gamma)'|W \sim \mathcal{N}(\mathbf{b}_0, \mathbf{B}_0^{-1}W)$, $W \sim \mathcal{IG}(n_0, d_0)$ and $V \sim \mathcal{IG}(\nu_0, \delta_0)$, where $\mathbf{b}_0 = (0.5, 25, 8)'$, $\mathbf{B}_0^{-1} = \text{diag}(0.25^2, 10^2, 4^2)$, and $n_0 = 2$, $d_0 = 2$, $\nu_0 = 2$, $\delta_0 = 2$. The conjugate priors allow use of Storvik's algorithm for filtering and parameter learning. The sufficient statistics for the parameters are $s_t = (\mathbf{B}_t, \mathbf{b}_t, n_t, d_t, \nu_t, \delta_t)$ and the updating recursions are

$$\begin{aligned} \mathbf{B}_t &= \mathbf{B}_{t-1} + \mathbf{F}_t\mathbf{F}_t', & \mathbf{b}_t &= \mathbf{B}_t^{-1}(\mathbf{B}_{t-1}\mathbf{b}_{t-1} + \mathbf{F}_t x_t), \\ n_t &= n_{t-1} + 1/2, & d_t &= d_{t-1} + (\mathbf{b}_{t-1}'\mathbf{B}_{t-1}\mathbf{b}_{t-1} + x_t^2 - \mathbf{b}_t'\mathbf{B}_t\mathbf{b}_t)/2, \\ \nu_t &= \nu_{t-1} + 1/2, & \delta_t &= \delta_{t-1} + (y_t - x_t^2/20)^2/2, \end{aligned}$$

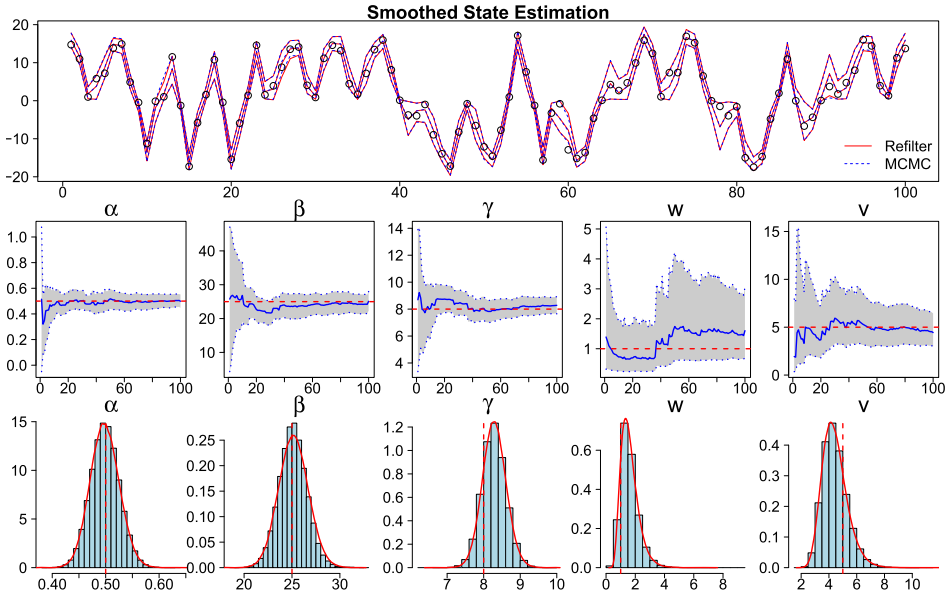

Figure 4: State and parameter estimation for the nonstationary growth model (see Section 4.2). Top row: smoothed means and 95% intervals for the states using the refiltering algorithm (solid lines) and long MCMC (dashed lines). Middle row: filtered means and 95% intervals for the parameters using Storvik (2002)'s algorithm. Bottom row: posterior distributions for the parameters at the last time step, $T = 100$, for the Storvik algorithm (histograms) and the long MCMC (density curves). The true parameter values are indicated by horizontal and vertical red lines in rows 2 and 3, respectively.

where $\mathbf{F}_t = (x_{t-1}, x_{t-1}/(1+x_{t-1}^2), \cos(1.2(t-1))'$. The parameter learning process and the posterior histograms of the parameters at time $T = 100$ are plotted in Figure 4. Notice from the figure that the 95% confidence bands for the parameters narrow quickly as time increases. From the histograms, we observe that the samples concentrate around the true parameter values. $N = 50000$ particles were used for Storvik's algorithm.

Furthermore, comparisons for the refiltering smoothing algorithm were made with $N_0 = 10,000$ and $n_0 = 1,000$, with a long MCMC using $N = 150,000$ iterations. The smoothing plot is also presented in Figure 4. The results from the two smoothing algorithms closely agree with each other.

