TI 2025-034/III
Tinbergen Institute Discussion Paper

Simulation Smoothing for State Space Models: An Extremum Monte Carlo Approach

Karim Moussa[1]

1 Vrije Universiteit Amsterdam, Tinbergen Institute

Tinbergen Institute is the graduate school and research institute in economics of Erasmus University Rotterdam, the University of Amsterdam and Vrije Universiteit Amsterdam.

Contact: discussionpapers@tinbergen.nl

More TI discussion papers can be downloaded at https://www.tinbergen.nl

Tinbergen Institute has two locations:

Tinbergen Institute Amsterdam
Gustav Mahlerplein 117
1082 MS Amsterdam
The Netherlands
Tel.: +31(0)20 598 4580

Tinbergen Institute Rotterdam
Burg. Oudlaan 50
3062 PA Rotterdam
The Netherlands
Tel.: +31(0)10 408 8900

Simulation Smoothing for State Space Models: An Extremum Monte Carlo Approach

K. Moussa*

Vrije Universiteit Amsterdam and Tinbergen Institute, the Netherlands

May 14, 2025

Abstract

This paper introduces a novel approach to simulation smoothing for nonlinear and non-Gaussian state space models. It allows for computing smoothed estimates of the states and nonlinear functions of the states, as well as visualizing the joint smoothing distribution. The approach combines extremum estimation with simulated data from the model to estimate the conditional distributions in the backward smoothing decomposition. The method is generally applicable and can be paired with various estimators of conditional distributions. Several applications to nonlinear models are presented for illustration. An empirical application based on a stochastic volatility model with stable errors highlights the flexibility of the approach.

Keywords: Amortized inference, Fixed-interval smoothing, Importance sampling, Latent variables, Stable distribution, Stochastic volatility.

*E-mail: k.moussa@vu.nl

1 Introduction

The estimation of variables subject to measurement noise has a long history in statistics, tracing back at least to Galileo's concept of random observational errors (Galilei 1632; Hald 1986). When the data are ordered in time, the problem is often formulated using state space models (SSMs). These models treat observations as noisy measurements of latent states, which govern the dynamics of the system under study. Applications of SSMs are ubiquitous, with numerous examples in economics and finance, neuroscience, epidemiology, astronomy, target tracking, and many other fields (Durbin and Koopman 2012; Chopin and Papaspiliopoulos 2020).

Given the SSM, an important objective is to perform inference on the unobserved states. We focus on *simulation smoothing* (e.g., De Jong and Shephard 1995; Durbin and Koopman 2002; Godsill, Doucet, and West 2004; McCausland, Miller, and Pelletier 2011), where the goal is to draw paths of the states conditional on the observed data. The simulated paths can be used to compute estimates of the states, including their conditional means, quantiles, and other characteristics of the so-called smoothing distribution. Furthermore, the paths provide a means to visualize and study the joint behavior of the states given the data. As noted by Godsill et al. (2004, p. 165), "*Generating sample realizations is the most efficient, effective, and intuitive approach to studying complicated multivariate joint distributions.*"

This paper introduces a novel approach to simulation smoothing for nonlinear and non-Gaussian SSMs that is based on extremum Monte Carlo (XMC; Moussa, Blasques, and Koopman 2023). The method combines extremum estimation (Amemiya 1985) with simulated data from the model to estimate the conditional distributions in the backward decomposition of the smoothing distribution (Carter & Kohn 1994; Frühwirth-Schnatter 1994). The approach is generally applicable, with the main requirement being the ability to simulate from the SSM of interest. It can be implemented with various estimators of conditional distributions, each leading to a different version of the simulation smoother with its own statistical and computational properties.

The rest of this paper is organized as follows. Section 2 describes the problem of simulation smoothing within the SSM framework. Section 3 introduces the XMC simulation smoother. Section 4 discusses the use of importance sampling to refine the smoothed estimates and paths. Section 5 presents applications to illustrate the simulation smoother. Section 6 provides discussion. The appendix contains supplementary material.

2 State Space Framework and Backward Sampling

Using $x_t \in \mathbb{R}^{N_x}$ to denote the state vector at time t and $y_t \in \mathbb{R}^{N_y}$ for the corresponding vector of measurements (or observations), we consider the general SSM defined by

$$\begin{aligned} x_{t+1} &= s_t(x_t, \varepsilon_t^x), & x_1 &\sim p(x_1), \\ y_t &= m_t(x_t, \varepsilon_t^y), & (\varepsilon_t^x, \varepsilon_t^y) &\sim p(\varepsilon_t^x, \varepsilon_t^y), \end{aligned} \tag{1}$$

for discrete times $t = 1, \ldots, T$, where $T \in \mathbb{N}$ is the length of the time series. Here, s_t and m_t denote the state transition and measurement functions, respectively, and $p(z)$ represents the probability density or mass function of a random variable z. These functions may depend on static parameters, which are assumed to be fixed and given. The noise terms ε_t^x and ε_t^y are assumed to be mutually and serially independent, and independent of the initial state x_1.

In this context, simulation smoothing refers to drawing paths of the states conditional on all available observations,

$$x_{1:T} \sim p(x_{1:T}|y_{1:T}). \tag{2}$$

This task is often performed by utilizing the following backward decomposition of the smoothing distribution,

$$p(x_{1:T}|y_{1:T}) = \prod_{t=1}^{T} p(x_t|\mathcal{F}_t), \qquad \mathcal{F}_t = \begin{cases} y_{1:T} & \text{if} \quad t = T, \\ (y_{1:t}, x_{t+1}) & \text{if} \quad t < T. \end{cases} \tag{3}$$

The above factorization follows from Bayes' rule, where the independence assumptions on the noise terms ensure that the states follow a Markov process and future measurements can be discarded given x_{t+1}. As a result, simulation smoothing can be accomplished by sequentially drawing from the components $p(x_t|\mathcal{F}_t)$, starting at the final time T and proceeding backward in time. This yields the backward sampling procedure in Algorithm 1.

Algorithm 1 Backward Sampling for Simulation Smoothing

To draw a path $x_{1:T}$ from the joint smoothing density $p(x_{1:T}|y_{1:T})$:

1. **Draw states at time** T.
   Draw $x_T \sim p(x_T|y_{1:T})$.
2. **Draw remaining states moving backward in time**.
   For $t = T-1, \ldots, 1$:
   Draw $x_t \sim p(x_t|y_{1:t}, x_{t+1})$.

The backward sampling approach is directly applicable when the measurement and state transition functions m_t and s_t in (1) are linear, and all distributions involved are Gaussian (Carter & Kohn 1994; Frühwirth-Schnatter 1994). However, for nonlinear and non-Gaussian SSMs, which are common in practice (e.g., Doucet, De Freitas, & Gordon 2001), it is generally not possible to sample directly from the components $p(x_t|\mathcal{F}_t)$ in (3). Consequently, approximations are required.

In this more general setting, simulation smoothing is typically carried out using Markov chain Monte Carlo (MCMC) or sequential Monte Carlo (SMC) methods. These methods are widely used, but they can be computationally intensive, particularly when a large number of state paths is required. For example, most SMC smoothing algorithms have a complexity of $O(M^2)$, with M the number of particles (Chopin & Papaspiliopoulos 2020, Ch. 12). Additionally, both MCMC and SMC methods involve repeated evaluations of the density functions $p(y_t|x_t)$ and $p(x_t|x_{t-1})$, which can be computationally expensive or unavailable in some models (e.g., Lombardi and Calzolari 2009; Chopin and Papaspiliopoulos 2020). The simulation smoother presented in the following section relies on XMC in order to address some of these challenges.

3 Simulation Smoothing by Extremum Monte Carlo

3.1 Background

To settle ideas, we begin by discussing the idea behind the simulation smoother in a simplified setting. Consider two random variables, X and Y, and suppose our aim is to estimate the conditional density function $p(X|Y)$. Assume we can draw N cases of both variables,

$$X^{(i)},\ Y^{(i)}, \qquad i = 1, \ldots, N,$$

Let $\mathbb{F}_N$ denote a set of conditional densities $f(X|Y)$ from which sampling is possible, and let L be a corresponding loss function. The conditional density $p(X|Y)$ can then be estimated via the extremum estimator

$$\widehat{f}^N \in \underset{f \in \mathbb{F}_N}{\arg\min} \frac{1}{N} \sum_{i=1}^{N} L\left(X^{(i)}, Y^{(i)}, f\right). \tag{4}$$

As a simple example, consider the parametric space of Gaussian conditional densities

$$\mathbb{F}_N = \left\{ f_{\mathrm{G}}\left(X;\, a + bY,\, \sigma^2\right) \,\middle|\, a, b \in \mathbb{R},\, \sigma^2 > 0 \right\},$$

where $f_{\mathrm{G}}(X; \mu, \sigma^2)$ denotes the Gaussian density with mean μ and variance σ^2. If we choose the loss function to be the negative log density,

$$L\left(X^{(i)}, Y^{(i)}, f\right) = -\log f\left(X^{(i)} | Y^{(i)}\right),$$

then the estimator in (4) corresponds to the maximum likelihood estimator.

