Blasques, Francisco; Koopman, Siem Jan; Moussa, Karim

Working Paper

Extremum Monte Carlo Filters: Real-Time Signal Extraction via Simulation and Regression

Tinbergen Institute Discussion Paper, No. TI 2023-016/III

Provided in Cooperation with:
Tinbergen Institute, Amsterdam and Rotterdam

Suggested Citation: Blasques, Francisco; Koopman, Siem Jan; Moussa, Karim (2023) : Extremum Monte Carlo Filters: Real-Time Signal Extraction via Simulation and Regression, Tinbergen Institute Discussion Paper, No. TI 2023-016/III, Tinbergen Institute, Amsterdam and Rotterdam

This Version is available at:
https://hdl.handle.net/10419/273827

TI 2023-016/III
Tinbergen Institute Discussion Paper

Extremum Monte Carlo Filters: Real-Time Signal Extraction via Simulation and Regression

Francisco Blasques[1]
Siem Jan Koopman[1]
Karim Moussa[1]

1 Vrije Universiteit Amsterdam and Tinbergen Institute

Extremum Monte Carlo Filters:

Real-Time Signal Extraction via Simulation and Regression

Francisco Blasques, Siem Jan Koopman, Karim Moussa*

Vrije Universiteit Amsterdam and Tinbergen Institute, the Netherlands

First version: November 18, 2021
This version: March 23, 2023

Abstract

This paper introduces a novel simulation-based filtering method for general state space models. It allows for the computation of time-varying conditional means, quantiles, and modes, but also for the prediction of latent variables in general. The method relies on generating artificial samples of data from the joint distribution implied by the model and on estimating the conditional quantities of interest via extremum estimation. We call this procedure *Extremum Monte Carlo* and define a corresponding class of filters for signal extraction. The method can be applied to any model from which data can be simulated and is not liable to the curse of dimensionality. Furthermore, the use of extremum estimation allows for a wide range of conditioning sets, including data with missing entries and unequal spacing. The filtering method also places the computational burden predominantly in the off-line phase, which makes it particularly suitable for real-time applications. We present illustrations for some challenging problems characterized by nonlinearity, high-dimensionality, and intractable density functions.

Keywords: Nonlinear non-Gaussian state space models, Least squares Monte Carlo, Real-time filtering, Intractable densities, Curse of dimensionality.

*Corresponding author. E-mail: k.moussa.research@gmail.com. Blasques thanks the Dutch Research Council (VI.Vidi.195.099) for financial support. Koopman acknowledges support from Aarhus University, Denmark, and funding of the Danish National Research Foundation (DNRF78). All reported results are generated by a computer with an Intel i5 quad-core processor with 3.3 GHz clock frequency. The software code is written in `Python` and is optimized by calling various functions from pre-compiled `C/C++` code.

1 Introduction

State space models (SSMs) decompose observed time series into two unobserved parts: the states (or signal) which are the true objects of interest, and the noise which complicates the extraction of the signal from the data. The state space modeling approach has become pervasive in both the scientific and industry domains, with applications in fields varying from financial econometrics and forecasting to robotics. By defining $x_t \in \mathbb{R}^{N_x}$ as the state vector at time t and $y_t \in \mathbb{R}^{N_y}$ as the corresponding vector of observed variables (or measurements), together with the noise vectors ε_t^x and ε_t^y, we can represent the SSM by

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

for $t = 1, \ldots, T$, with $N_x, N_y, T \in \mathbb{N}$, where T is the length of the time series. All variables may be either continuous or discrete, and we use $p(\cdot)$ to denote the probability density or the mass function for the corresponding variables. We emphasize the generality of SSM (1) by noting that the only restriction we impose on the functions m_t, s_t, $p(x_1)$, and $p(\varepsilon_t^x, \varepsilon_t^y)$ is that one is able to simulate from the model. For example, the noise vectors ε_t^x and ε_t^y are allowed to be mutually and serially dependent, while the state transition function s_t may depend on the path $y_{1:t} = \{y_j\}_{j=1}^t$ and the measurement function m_t may depend on $y_{1:t-1}$. Furthermore, all functions can depend on exogenous variables and on the vector of static parameters (or hyperparameters) θ. The above dependencies are suppressed in the notation for conciseness.

Once the static parameters θ have been provided or estimated, the interest is often shifted towards signal extraction, which may be performed via the conditional expectation of the states,

$$\mathbb{E}\left[x_t | Y_t\right], \tag{2}$$

for $t = 1, \ldots, T$, where Y_t denotes the conditioning set. Common choices are $Y_t = y_{1:t}$ for filtering, $Y_t = y_{1:t-k}$ for k-period forecasting, and $Y_t = y_{1:t+k}$ for smoothing, with $k \in \mathbb{N}$. If the SSM is linear and Gaussian, that is, m_t and s_t are linear functions, and all densities p are Gaussian, the conditional expectations in (2) can be computed recursively by the well-known Kalman filter (KF; Kalman, 1960). A simple example is the following Gaussian local level (LL) model

$$\begin{aligned} y_t &= x_t + \varepsilon_t^y, & \varepsilon_t^y &\sim \mathrm{N}(0, \sigma_y^2), \\ x_{t+1} &= x_t + \varepsilon_t^x, & \varepsilon_t^x &\sim \mathrm{N}(0, \sigma_x^2), \end{aligned} \tag{3}$$

with $x_1 \sim \mathrm{N}(\mu_1, \sigma_1^2)$ for some $\mu_1 \in \mathbb{R}$ and $\sigma_1, \sigma_x, \sigma_y > 0$, and the scalar noise terms ε_t^x and ε_t^y are assumed to be mutually and serially independent, as well as independent from x_1. The LL model is a special case of the SSM in (1) with $m_t(x_t, \varepsilon_t^y) = x_t + \varepsilon_t^y$ and $s_t(x_t, \varepsilon_t^x) = x_t + \varepsilon_t^x$, normal density $p_{\mathrm{N}}(\varepsilon_t^x, \varepsilon_t^y) = p_{\mathrm{N}}(\varepsilon_t^x) p_{\mathrm{N}}(\varepsilon_t^y)$, and $\theta = (\mu_1, \sigma_1, \sigma_x, \sigma_y)'$.

Figure 1 provides an illustration by applying the LL model to measurements of the annual flow volume of the Nile taken at Aswan from 1871 to 1970 (Durbin & Koopman, 2012, Ch. 2). After setting the static parameters to the maximum likelihood estimates $\sigma_x = 38.329$ and $\sigma_y = 122.877$, with $\mu_1 = 0$ and $\sigma_1^2 = 10^7$ for approximate diffuse initialization, the expectations $\mathbb{E}[x_t | y_{1:t}]$ for $t = 1, \ldots, T$ can be obtained by the KF.

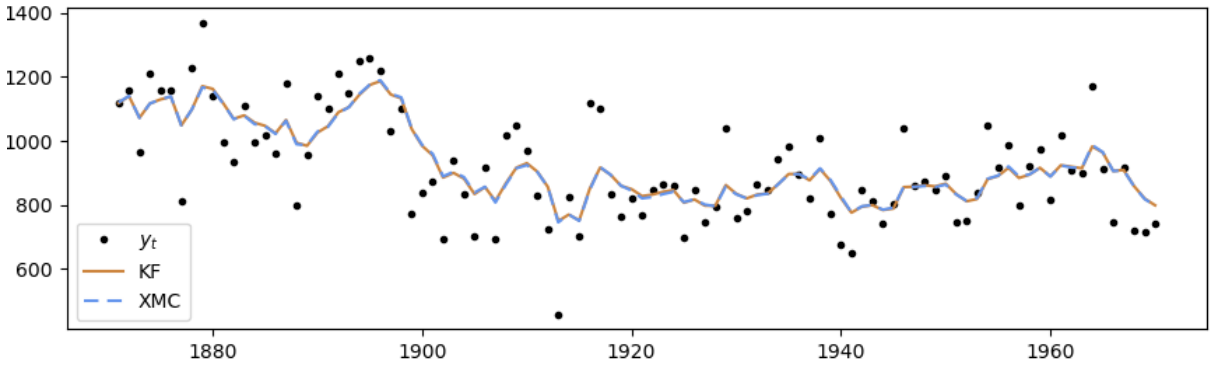

Figure 1: Analysis of the annual flow volume measurements y_t of the Nile (discharge at Aswan in $10^8 m^3$) from 1871 to 1970 based on the LL model in (3): signals extracted via $\mathbb{E}[x_t|y_{1:t}]$ by the Kalman filter (KF) and linear Extremum Monte Carlo filter (XMC) with $N = 10^4$ paths.

In practice, however, the convenient linear Gaussian assumption appears to hold by exception, rather than the rule, so that the generalities of nonlinearity and non-Gaussian noise are often needed; see Doucet, De Freitas, and Gordon (2001) and Zeng and Wu (2013) for multiple examples in engineering and economics. For many nonlinear and non-Gaussian SSMs, the computation of the conditional expectation in (2) is a challenging task and is often tackled by particle filtering methods; see Gordon, Salmond, and Smith (1993) and Pitt and Shephard (1999). However, filtering remains a challenging task for cases with, for example, high-dimensional state vectors x_t, limited tractability of the SSM, and the requirement to evaluate the expectation sequentially in real time.

In this study, we propose a novel simulation-based filtering method that relies on generating artificial samples of data from the joint distribution $p(x_{1:T}, y_{1:T})$ implied by the SSM in (1) and estimating the conditional expectations in (2) via extremum estimation (e.g., Amemiya, 1985; Hayashi, 2000). We call this procedure *Extremum Monte Carlo* (XMC) and use it to define a corresponding class of filters for signal extraction. The XMC method is mainly a filtering technique for SSMs and is therefore related to the KF and its nonlinear/non-Gaussian extensions. It is also related to the least squares Monte Carlo method (LSMC; Longstaff & Schwartz, 2001), which was developed for the valuation of American options in financial trading. A crucial step in the LSMC algorithm is the approximation of the conditional expectation $\mathbb{E}[X|Y]$ by simulation of the random variables X and Y from their joint distribution,

$$\left(X^{(i)}, Y^{(i)}\right) \sim p(X, Y), \qquad i = 1, \ldots, N.$$

The variates are then used as data in a least squares regression to compute

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

where $L(u) = u^2$ is the squared error loss function and $f(\cdot)$ can be any function from some suitable function space $\mathbb{F}_N$. Finally, the function estimate is used to predict X for any Y value of interest by

$$\widehat{f}^N(y) \approx \mathbb{E}[X|Y = y].$$

The LSMC method is widely adopted for the valuation of derivatives with early-exercise features, as well as credit valuation adjustments (Green, 2015), and it has found many other applications. Examples range from solving backward stochastic differential equations (Gobet, Lemor, and Warin 2005; Bender and Steiner 2012), to the estimation of complex *un*conditional moments for various dynamic volatility models (Engle, 2002). In addition, the method is increasingly being used in portfolio optimization (e.g., Denault & Simonato, 2017; R. Zhang et al., 2019), where it is called "simulation-and-regression." Naturally, this procedure can be generalized by allowing for loss functions $L(u)$ other than the squared error loss. Important examples are the tilted absolute error loss, $L_\tau(u) = u(\tau - 1_{\{u<\tau\}})$, with prediction error $u = X - f(Y)$ to estimate the conditional τ-quantile for $\tau \in (0, 1)$, and the all-or-nothing loss, $L_\delta(u) = 1_{\{|u|\geq\delta\}}$, with tolerance level $\delta > 0$ to approximate the conditional mode, which corresponds to the limit $\delta \to 0$.