Using 200 simulated datasets, Table 2 gives a summary of the overall performance of the three smoothing algorithms compared to a long MCMC. A significant decrease in the mean absolute error for the new smoothing methods relative to PLS is detected. The plot of the standardized absolute errors over time of the three smoothing algorithms (not shown) illustrates the same patterns as for the AR(1) model: the main improvement of the two new smoothing algorithms over PLS is evident at the beginning of the time

Method	Time (s)	Iterations M	Particles N	MAE* States	MAE* Params
Long MCMC	897	150000	–	–	–
MCMC	123	18796	–	0.174	0.412
PMMH	123	1246	500	0.146	0.178
PLS	123	–	5000	0.362	0.166
PLSa	127	–	2500	0.197	0.166
Refiltering	**123**	–	**10000/1000**	**0.104**	**0.166**

Table 2: Comparison of smoothing algorithms for the nonlinear growth model (see Section 4.2). Results are based on 200 simulated datasets.

series. By comparing the results from Refiltering, PMMH, and short MCMC, we notice that Refiltering dominates the latter two for both state and parameter estimation.

Note that for this model, the posterior distribution of the states is often multimodal, and it is difficult to distinguish between the positive and negative sign of the states based on the data. Thus, it is difficult to assign initial values for the states for the MCMC algorithm based only on the observations. With bad starting values for the states, the MCMC chain takes much longer to converge. In contrast, the SMC smoothing algorithms are not based on Markov chains and thus avoid issues with starting values.

4.3 Chaotic Model

The third example is the nonlinear Ricker model (Fasiolo, Pya, and Wood, 2016):

$$\begin{aligned} y_t &\sim Pois(\phi N_t), \\ N_t &= rN_{t-1}e^{-N_{t-1}+z_t}, \ z_t \sim \mathcal{N}(0, \sigma^2). \end{aligned}$$

This model is used in ecology, where N_t stands for the population density at time t, r is the growth rate, and y_t is the observed population size at time t. This model is characterized by its sensitivity to parameter variations: small increments in r will lead to significant oscillations in the likelihood function. As a result, the likelihood function can be intractable in certain areas of the parameter space and parameter estimation via maximum likelihood methods is challenging. Fasiolo, Pya, and Wood (2016) described the pathological likelihood function for this model and compared various maximum likelihood estimation and Bayesian parameter estimation approaches. A time series of 100 observations is generated from this model with true parameter values $r = e^{3.8}$, $\sigma^2 = 0.3$ and $\phi = 10$.

To estimate this model within our framework, we first make the transformations $x_t = \log(N_t)$ and $\mu = \log(r)$. The observation and evolution equations then become

$$\begin{aligned} y_t &\sim Pois(\phi e^{x_t}), \\ x_t &= \mu + x_{t-1} - e^{x_{t-1}} + z_t. \end{aligned}$$

The model includes three unknown parameters, $\theta = (\phi, \mu, \sigma^2)$. We assume conjugate priors, with $\phi \sim Gamma(a_0, b_0)$, $\mu|\sigma^2 \sim \mathcal{N}(m_0, c_0^{-1}\sigma^2)$ and $\sigma^2 \sim \mathcal{IG}(n_0, d_0)$, with

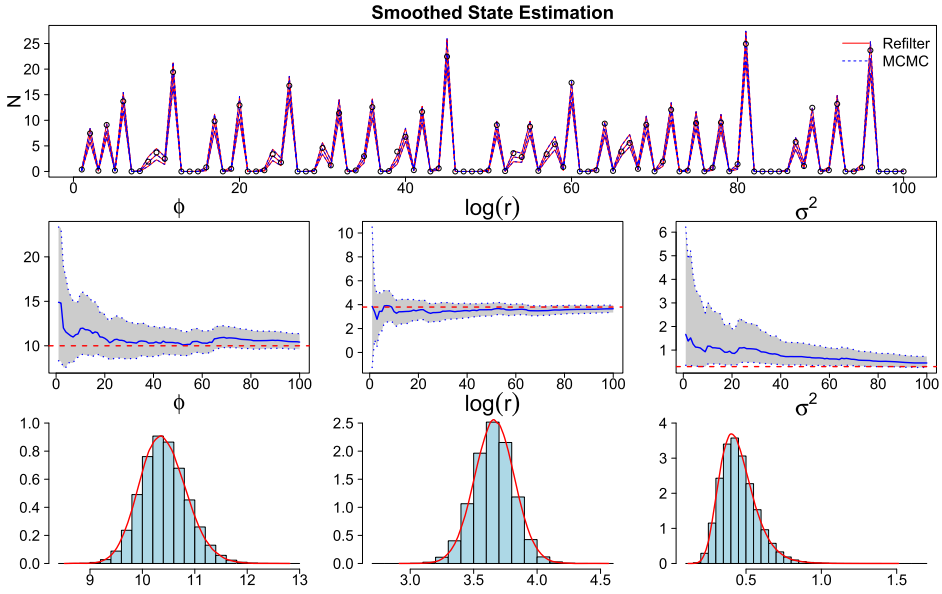

Figure 5: State and parameter estimation for the chaotic model (see Section 4.3). Top row: smoothed means and 95% intervals for the states using the refiltering algorithm (solid lines) and long MCMC (dashed lines). Middle row: filtered means and 95% intervals for the parameters using Storvik (2002)'s algorithm. Bottom row: posterior distributions for the parameters at the last time step for the Storvik algorithm (histograms) and the long MCMC (density curves). True parameter values are shown as horizontal lines in row 2.