Once the conditional density has been estimated, it can be used to generate D draws conditional on a specific value y of Y,

$$X^{[j]} \sim \widehat{f}^N(X|y) \approx p(X|Y = y), \qquad j = 1, \ldots, D.$$

An elementary version of the XMC simulation smoother is obtained by applying the above procedure for $t = T, \ldots, 1$ with $X = x_t$ and $Y = \mathcal{F}_t$ to perform backward sampling, where $\mathcal{F}_t$ is the information set at time t defined in (3). However, several key modifications are necessary to enhance its practical utility. First, the Gaussian conditional density estimator from the example may not be suitable for the given application. To broaden the applicability, the actual simulation smoother employs general (semi-)nonparametric estimators. Second, since the information set $\mathcal{F}_t$ contains the observations $y_{1:t}$, the approach would be inefficient when t is large. To address this, the covariates are restricted to a subset $\mathcal{C}_t \subseteq \mathcal{F}_t$. Lastly, when the covariates $\mathcal{C}_t$ correspond to rolling windows, it becomes possible to reuse previously estimated density functions at different time points, which can lead to large computational savings. The details of these modifications are discussed in the next sections.

3.2 Simulation Smoothing Algorithm

Algorithm 2 presents the XMC simulation smoother. The algorithm takes as input a given SSM, a set $\mathbb{F}_N$ of conditional densities, and a corresponding loss function L. The set $\mathbb{F}_N$ may also include probability mass functions, characteristic functions, cumulative distribution functions, and other suitable representations of a conditional distribution, in which case the method remains directly applicable. The output of the algorithm consists of D paths drawn from the estimated smoothing distribution, which are generated in three steps: simulation, fitting, and backward sampling.

In the first step, the SSM is used to simulate N paths of the latent states $x_{1:T}^{(i)}$ and corresponding measurements $y_{1:T}^{(i)},\ i = 1, \ldots, N$. The simulated data are then split into training and validation samples, with $c_{\text{val}} \in (0, 1)$ representing the fraction allocated to the validation sample. In the second step, these data are used to fit the simulation smoother to the given SSM. This is done by solving the optimization problem in (5), first at times T and $T - 1$ for various candidate values of the tuning parameters (regularization), which are present in most estimators of conditional distributions. Using the selected tuning

Algorithm 2 Extremum Monte Carlo Simulation Smoother

1. **Simulate**: Use the SSM in (1) to simulate N paths of the states and observations,

$$x_{1:T}^{(i)},\ y_{1:T}^{(i)}, \qquad i = 1, \ldots, N.$$

2. **Fit**:

   (a) *Split data*: Set $c_{\text{val}} \in (0, 1)$ and split the data into training and validation samples with sizes

$$N_{\text{tr}} = N - N_{\text{val}} \qquad \text{and} \qquad N_{\text{val}} = [c_{\text{val}} N].$$

   (b) *Regularization*: For a set of candidate tuning parameters, perform the following extremum estimation at times $t = T$ and $t = T - 1$:

$$\widehat{f}_t^N \in \underset{f \in \mathbb{F}_N}{\arg\min} \frac{1}{N_{\text{tr}}} \sum_{i=1}^{N_{\text{tr}}} L\left(x_t^{(i)}, \mathcal{C}_t^{(i)}, f\right), \tag{5}$$

   with covariates $\mathcal{C}_t^{(i)} \subseteq \mathcal{F}_t^{(i)}$, where $\mathcal{F}_t^{(i)}$ represents the information set in (3) for the i-th simulated sample, $\mathbb{F}_N$ is a set of conditional densities $f(x_t|\mathcal{C}_t)$, and L is a loss function. Select the tuning parameters that minimize the average loss for the validation sample.

   (c) *Estimation*: Use the selected tuning parameters to perform the estimation in (5) for all times $t = T, \ldots, 1$ to obtain the function estimates $\{\widehat{f}_t^N\}_{t=1}^T$.

3. **Backward sampling**: Draw D smoothed paths of the states by using Algorithm 1 with $p(x_t|\mathcal{F}_t) \coloneqq \widehat{f}_t^N(x_t|\mathcal{C}_t)$ for $t = T, \ldots, 1$, with covariates $\mathcal{C}_t$ based on the actual data:

$$x_{1:T}^{[j]} \sim \prod_{t=1}^{T} \widehat{f}_t^N\left(x_t|\mathcal{C}_t\right) \approx p(x_{1:T}|y_{1:T}), \qquad j = 1, \ldots, D.$$

parameters, the optimization is performed at all time points to obtain the density estimates $\widehat{f}_t^N(x_t|\mathcal{C}_t) \approx p(x_t|\mathcal{F}_t)$ for $t = T, \ldots, 1$. The final step applies backward sampling using these estimates to generate D smoothed paths. We now discuss some of these steps in more detail.

In the second step, the covariates used in the optimizations, $\mathcal{C}_t^{(i)} \subseteq \mathcal{F}_t^{(i)}$, are defined to include the $W \in \{1, \ldots, T\}$ measurements nearest to time t, where the window size W is considered a tuning parameter. Based on the information sets in (3), we have

$$\mathcal{C}_t = \begin{cases} y_{\underset{\sim}{t}:t} & \text{if} \quad t = T, \\ \left(y_{\underset{\sim}{t}:t}, x_{t+1}\right) & \text{if} \quad t < T, \end{cases} \qquad \underset{\sim}{t} = \max\left\{t - W + 1, 1\right\}, \tag{6}$$

for $t = T, \ldots, 1$. The superscripts (i) indicate that the elements of $\mathcal{C}_t^{(i)}$ correspond to the

i-th simulated path from the training sample, where $i = 1, \ldots, N_{\text{tr}}$. Thus, $\mathcal{C}_t^{(i)} = y_{\underset{\sim}{t:t}}^{(i)}$ when $t = T$, and $\mathcal{C}_t^{(i)} = \left(y_{\underset{\sim}{t:t}}^{(i)}, x_{t+1}^{(i)}\right)$ when $t < T$.

The window size is selected in the regularization step, along with other tuning parameters specific to the chosen conditional density estimator. This is done by minimizing the average loss for the validation sample over a set of candidate tuning parameters generated using a Bayesian optimization procedure (Bergstra, Yamins, & Cox 2013). To save computations, the tuning parameters optimized at time $T-1$ are reused for times $t < T-1$, as the minimization problems in (5) are similar. At time T, however, there is no future state to condition on, so different tuning parameters may be required.

In the estimation step, the selected tuning parameters are used estimate the conditional densities for all times $t = T, \ldots, 1$. The resulting estimates $\left\{\widehat{f}_t^N\right\}_{t=1}^T$ are then used in the final step to perform backward sampling, where they are evaluated with covariates $\mathcal{C}_t$ based on the actual measurements, which is indicated by the absence of a superscript. Specifically, Algorithm 1 is applied with $p(x_t|\mathcal{F}_t) \coloneqq \widehat{f}_t^N(x_t|\mathcal{C}_t)$ for $t = T, \ldots, 1$, to draw D paths $x_{1:T}^{[j]},\ j = 1, \ldots, D$, from the estimated smoothing distribution.

For each choice of conditional density (or distribution) estimator, Algorithm 2 defines a corresponding version of the simulation smoother. To ensure broad applicability, the estimator must be sufficiently general to approximate most smoothing distributions of practical interest. Additionally, it should be capable of handling a potentially large number of covariates, and sampling from the estimated distributions should be computationally inexpensive to maintain the efficiency of the simulation smoother. In the applications, the simulation smoother will be illustrated using the following two widely used estimators.

The first estimator we consider is the mixture density network (Bishop 1994, 2006), which is a mixture of normal densities where the parameters are the output of a neural network. In our case, the covariates $\mathcal{C}_t$ serve as the input of the network. The universal approximation properties of neural networks and mixtures of normal densities enable wide applicability, and sampling from the estimated distribution is straightforward. We allow the

number of layers to exceed one and treat this as a tuning parameter, consistent with modern applications (Zen and Senior 2014; Zhang et al. 2020; He and Wang 2020). This results in the deep mixture density network (DMDN) version of the XMC simulation smoother. Further background on the DMDN estimator can be found in Appendix A.1.

The second estimator relies on regression forests (RFs), which are ensembles of regression trees (Breiman 2001; Meinshausen 2006). It builds on the observation by Meinshausen (2006) that the predictions of a random forest can be expressed as a weighted average of the response values in the training sample, where the weights enable estimation of the full conditional distribution. In our case, the weight assigned to each response value $x_t^{(i)}$ represents the probability $P(x_t = x_t^{(i)}|\mathcal{C}_t)$ for $i = 1, \ldots, N$, providing a discrete approximation to the target distribution when the states are continuous. Simulation is performed via weighted resampling with replacement. This approach to conditional simulation was recently applied by van der Westhuizen, Heuvelink, and Hofmeyr (2023) in the context of digital soil mapping. We use it for estimation and simulation of the conditional distributions in the backward decomposition in (3), leading to the RF version of the XMC simulation smoother. Additional details on the RF estimator are provided in Appendix A.2.