The key idea behind the XMC method is to apply the above procedure repeatedly for times $t = 1, \ldots, T$ by setting $X = x_t$ and $Y = \widetilde{Y}_t \subseteq Y_t$, where the covariates $\widetilde{Y}_t$ are an appropriate subset of the conditioning set. In essence, we first use the SSM in (1) to simulate paths of the states and observations, and regress the former onto subsets of the latter. The estimated regression functions are then evaluated at the observed data to predict the states. Each choice of function estimator and loss function yields a different filter, hence the method defines a corresponding class of *Extremum Monte Carlo filters*. To make matters concrete, we now return to the LL model illustration. Noting that the KF is linear in the observations (Durbin & Koopman, 2012, Ch. 2), we can attempt to mimic this filter by applying the XMC method with linear regression to minimize the squared error loss for a sample of N simulated paths. Figure 1 shows the predictions of the states x_t from the resulting linear XMC filter for $N = 10^4$, which are seen to coincide with the KF.

The proposed filtering method has various beneficial characteristics. It is applicable to any model that can be simulated, including high-dimensional models, complex models with limited tractability, and models not nested in SSM (1). Furthermore, the use of extremum estimation allows for any conditioning set of choice, including unequally-spaced and unbalanced data sets with missing entries. In addition, the method is fast in real time because the bulk of the computations (simulation and estimation) can be done in advance, such that the real-time computations only involve evaluations of the estimated regression functions.

The XMC method also fills a gap in the simulation-based estimation literature as it enables signal extraction when the static parameters are estimated by the method of simulated moments (McFadden, 1989) or indirect inference (Gourieroux, Monfort, & Renault, 1993). If the model of choice contains latent variables, their extraction is important in many applications. The XMC filter therefore naturally complements the above estimation methods since they only have minimal requirements in addition to the ability to simulate from the joint distribution implied by the model.

The remainder of this paper is structured as follows. Section 2 introduces the XMC method together with several corresponding filters. Section 3 presents four applications to facilitate the discussion of the key characteristics of the XMC method. Section 4 considers filter properties for the case when either the number of simulated paths N or the time series length T becomes large. Section 5 concludes. The proofs and supplementary material are placed in the Appendix.

2 The Extremum Monte Carlo method

Algorithm 1 presents the XMC method for a given instance of the SSM in (1). The method consists of three fundamental steps: simulation, fitting, and prediction. For conciseness we assume the state to be a scalar, as the vector case is handled simply by performing the last two steps separately for each element in x_t. The simulation step ensures that N paths are available for both x_t and y_t. The generated data are then split in two parts. A training sample, which is directly used in the regressions, and a validation sample, which is used to regularize the tuning parameters of the chosen regression method. The validation sample fraction does not require optimizing because the data can be generated at will.

The regularization is performed by selecting the minimizer of the validation loss from several candidate tuning parameters generated by a Bayesian optimization procedure (Bergstra, Yamins, & Cox, 2013). While this could be done for each time separately, in practice it is sufficient to determine the tuning parameters at some suitable time-point $t = t^*$. We consider the window size $W \in \{1, \ldots, T\}$ as a tuning parameter and define the covariate set $\widetilde{Y}_t$ to consist of the W observations from Y_t, the conditioning set, that are nearest to time t. For example, with filtering ($Y_t = y_{1:t}$), we define the covariate set

Algorithm 1 Extremum Monte Carlo filtering method.

1. **Simulate**: Use the SSM to simulate N paths from the joint distribution,

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

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 regression at time $t = t^*$:

$$\widehat{f}_t^N \in \underset{f \in \mathbb{F}_N}{\arg\min} \frac{1}{N_{\text{tr}}} \sum_{i=1}^{N_{\text{tr}}} L\left(x_t^{(i)} - f\left(\widetilde{Y}_t^{(i)}\right)\right), \tag{4}$$

   with function space $\mathbb{F}_N$ and covariates $\widetilde{Y}_t^{(i)} \subseteq Y_t^{(i)}$. Select the tuning parameters that minimize the corresponding loss for the validation sample.

   (c) *Regression*: Use the regularized tuning parameters to perform the regression in (4) at all times $t = 1, \ldots, T$ to obtain the function estimates $\{\widehat{f}_t^N\}_{t=1}^T$.

3. **Predict**: Evaluate the estimated regression functions at the observed data $\widetilde{Y}_t$ for $t = 1, \ldots, T$ to predict the states:

$$\widehat{x}_t = \widehat{f}_t^N(\widetilde{Y}_t).$$

by

$$\widetilde{Y}_t = y_{\underset{\sim}{t}:t}, \qquad \text{with} \qquad \underset{\sim}{t} = \max\{t - W + 1, 1\}. \tag{5}$$

The autoregressive structure in SSM (1) implies that these observations are generally the most informative on x_t. The choice of t^* should be made such that it prevents underestimation of the window size. The validation loss is therefore best measured where most observations are available, hence $t^* = T$ is a natural choice for filtering and forecasting.

By choosing a specific loss function and regression method, Algorithm 1 defines a corresponding XMC filter, which can be chosen according to preference, provided that a minimizer of the expected loss exists. The objective function in (4) can be generalized to include weights. Furthermore, we note that the XMC method could also be used to predict functions of the states and to forecast the observations, simply by changing the dependent variable in the regressions.

The preferred choice of regression method will vary with the signal extraction problem. For the LL model example in the introduction, a linear regression is suitable. In other cases, a general function estimator can be more appropriate. We therefore consider several nonlinear regression methods: tree-based gradient boosting (GB), the random forest (RF), and the quantile regression forest (QRF). The latter is particularly suitable for the estimation of quantiles. A short discussion of these methods can be found in Appendix A.

To analyze the computational complexity of the method, we focus on the regression step in Algorithm 1, which is generally the dominant runtime factor. For both the number of states N_x and the time series length T, the complexity is linear because each regression is performed separately. Crucially, this separability implies that the total runtime can be made to approximate that of the longest among the regressions by increasing the number of physical cores. In addition, the fact that the estimates are not defined recursively ensures that potential errors in the measurements or predictions do not propagate. On the other hand, the scaling in the number of paths N and covariates $C := WN_y$ depends on the chosen regression method and corresponding implementation. Table 1 shows current estimates of the computational complexity for several XMC filters.

3 Illustrations and discussion

This section develops four applications to illustrate and discuss several of the filter's key characteristics. We focus on filtering via the conditional means of the states $\mathbb{E}[x_t|y_{1:t}]$ for $t = 1, \ldots, T$; illustrations of forecasting, handling missing data, and quantile estimation

Table 1: Computational complexity of the regression step for several XMC filters. The estimates are based on the QR decomposition for linear regression with least squares, the complexity of $O(CN\log(N))$ for a single tree with $C = WN_y$ covariates, and the common choice of $\sqrt{C}$ split variables for RFs (Hastie et al., 2009).

XMC filter	Linear	GB	RF and QRF
Complexity	$O(N_x TNC^2)$	$O(N_x TCN\log(N))$	$O(N_x T\sqrt{C}N\log(N))$

can be found in Appendix B. In all applications the noise terms ε_t^x and ε_t^y are assumed to be mutually and serially independent, as well as independent of the initial state x_1.

3.1 Nonlinear filtering

Consider the following nonlinear model for a univariate time series y_t as given by

$$\begin{aligned} y_t &= \frac{x_t^2}{20} + \varepsilon_t^y, & \varepsilon_t^y &\sim \mathrm{N}(0, \sigma_y^2), \\ x_{t+1} &= \frac{1}{2}x_t + \frac{25x_t}{1+x_t^2} + 8\cos(1.2(t+1)) + \varepsilon_t^x, & \varepsilon_t^x &\sim \mathrm{N}(0, \sigma_x^2), \end{aligned} \tag{6}$$

with $x_1 \sim \mathrm{N}(0,1)$ and the static parameters set to $\sigma_x^2 = 0.1$ and $\sigma_y^2 = 1$ as in Kitagawa (1996). This model is a special case of the SSM in (1) and is often used for illustrating the performance of nonlinear filters. We used the above model to simulate a path of length $T = 100$ and predicted the states using the corresponding observations with several XMC filters for $N = 10^5$ paths.

For comparison we used the bootstrap filter (BF; Gordon et al., 1993), which is a standard version of the particle filter. This method requires an importance sampler to draw N values of the states $x_t^{(i)}, i = 1, \ldots, N$, which are called the particles. With the BF, the particles are drawn using $x_1^{(i)} \sim p(x_1)$ and $x_{t+1}^{(i)} = s_t\big(x_t^{(i)}, z^{(i)}\big)$ with $z^{(i)}$ a draw of ε_t^x, and they are weighted to form a discrete approximation to the density $p(x_t|y_{1:t})$. This yields the approximation $\mathbb{E}\left[x_t|y_{1:t}\right] \approx \sum_{i=1}^N \omega_t^{(i)} x_t^{(i)}$ with convex weights $\omega_t^{(i)} \propto \omega_{t-1}^{(i)} p(y_t|x_t^{(i)})$ and $\omega_0^{(i)} = 1/N$. To prevent the weights from degenerating, the particles were resampled whenever the effective sample size, as defined by

$$\mathrm{ESS}_t = \frac{1}{\sum_{i=1}^N \left(\omega_t^{(i)}\right)^2} \in [1, N], \tag{7}$$

dropped below $N/2$. We set the number of particles to 10^7 to ensure a highly accurate approximation to the filtering expectations.