$a_0 = 15, b_0 = 1, m_0 = 5, c_0 = 0.1, n_0 = 2, d_0 = 2$. The sufficient statistics are $s_t = (a_t, b_t, m_t, c_t, n_t, d_t)$ and the updating recursions are

$$\begin{aligned} a_t &= a_{t-1} + y_t, & b_t &= b_{t-1} + e^{x_t}, \\ c_t &= c_{t-1} + 1, & m_t &= c_t^{-1}(c_{t-1}m_{t-1} + x_t - x_{t-1} + e^{x_{t-1}}), \\ n_t &= n_{t-1} + 1/2, & d_t &= d_{t-1} + [c_{t-1}m_{t-1}^2 + (x_t - x_{t-1} + e^{x_{t-1}})^2 - c_t m_t^2]/2. \end{aligned}$$

The smoothed states, parameter learning bands and posterior distributions at time $T = 100$ are summarized in Figure 5. A total of $N = 100,000$ particles were used for the filtering. For the simulation, the true parameter values were learned quickly and the posterior distribution of the parameter converges to the true values. Figure 5 also provides a comparison of refiltering with $N_0 = 10,000$ and $n_0 = 1,000$ to a long MCMC with 150,000 iterations for smoothing. The smoothing results from the two methods shown in the figure are almost identical.

To make a comparison of the three SMC smoothing methods, 200 different simulations were performed. Plots of the MAE* values over time (not shown) were examined

Method	Time (s)	Iterations M	Particles N	MAE* States	MAE* Params
Long MCMC	504	150000	–	–	–
MCMC	69	18804	–	0.055	0.097
PMMH	69	633	500	0.132	0.197
PLS	64	–	4600	0.112	0.141
PLSa	69	–	2700	0.072	0.141
Refiltering	**69**	–	**10000/1000**	**0.048**	**0.141**

Table 3: Smoothing comparison for the chaotic model (see Section 4.3). Results are based on 200 simulated datasets.

and Table 3 presents numerical summaries based on the simulations. From the MAE* plot, similar patterns can be found for this case compared to the previous two examples: PLSa and refiltering dominate PLS early in the time series (up to about time $t = 80$), and the three methods coincide afterwards. From Table 3, a significant decrease in MAE* can be noticed for the two new methods compared to PLS.

When comparing Refiltering, MCMC and PMMH, we find that Refiltering dominates the other two methods in terms of state estimation. To summarize, the simulation results of the three models studied in this section show that Refiltering is as accurate or more accurate than MCMC and PMMH methods for both state and parameter estimation.

4.4 Marginal Likelihood Selection

As noted by Carvalho et al. (2010), the marginal likelihood can be computed trivially from the output of SMC-based Bayesian filtering and learning algorithms (e.g., Storvik, 2002; Carvalho et al., 2010). Define $\omega_t^j = p(y_t|x_t^j, \theta^j, \mathcal{M})$, where $(x_t^j, \theta^j) \sim p(x_t, \theta|y^{t-1}, \mathcal{M})$ for a given model $\mathcal{M}$. Then, the log marginal likelihood for model $\mathcal{M}$ is estimated by

$$\log(f(y|\mathcal{M})) \approx \sum_{t=1}^{T} \log\left(\sum_{j=1}^{N} \omega_t^j\right) - T\log(N).$$

By comparison, many different MCMC-based estimates of the marginal likelihood have been proposed. One of the most commonly used and straightforward methods is the harmonic mean estimator (Newton and Raftery, 1994), which can be computed based on the joint distribution of the data:

$$\log(f(y|\mathcal{M})) \approx \log\left(\frac{1}{\frac{1}{N}\sum_{j=1}^{N} \frac{1}{p(y|\psi^j, \mathcal{M})}}\right),$$

where $y = (y_1, \ldots, y_T)$ are the observations, N is the total number of MCMC iterations, and $\psi^j = (x^j, \theta^j), j = 1, \ldots, N$ are the posterior draws of the states and parameters.