Notably, Algorithm 2 can be applied with other estimators of conditional distributions, including parametric approaches, kernel density estimators (KDEs), Gaussian processes (Williams & Rasmussen 2006), normalizing flows (Kobyzev, Prince, & Brubaker 2020), and other methods that allow for simulation.

3.3 Steady State Approach

In many cases, the density estimates can be reused for simulation at different time points, allowing the estimation costs to be amortized (Stuhlmüller, Taylor, & Goodman 2013). This can lead to substantial computational savings, particularly for long time series. By analogy to the XMC filter (Moussa et al. 2023), we refer to this extension of Algorithm 2 as the *steady state* approach to simulation smoothing.

As shown in Appendix B.1, the range of time points at which density functions can be reused is given by

$$W \leq t \leq T-1. \tag{7}$$

The upper bound reflects the requirement of a future state for conditioning, while the lower bound ensures that W past observations are available. A suitable time t_{ss} for the density function $\widehat{f}_{t_{\text{ss}}}^{N}$ to be reused can be determined in several ways. The choice $t_{\text{ss}} = T-1$ minimizes computations, but it assumes that the density estimate at time $T-1$ generalizes well to the previous time points. Alternatively, the validation sample can be used to iteratively evaluate the suitability of the density estimates during the estimation step in Algorithm 2. Further details can be found in Appendix B.2.

3.4 Computational Considerations

The runtime of Algorithm 2 can be reduced by parallelizing the computations in the estimation step, as the minimizations in (5) can be carried out simultaneously for different times. However, the sequential approach can be useful when the function estimates are memory-intensive. In such cases, the sampling step in Algorithm 2 can be performed immediately after the estimation step for each time t, discarding the function estimates directly after each draw. This approach requires storing only a single density estimate at a time.

Table 1 presents current estimates of the computational complexity for the XMC simulation smoother in Algorithm 2, where N denotes the number of simulated paths used for fitting and D the number of draws from the estimated smoothing distribution. The complexities labeled "Estimation" and "Sampling" correspond to a single time iteration in the estimation and backward sampling steps, respectively. The complexity estimates for the estimation step are based on Chapters 9.7 and 11.10 of Hastie, Tibshirani, and Friedman (2009).

For the RF estimator, a single draw at time t has a complexity of $O(N)$, corresponding to resampling from a multinomial distribution (e.g., Li, Bolic, & Djuric 2015). For $t < T$, each of the D draws is based on a different multinomial distribution, as the j-th draw is

Table 1: Current estimates of the computational complexity for the XMC simulation smoother in Algorithm 2, where N denotes the number of simulated paths used for fitting and D the number of draws from the estimated smoothing distribution. The complexities labeled "Estimation" and "Sampling" correspond to a single time iteration in the estimation and backward sampling steps.

Version	Estimation	Sampling	Simulation smoother
DMDN	$O(N)$	$O(D)$	$O\big[T(N+D)\big]$
RF	$O\big[N\log(N)\big]$	$O(ND)$	$O\big[TN(\log(N)+D)\big]$

conditional on the future state $x_{t+1}^{[j]},\ j = 1, \ldots, D$. Thus, the complexity at a single time step is $O(ND)$, and the overall complexity is $O\big[TN(\log(N) + D)\big]$.

In contrast, for the DMDN estimator, the costs of sampling are independent of N because the training data are used solely for estimating the neural network. Consequently, simulation has a complexity of $O(D)$, resulting in an overall complexity of $O\big[T(N+D)\big]$ for the DMDN-XMC simulation smoother. This makes large values of both N and D feasible, enabling accurate simulation smoothing in a broad range of practical applications.

4 Importance Sampling

For conciseness, let $x = x_{1:T}$ and $y = y_{1:T}$ denote the full paths of the states and measurements. Suppose the joint density $p(x, y)$ can be evaluated for any pair (x, y), and the estimated smoothing density

$$\widehat{f}(x|y) = \prod_{t=1}^{T} \widehat{f}_t^N(x_t|\mathcal{C}_t^N), \qquad N \in \mathbb{N},$$

is positive on the support of $p(x|y) = p(x, y)/p(y)$. In this case, importance sampling techniques (e.g., Durbin & Koopman 2012, Ch. 11) can be used to refine the smoothed estimates and paths. When the focus is on estimating some function g of the states, importance sampling is based on the identity

$$\mathbb{E}[g(x)|y] = \int g(x)p(x|y)dx = \int g(x)\frac{p(x|y)}{\widehat{f}(x|y)}\widehat{f}(x|y)dx \propto \int g(x)\frac{p(x,y)}{\widehat{f}(x|y)}\widehat{f}(x|y)dx.$$

Thus, the expectation can be estimated via Monte Carlo integration using D draws from the importance density $\widehat{f}$,

$$\sum_{i=1}^{D} \widetilde{w}_i \, g\left(x^{[i]}\right) \approx \mathbb{E}[g(x)|y], \tag{8}$$

where $x^{[i]} \sim \widehat{f}(x|y),\ i = 1, \ldots, D$, and the normalized importance weights $\widetilde{w}_i$ are given by

$$\widetilde{w}_i = \frac{w_i}{\sum_{j=1}^{D} w_j}, \qquad \text{with} \qquad w_i = \frac{p(x^{[i]}, y)}{\widehat{f}(x^{[i]}|y)}. \tag{9}$$

Under the mild assumption that the expectation $\mathbb{E}[g(x)|y]$ and the marginal likelihood $p(y)$ exist, Theorem 1 in Geweke (1989) ensures that the estimator in (8) based on D independent draws converges almost surely to the target expectation as $D \to \infty$.

The importance weights can also be used to resample the paths with corresponding probabilities $\widetilde{w}_i$, a technique known as sampling importance resampling (SIR; Rubin 1987). Under the same assumptions as above, letting $D \to \infty$ ensures that the SIR paths are consistent with the smoothing density $p(x|y)$, as shown by Smith and Gelfand (1992). This approach was also considered by Kim, Shephard, and Chib (1998) for refining posterior draws of the states in Bayesian inference for a stochastic volatility model.

5 Applications

This section presents several applications of the XMC simulation smoother to illustrate its main features. The validation sample fraction is set to $c_{\text{val}} = 0.1$ throughout.

5.1 Simulation Smoothing for a Nonlinear Model

We consider the nonlinear model given by

$$\begin{aligned} x_{t+1} &= \frac{1}{2}x_t + \frac{25x_t}{1+x_t^2} + 8\cos\left[1.2(t+1)\right] + \varepsilon_t^x, & \varepsilon_t^x &\sim \mathrm{N}(0, \sigma_x^2), \\ y_t &= \frac{x_t^2}{20} + \varepsilon_t^y, & \varepsilon_t^y &\sim \mathrm{N}(0, \sigma_y^2), \end{aligned} \tag{10}$$

with $x_1 \sim \mathrm{N}(0, 1)$. The parameters are set to $\sigma_x^2 = 0.1$ and $\sigma_y^2 = 1$ as in Kitagawa (1996). This model serves as a common test case for filtering and smoothing methods (e.g., Gordon, Salmond, and Smith 1993; Kitagawa 1996; Doucet, Godsill, and Andrieu 2000).

Figure 1 (a) shows a simulated path of the measurements from the nonlinear model in (10). Figure 1 (b) presents estimates of the corresponding smoothing means $\mathbb{E}[x_t|y_{1:T}]$ based on the DMDN and RF-XMC simulation smoothers with $N = D = 10^4$. For comparison, the figure also includes estimates based on particle smoothing, obtained using forward filter backward smoothing (Doucet et al. 2000) with 10^4 particles. The close agreement among the three methods supports the accuracy of the XMC simulation smoothers.

For further analysis, we examine the simulated paths and their distributions. Figure 2 displays the true path of the states (indicated by dots) that was used to generate the mea-