Figure 2 (a) presents the simulated observations, Part (b) shows the true and filtered states based on the BF and GB-XMC filter, while Part (c) shows the differences with the BF for several XMC filters. From the difference in scales between Parts (b) and (c) we find that the GB and RF-XMC filters are generally adequate, while the linear filter (Lin-XMC) is not. The latter remains unchanged when N is increased, which indicates that the expectations $\mathbb{E}\left[x_t|y_{1:t}\right]$ are inherently nonlinear in the observations. This example demonstrates the need for general regression methods like GB and RF in the XMC method.

3.2 Real-time speed and accuracy

This section discusses the relation between real-time speed and accuracy of the XMC method. In real-time applications the observations $y_{1:t}$ are not known in advance, which means that the estimated regression functions must be accurate on most of their domain. It is therefore expected that a larger number of draws N is needed to achieve the same

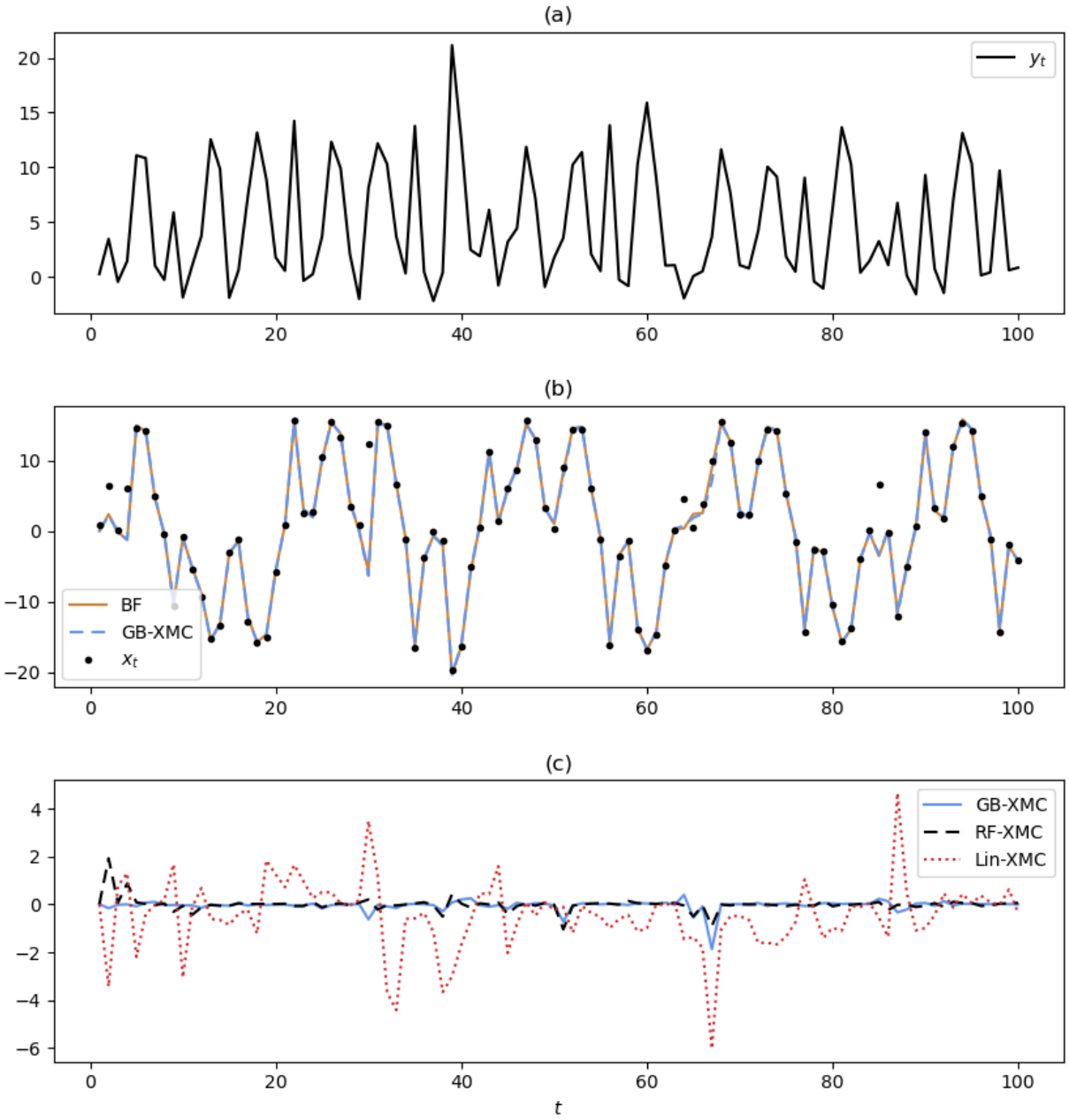

Figure 2: Analysis of a simulated path from the nonlinear model in (6): (a) observations; (b) true and filtered states by the BF and GB-XMC filter; (c) differences with BF for several XMC filters. The BF is based on 10^7 particles; the XMC filters are based on 10^5 simulated paths.

accuracy as other simulation-based methods that provide direct point estimates (e.g., particle filters). However, an important property of the XMC method is that most of the computations take place in the simulation and regression steps, which can be performed off-line. The on-line (or real-time) phase then only consists of the prediction step, which is computationally light. Furthermore, it is expected that the impact of increasing N on computing the predictions is small, so that the desired level of accuracy can be achieved without compromising the method's speed in real time.

To illustrate this last point, we performed a simulation study using the nonlinear

Table 2: Results from simulation study based on the nonlinear model in (6): overall root mean squared error (RMSE) and runtimes (computer execution time in seconds) based on 10^4 test paths for the bootstrap filter (BF) and gradient boosting XMC filter methods with various number of draws N.

$\log_{10}(N)$	3		4		5	
Method	BF	XMC	BF	XMC	BF	XMC
RMSE	1.688	1.858	1.664	1.709	1.662	1.674
runtime off-line	-	72.1	-	437.2	-	4240.6
runtime on-line	361.3	**3.0**	1295.7	**2.4**	13723.2	**3.4**

model in (6). The root mean squared error (RMSE) and runtimes of the GB-XMC filter are compared with those of the BF for various values of N. Table 2 shows the results from the simulation study, in which the path length was set to $T = 100$, and 10^4 test paths were used to estimate the performance. As expected, the BF was more accurate than the XMC filter for an equal number of draws. Most noteworthy about the results is that the time spent in the on-line phase by the XMC filter is several orders of magnitude smaller. These times are impacted by the number of draws only indirectly—via the estimated regression functions, with runtimes that are not necessarily increasing in N. In this case, the runtime is larger for $N = 10^3$ than for $N = 10^4$ because the GB method has a tendency to overfit the training data for small samples, which results in function estimates that are more complex and, therefore, more expensive to evaluate. By contrast, the BF incurs all its computing costs in real time, and the runtimes are directly impacted by the number of draws. The results illustrate that particle filters have an inherent trade-off between real-time speed and accuracy, whereas XMC filters do not have this trade-off. Given that the computational "bottleneck" in Algorithm 1 can be executed off-line, the XMC method is particularly suitable for real-time applications.

3.3 Intractable model densities

The stochastic volatility (SV) model is often used for the modeling of time series of daily financial returns. We consider the SV model with stable measurement noise (e.g., Vankov, Guindani, & Ensor, 2019) given by

$$\begin{aligned} y_t &= \exp(x_t/2)\varepsilon_t^y, & \varepsilon_t^y &\sim \mathrm{S}(\alpha, \beta), \\ x_{t+1} &= \mu + \phi(x_t - \mu) + \sigma_x \varepsilon_t^x, & \varepsilon_t^x &\sim \mathrm{N}(0, 1), \end{aligned} \tag{8}$$

where x_t represents the unobserved log variance, with initialisation $x_1 \sim \mathrm{N}(\mu, \sigma_x^2/(1-\phi^2))$ and static parameters $\mu \in \mathbb{R}$, $|\phi| < 1$, and $\sigma_x > 0$. Furthermore, $\mathrm{S}(\alpha, \beta)$ denotes the first parametrization of the standard univariate stable distribution as in Nolan (2009), with tail index parameter $\alpha \in (0, 2]$ and asymmetry parameter $\beta \in [-1, 1]$. Except for a few specific choices of the parameters, the density is not available in closed form, hence the characteristic function is used to describe the distribution:

$$\mathbb{E}[\exp(iu\varepsilon_t^y)] = \begin{cases} \exp\left(-|u|\left[1 + i\beta\frac{2}{\pi}(\operatorname{sgn} u)\log|u|\right]\right) & \text{if } \alpha = 1, \\ \exp\left(-|u|^\alpha\left[1 - i\beta\tan(\frac{\pi\alpha}{2})(\operatorname{sgn} u)\right]\right) & \text{otherwise.} \end{cases}$$

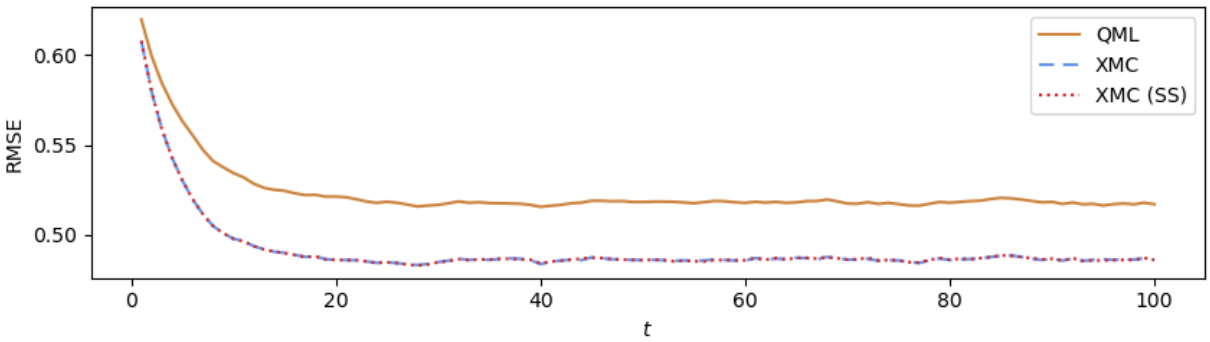

Figure 3: Results from simulation study based on the SV model in (8): the RMSEs over time are depicted for the quasi-maximum likelihood (QML) filter and the gradient boosting XMC filter. The latter is applied with $N = 10^5$ simulated paths, and with and without the steady state (SS) modification of Section 4.2.

The stable distribution has finite moments of any order less than α, and an important property of the 1-parametrization is $\mathbb{E}[\varepsilon_t^y] = 0$ if $\alpha > 1$, such that the first moment is not impacted by the tail and asymmetry parameters.