An important advantage of SMC over MCMC is that the estimation of the marginal likelihood from SMC output is numerically stable. As shown in Table 4, in our simulation studies of Section 4, we found that the SMC-based marginal likelihood estimator

	AR(1) + Noise		Nonlinear Growth		Chaotic Model	
N	Storvik	MCMC	Storvik	MCMC	Storvik	MCMC
1,000	-176.44	-164.40	-212.91	-253.27	-286.34	-193.37
5,000	-176.71	-171.74	-212.51	-169.92	-284.88	-200.35
10,000	-176.71	-169.88	-211.66	-170.42	-284.96	-195.64
50,000	-176.87	-169.86	-212.45	-170.91	-285.16	-199.54
100,000	-176.91	-170.84	-212.10	-162.56	-284.98	-196.61
500,000	-176.88	-173.56	-212.30	-164.06	-285.10	-197.99

Table 4: For each of the models considered in Sections 4.1–4.3, this table provides comparisons of the log marginal likelihood computed using Storvik (2002) and MCMC for different Monte Carlo sizes (N). The results for each model are based on one dataset.

converges quickly as N increases, while for MCMC, the harmonic mean estimator is unstable even for N as large as 500,000 in all three models.

4.5 Stochastic Volatility Model for S&P 500 Returns

To illustrate the proposed methods on real data, we analyze daily returns on Standard & Poor's (S&P 500) index from January 2008–March 2009, during the financial crisis. We consider the following stochastic volatility model:

$$\begin{aligned} y_t &= \mu + \exp(x_t/2)\epsilon_t, \quad \epsilon_t \sim \mathcal{N}(0,1), \\ x_t &= \alpha + \beta x_{t-1} + \eta_t, \quad \eta_t \sim \mathcal{N}(0, W), \end{aligned}$$

where $y_t = \log(P_t/P_{t-1})$ are the daily returns and P_t are the daily prices on the S&P 500 index, μ is the expected return, and x_t is the unobserved log-variance state variable, which follows an AR(1) process with constant term α. The coefficient β measures the autocorrelation in the logged squared returns, and W is the so-called volatility of volatility. This model has been widely used to analyze financial time series with volatility clustering (for example, Jacquier, Polson, and Rossi, 1994; Kim, Shephard, and Chib, 1998). Here, we compare smoothing and parameter estimation results from the PLS, PLSa, Refiltering and MCMC algorithms.

Conjugate priors are assumed for the unknown parameters $\theta = (\mu, \alpha, \beta, W)$. Specifically, we assume $\mu \sim \mathcal{N}(\mu_0, \sigma_0^2)$, $(\alpha, \beta)'|W \sim \mathcal{N}(\mathbf{b}_0, \mathbf{B}_0^{-1}W)$, and $W \sim \mathcal{IG}(n_0, d_0)$. The hyperparameters are $\mu_0 = 0$, $\sigma_0^2 = 1$, $\mathbf{b}_0 = (0, 0.9)'$, $\mathbf{B}_0 = \text{diag}(1,1)$, $n_0 = 2$, $d_0 = 2$. The refiltering algorithm is run with $N = 10,000$ and $N_0 = 1000$. The parameter learning and state smoothing estimates are compared to the single-state updating MCMC scheme of Jacquier et al. (1994) and PMMH algorithms. Table 5 provides comparisons of the smoothing results for the S&P 500 returns for PLS, PLSa, Refiltering and PMMH. The first two columns give values of the number of iterations and number of particles respectively and for each method. We notice that in terms of the MAE for state estimation, Refiltering provides the lowest value. On the other hand, the MAE

Method	Time (s)	Iterations M	Particles N	MAE* States	MAE* Params
Long MCMC	70	1000000	–	–	–
MCMC	8	100000	–	0.035	0.021
PMMH-1000	269	1200	1000	0.066	0.258
PMMH-500	267	3300	500	0.055	0.133
PMMH-250	273	8000	250	0.056	0.192
PLS	270	–	1000	0.471	0.354
PLSa	267	–	500	0.282	0.354
Refiltering	**256**	–	**5000/1000**	**0.030**	**0.354**

Table 5: Smoothing results for the stochastic volatility model for daily S&P 500 returns (see Section 4.4). While parallelization was not used for the PLS, PLSa and Refiltering algorithms, it could be used to reduce run times as in the previous examples.

for the parameter estimates is the lowest for PMMH with 500 particles. The run times were set as near as possible for PMMH, PLS, PLSa and Refiltering.

The sequential learning plots in Figure 6 show an abrupt change in the parameters, particularly α and β, in September 2008. The timing corresponds to the days following the collapse of Lehman Brothers, when stock prices dropped and volatility increased sharply (i.e., the start of the financial crisis). Figure 7 shows the filtered and smoothed volatilities for each algorithm. These plots show clear evidence that PLS and MCMC do not match especially from September-November 2008, when the volatility changes abruptly. PLSa more closely follows the MCMC results and among Refiltering, PLS and PLSa, Refiltering is by far the most accurate relative to long MCMC. The smoothed posterior state estimates of stochastic volatility obtained with the PMMH smoother closely matches the estimates obtained with long MCMC.

Figures 8–10 in the Supplemental Material show results for a situation where $T = 3000$. The refiltering algorithm is the method that provides estimates of the log-volatility state that most closely resembles the ones obtained from a long MCMC.