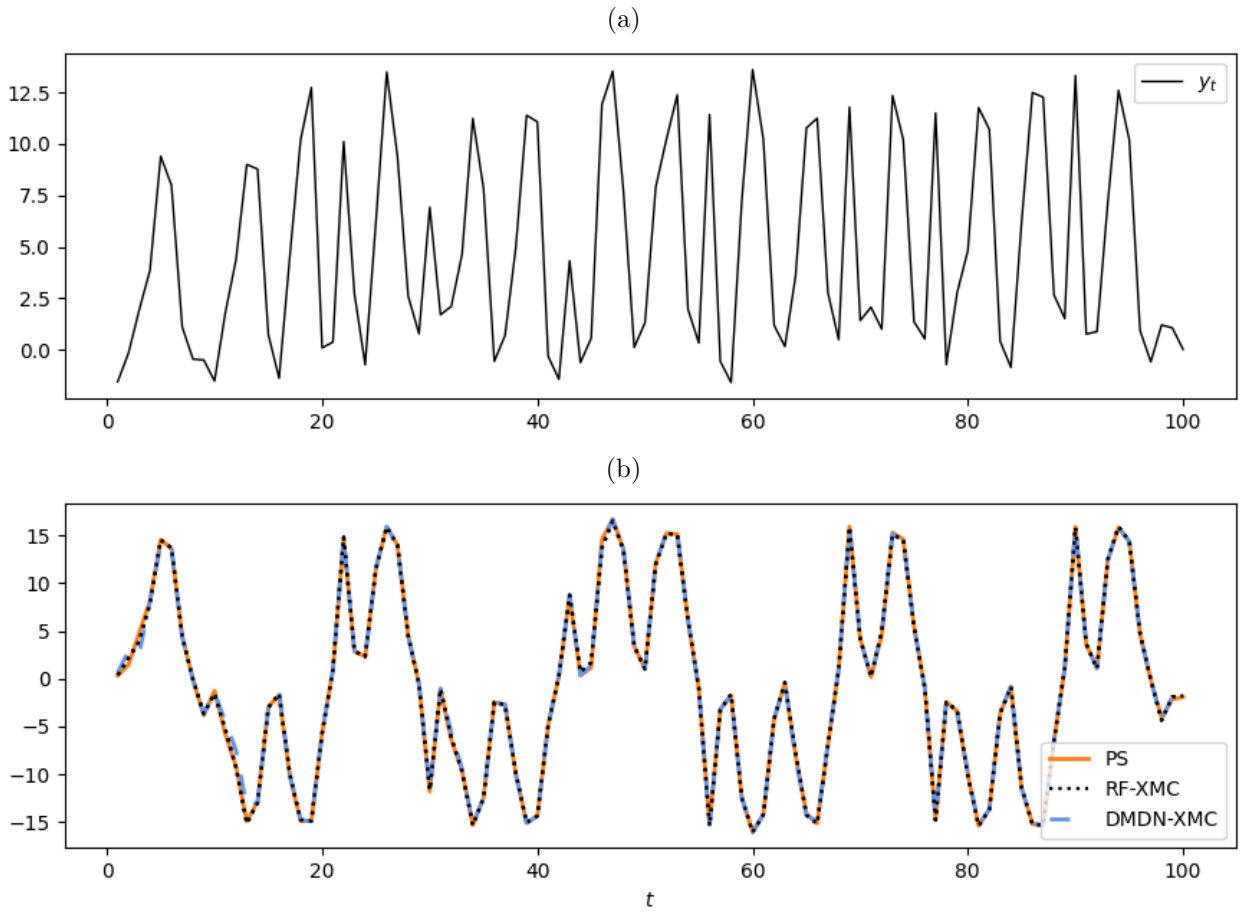

Figure 1: Simulated measurements and estimates of the corresponding smoothing means $\mathbb{E}[x_t|y_{1:T}]$ based on the nonlinear model in (10): (a) simulated path of the measurements; (b) estimates of the smoothing means based on particle smoothing (PS) with 10^4 particles, and the RF and DMDN versions of the XMC simulation smoother with $N = D = 10^4$.

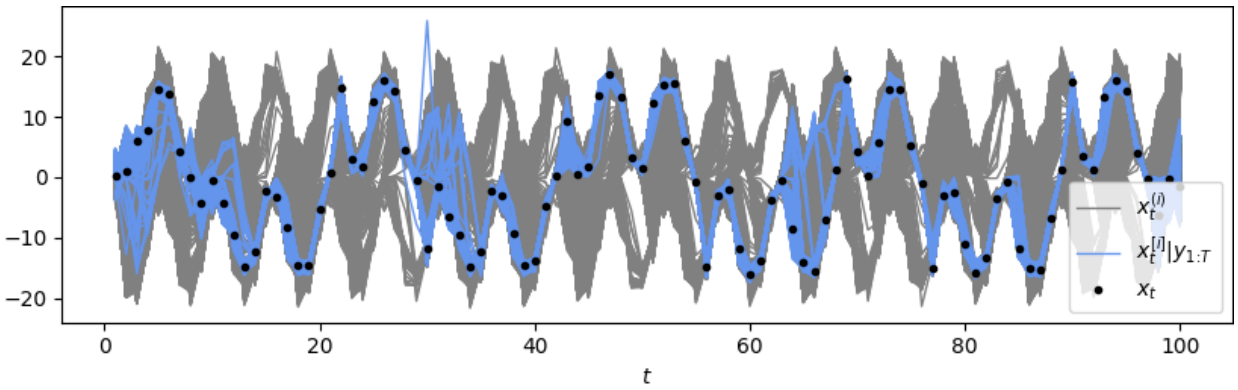

Figure 2: True states (dots) used to generate the measurements in Figure 1, shown alongside $D = 10^5$ smoothed paths from the DMDN-XMC simulation smoother with $N = 10^6$ (blue lines), as well as 10^5 draws from the unconditional distribution of the states (gray lines).

surements in Figure 1 (a). Based on these measurements, the DMDN-XMC simulation smoother was applied with $N = 10^6$ to generate $D = 10^5$ smoothed paths of the states, shown in blue. For comparison, an equal number of paths drawn from the unconditional distribution of the states are shown in gray. Unlike the unconditional paths, the smoothed paths are tightly concentrated around the true trajectory, reflecting the information contained in the observations about the latent states.

At several time points, the paths suggest that the marginal distributions are multimodal, which is a known characteristic of the nonlinear model (Doucet et al. 2001, Ch. 1). This presents challenges for several established approaches to simulation smoothing. For instance, Gaussian approximations to the smoothing distribution (Durbin & Koopman 1997; Shephard & Pitt 1997) are not suitable in this setting, as they neglect one or more modes of the target density, while MCMC methods are prone to becoming trapped in one of the modes (e.g., Hoogerheide, Van Dijk, & Van Oest 2009). Notably, the XMC simulation smoother generates independent samples, and a sufficiently flexible estimator could be used to account for the modes of the target distribution.

For illustration, Figure 3 shows the marginal densities $p(x_t)$ and $p(x_t|y_{1:T})$ at $t = 9$ and $t = 58$. These densities are estimated using a Gaussian KDE based on the simulated paths shown in Figure 2. At $t = 9$, the smoothing density exhibits a large shift in the modes relative to the unconditional density. At $t = 58$, the marginal smoothing density is

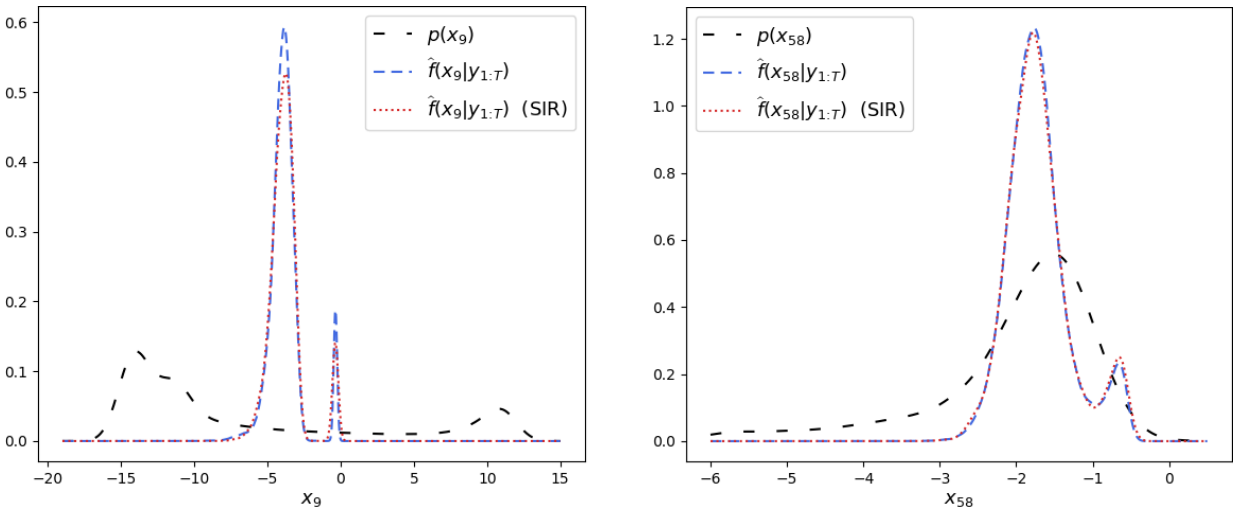

Figure 3: Marginal density estimates at $t = 9$ and $t = 58$, based on the nonlinear model in (10) and the simulated measurements shown in Figure 1 (a). Estimates are shown for the unconditional density and the smoothing density based on the paths in Figure 2, and the density corresponding to the SIR extension of the simulation smoother described in Section 4.

bimodal, while the unconditional density is unimodal. These examples highlight the value of smoothing for accurate estimation of the states and their distributions.

Figure 3 also displays the KDE based on the SIR extension of the simulation smoother described in Section 4. At both times, the SIR correction is small and has little impact on the location of the modes. This pattern holds at other time points (not shown), supporting the accuracy of the basic version of the simulation smoother.