We perform a simulation study using the SV model in (8) with the parameter choice from Vankov et al. (2019), that is, $\mu = -0.2, \phi = 0.95, \sigma_x = 0.2, \alpha = 1.75$, and $\beta = 0.1$. The path length is set to $T = 100$, the number of test paths for evaluating the performance of the filters is set to 10^5, and the number of paths for the XMC method is also set to $N = 10^5$.

To compare the performance of the XMC method, we consider the quasi-maximum likelihood (QML) filter of Harvey, Ruiz, and Shephard (1994) which remains valid without a tractable observation density. The method is based on transforming the observations by $\widetilde{y}_t = \log y_t^2$ to cast the SV model into a linear state space form, that is,

$$\begin{aligned} \widetilde{y}_t &= x_t + 2\widetilde{\varepsilon}_t^y, \\ x_{t+1} &= (1-\phi)\mu + \phi x_t + \sigma_x \varepsilon_t^x, \end{aligned}$$

with $\widetilde{\varepsilon}_t^y = \log|\varepsilon_t^y|$. Although $\widetilde{\varepsilon}_t^y$ is not normally distributed, one can assume it is, so that the KF can be used to act as an approximate filter for x_t. This normal approximation matches the first two moments of $\widetilde{\varepsilon}_t^y$, which are given in Lemma 3.19 of Nolan (2009).

Figure 3 shows the RMSE at different time points for the QML (brown) and GB-XMC (blue) filtered states; the red graph will be discussed in Section 4.2. It can be seen that shortly after $t = 20$, the RMSE stops decreasing for both filters, suggesting a relatively small contribution from further lags. The overall RMSE is 0.492 for the XMC filter, a substantial improvement over the RMSE of 0.524 attained by the QML filter.

3.4 High-dimensional filtering

To investigate the performance of the XMC method for varying sizes of the state vector, we conduct a simulation study using the following vector autoregressive model with noise (VARN) as given by

$$\begin{aligned} y_t &= x_t + \varepsilon_t^y, & \varepsilon_t^y &\sim \text{Cauchy}(0, I_d), \\ x_{t+1} &= Ax_t + \varepsilon_t^x, & \varepsilon_t^x &\sim \text{N}(0, I_d), \end{aligned} \tag{9}$$

with vector variables $x_t, y_t, \varepsilon_t^x, \varepsilon_t^y \in \mathbb{R}^d$ for some $d \in \mathbb{N}$, I_d is the $d \times d$ identity matrix, and we set $A = 0.9 \cdot I_d$ with $x_1 \sim \mathrm{N}(0, (1 - 0.9^2)^{-1} I_d)$. The measurement noise vector ε_t^y is multivariate Cauchy with location zero and scale matrix I_d, which is proposed by van Leeuwen (2003) to avoid degeneracy of the particle filter weights in the context of a high-dimensional SSM. This specification exploits the fact that if the observations become less informative on the states, their posterior distribution will be more similar to the prior, thereby decreasing the variance of the weights. The model in (9) is a dynamic version of the multivariate Cauchy model used by Bengtsson, Bickel, and Li (2008) to show that, in comparison with the Gaussian case, these fat-tailed observation errors are able to considerably slow down the degeneracy of the particle weights as $d \to \infty$.

We simulate paths of the states and observations of length $T = 50$ using the VARN model with dimensions $d \in \{25, 50, 100, 200\}$ and filter the states using the corresponding observations. The number of test paths is set to $10^5/d$, such that all results are based on 10^5 state elements. The performance of the XMC filter is compared with the bootstrap filter of Gordon et al. (1993). For both methods, we set the number of draws to $N = 10^4$, which is a-typically large for particle filters. As discussed in Section 3.2, these quantities cannot be compared directly since only the number of particles increases the real-time duration of the corresponding filter. Hence, the results are indicative of the real-time accuracy that can be attained by the BF, but they merely represent a lower bound on the accuracy of the XMC filter.

Figure 4 shows the RMSE (based on the average MSE over the state elements) against time attained by the BF and GB-XMC filter for the selected model dimensions. As d increases, the BF becomes less accurate and its RMSE grows towards (and eventually exceeds) the RMSE of the unconditional mean. The performance of the XMC filter remains markedly steady as the model dimension increases. Table 3 shows the overall RMSE for both methods, as well as the average value of ESS_t/N for the BF based on (7). Up to $d = 100$, the latter number decreases with the dimension, which explains why the performance of the BF deteriorates despite the extra available information: the variance of the estimates increases because the weight becomes more concentrated on a smaller share of the particles. Starting at $d = 100$, there are several occurrences in which all BF particle weights evaluated to zero; in such cases, we resort to resetting the weights to $1/N$, which is justified by the fact that all weights are numerically equivalent before the reset. The increased number of resets at $d = 200$ causes the average ESS_t/N value to increase again but, as expected, there is no corresponding improvement in accuracy.

The deterioration of the BF is bound to occur for all particle filters because the weights are proportional to $p(y_t|x^{(i)})$, and therefore, to those of the BF. More generally, the

Table 3: Results from simulation study based on the VARN model in (9): overall RMSE of the bootstrap filter (BF) and the gradient boosting XMC filter for selected model dimensions, and the average value of ESS_t/N for the BF based on (7).

d	25		50		100		200	
Method	BF	XMC	BF	XMC	BF	XMC	BF	XMC
RMSE	1.257	**1.199**	1.511	**1.195**	1.828	**1.189**	2.148	**1.196**
Average ESS_t/N	0.098	-	0.072	-	0.053	-	0.088	-

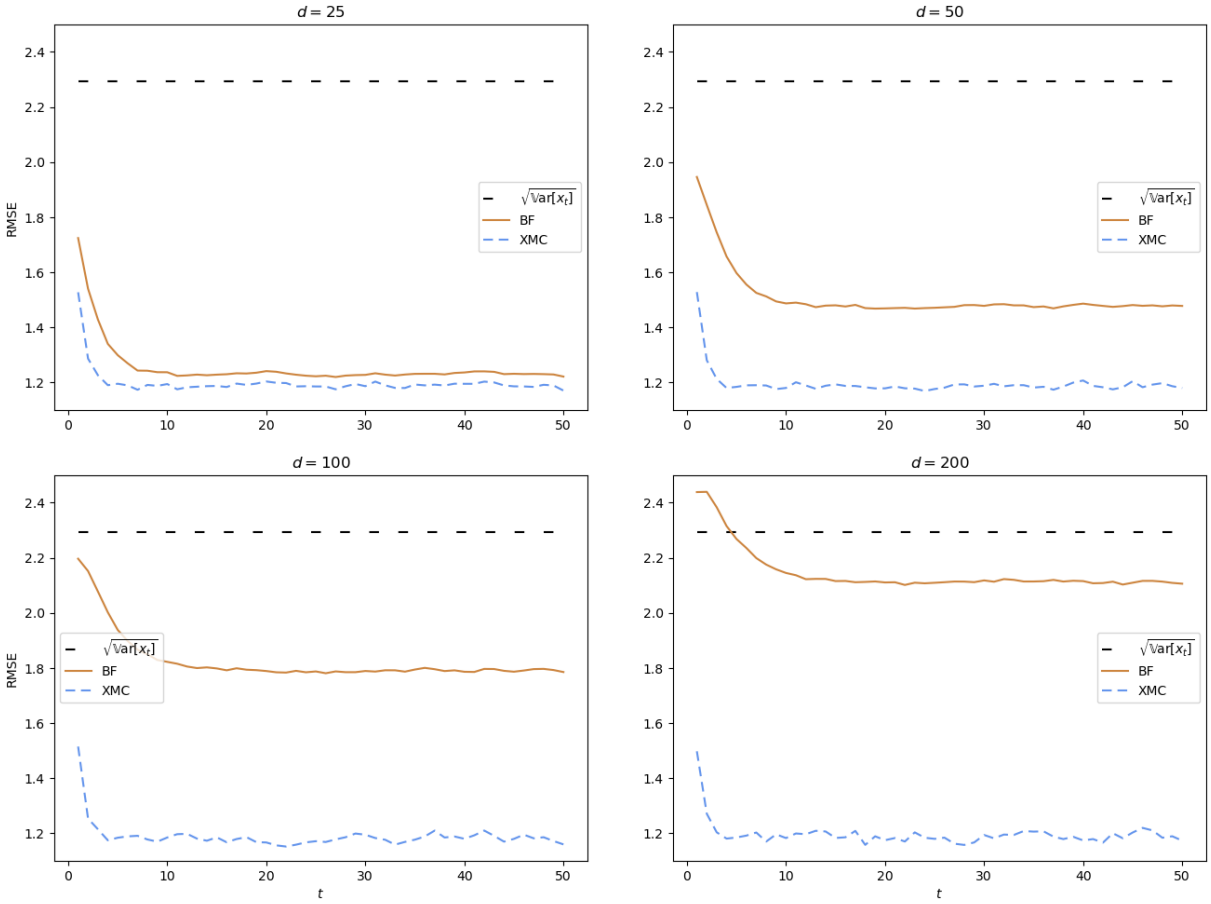

Figure 4: Results from simulation study for the VARN model in (9): the RMSE against time attained by the bootstrap filter (BF) and the gradient boosting XMC filter for selected model dimensions d. The unconditional standard deviation of the states $\sqrt{\mathbb{V}\text{ar}\left[x_t\right]}$ is the RMSE of the unconditional mean.

weights degeneracy problem is a manifestation of the well-known curse of dimensionality for importance sampling. On the other hand, the accuracy of the XMC filter keeps improving until $d = 100$ and remains almost unchanged for $d = 200$. Since each element of the state vector is treated separately, increasing its size generally does not have a negative impact on the filter's accuracy. In addition, the use of tree-based methods makes it possible to handle large numbers of covariates, even if many of them are uninformative. By relying on regression instead of importance sampling, the XMC method is able to overcome the curse of dimensionality.

4 Filter properties

In this section, we consider filter properties for the case when the number of simulated paths N or the time series length T becomes large. Section 4.1 provides conditions under which an XMC filter defined by Algorithm 1 converges to an optimal filter as N diverges to infinity. In this analysis T is assumed to be finite, as is the case in all applications. The case of large (but finite) T is considered from a practical perspective in Section

4.2, which introduces an extension to Algorithm 1 that may yield large computational savings for long time series. The latter are common with data on natural phenomena (e.g., astronomical and meteorological data) and in fields where measurements with daily or higher frequencies are often available, such as financial econometrics.

4.1 Convergence

For the purpose of generality, our filter convergence analysis takes place in the context of a general signal extraction problem characterized by the instance of the SSM in (1), the conditioning sets $Y_t \subseteq y_{1:T}$ for $t = 1, \dots, T$, and the loss function L; these specifications will be considered *given*. A filtering example ($Y_t = y_{1:T}$) with the Gaussian LL model in (3) will be used to settle ideas.