5 Conclusions

In this paper, we have proposed two new particle smoothing algorithms that simultaneously deal with state and parameter learning in general state-space models. The first is a modification of the PLS algorithm of Carvalho et al. (2010), which includes an adjustment term to their backward resampling weights. The second, called Refiltering, is a two-step algorithm that includes a parameter learning step followed by a forward-backward algorithm for smoothing. The Refiltering algorithm is well suited for parallel implementation, since the smoothing step requires essentially no communication between processors.

We implemented the new methods on four examples: a benchmark autoregressive plus noise model, a nonlinear growth model, a chaotic model, and a stochastic volatility model from finance. We compared estimates for states and parameters with the

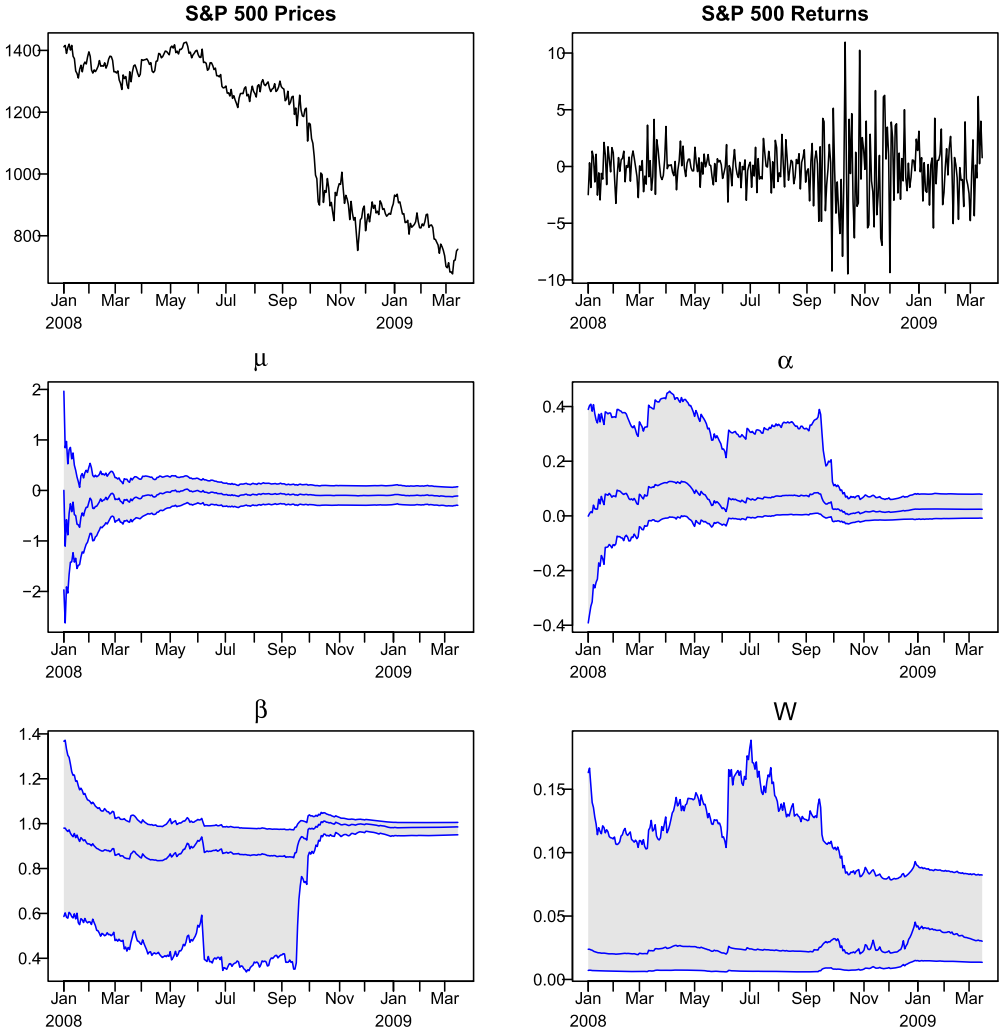

Figure 6: Parameter estimation for the stochastic volatility model for daily S&P 500 returns (see Section 4.4). Top row: Daily prices and returns on the S&P 500 index from January 2008 to March 2009. Middle and bottom rows: filtered medians and 95% intervals for the parameters μ, α, β and W. Results are based on Storvik's SIR algorithm with 50,000 particles.

widely-used PLS algorithm, Particle MCMC, and a full MCMC implementation. For all examples, the new smoothing methods showed significant improvement over PLS, and the Refiltering algorithm is shown to be highly competitive with smoothing algorithms based on MCMC and Particle MCMC methods. Overall, our proposed methods are quite general and apply to a wide class of nonlinear, non-Gaussian state-space models for retrospective state and parameter estimation.