An important feature of simulation smoothers is that the state paths allow for visualization of the joint smoothing distribution. To illustrate this, the left pane of Figure 4 presents

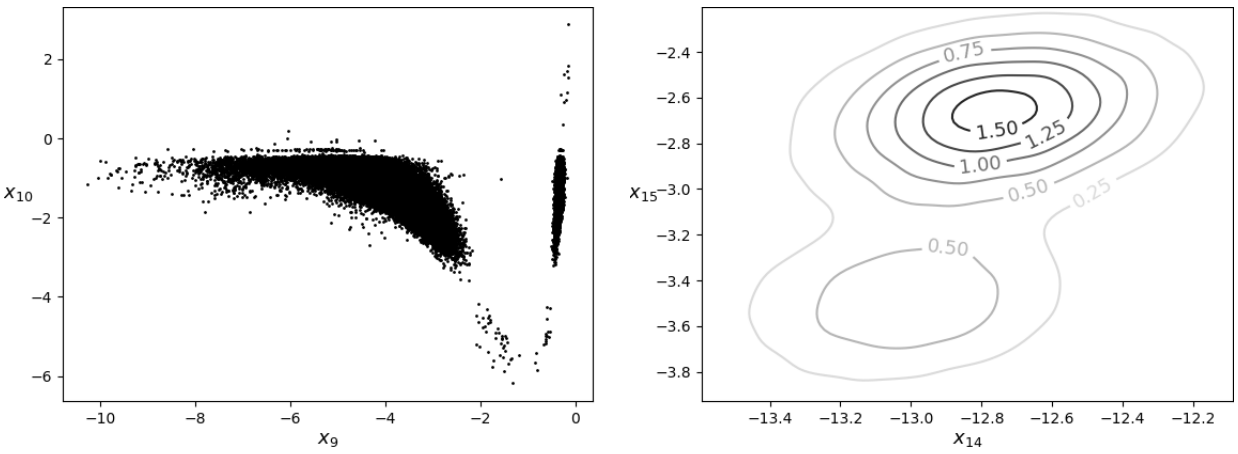

Figure 4: Visualization of the joint smoothing distribution based on the nonlinear model in (10) and the simulated measurements shown in Figure 1 (a). Left: scatter plot of the variates x_9 and x_{10} from the DMDN-XMC simulation smoother. Right: contour plot of the density $\widehat{f}(x_{14}, x_{15}|y_{1:T})$.

a scatter plot of the variates x_9 and x_{10} from the DMDN-XMC simulation smoother, revealing that the smoothing distribution is multimodal and non-elliptical. For a more detailed view, contour or surface plots could be used. The right pane of Figure 4 shows a contour plot of the joint density $\widehat{f}(x_{14}, x_{15}|y_{1:T})$, which is bimodal, with a global mode around $(x_{14}, x_{15}) = (-12.8, -2.7)$ and a local mode around $(x_{14}, x_{15}) = (-13, -3.5)$.

5.2 Empirical Application: Smoothing Time-Varying Volatility

This section applies the XMC simulation smoother to estimate the time-varying volatility of daily stock prices for Tesla. Our analysis is based on the stochastic volatility (SV) model (Lombardi and Calzolari 2009; Vankov, Guindani, and Ensor 2019) given by

$$\begin{aligned} x_{t+1} &= \mu_x + \phi_x(x_t - \mu_x) + \sigma_x \varepsilon_t^x, & \varepsilon_t^x &\sim \mathrm{N}(0,1), \\ y_t &= \exp(x_t/2)\varepsilon_t^y, & \varepsilon_t^y &\sim \mathrm{S}(\alpha, \beta), \end{aligned} \tag{11}$$

with $x_1 \sim \mathrm{N}\big[\mu_x, \sigma_x^2/(1-\phi_x^2)\big]$ and parameters $\mu_x \in \mathbb{R}$, $|\phi_x| < 1$, and $\sigma_x > 0$. The observation y_t represents the return on a financial asset at time t, and x_t determines its scale (or volatility). Additionally, $\mathrm{S}(\alpha, \beta)$ denotes the first parameterization of the stable distribution as in Nolan (2009), with tail index $\alpha \in (0, 2]$ and asymmetry parameter $\beta \in [-1, 1]$.

The stable distribution has heavy tails for $\alpha < 2$, and the above model has been used in applications characterized by extreme movements, including currency crises (Lombardi & Calzolari 2009), propane spot prices (Vankov et al. 2019), and Bitcoin (Blasques, Koopman, & Moussa 2024). We consider its application to the daily log returns of Tesla, defined as $100 \cdot \log(P_t/P_{t-1})$, where P_t is the closing price of Tesla stock at day t, adjusted for stock splits. The observations correspond y_t to the centered log returns shown in Figure 5, ranging from January 4, 2021, to January 3, 2022. The data were obtained from *WRDS* (ticker: TSLA).

A complicating factor is that, apart from a few specific parameter choices, such as $\alpha = 2$, which yields the normal distribution, the stable density $p(\varepsilon_t^y)$ does not have a closed-form expression. This precludes the direct use of many standard smoothing methods.

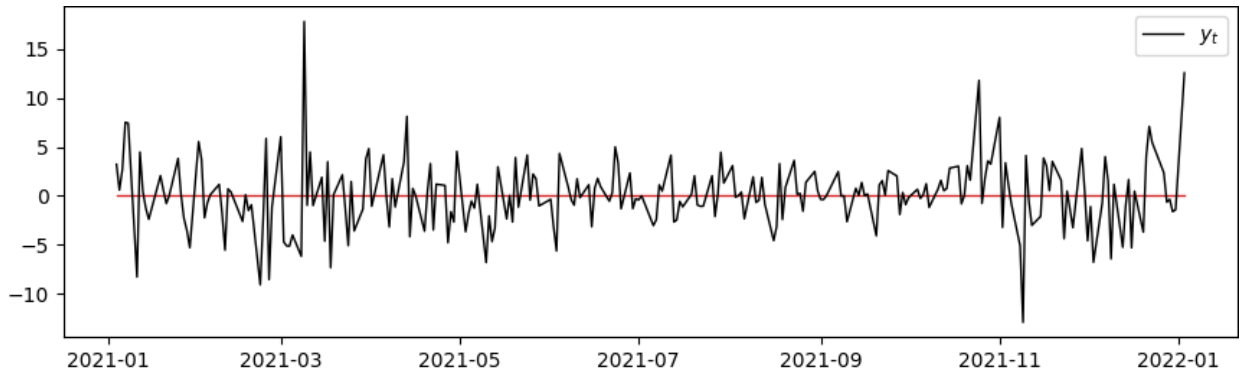

Figure 5: Centered daily log returns of Tesla stock from January 4, 2021, to January 3, 2022.

An important feature of the XMC simulation smoother is that it can be applied when the SSM allows for simulation. Generating draws from the stable distribution is straightforward using the method of Chambers, Mallows, and Stuck (1976), which makes the proposed simulation smoother directly applicable to the SV model in (11).

For illustration, the DMDN-XMC simulation smoother was applied to the SV model and Tesla data set described above, with $N = 10^6$ and $D = 10^4$. The static parameters were determined via indirect inference (Gourieroux, Monfort, & Renault 1993), using the estimator from Blasques et al. (2024) based on an extended sample (June 30, 2010 - March 28, 2025). The estimates are given by $\alpha = 1.819$, $\beta = -0.050$, $\mu_x = 1.632$, $\phi_x = 0.989$, and $\sigma_x = 0.140$. Figure 6 (a) shows the estimated $10\%, 50\%$, and 90% smoothing quantiles corresponding to $p(x_t|y_{1:T})$. The state quantiles are higher near the ends of the sample, in line with the log returns in Figure 5, which display a relatively calm middle region.

Figure 6 (b) presents the corresponding estimates obtained using the steady state version of the DMDN-XMC simulation smoother, as described in Section 3.3. The steady state was reached at the first opportunity ($t_{\text{ss}} = 252$), which allowed for bypassing 91.3% of the optimizations in the estimation step of Algorithm 2. The estimates from the steady state simulation smoother closely match those from the basic version, demonstrating that the computational savings did not come at the expense of accuracy.

Simulation smoothing also enables the estimation of nonlinear functions of the states. As illustration, the DMDN and RF versions of the steady state XMC simulation smoother were applied with $D = 10^4$ and $N = 10^6$ (DMDN) or $N = 10^5$ (RF) to estimate the