In the following, we use $\mathcal{Y}$ to denote the collection of all paths $y_{1:T}$ that are realizable in the sense that $p(y_{1:T}) > 0$ (Frühwirth-Schnatter, 1994). In addition, we assume the existence of an optimal filter.

Assumption 1 (Existence of optimal filter)**.** *There exists an optimal filter* $\{f_t^*(Y_t)\}_{t=1}^T$ *such that for all* $y_{1:T} \in \mathcal{Y}$ *and* $t = 1, \dots, T$*,*

$$f_t^*(Y_t) \in \arg\min_{c \in \mathbb{R}} \mathbb{E}\left[L\left(x_t - c\right) \middle| Y_t\right]. \tag{10}$$

An optimal filter is thus defined as a sequence of prediction functions f_t^* that are pointwise minimizers of the expected loss, where the "points" are the conditioning sets Y_t. Here it should be noted that $Y_t \subseteq y_{1:T}$ is fixed whenever $y_{1:T}$ is, a property that we will use repeatedly. In general, the minimizers f_t^* need not be unique, and the same therefore holds for the optimal filter. This may happen, for example, with the all-or-nothing loss if $p(x_t|Y_t)$ is multimodal; it may then be beneficial for the stability of the filter to use a selection function for choosing between the minimizers.

Example (Optimal filter)**.** *For a filtering problem with the squared error loss, the objective function in (10) becomes*

$$\mathbb{E}\left[L\left(x_t - c\right) \middle| Y_t\right] = \mathbb{E}[(x_t - c)^2 | y_{1:t}].$$

If this objective function exists, it is well known that the corresponding minimizer f_t^* *is given by the conditional expectation of* x_t *given* $y_{1:t}$*,*

$$f_t^*(Y_t) = \mathbb{E}[x_t | y_{1:t}].$$

For linear Gaussian SSMs, the optimal filter is given by the KF, which, for the Gaussian LL model in (3), reduces to the recursion

$$\begin{aligned} \mathbb{E}[x_t|y_{1:t}] &= (1 - K_t)\,\mathbb{E}[x_t|y_{1:t-1}] + K_t y_t, & \mathbb{V}\mathrm{ar}\left[x_t|y_{1:t}\right] &= \mathbb{V}\mathrm{ar}\left[x_t|y_{1:t-1}\right](1 - K_t), \\ \mathbb{E}[x_{t+1}|y_{1:t}] &= \mathbb{E}[x_t|y_{1:t}], & \mathbb{V}\mathrm{ar}\left[x_{t+1}|y_{1:t}\right] &= \mathbb{V}\mathrm{ar}\left[x_t|y_{1:t}\right] + \sigma_x^2, \end{aligned} \tag{11}$$

for $t = 1, \dots, T$*, with* $K_t = (1 + \sigma_y^2 / \mathbb{V}\mathrm{ar}\left[x_t|y_{1:t-1}\right])^{-1}$ *and* $y_{1:0} := \emptyset$ *(Durbin & Koopman, 2012, Ch. 2).*

To enable a formal discussion of the XMC filter and its convergence, we will use the following definition for the covariate sets.

Definition 1 (Covariate set)**.** A *covariate set* $\widetilde{Y}_t \subseteq Y_t$ is a subset of the conditioning set at time t. The power set $\mathcal{P}(Y_t)$ is the collection of all feasible covariate sets at time t.

Like the conditioning set, each feasible covariate set $\widetilde{Y}_t \in \mathcal{P}(Y_t)$ consists of elements from $y_{1:T}$, which are therefore fixed whenever $y_{1:T}$ is, and random otherwise.

In general, the optimal size of the covariate set used in the regressions varies with N, an illustration of which is given in Appendix B.2. The XMC filter accounts for this dependency by using regularized covariate sets $\widetilde{Y}_t^N \in \mathcal{P}(Y_t)$, which are defined in terms of a window size (e.g., via (5) for filtering) that minimizes the loss for a separate validation sample of size $N_{\text{val}} = [c_{\text{val}} N]$ with, say, $c_{\text{val}} = 0.1$. The covariate sets can of course be regularized in other ways, such as by adding a penalty term to the objective function for the window size. In this section we assume only that the regularized covariate sets are such that they converge to the corresponding conditioning sets.

Assumption 2 (Convergence of regularized covariate sets)**.** *For $t = 1, \dots, T$ we have*

$$\lim_{N\to\infty} \widetilde{Y}_t = Y_t \qquad \textit{almost surely.}$$

Having defined the regularized window size, the XMC filter can be represented as the set of prediction functions

$$\left\{ \widehat{f}_t^N\left(\widetilde{Y}_t^N\right) \right\}_{t=1}^T,$$

and we note that each term defines a different function estimator depending on the composition of $\widetilde{Y}_t^N$, the regularized covariate set. This is illustrated in the next example.

Example (Linear XMC filter)**.** *Consider the XMC filter defined by using a linear regression function with parameters estimated by the least squares (LS) method,*

$$\widehat{f}_t^N\left(\widetilde{Y}_t^N\right) = \widehat{\beta}_0^{LS} + \sum_{y_j \in \widetilde{Y}_t^N} \widehat{\beta}_j^{LS} y_j, \tag{12}$$

for $t = 1, \dots, T$, with $\widetilde{Y}_t^N$ such that Assumption 2 holds. Each feasible value of $\widetilde{Y}_t^N \in \mathcal{P}(Y_t)$ defines a different function estimator. For instance, with filtering at time $t = 2$ we have the possible estimators

$$\begin{aligned} \widehat{f}_2^N(\emptyset) &= \widehat{\beta}_{0,0}^{LS}, & \widehat{f}_2^N(\{y_1\}) &= \widehat{\beta}_{0,1}^{LS} + \widehat{\beta}_{1,1}^{LS} y_1, \\ \widehat{f}_2^N(\{y_2\}) &= \widehat{\beta}_{0,2}^{LS} + \widehat{\beta}_{2,2}^{LS} y_2, & \widehat{f}_2^N(\{y_1, y_2\}) &= \widehat{\beta}_{0,12}^{LS} + \widehat{\beta}_{1,12}^{LS} y_1 + \widehat{\beta}_{2,12}^{LS} y_2, \end{aligned}$$

each of which could be used to predict the state x_2.

We shall say that the XMC filter *converges to an optimal filter* $\{f_t^*\}_{t=1}^T$ if for all realizable paths $y_{1:T} \in \mathcal{Y}$,

$$\sup_{t=1,\dots,T} \left| \widehat{f}_t^N\left(\widetilde{Y}_t^N\right) - f_t^*(Y_t) \right| \xrightarrow{\text{a.s.}} 0 \qquad \text{as} \qquad N \to \infty. \tag{13}$$

The above convergence is thus pointwise over the realizable paths (i.e., the observed data) rather than uniform, where the latter is typically not manageable due to $\mathcal{Y}$ being non-compact; in most applications $\mathcal{Y}$ coincides with $\mathbb{R}^{N_y \times T}$. On the other hand, the almost sure mode of convergence pertains to the randomness that stems from the simulation step in Algorithm 1, which should become irrelevant as $N \to \infty$.

Lastly, consider the limit functions

$$f_t^\infty(\widetilde{Y}_t) = \lim_{N \to \infty} \widehat{f}_t^N(\widetilde{Y}_t) \tag{14}$$

for $t = 1, \dots, T$, where the notation emphasizes that the covariate sets $\widetilde{Y}_t \in \mathcal{P}(Y_t)$ remain invariant with respect to N. We then require that the limits $f_t^\infty(\widetilde{Y}_t)$ coincide with a minimizer when $\widetilde{Y}_t = Y_t$.

Assumption 3 (Convergence of function estimators)**.** *For all $y_{1:T} \in \mathcal{Y}$ and $t = 1, \dots, T$ it holds that*

$$f_t^\infty(Y_t) = f_t^*(Y_t) \qquad \text{almost surely,} \tag{15}$$

for some minimizer f_t^* of (10).

The following lemma is then a straightforward result.

Lemma 1 (Filter convergence)**.** *Under Assumptions 1–3 the XMC filter converges almost surely to an optimal filter $\{f_t^*\}_{t=1}^T$, so that the condition in (13) is satisfied for all $y_{1:T} \in \mathcal{Y}$.*

The next example provides an application of Lemma 1 to illustrate filter convergence in the context of the Gaussian LL model.

Example (Filter convergence)**.** *Noting that the conditional variances in (11) do not depend on y_t, it follows that the conditional expectation computed by the KF is a linear function of the observations, so that $f_t^*(Y_t) = f_t^*(y_{1:t}) = \beta_0 + \sum_{j=1}^t \beta_j y_j$ for appropriate constants $\beta_k \in \mathbb{R}$, $k = 0, \dots, t$. By Lemma 1, a linear XMC filter defined by (12) converges to the KF as $N \to \infty$ for any choice of regularized covariate sets that satisfy Assumption 2.*

Lemma 1 establishes filter convergence in a general setting where the method of regularization is left unspecified. The result therefore also applies to alternative approaches, such as penalization of the objective function, provided that Assumption 2 holds. With the loss-based approach from Section 2, we can expect this assumption to hold for most signal extraction problems of practical interest. A discussion and formal result of convergence for our specific regularization method requires introducing corresponding notational machinery and is therefore placed in Appendix C.

Part of the generality of Lemma 1 is due to its use of Assumption 3, which enabled a separation of the convergence analysis for the filter from the specific regression method used; see Appendix A for references to analyses of the latter type. The related question of which regression method is optimal with *finite* N remains a challenging one because the answer will vary with the signal extraction problem, and most nonlinear non-Gaussian SSMs admit little knowledge about the form of the corresponding optimal filter(s). In this light, we note that convergence (or consistency) will typically be a greater concern than other estimator properties because the simulated data can be generated at will and,

as discussed in Section 3.2, increasing N does not effectively impact the filter's speed in real time. In the absence of further knowledge, a prudent strategy is therefore to use general regression models (e.g., GB, RF) with the aim of establishing filter convergence, as was done in the illustrations.

4.2 Steady state

This section describes a *steady state* (SS) extension to Algorithm 1 that may yield large computational savings for long time series. The idea is to stop performing regressions after some time t_{ss} and use the function estimate $\widehat{f}_{t_{\text{ss}}}^N$ for prediction at the remaining times $t > t_{\text{ss}}$. A minimal requirement for such approach to be sensible is that the covariate sets change over time only through shifts of their elements. More specifically, this means that there exists some $t_W \in \mathbb{N}$ such that

$$\widetilde{Y}_{t+1} = \left\{ y_{j+1} \middle| y_j \in \widetilde{Y}_t \right\} \qquad \forall\, t \geq t_W. \tag{16}$$

For example, the above condition is satisfied with $t_W = W$ for filtering with covariate sets defined by (5). Each time $t \geq t_W$ is then a potential candidate for t_{ss}.