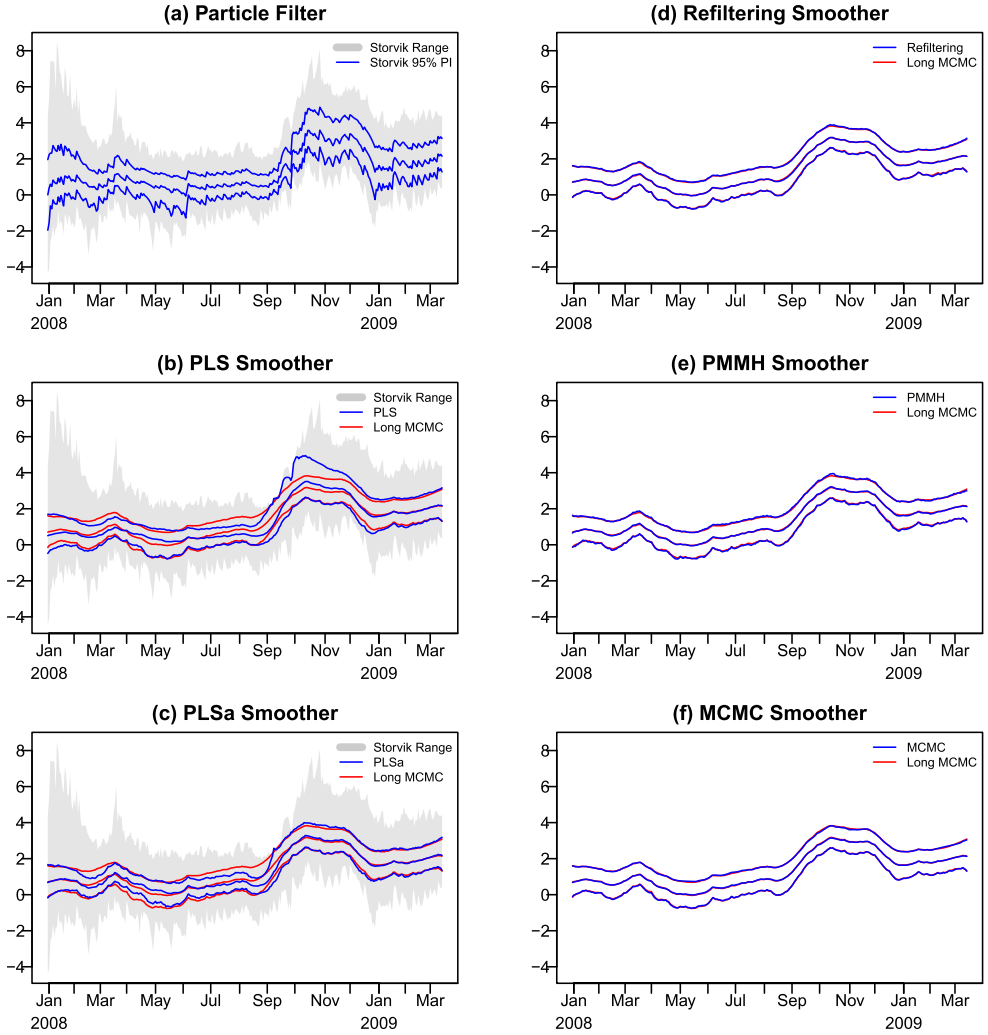

Figure 7: State estimation for the stochastic volatility model for daily S&P 500 returns, January 2008 – March 2009 (see Section 4.4). Panel (a) shows the filtered range, median and 95% intervals for x_t from Storvik's algorithm. Panels (b)–(f) show the smoothed mean and 95% intervals for x_t for the PLS, PLSa, Refiltering, PMMH, MCMC algorithms versus the long MCMC algorithm. Gray bands in (a)–(c) are the filtered range from Storvik's algorithm.

Finally, we note that all of the examples considered in this paper are low-dimensional, with state and observation that are one-dimensional. It is well known that particle filters (or sequential Monte Carlo methods based on importance sampling) fail with high-dimensional states and observations, leading to unbalanced weights and particle degen-

eracy (see Snyder, Bengtsson, Bickel, and Anderson, 2008). Similar degeneracy problems are likely to occur for high-dimensional parameters. An important area of current research is designing state and parameter estimation algorithms for high-dimensional models. We believe that "exact" methods based on importance sampling will fail, and suggest that approximate methods such as the ensemble Kalman filter (EnKF, Evensen, 1994; Katzfuss, Stroud, and Wikle, 2016) will provide more accurate and stable results. These methods are already widely used in geophysics, and EnKF state and parameter estimation methods have proved to be successful in many high-dimensional examples (e.g., Stroud and Bengtsson, 2007; Stroud, Stein, Lesht, Schwab, and Beletsky, 2010; Stroud, Katzfuss, and Wikle, 2017).

Supplementary Material

Supplementary Material of the Sequential Monte Carlo Smoothing with Parameter Estimation (DOI: 10.1214/17-BA1088SUPP; .pdf). Supplementary material A provides summaries of the algorithms (MCMC, Particle Filter and PMMH) referenced in the paper. Supplementary material B provides graphical summaries for different estimation methods for the stochastic volatility model with $T = 3000$.