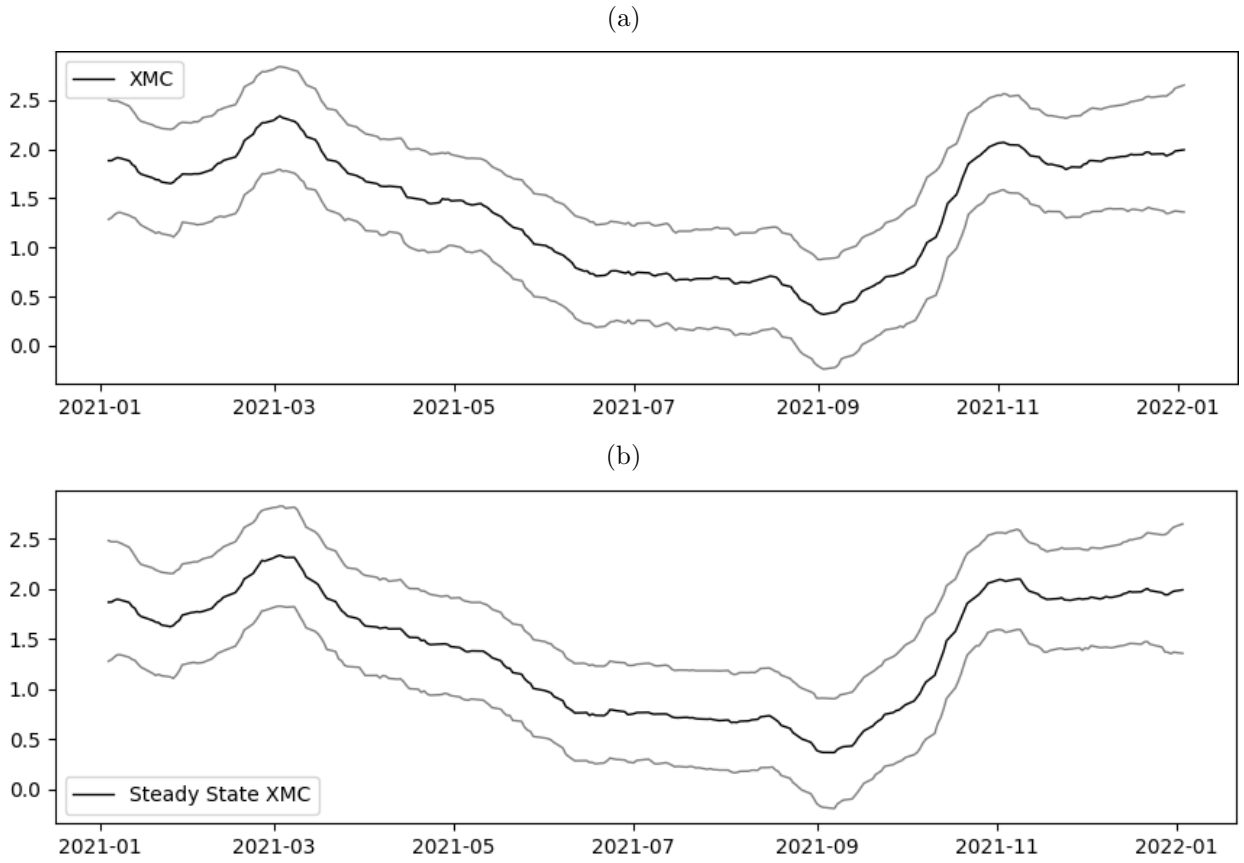

Figure 6: Marginal smoothing quantiles corresponding to $p(x_t|y_{1:T})$ for cumulative probabilities $10\%, 50\%$, and 90% based on the SV model in (11): (a) estimates from the DMDN-XMC simulation smoother with $N = 10^6$ and $D = 10^4$; (b) analogous estimates from the steady state version of the simulation smoother. The steady state was reached at time $t_{\text{ss}} = 252$, selected using the validation sample approach from Appendix B.2 with $c_{\text{ss}} = 0$.

volatility of the log returns, defined as

$$\sigma_t = \exp(x_t/2).$$

Figure 7 displays the first 10 smoothed volatility paths, along with estimates of the corresponding smoothing means $\mathbb{E}[\sigma_t|y_{1:T}]$ based on all $D = 10^4$ paths. The estimated smoothing means are similar for both versions of the simulation smoother, supporting their accuracy. Volatility is estimated to be higher near the ends of the sample, which is consistent with the pattern in the absolute values of the log returns shown in Figure 7.

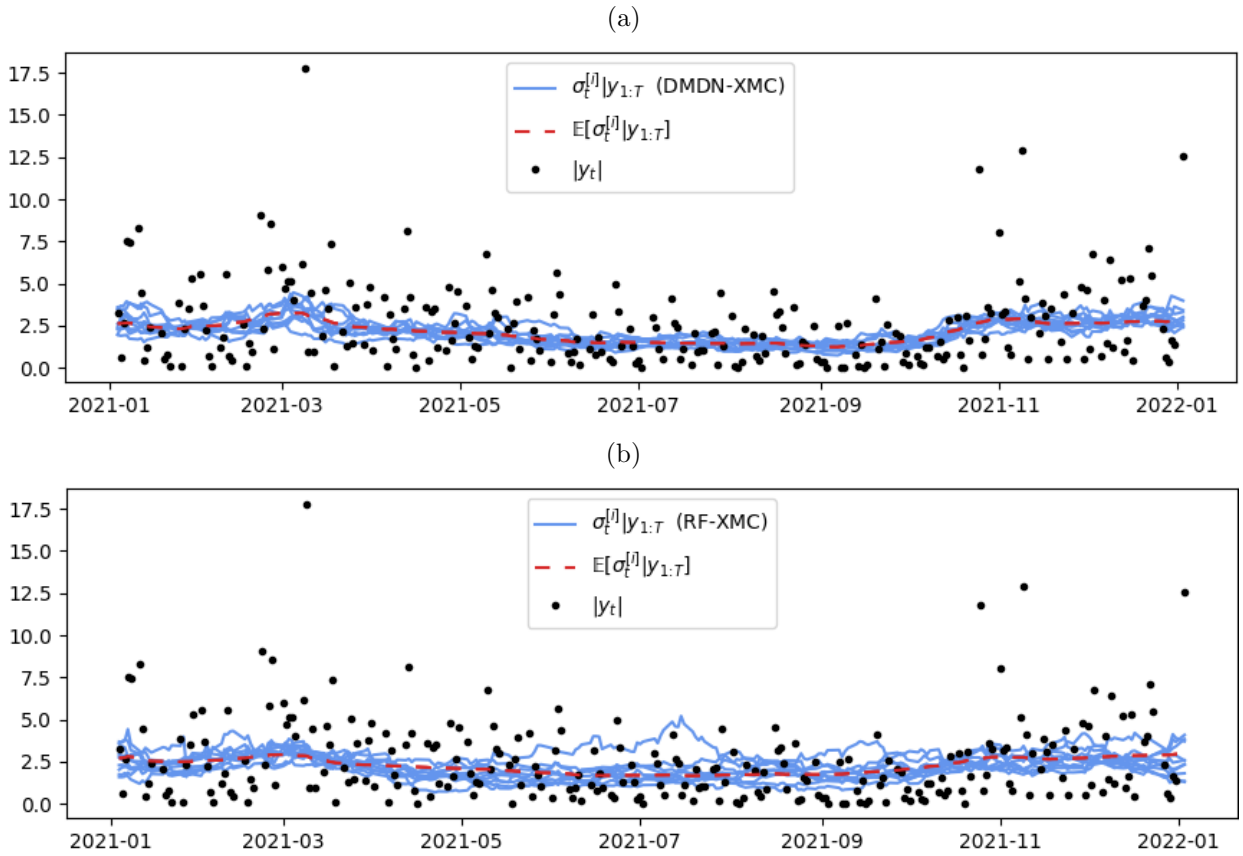

Figure 7: Paths of the volatility $\sigma_t = \exp(x_t/2)$ from the XMC simulation smoother based on the SV model in (11) and the Tesla data: (a) mean volatilities and first 10 out of $D = 10^4$ paths used to compute them for the DMDN-XMC simulation smoother; (b) analogous quantities for the RF version. The steady state approach was used in both cases, with times $t_{\text{ss}} = 252$ (DMDN) and $t_{\text{ss}} = 230$ (RF) selected using the validation sample approach from Appendix B.2 with $c_{\text{ss}} = 0$.

6 Discussion

This section provides a discussion of related work and extensions of the proposed method. The simulation smoother builds on the XMC filtering method for SSMs introduced in Moussa et al. (2023), which uses regression with simulated data to predict latent states. The focus of that method is on computing point estimates of the states, such as the filtering or smoothing means. In contrast, this study addresses the simulation of paths from the smoothing distribution of the states. As illustrated in the applications, these paths enable the simultaneous computation of multiple smoothing estimates of interest. Moreover, they allow for visualization of the joint behavior of the states conditional on the observed data.

The XMC simulation smoother has a natural connection to the literature on Monte Carlo methods for smoothing in SSMs; see Durbin and Koopman (2012, Ch. 11) and Chopin

and Papaspiliopoulos (2020, Ch. 12), as well as the references therein. Additionally, the steady state approach from Section 3.3 links the simulation smoother to amortized inference (Stuhlmüller et al. 2013; Gershman and Goodman 2014), where Bayesian inference is based on posterior densities estimated using neural networks that are trained with simulated data from a prior distribution. In this approach, the computational cost of estimation is amortized by reusing the estimated posterior density across multiple data samples. Notably, the steady state simulation smoother amortizes costs by reusing estimated densities at different time points within the same sample.

Two related amortized inference approaches are proposed by Paige and Wood (2016), who explore its use in Bayesian networks, and Lin and Eisner (2018), who apply it to natural language processing. Both focus on constructing importance samplers for SMC. Unlike these approaches, the present work focuses on simulation smoothing for SSMs, and the estimation approach is either direct or based on pathwise importance sampling, as described in Section 4. Special attention is given to data reduction through the use of selected measurements as covariates—an aspect not discussed in the aforementioned works, but crucial for maintaining computational feasibility in medium-to-long time series. Another distinguishing feature of the simulation smoother is that Algorithm 2 can be paired with various estimators of conditional distributions, allowing for customization based on specific preferences or the application at hand.