To determine whether the impact of the SS approach is acceptable, we propose an intuitive estimate of the largest increase in the loss by noting that the regression functions are expected to differ more from each other the further in time they are apart. We therefore check for $t = t_W, \ldots, T$ if

$$\sum_{i=1}^{N_{\text{val}}} L\left(x_T^{\langle i \rangle} - \widehat{f}_t^N\left(\widetilde{Y}_T^{\langle i \rangle}\right)\right) \leq (1 + c_{\text{ss}}) \sum_{i=1}^{N_{\text{val}}} L\left(x_T^{\langle i \rangle} - \widehat{f}_T^N\left(\widetilde{Y}_T^{\langle i \rangle}\right)\right) \tag{17}$$

holds, with $c_{\text{ss}} \geq 0$ a chosen tolerance level and superscripts $\langle i \rangle$ indicating cases from the validation sample $(x_{1:T}^{\langle i \rangle}, y_{1:T}^{\langle i \rangle}), i = 1, \ldots, N_{\text{val}}$. If the condition in (17) is satisfied at time t, we say that an SS has been reached and set $t_{\text{ss}} = t$, after which the estimate $\widehat{f}_{t_{\text{ss}}}^N$ can be used to circumvent the remaining regressions.

We illustrate the SS approach by applying it in the simulation study for the SV model from Section 3.3. Figure 3 shows the performance of the SS and regular versions of the GB-XMC filter and the QML filter. The SS estimate $\widehat{f}_{t_{\text{ss}}}^N$ corresponds to $t_{\text{ss}} = 25$ ($c_{\text{ss}} = 0; W = 21$), such that only a small part of the regressions had to be performed. The SS approach is seen to have no effective impact on the filter's accuracy. This result is as expected because the SV model in (8) is strictly stationary, which implies that the limit estimators $f_t^\infty(\widetilde{Y}_t)$ for covariate sets satisfying the condition in (16) are equivalent for all $t \geq t_W$. The proposed approach should, however, be applicable more generally. For example, applying the SS approach to the filtering exercise with the non-stationary LL model in (3) yielded results identical to Figure 1 (which are therefore omitted) with $t_{\text{ss}} = 21$ for $c_{\text{ss}} = 0$. A more precise specification of the models for which the SS concept is suitable will be left to future research.

5 Conclusion

This paper introduces a novel simulation-based filtering method for computing time-varying conditional means, quantiles, and modes, and for predicting latent variables in

general. The XMC method relies on generating artificial samples of data from the joint distribution of the model and estimating quantities of interest via extremum estimation. Convergence to an optimal filter is shown to hold under mild regularity conditions, and the filtering characteristics are illustrated in the applications. The method is applicable to any model from which data can be simulated and is not liable to the curse of dimensionality. Furthermore, the use of extremum estimation allows for any conditioning set, including data sets with missing entries and unequal spacing. The filtering method also places the computational burden predominantly in the off-line phase, which makes it particularly suitable for real-time applications.

References

Amemiya, T. (1985). *Advanced econometrics*. Harvard University Press, Cambridge MA.

Bender, C., & Steiner, J. (2012). Least-squares Monte Carlo for backward SDEs. In *Numerical methods in finance* (pp. 257–289). Springer.

Bengtsson, T., Bickel, P., & Li, B. (2008). Curse-of-dimensionality revisited: Collapse of the particle filter in very large scale systems. In *Probability and Statistics: Essays in Honor of David A. Freedman* (pp. 316–334). Institute of Mathematical Statistics.

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).

Biau, G., & Cadre, B. (2021). Optimization by gradient boosting. In *Advances in contemporary statistics and econometrics* (pp. 23–44). Springer.

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

Denault, M., & Simonato, J.-G. (2017). Dynamic portfolio choices by simulation-and-regression: Revisiting the issue of value function vs portfolio weight recursions. *Computers & Operations Research*, *79*, 174–189.

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

Durbin, J., & Koopman, S. J. (2012). *Time series analysis by state space methods*. Oxford university press.

Engle, R. (2002). New frontiers for arch models. *Journal of Applied Econometrics*, *17*(5), 425–446.

Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. *Annals of Statistics*, 1189–1232.

Frühwirth-Schnatter, S. (1994). Data augmentation and dynamic linear models. *Journal of Time Series Analysis*, *15*(2), 183–202.

Gobet, E., Lemor, J.-P., & Warin, X. (2005). A regression-based monte carlo method to solve backward stochastic differential equations. *The Annals of Applied Probability*, *15*(3), 2172–2202.

Gordon, N. J., Salmond, D. J., & Smith, A. F. (1993). Novel approach to nonlinear/non-gaussian bayesian state estimation. In *IEE 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.

Green, A. (2015). *Xva: credit, funding and capital valuation adjustments*. John Wiley & Sons.

Harvey, A. C., Ruiz, E., & Shephard, N. (1994). Multivariate stochastic variance models. *The Review of Economic Studies*, *61*(2), 247–264.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The elements of statistical learning: data mining, inference, and prediction*. Springer Science & Business Media.

Hayashi, F. (2000). *Econometrics*. Princeton.

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Transactions of the ASME–Journal of Basic Engineering*, *82*(Series D), 35–45.

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.

Longstaff, F. A., & Schwartz, E. S. (2001). Valuing american options by simulation: a simple least-squares approach. *The Review of Financial Studies*, *14*(1), 113–147.

McFadden, D. (1989). A method of simulated moments for estimation of discrete response models without numerical integration. *Econometrica*, 995–1026.

Meinshausen, N. (2006). Quantile regression forests. *Journal of Machine Learning Research*, *7*(6).

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

Pitt, M. K., & Shephard, N. (1999). Filtering via simulation: Auxiliary particle filters. *Journal of the American Statistical Association*, *94*(446), 590–599.

Scornet, E., Biau, G., & Vert, J.-P. (2015). Consistency of random forests. *The Annals of Statistics*, *43*(4), 1716–1741.

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.

van Leeuwen, P. J. (2003). A variance-minimizing filter for large-scale applications. *Monthly Weather Review*, *131*(9), 2071–2084.

Zeng, Y., & Wu, S. (2013). *State-space models: Applications in economics and finance* (Vol. 1). Springer.

Zhang, R., Langrené, N., Tian, Y., Zhu, Z., Klebaner, F., & Hamza, K. (2019). Dynamic portfolio optimization with liquidity cost and market impact: a simulation-and-regression approach. *Quantitative Finance*, *19*(3), 519–532.

Zhang, T., & Yu, B. (2005). Boosting with early stopping: Convergence and consistency. *The Annals of Statistics*, *33*(4), 1538–1579.

Appendix A Regression methods

A.1 Gradient boosting

Gradient boosting (GB; Friedman, 2001) is currently considered the standard "off-the-shelf" solution to data mining problems. The idea is as follows. Suppose that for the random variables X and Y a sample $(X^{(i)}, Y^{(i)}), i = 1, \ldots, N$ is available, and we wish to estimate a function $f^*(Y)$ that minimizes the expected loss with respect to X. Starting with some initial function f_0 and denoting by $\tau(Y; \psi)$ a regression tree with parameters $\psi \in \Psi$, the tree-based GB estimate of f^* is defined by applying for $m = 1, \ldots, M$:

$$\begin{aligned} \widehat{\psi}_m &\in \arg\min_{\psi \in \Psi} \sum_{i=1}^{N} \left(-\frac{\partial L(X^{(i)} - \widehat{X})}{\partial \widehat{X}}\bigg|_{\widehat{X} = f_{m-1}(Y^{(i)})} - \tau(Y^{(i)}; \psi) \right)^2, \\ f_m(Y) &:= f_{m-1}(Y) + \lambda \tau(Y; \widehat{\psi}_m), \end{aligned} \tag{18}$$

with learning rate parameter $\lambda \in (0, 1)$ that is used to control how rapidly the function f_m is updated. The idea is thus to iteratively fit regression trees to the negative gradient of the loss function—the direction of steepest descent—and to use an average of the trees as estimate of the regression function. For the normalized squared error loss $L(u) = u^2/2$, the first part of the boosting scheme in (18) becomes

$$\widehat{\psi}_m \in \arg\min_{\psi \in \Psi} \sum_{i=1}^{N} \left(X^{(i)} - f_{m-1}\left(Y^{(i)}\right) - \tau(Y^{(i)}; \psi) \right)^2,$$

which amounts to iteratively fitting a regression tree to the residuals of the previous step. The above special case provides an intuition for why GB is found in practice to work well in many different applications. In addition, the use of regression trees provides an internal feature selection process to deal with large numbers of covariates, even if only a small part of them is informative. Some convergence results have been established, for example by Biau and Cadre (2021), who use an $\mathcal{L}^2$-penalization for regularization, and T. Zhang and Yu (2005), who consider a general boosting procedure with early stopping.

A.2 Random forest and quantile regression forest

The random forest (RF; Breiman, 2001) is defined as an average of a large number of decorrelated regression trees that are grown using bootstrapped samples of the data. The idea is that regression trees are characterized by a low bias and high variance, and because the trees are identically distributed their average retains this low bias while benefiting from a reduced variance. A distinctive step in the RF procedure is that when growing a tree, each split decision in a terminal node is made using a subset of randomly selected covariates to reduce the correlation between the trees. Given a sample $(X^{(i)}, Y^{(i)}), i = 1, \ldots, N$ of random variables X and Y, the RF predictions of X can be represented directly in terms of the data by

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

The weights are defined as an average,

$$w_i(y) = \frac{1}{K}\sum_{k=1}^{K} w_i^k(y), \tag{19}$$

for K regression trees, each of which predicts the dependent variable X by taking the average over the corresponding variates in the leaf to which y is assigned. This may be represented by

$$w_i^k(y) = \frac{1_{\{i\in\mathbb{L}_k(y)\}}}{|\mathbb{L}_k(y)|},$$

with $\mathbb{L}_k(y)$ denoting the set of indices j corresponding to the values $Y^{(j)}$ in the leaf to which y is assigned and $|\mathbb{L}_k(y)|$ its number of elements. Convergence of the RF has been established by Scornet, Biau, and Vert (2015) in the context of additive models.