References

Andrieu, C., Doucet, A., and Holenstein, R. (2010). "Particle Markov Chain Monte Carlo Methods (with discussion)." *Journal of the Royal Statistical Society, Series B (Statistical Methodology)*, 72(3): 269–342. MR2758115. doi: https://doi.org/10.1111/j.1467-9868.2009.00736.x. 1138

Briers, M., Doucet, A., and Maskell, S. (2010). "Smoothing Algorithms for State-Space Models." *Annals of the Institute of Statistical Mathematics*, 62(1): 61–89. MR2577439. doi: https://doi.org/10.1007/s10463-009-0236-2. 1138, 1149

Carlin, B. P., Polson, N. G., and Stoffer, D. S. (1992). "A Monte Carlo Approach to Nonnormal and Nonlinear State-Space Modeling." *Journal of the American Statistical Association*, 87(418): 493–500. 1138, 1149

Carter, C. K. and Kohn, R. (1994). "On Gibbs Sampling for State Space Models." *Biometrika*, 81(3): 541–553. 1138, 1145

Carvalho, C. M., Johannes, M. S., Lopes, H. F., and Polson, N. G. (2010). "Particle Learning and Smoothing." *Statistical Science*, 25(1): 88–106. 1137, 1138, 1142, 1146, 1148, 1153, 1155

de Jong, P. and Shephard, N. (1995). "The Simulation Smoother for Time Series Models." *Biometrika*, 82(2): 339–350. MR1354233. doi: https://doi.org/10.1093/biomet/82.2.339. 1138

Evensen, G. (1994). "Sequential Data Assimilation with a Nonlinear Quasi-Geostrophic Model using Monte Carlo Methods to Forecast Error Statistics." *Journal of Geophysical Research: Oceans*, 99(C5): 10143–10162. 1158

Fasiolo, M., Pya, N., and Wood, S. N. (2016). "A Comparison of Inferential Methods for Highly Non-Linear State Space Models in Ecology and Epidemiology." *Statistical Science*, 31(1): 96–118. MR3458595. doi: https://doi.org/10.1214/15-STS534. 1151

Fearnhead, P. (2002). "Markov Chain Monte Carlo, Sufficient Statistics, and Particle Filters." *Journal of Computational and Graphical Statistics*, 11(4): 848–862. 1138

Fearnhead, P., Wyncoll, D., and Tawn, J. (2010). "A Sequential Smoothing Algorithm with Linear Computational Cost." *Biometrika*, 97(2): 447–464. 1138

Frühwirth-Schnatter, S. (1994). "Data Augmentation and Dynamic Linear Models." *Journal of Time Series Analysis*, 15(2): 183–202. 1138, 1145

Gamerman, D. (1998). "Markov Chain Monte Carlo for Dynamic Generalized Linear Models." *Biometrika*, 85: 215–227. MR1627273. doi: https://doi.org/10.1093/biomet/85.1.215. 1138

Geweke, J. and Tanizaki, H. (2001). "Bayesian Estimation of State-Space Models using the Metropolis–Hastings Algorithm within Gibbs Sampling." *Computational Statistics and Data Analysis*, 37(2): 151 – 170. 1138

Godsill, S. J., Doucet, A., and West, M. (2004). "Monte Carlo Smoothing for Nonlinear Time Series." *Journal of the American Statistical Association*, 99(465): 156–168. 1138, 1142, 1145, 1146

Gordon, N. J., Salmond, D. J., and Smith, A. F. M. (1993). "Novel Approach to Nonlinear/Non-Gaussian Bayesian State Estimation." *IEE Proceedings F, Radar and Signal Processing*, 140(2): 107–113. 1137, 1140, 1149

Jacquier, E., Polson, N. G., and Rossi, P. E. (1994). "Bayesian Analysis of Stochastic Volatility Models." *Journal of Economic and Business Statistics*, 12: 371–417. 1154

Kantas, N., Doucet, A., Singh, S. S., Maciejowski, J., and Chopin, N. (2015). "On Particle Methods for Parameter Estimation in State-Space Models." *Statistical Science*, 30(3): 328–351. 1138

Katzfuss, M., Stroud, J. R., and Wikle, C. K. (2016). "Understanding the Ensemble Kalman Filter." *The American Statistician*, 70: 350–357. 1158

Kim, S., Shephard, N., and Chib, S. (1998). "Stochastic Volatility: Likelihood Inference and Comparison with ARCH Models." *Review of Economic Studies*, 65: 361–393. 1154

Kitagawa, G. (1996). "Monte Carlo Filter and Smoother for Non-Gaussian Nonlinear State Space Models." *Journal of Computational and Graphical Statistics*, 5(1): 1–25. 1138

Kitagawa, G. (1998). "A Self-Organizing State-Space Model." *Journal of the American Statistical Association*, 93(443): 1203–1215. 1137, 1138, 1140