This paper presents a novel approach to simulation smoothing for nonlinear and non-Gaussian SSMs. One of the key strengths of the proposed method is that it does not require redesigning for specific models. Furthermore, it is generally applicable, with the main requirement being the ability to simulate from the SSM of interest. Future work could explore whether the approach can be leveraged for parameter estimation. For example, the method might be used to construct importance samplers for simulated maximum likelihood or to provide candidate densities for MCMC. Such extensions could further broaden the applicability of the XMC simulation smoother.

Acknowledgments

This paper is based on Chapter 4 of my PhD dissertation (Moussa 2024). An earlier version of the paper named "Simulation Smoothing: An Extremum Monte Carlo Approach" was presented at the 2023 Econometrics Brownbag Seminar at the Vrije Universiteit Amsterdam, the 2024 RCEA conference in London, and the 2024 QFFE conference in Marseille. I am grateful for comments from Siem Jan Koopman, Francisco Blasques, Neil Shephard, Simon Godsill, Nalan Bastürk, Frank van der Meulen, Lennart Hoogerheide, Mariia Artemova, and Lukas Hoesch, as well as the feedback from various conference participants.

Funding Details

No funding was received for this study.

Disclosure Statement

The author declares no competing interests.

References

Amemiya, T. (1985). *Advanced Econometrics*. Harvard University Press.

Bergstra, J., Yamins, D., & Cox, D. (2013). Making a Science of Model Search: Hyperparameter Optimization in Hundreds of Dimensions for Vision Architectures. In *International Conference on Machine Learning* (pp. 115–123).

Bishop, C. M. (1994). Mixture Density Networks.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning* (Vol. 4) (No. 4). Springer.

Blasques, F., Koopman, S. J., & Moussa, K. (2024). Asymmetric Stable Stochastic Volatility Models: Estimation, Filtering, and Forecasting. *Journal of Time Series Analysis*.

Breiman, L. (2001). Random Forests. *Machine Learning*, *45*(1), 5–32.

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

Chambers, J. M., Mallows, C. L., & Stuck, B. (1976). A Method for Simulating Stable Random Variables. *Journal of the American Statistical Association*, *71*(354), 340–344.

Chopin, N., & Papaspiliopoulos, O. (2020). *An Introduction to Sequential Monte Carlo*. Springer.

De Jong, P., & Shephard, N. (1995). The Simulation Smoother for Time Series Models. *Biometrika*, *82*(2), 339–350.

Doucet, A., De Freitas, N., & Gordon, N. J. (2001). *Sequential Monte Carlo Methods in Practice* (Vol. 1) (No. 2). Springer.

Doucet, A., Godsill, S., & Andrieu, C. (2000). On Sequential Monte Carlo Sampling Methods for Bayesian Filtering. *Statistics and computing*, *10*(3), 197–208.

Durbin, J., & Koopman, S. J. (1997). Monte Carlo Maximum Likelihood Estimation for non-Gaussian State Space Models. *Biometrika*, *84*(3), 669–684.

Durbin, J., & Koopman, S. J. (2002). A Simple and Efficient Simulation Smoother for State Space Time Series Analysis. *Biometrika*, *89*(3), 603–616.

Durbin, J., & Koopman, S. J. (2012). *Time Series Analysis by State Space Methods*. Oxford University Press.

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

Galilei, G. (1632). *Dialogo Sopra I Due Massimi Sistemi Del Mondo, Tolemaico, E Copernicano*.

Gershman, S., & Goodman, N. (2014). Amortized Inference in Probabilistic Reasoning. In *Proceedings of the Annual Meeting of the Cognitive Science Society* (Vol. 36).

Geweke, J. (1989). Bayesian Inference in Econometric Models Using Monte Carlo Integration. *Econometrica*, 1317–1339.

Godsill, S. J., Doucet, A., & West, M. (2004). Monte Carlo Smoothing for Nonlinear Time Series. *Journal of the American Statistical Association*, *99*(465), 156–168.

Gordon, N. J., Salmond, D. J., & Smith, A. F. (1993). Novel Approach to Nonlinear/Non-Gaussian Bayesian State Estimation. In *IEEE Proceedings F (Radar and Signal Processing)* (Vol. 140, pp. 107–113).

Gourieroux, C., Monfort, A., & Renault, E. (1993). Indirect Inference. *Journal of applied econometrics*, *8*(S1), S85–S118.

Hald, A. (1986). Galileo's Statistical Analysis of Astronomical Observations. *International Statistical Review/Revue Internationale de Statistique*, 211–220.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer.

He, Y., & Wang, J. (2020). Deep Mixture Density Network for Probabilistic Object Detection. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (pp. 10550–10555).

Hoogerheide, L. F., Van Dijk, H. K., & Van Oest, R. D. (2009). Simulation Based Bayesian Econometric Inference: Principles and Some Recent Computational Advances. *Handbook of Computational Econometrics*, 215–280.

Kim, S., Shephard, N., & Chib, S. (1998). Stochastic Volatility: Likelihood Inference and Comparison with ARCH Models. *The Review of Economic Studies*, *65*(3), 361–393.

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.

Kobyzev, I., Prince, S. J., & Brubaker, M. A. (2020). Normalizing Flows: An Introduction and Review of Current Methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *43*(11), 3964–3979.

Li, T., Bolic, M., & Djuric, P. M. (2015). Resampling Methods for Particle Filtering: Classification, Implementation, and Strategies. *IEEE Signal Processing Magazine*, *32*(3), 70–86.

Lin, C.-C., & Eisner, J. (2018). Neural Particle Smoothing for Sampling from Conditional Sequence Models. *arXiv Preprint arXiv:1804.10747*.

Lombardi, M. J., & Calzolari, G. (2009). Indirect Estimation of α-Stable Stochastic Volatility Models. *Computational Statistics & Data Analysis*, *53*(6), 2298–2308.

McCausland, W. J., Miller, S., & Pelletier, D. (2011). Simulation Smoothing for State–Space Models: A Computational Efficiency Analysis. *Computational Statistics & Data Analysis*, *55*(1), 199–212.

Meinshausen, N. (2006). Quantile Regression Forests. *Journal of Machine Learning Research*, *7*(6), 983–999.

Moussa, K. (2024). *Signal Extraction by the Extremum Monte Carlo Method* (PhD Dissertation). Vrije Universiteit Amsterdam and Tinbergen Institute, the Netherlands.

Moussa, K., Blasques, F., & Koopman, S. J. (2023). Extremum Monte Carlo Filters: Signal Extraction via Simulation and Regression. *Discussion Paper Tinbergen Institute TI 2023-016/III*. Retrieved from `https://papers.tinbergen.nl/23016.pdf`

Nolan, J. P. (2009). Univariate Stable Distributions. *Stable Distributions: Models for Heavy Tailed Data*, *22*(1), 79–86.

Paige, B., & Wood, F. (2016). Inference Networks for Sequential Monte Carlo in Graphical Models. In *International Conference on Machine Learning* (pp. 3040–3049).

Rubin, D. B. (1987). The Calculation of Posterior Distributions by Data Augmentation: Comment: A Noniterative Sampling/Importance Resampling Alternative to the Data Augmentation Algorithm for Creating a Few Imputations When Fractions of Missing Information Are Modest: The SIR Algorithm. *Journal of the American Statistical Association*, *82*(398), 543–546.

Shephard, N., & Pitt, M. K. (1997). Likelihood Analysis of Non-Gaussian Measurement Time Series. *Biometrika*, *84*(3), 653–667.

Smith, A. F., & Gelfand, A. E. (1992). Bayesian Statistics Without Tears: A Sampling–Resampling Perspective. *The American Statistician*, *46*(2), 84–88.

Stuhlmüller, A., Taylor, J., & Goodman, N. (2013). Learning Stochastic Inverses. *Advances in Neural Information Processing Systems*, *26*.

van der Westhuizen, S., Heuvelink, G. B., & Hofmeyr, D. P. (2023). Multivariate Random Forest for Digital Soil Mapping. *Geoderma*, *431*, 116365.

Vankov, E. R., Guindani, M., & Ensor, K. B. (2019). Filtering and Estimation for a Class of Stochastic Volatility Models with Intractable Likelihoods. *Bayesian Analysis*, *14*(1), 29–52.

Williams, C. K., & Rasmussen, C. E. (2006). *Gaussian Processes for Machine Learning* (Vol. 2) (No. 3). MIT Press Cambridge, MA.

Zen, H., & Senior, A. (2014). Deep Mixture Density Networks for Acoustic Modeling in Statistical Parametric Speech Synthesis. In *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 3844–3848).

Zhang, H., Liu, Y., Yan, J., Han, S., Li, L., & Long, Q. (2020). Improved Deep Mixture Density Network for Regional Wind Power Probabilistic Forecasting. *IEEE Transactions on Power Systems*, *35*(4), 2549–2560.