The quantile regression forest (QRF; Meinshausen, 2006) exploits the RF to perform quantile regression. Noting that for random variables X and Y the conditional CDF is defined by

$$P(X \leq x|Y = y) = \mathbb{E}[1_{\{X\leq x\}}|Y = y],$$

the QRF approximates the right-hand side by

$$\sum_{i=1}^{N} w_i(y)\ 1_{\{X^{(i)}\leq x\}},$$

with the weights as defined in (19). The quantile estimates can then be computed by inverting the above CDF estimate. This approach requires only a single RF to be fit for estimating any number of quantiles with monotonicity in the cumulative probabilities. Convergence of the QRF method is discussed in Meinshausen (2006, Sec. 4).

Appendix B Additional illustrations

This section contains several additional illustrations regarding covariate set regularization and filter convergence. The illustrations are based on the Gaussian LL model in (3) applied to measurements of the annual flow volume of the Nile taken at Aswan from 1871 to 1970. The static parameters were set to the maximum likelihood estimates $\sigma_x = 38.329$ and $\sigma_y = 122.877$, with $\mu_1 = 0$ and $\sigma_1^2 = 10^7$ to approximate diffuse initialization. More information on this application can be found in Durbin and Koopman (2012, Ch. 2).

B.1 Filter convergence

The use of regression easily accommodates prediction based on other conditioning sets than the one used for filtering. For example, Figure 5 shows the 1-period forecasts of the linear XMC filter ($N = 10^4$), which coincide with the ones based on the KF. At $t = 1$, the prediction is unconditional, resulting in the value $\mu_1 = 0$, while for $t > 1$ the forecasts equal the lagged predictions from filtering, $\mathbb{E}[x_{t-1}|y_{1:t-1}]$.

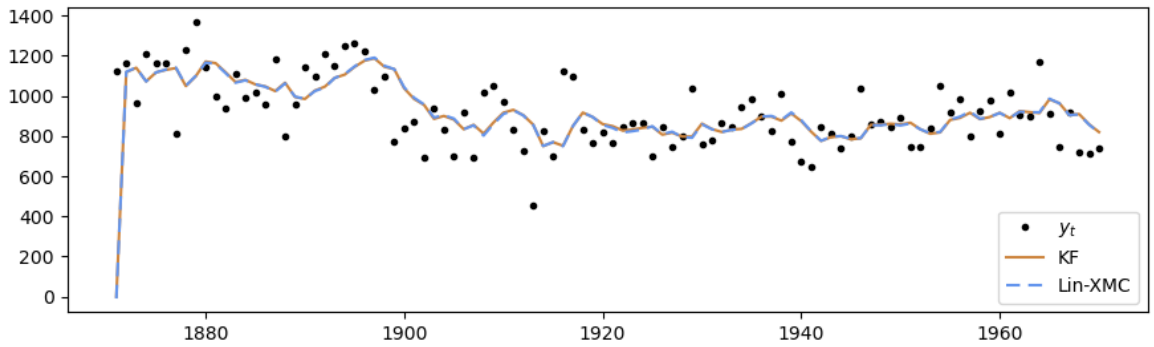

Figure 5: Forecasting analysis of the Nile data based on the LL model in (3): 1-period forecasts of the state from the KF and linear XMC filter with $N = 10^4$.

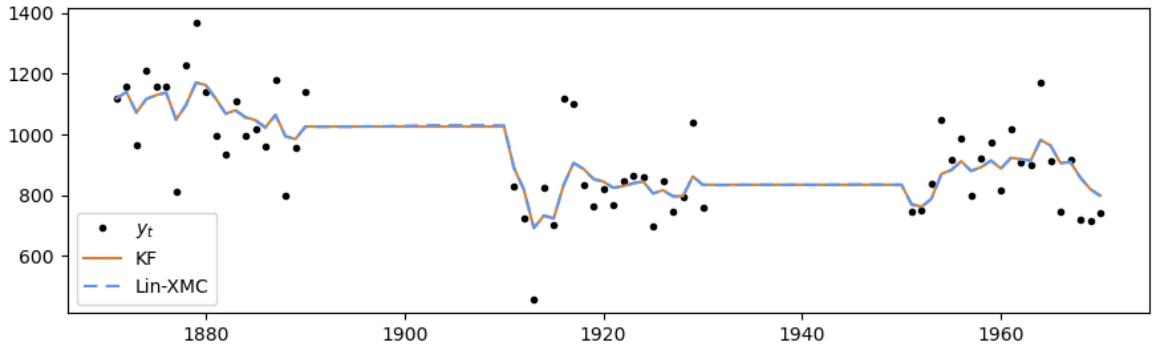

Figure 6: Filtering analysis of the partial Nile data based on the LL model in (3): filtered states from the KF and linear XMC filter with $N = 10^5$.

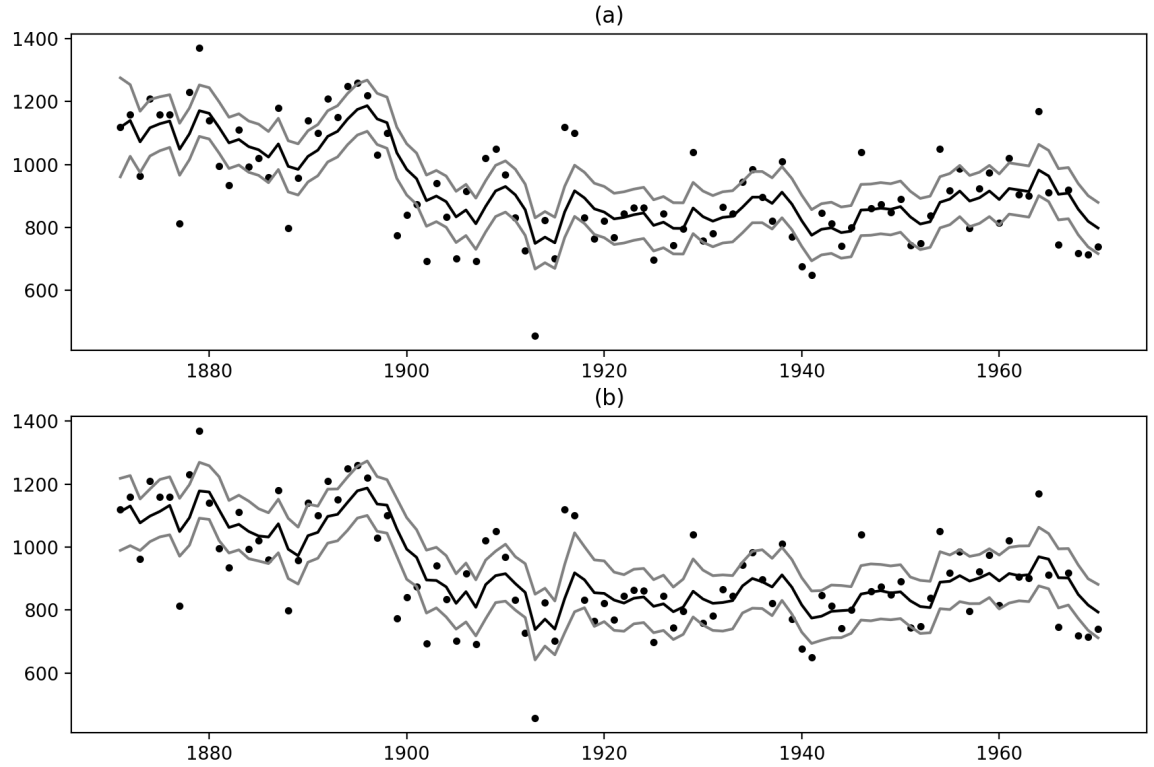

Figure 7: Filtering analysis of the Nile data based on the LL model in (3): state quantiles for cumulative probabilities $10\%, 50\%$, and 90%: (a) KF; (b) QRF-XMC filter with $N = 10^6$.

Another type of conditioning set results from the occurrence of missing data, which is common in practice. The XMC filter handles this issue by simply omitting the corresponding covariates when performing the regression. We illustrate the handling of missing data by treating the observations at time points $21, \ldots, 40$ and $61, \ldots, 80$ as missing. To deal with these longer sequences of missing data, the window size was set to 40. Figure 6 shows the filtered states from the linear XMC filter with $N = 10^5$ paths. The predictions are seen to coincide with those of the KF, which is able to treat missing data exactly (Durbin & Koopman, 2012, Ch. 4.10).

Lastly, Figure 7 (a) shows the filtered $10\%, 50\%$, and 90% state quantiles based on the KF. The corresponding estimates of the QRF-XMC filter with $N = 10^6$ are shown in Figure 7 (b), which are seen to be close to the true quantiles. As expected, the median estimates appear to be slightly more accurate than the more extreme quantiles.

B.2 Covariate set regularization

To investigate how the filter's performance is impacted by the window size, we performed a simulation study using the LL model in (3) with $T = 100$. We focus on the accuracy of the filtered state at the last time point, $\widehat{x}_T$, as a function of the window size, or equivalently, of the lower covariate set endpoint $\underset{\sim}{T} = T - W + 1$ with $\widetilde{Y}_t$ defined by (5). In particular, the RMSE of $\widehat{x}_T$ was computed for the linear XMC filter with $N \in \{10^3, 10^4\}$ based on a test sample of 10^5 paths and ten repetitions of Algorithm 1 for different seeds.

Figure 8 shows the results of the simulation study. As expected, the RMSE decreases with N, and it is seen to be non-monotonic in the lower endpoint. Adding recent observations as covariates initially improves the performance, but after some point the increase in variance from having to estimate more parameters outweighs the decrease in the bias with respect to $\mathbb{E}[x_T|y_{1:T}]$. Regarding the bias, we note that there are clear diminishing returns to adding covariates because the observations are dependent and decreasingly informative the more remote they are from the state. The optimal covariate window is seen to vary with N, which indicates that an increase in the complexity of the regression method is warranted once more data are available. For comparison, the RMSE is also shown for the KF, which computes $\mathbb{E}[x_T|y_{1:T}]$ exactly using the recursion in (11). For $N = 10^4$, the performance of the linear XMC filter with $\underset{\sim}{T} = 87$ ($W = 14$) is almost in-

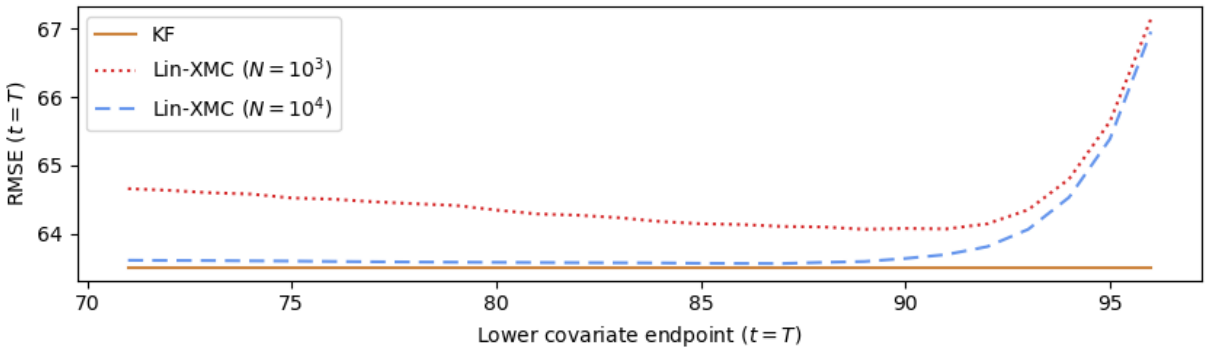

Figure 8: RMSE of $\widehat{x}_T$ in the Gaussian LL model based on the predictions for 10^5 simulated test paths. The results are shown for the KF and linear XMC filter with $N \in \{10^3, 10^4\}$ for various values of the lower covariate set endpoint, and the upper endpoint set to $T = 100$.

distinguishable from that of the optimal filter. The RMSE increases if the lower endpoint is altered, which underlines the importance of covariate set regularization.