Kitagawa, G. and Sato, S. (2001). "Monte Carlo Smoothing and Self-Organising State-Space Model." In *Sequential Monte Carlo Methods in Practice*, Statistics for Engineering and Information Science, 177–195. Springer:New York. MR1847792. 1140

Liu, J. and West, M. (2001). "Combined Parameter and State Estimation in Simulation-Based Filtering." In Doucet, A., de Freitas, N., and Gordon, N. (eds.), *Sequential Monte Carlo Methods in Practice*, Statistics for Engineering and Information Science, 197–223. Springer:New York. MR1847793. 1138, 1140, 1144, 1145

Lopes, H. F., Carvalho, C. M., Johannes, M. S., and Polson, N. G. (2011). "Particle Learning for Sequential Bayesian Computation (with discussion)." In Bernardo, J. M., Bayarri, M. J., Berger, J. O., Dawid, A. P., Heckerman, D., Smith, A. F. M., and West, M. (eds.), *Bayesian Statistics 9*, 317–360. Oxford University Press: Oxford. 1146

Lopes, H. F. and Tsay, R. S. (2011). "Particle filters and Bayesian inference in financial econometrics." *Journal of Forecasting*, 30(1): 168–209. MR2758809. doi: https://doi.org/10.1002/for.1195. 1138

Newton, M. A. and Raftery, A. E. (1994). "Approximate Bayesian Inference with the Weighted Likelihood Bootstrap (with discussion)." *Journal of the Royal Statistical Society, Series B (Statistical Methodology)*, 56(1): 3–48. 1153

Niemi, J. and West, M. (2010). "Adaptive Mixture Modeling Metropolis Methods for Bayesian Analysis of Nonlinear State-Space Models." *Journal of Computational and Graphical Statistics*, 19(2): 260–280. 1138

Pitt, M. and Shephard, N. (1999). "Filtering via Simulation: Auxiliary Particle Filters." *Journal of the American Statistical Association*, 94(446): 590–599. 1137

Polson, N. G., Stroud, J. R., and Müller, P. (2008). "Practical Filtering with Sequential Parameter Learning." *Journal of the Royal Statistical Society, Series B (Statistical Methodology)*, 70(2): 413–428. MR2424760. doi: https://doi.org/10.1111/j.1467-9868.2007.00642.x. 1146

Scipione, C. M. and Berliner, L. M. (1992). "Bayesian Inference in Nonlinear Dynamical Systems." Technical report, Ohio State University, Department of Statistics. 1138

Scott, S. L. (2002). "Bayesian Methods for Hidden Markov Models: Recursive Computing in the 21st Century." *Journal of the American Statistical Association*, 97: 337–351. 1146

Shephard, N. and Pitt, M. K. (1997). "Likelihood Analysis of Non-Gaussian Measurement Time Series." *Biometrika*, 84: 653–667. 1138

Silverman, B. (1986). *Density Estimation for Statistics and Data Analysis*. Chapman Hall/CRC Press. 1144

Snyder, C., Bengtsson, T., Bickel, P., and Anderson, J. (2008). "Obstacles to High-Dimensional Particle Filtering." *Monthly Weather Review*, 136: 4629–4640. 1158

Storvik, G. (2002). "Particle Filters for State-Space Models with the Presence of Unknown Static Parameters." *IEEE Transactions on Signal Processing*, 50(2): 281–289. 1138, 1141, 1146, 1147, 1150, 1152, 1153, 1154

Stroud, J. R. and Bengtsson, T. (2007). "Sequential State and Variance Estimation within the Ensemble Kalman Filter." *Monthly Weather Review*, 135: 3194–3208. 1158

Stroud, J. R., Katzfuss, M., and Wikle, C. K. (2017). "A Bayesian Adaptive Ensemble Kalman Filter for Sequential State and Parameter Estimation." *Monthly Weather Review*. 1158

Stroud, J. R., Müller, P., and Polson, N. G. (2003). "Nonlinear State-Space Models with State-Dependent Variances." *Journal of the American Statistical Association*, 98(462): 377–386. 1138

Stroud, J. R., Stein, M. L., Lesht, B. M., Schwab, D. J., and Beletsky, D. (2010). "An Ensemble Kalman Filter and Smoother for Satellite Data Assimilation." *Journal of the American Statistical Association*, 105: 978–990. 1158

Yang, B., Stroud, J. R., and Huerta, G. (2017). "Supplementary Material for Sequential Monte Carlo Smoothing with Parameter Estimation." *Bayesian Analysis*. doi: https://doi.org/10.1214/17-BA1088SUPP. 1140

Acknowledgments

J. Stroud and B. Yang gratefully acknowledge funding from the McDonough School of Business. J. Stroud gratefully acknowledges funding from the Stallkamp Teaching Skills Program Fund at Georgetown University. We thank the Editor-in-Chief, Editor, Associate Editor, and two referees for excellent comments and suggestions, which greatly improved the manuscript. We also thank Dave Higdon and Refik Soyer for helpful comments. See Supplement A and Supplement B for the supplementary material.