Appendix

A Conditional Distribution Estimators

A.1 Deep Mixture Density Network

We start by discussing the single-layer mixture density network (Bishop 1994, 2006), after which the deep mixture density network (DMDN) is considered. The goal is to estimate the conditional density $p(X|Y)$, where $X \in \mathbb{R}^{N_X}$ and $Y \in \mathbb{R}^{N_Y}$ are random vectors. Consider the mixture of C Gaussian densities given by

$$f(X) = \sum_{j=1}^{C} \omega_j \, f_{\mathrm{G}}(X; \mu_j, \sigma_j^2 I_{N_X}), \qquad \sum_{j=1}^{C} \omega_j = 1, \quad \omega_j \geq 0,$$

where $f_{\mathrm{G}}(X; \mu, V)$ denotes the Gaussian density with mean vector μ and variance matrix V. The mixture components are parameterized by $\omega_j = \omega_j(y)$, $\mu_j = \mu_j(y)$, and $\sigma_j = \sigma_j(y)$, which are the outputs of a neural network (Hastie et al. 2009, Ch. 11) with inputs $y \in \mathbb{R}^{N_y}$. This results in a conditional density, which can be estimated via maximum likelihood given a sample $\left(X^{(i)}, Y^{(i)}\right)$, $i = 1, \ldots, N$. The conditional density estimator is then obtained as

$$\widehat{f}^N \in \underset{f \in \mathbb{F}_N}{\arg\min} -\frac{1}{N} \sum_{i=1}^{N} \log f\left(X^{(i)} | Y^{(i)}\right),$$

where $\mathbb{F}_N$ is the set of mixture density networks considered. The above minimization is performed using the ADAM optimizer (Kingma & Ba 2015).

The DMDN extends the mixture density network by introducing multiple hidden layers in the neural network that maps the inputs $y \in \mathbb{R}^N$ to the parameters of the mixture density. The number of layers, the number of neurons per layer, the number of mixture components, as well as the learning rate and the batch size for optimizing the objective function are all treated as tuning parameters. Once the conditional density has been estimated, sampling is straightforward: for a given input y, a value is simulated from the j-th density in the

mixture with probability $\omega_j(y)$.

A.2 Regression Forest

The regression forest estimator considered in the applications is based on the random forest (Breiman 2001), which is defined as an average of a large number of decorrelated regression trees. The trees are grown using bootstrapped samples of the data. Regression trees are known to have low bias but high variance; by averaging over many such identically distributed trees, the random forest retains this low bias while reducing the variance. To reduce the correlation between the trees, the split decisions are made using a subset of covariates that are randomly selected at each node.

Random forests are typically used for prediction by providing estimates of the conditional mean. However, Meinshausen (2006) showed that they can also be used to estimate the full conditional probability distribution. Given a sample $\left(X^{(i)}, Y^{(i)}\right),\ i = 1, \ldots, N$ from $p(X, Y)$, the random forest predictions of X can be represented as weighted averages of the response values,

$$\sum_{i=1}^{N} \omega_i(y) X^{(i)} \approx \mathbb{E}[X|Y = y], \qquad \sum_{i=1}^{N} \omega_i(y) = 1, \quad \omega_i(y) \geq 0.$$

The weights are defined as an average over the weights of K regression trees,

$$\omega_i(y) = \frac{1}{K} \sum_{k=1}^{K} \omega_i^k(y),$$

where each tree predicts the response X by taking the average over the corresponding variates in the leaf to which y is assigned. The individual tree weights are defined by

$$\omega_i^k(y) = \frac{\mathbb{I}\big(i \in \mathbb{L}_k(y)\big)}{|\mathbb{L}_k(y)|},$$

where $\mathbb{I}(A)$ is the indicator function of the event A, $\mathbb{L}_k(y)$ denotes the set of indices j of

the values $Y^{(j)}$ in the leaf to which y is assigned, and $|\mathbb{L}_k(y)|$ is its number of elements.

These weights define a probability mass function given by

$$\widehat{f}^N(x|Y=y) = \sum_{i=1}^{N} \omega_i(y)\, \mathbb{I}\left(X^{(i)} = x\right).$$

Drawing from this distribution amounts to resampling the variates $X^{(i)}$ with replacement, using the weights $\omega_i(y),\ i = 1, \ldots, N$, as probabilities.

The tuning parameters are the number of covariates considered at each split and the maximum depth of the trees. In the applications of the simulation smoother, the default value of $K = 100$ trees is used. For conditional simulation with vector-valued responses, a similar approach could be adopted using multivariate regression forests (De'Ath 2002; Segal and Xiao 2011; Cevid, Michel, Näf, Bühlmann, and Meinshausen 2022; van der Westhuizen et al. 2023).

B Details on Steady State Approach

B.1 Rolling Window Covariates

Reusing a density estimate $\widehat{f}^N_{t_{\text{ss}}}$ for simulation at times $t \neq t_{\text{ss}}$ requires that the covariates follow a rolling window structure, as described below. Let B denote the backshift operator, defined by $B^m z_t = z_{t-m}$ for $m \in \mathbb{Z}$ and any time series $\{z_t\}$. Then, with covariates $\mathcal{C}_t$ and $\mathcal{C}_{t_{\text{ss}}}$ defined via (6), we require that

$$\mathcal{C}_t = \left\{ B^{(t_{\text{ss}}-t)} z_k \,\middle|\, z_k \in \mathcal{C}_{t_{\text{ss}}} \right\}, \tag{B.1}$$

where each element z_k either corresponds to the next state, $x_{t_{\text{ss}}+1}$, or to an observation at or before time t_{ss}. Thus, the covariates for time t are obtained by shifting each covariate for time t_{ss} backward by $(t_{\text{ss}} - t)$ time units.

The condition in (B.1) defines the feasible range for t (and t_{ss}) as given in (7), where the

upper bound reflects the availability of a future state for conditioning, while the lower bound ensures that the observations $y_{\underset{\sim}{t}:t}$ used as covariates in $\mathcal{C}_t$ are consistent with (B.1). The latter follows from the definition of the covariates in (6), which implies that when $t \geq W$, it holds for any $m \in \mathbb{N}$ that

$$\underbrace{t+m}_{\sim} = \max\left\{(t+m) - W + 1, 1\right\} = t + m - W + 1 = \max\left\{t - W + 1, 1\right\} + m = \underset{\sim}{t} + m.$$

This shows that the lower index of the observations used as covariates at any time $t+m$ equals the lower index at time t, shifted by m time units. Consequently, the observations (and therefore the covariates) correspond to rolling windows.

B.2 Iterative Procedure for the Steady State Time

The following procedure uses the validation sample to iteratively evaluate whether a given density estimate is suitable for reuse at earlier time points. Let $c_{\text{ss}} \geq 0$ be a chosen tolerance level, and denote the cases from the validation sample by $(x^{\langle i \rangle}, y^{\langle i \rangle})$, $i = 1, \ldots, N_{\text{val}}$. At time t in the estimation step of Algorithm 2, we assess whether the density estimate $\widehat{f}_t^N$ generalizes well to time W—the most distant feasible past time—by comparing it to the corresponding estimate $\widehat{f}_W^N$. This yields a conservative estimate of the performance of $\widehat{f}_t^N$ at other time points.

Specifically, we check for $t = T-1, \ldots, W+1$ whether the following condition holds,

$$\sum_{i=1}^{N_{\text{val}}} L\left(x_W^{\langle i \rangle}, \mathcal{C}_W^{\langle i \rangle}, \widehat{f}_t^N\right) \leq (1 + c_{\text{ss}}) \sum_{i=1}^{N_{\text{val}}} L\left(x_W^{\langle i \rangle}, \mathcal{C}_W^{\langle i \rangle}, \widehat{f}_W^N\right).$$

If this condition is satisfied at time t, we say that a steady state has been reached and set $t_{\text{ss}} \coloneqq t$. The corresponding density estimate $\widehat{f}_{t_{\text{ss}}}^N$ can then be used to circumvent the optimizations in (5) for the remaining feasible times $t = t_{\text{ss}} - 1, \ldots, W+1$.

References

Bishop, C. M. (1994). Mixture Density Networks.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning* (Vol. 4) (No. 4). Springer.

Breiman, L. (2001). Random Forests. *Machine Learning*, *45*(1), 5–32.

Cevid, D., Michel, L., Näf, J., Bühlmann, P., & Meinshausen, N. (2022). Distributional Random Forests: Heterogeneity Adjustment and Multivariate Distributional Regression. *Journal of Machine Learning Research*, *23*(333), 1–79.

De'Ath, G. (2002). Multivariate Regression Trees: A New Technique for Modeling Species–Environment Relationships. *Ecology*, *83*(4), 1105–1117.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer.

Kingma, D. P., & Ba, J. (2015). Adam: A Method for Stochastic Optimization. *International Conference on Learning Representations (ICLR)*.

Meinshausen, N. (2006). Quantile Regression Forests. *Journal of Machine Learning Research*, *7*(6), 983–999.

Segal, M., & Xiao, Y. (2011). Multivariate Random Forests. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, *1*(1), 80–87.

van der Westhuizen, S., Heuvelink, G. B., & Hofmeyr, D. P. (2023). Multivariate Random Forest for Digital Soil Mapping. *Geoderma*, *431*, 116365.