Appendix C Window-based filter convergence

This section considers filter convergence for the specific regularization approach from Section 2, in which a window size parameter $W \in \{1, \ldots, T\}$ is used to govern all covariate sets. For example, with filtering this is done via (5). More generally, this approach considers the covariate sets $\widetilde{Y}_t = \widetilde{Y}_t(W)$ as given functions of W. We impose the following restrictions to ensure that these window-based covariate sets are growing in W and eventually coincide with the conditioning set.

Definition 2 (Window-based covariate sets)**.** For given conditioning sets Y_t, $t = 1, \ldots, T$, the *window-based covariate sets* $\widetilde{Y}_t(W)$ are such that for $W = 1, \ldots, T-1$ we have

$$\widetilde{Y}_t(W) \subseteq \widetilde{Y}_t(W+1) \subseteq \widetilde{Y}_t(|Y_t|) = Y_t,$$

where $|Y_t|$ denotes the size of the conditioning set.

Another element specific to the approach from Section 2 is that the regularized window is determined by minimizing the average loss at some suitable time point t^* based on a separate validation sample. It is well known that such use of independent samples ensures convergence to a minimizer of the expected loss. In this context, it is relevant to note that there are some SSMs for which not all of the observations are of added value to prediction. For example, if the state is an m-dependent process, such as the moving average model of order m,

$$x_t = \varepsilon_t^x + \sum_{j=1}^{m} \theta_j \varepsilon_{t-j}^x,$$

with mutually and serially independent noise terms ε_t^x and ε_t^y. In this case x_t is independent of y_{t-j} for $j > m$. The regularized window size, W^N, may then converge to a value smaller than $|Y_{t^*}|$ because some of the covariates are redundant. We introduce the following concept to deal with such cases.

Definition 3 (Infimum window size)**.** Let Assumption 1 hold, and let $\underline{W_t} \in \{1, \ldots, |Y_t|\}$ be the smallest value such that for all $W \geq \underline{W_t}$ there exists a minimizer f_t^* of (10) for which

$$f_t^\infty\big(\widetilde{Y}_t(W)\big) \overset{\text{a.s.}}{=} f_t^*(Y_t) \qquad \forall\ y_{1:T} \in \mathcal{Y},$$

with f_t^∞ as defined in (14). Then $\underline{W_t}$ is called the *infimum window size* for time t.

The infimum window size is essentially the smallest value to which a regularized window size could converge if it is based on minimizing the validation loss at time t; if there are no redundant observations, it follows that $\underline{W_t} = |Y_t|$. The regularization approach from Section 2 can then be characterized by the condition

$$W^\infty := \lim_{N\to\infty} W^N \in \big\{\underline{W_{t^*}}, \ldots, T\big\} \qquad \text{almost surely.} \tag{20}$$

The following result provides several sufficient conditions under which window-based XMC filters converge to an optimal filter.

Lemma 2 (Window-based filter convergence)**.** *Let Assumption 1 hold, and let the regularized covariate sets be defined in terms of a regularized window size W^N, so that $\widetilde{Y}_t^N = \widetilde{Y}_t(W^N)$ for $t = 1, \ldots, T$. Then convergence to an optimal filter is implied by any of the following conditions:*

(a) *Assumption 3 holds and $W^\infty \in \left\{ \max_{t=1,\ldots,T} |Y_t|, \ldots, T \right\}$ almost surely.*

(b) *The condition in (20) is satisfied and $\underline{W_{t^*}} = \max_{t=1,\ldots,T} \underline{W_t}$.*

(c) *The condition in (20) is satisfied and $\underline{W_{t^*}} = |Y_{t^*}| = \max_{t=1,\ldots,T} |Y_t|$.*

In Lemma 2, Condition (a) requires that the limit of the regularized window size is large enough to exceed all conditioning window sizes. It then follows by Definition 2 that each covariate set coincides with its corresponding conditioning set, such that Assumption 2 holds. This condition is easily verified for deterministic window sizes that grow at a prespecified rate.

Condition (b) and (c) pertain specifically to loss-based window sizes. The proposal to set $t^* = T$ for filtering and forecasting implies that $Y_t \subseteq Y_{t^*}$ for $t = 1, \ldots, T$. This choice is therefore the least restrictive on the size of the infimum window. For example, with filtering, the only possible value for $\underline{W_1}$ is 1, while $\underline{W_T}$ could take on the values $1, \ldots, T$ depending on the SSM. Condition (b) requires that the largest infimum window is located at time t^*, which, based on the above, we can expect to hold for virtually any filtering and forecasting problem of practical interest.

Condition (c) is less general, but simpler to verify in practice. The requirement of $|Y_{t^*}| = \max_{t=1,\ldots,T} |Y_t|$ means that the largest conditioning set is located at time t^*, which is satisfied by the choice of $t^* = T$ for filtering and forecasting. The requirement that $\underline{W_{t^*}} = |Y_{t^*}|$ is met when all observations from the conditioning set are of added value to prediction, which we can expect to hold for any SSM with autoregressive dynamics for the states.

Appendix D Proofs

The following auxiliary results will be used to prove Lemma 1 and 2.

Lemma 3 (Finite-N convergence of regularized covariate sets)**.** *Under Assumption 2, there exists some $M \in \mathbb{N}$ such that for $t = 1, \ldots, T$,*

$$\widetilde{Y}_t^N \overset{\text{a.s.}}{=} Y_t \qquad \forall\ N > M.$$

Proof. The result follows trivially by noting that if for some finite set Z a sequence $(z_N)_{N \in \mathbb{N}}$, $z_N \in Z$, convergences to a point $z_\infty \in Z$, then it holds that $\forall\ \epsilon > 0\ \exists\ N_\epsilon \in \mathbb{N}$: $z_N \in B_\epsilon(z_\infty)\ \forall\ N > N_\epsilon$, with $B_\epsilon(z_\infty)$ an open ball having center z_∞ and radius ϵ. Because Z is finite, it is possible (for any metric) to choose ϵ small enough such that $B_\epsilon(z_\infty) = \{z_\infty\}$. The result then follows by setting $Z = \mathcal{P}(Y_t)$ with $z_N = \widetilde{Y}_t^N$ and $z_\infty = Y_t$.

□

Similarly, we have the following result for the regularized window size. The proof follows the same argument.

Corollary 1 (Finite-N convergence of regularized window size)**.** *Suppose that* $W^\infty := \lim_{N\to\infty} W^N \in \{1, \ldots T\}$ *almost surely, then there exists some* $M \in \mathbb{N}$ *such that*

$$W^N \overset{\text{a.s.}}{=} W^\infty \qquad \forall\ N > M.$$

D.1 Proof of Lemma 1

By Assumptions 1-3, there exists an optimal filter $\{f_t^*\}_{t=1}^T$ such that for all realizable paths $y_{1:T} \in \mathcal{Y}$ and $t = 1, \ldots, T$ we have that as $N \to \infty$,

$$\left|\widehat{f}_t^N\left(\widetilde{Y}_t^N\right) - f_t^*\left(Y_t\right)\right| = \left|\widehat{f}_t^N\left(Y_t\right) - f_t^*\left(Y_t\right)\right| \to \left|f_t^\infty\left(Y_t\right) - f_t^*\left(Y_t\right)\right| = 0 \qquad \text{almost surely,}$$

where the first equality follows from Lemma 3, the limit function f_t^∞ is as defined in (14), and the final expression is zero by Assumption 3.

□

D.2 Proof of Lemma 2

(a) Because $W^N \in \{1, \ldots, T\}$, we have by Definition 2 and Corollary 1 that

$$\lim_{N\to\infty} \widetilde{Y}_t^N = \lim_{N\to\infty} \widetilde{Y}_t(W^N) = \widetilde{Y}_t(W^\infty) = Y_t \qquad \text{almost surely,}$$

for $t = 1, \ldots, T$, which implies Assumption 2. Lemma 1 then applies because Assumptions 1-3 hold.

(b) There exists an optimal filter $\{f_t^*\}_{t=1}^T$ such that for all realizable paths $y_{1:T} \in \mathcal{Y}$ and $t = 1, \ldots, T$ we have that as $N \to \infty$,

$$\begin{aligned}\left|\widehat{f}_t^N\left(\widetilde{Y}_t^N\right) - f_t^*\left(Y_t\right)\right| = \left|\widehat{f}_t^N\left(\widetilde{Y}_t\left(W^N\right)\right) - f_t^*\left(Y_t\right)\right| &= \left|\widehat{f}_t^N\left(\widetilde{Y}_t\left(W^\infty\right)\right) - f_t^*\left(Y_t\right)\right| \\ &\to \left|f_t^\infty\left(\widetilde{Y}_t\left(W^\infty\right)\right) - f_t^*(Y_t)\right| = 0 \qquad \text{almost surely,}\end{aligned}$$

where the first equality holds by definition of the regularized covariate sets in terms of the regularized window size, the second follows from Corollary 1, the limit function f_t^∞ is as defined in (14), and the final expression is zero by Definition 3 because $W^\infty \geq \underline{W_{t^*}} = \max_{t=1,\ldots,T} \underline{W_t}$ due to (20).

(c) $W^\infty \geq \underline{W_{t^*}} = \max_{t=1,\ldots,T} |Y_t|$ implies Condition (a). Alternatively, $\underline{W_{t^*}} = \max_{t=1,\ldots,T} |Y_t| \geq \max_{t=1,\ldots,T} \underline{W_t}$ because $|Y_t| \geq \underline{W_t}$ for $t = 1, \ldots, T$, which implies Condition (b).

□