Kernel Smoothing Conditional Particle Filter With Ancestor Sampling

Salima El Kolei, Fabien Navarro

Kernel Smoothing Conditional Particle Filter with Ancestor Sampling

Salima El Kolei, *CREST-ENSAI,* Fabien Navarro, *Université Paris 1 Panthéon-Sorbonne, SAMM*

***Abstract*—We introduce a new method for simultaneous estimation of parameters and latent process dynamics in non-linear and non-Gaussian state space models. Combining kernel smoothing, conditional particle filters, and ancestor sampling, our approach builds upon foundational insights from prior research. We dynamically adjust the kernel bandwidth using the Kullback-Leibler divergence criterion between the filtering and prediction distributions, ensuring robust exploration of the parameter space. This technique effectively addresses variance inflation issue and mitigates filter divergence and degeneracy. Furthermore, our method is versatile, compatible with a diverse range of kernels, broadening its applicability. It achieves competitive or superior performance while requiring significantly fewer particles than comparable methods, substantially reducing the computational load. Moreover, our approach can be extended to models with complex dependencies. Through comprehensive numerical simulations, we compare our method with current state-of-the-art algorithms, demonstrating that our approach achieves comparable, if not superior, performance while reducing the computational burden, particularly in high-dimensional settings.**

***Index Terms*—State estimation, Parameter estimation, Conditional particle filters, Ancestor sampling, Non-linear State space models**

List of Abbreviations:

AS	Ancestor sampling
CPF	Conditional particle filter
EM	Expectation-maximization
FFBS	Forward filtering/backward simulation
GSPF	Gaussian smoothing particle filter
KCPF-AS	Kernel smoothing CPF with AS
KLD	Kullback–Leibler divergence
MCMC	Markov chain Monte Carlo
MSE	Mean square error
PF	Particle filter
PGAS	Particle Gibbs with ancestor sampling
PGBS	Particle Gibbs with backward simulation
PMMH	Particle marginal Metropolis–Hastings
PSEM	Particle smoother EM
SAEM	Stochastic Approximation of EM
SMC	Sequential Monte Carlo
SSM	State-space model

I. Introduction

STATE-SPACE MODELS (SSMs) are versatile for capturing dependencies in time series data. With the rise of computational advancements, complex nonlinear SSMs have gained ground in process industries [9, 48]. For effective monitoring and control, simultaneous online estimation of the state process and unknown parameters becomes crucial. The primary hurdle is the intractable nature of the likelihood function for nonlinear and non-Gaussian hidden models. As a result, traditional maximum likelihood estimators confront difficulties due to the need to maximize such intractable integrals.

Numerical approaches, such as gradient algorithms and expectation-maximization (EM) algorithms, have been widely used in the statistical literature [58, 17], with stochastic approximation EM (SAEM) offering improved convergence and computational efficiency [24, 38]. This is especially relevant for Sequential Monte Carlo (SMC) solutions, which are generally computationally demanding. Within the EM framework for SSMs, a key challenge is computing the optimal filtering, which refers to the distribution of the latent process given the observations. To tackle this challenge, researchers turned to SMC filters or Monte Carlo Markov Chain (MCMC) algorithms to compute the E-step [46, 50].

Numerous online estimation methods have been discussed in literature, evidenced by contributions from [31, 40, 2, 10, 11, 14], among others. An ad hoc methodology, combining state and parameter estimation, was proposed in [31]. While innovative, this approach encounters challenges with subsampling from the prior. This problem occurs when the prior sample inadequately represents regions of high posterior density, potentially leading to deterioration in parameter estimation. Various solutions were proposed, including the work in [2]. The authors tackled this issue by introducing an artificial dynamic for the parameters, a solution that, however, can lead to the variance inflation and the filter divergence. To mitigate this bias and avoid the divergence of the filter, researchers have explored regularization techniques employing kernel methods, as detailed in [40, 6, 45, 33, 49]. Specifically, [45] proposed a regularization strategy tailored for latent dynamic systems in the absence of noise. The authors of [6] advance this methodology into parameter estimation, demonstrating that regularization effectively implements artificial parameter dynamics to mitigate the bias and variance inflation, and provide convergence results based on their analysis. This approach proved useful in diverse applications. For instance, [57] proposed an optimal Gaussian kernel bandwidth adjustment, in the presence of missing data. However, these approaches often require a high number of particles for convergence, limiting their applicability to complex and nonlinear data assimilation problems.

From 2010 onwards, Particle MCMC methods introduced in [1] signaled a new direction. This approach combines MCMC and SMC techniques for parameter and state estimation for complex SSMs. Employed as a building block for the SMC^2 algorithm [11], PMCMC consists in a MCMC algorithm to approximate the posterior distribution from observed data, overcoming the intractability of direct likelihood computations

by utilizing conditional particle filters (CPFs) for approximation. In contrast to conventional SMC samplers, CPFs operate iteratively, embedding one conditioning trajectory into a standard PF at each iteration. This approach enables efficient exploration of the state space. The underlying principle is that if the conditioning trajectory significantly deviates from the true signal, it will likely be eliminated during the resampling step of the filtering process due to its low probability of selection. Conversely, if the conditioning trajectory closely aligns with the true signal, it has a higher probability of being selected and propagated to subsequent time steps. Hence, an appropriately chosen conditioning trajectory ensures that the CPF will explore the state space in its neighborhood, potentially guiding the filter toward attractive region of the state space. Building on the innovations of PMCMC and SMC^2, the nested PF were introduced for state and parameter estimation by [14]. Unlike the SMC^2 approach, the nested PF allows recursive parameter estimation, which considerably reduces the complexity of the method. However, this benefit is balanced by the need for a higher number of particles in the nested PF. The parameter estimation is done with a kernel PF embedded in a standard PF. The authors show different forms of convergence of their approach and propose an example where the number of particles is considerably large.

An enhancement in the CPF paradigm is the ancestor sampling (AS) procedure [38], which confronts issues related to the Markov kernel's mixing properties and degeneracy in CPF [38, 12]. Its introduction fostered better kernel mixing, allowing for practical online computations with a limited number of particles. To achieve state and parameter estimation, the integration of AS with classical methods like CPF-SAEM [35] and particle smoothing EM algorithm (PSEM) with a forward filtering/backward simulation (FFBS) smoother emerged [50]. Within the framework of particle MCMC, a systematic way of combining AS and MCMC, the particle Gibbs ancestor sampling (PGAS) has been proposed in [37] as a conceptually similar approach to particle Gibbs with backward simulation (PGBS) [39]. PGAS utilizes a forward sweep instead of separate forward and backward sweeps employed by PGBS. While these algorithms can belong within online estimation methods, their performance rely on a high number of iterations and backward simulations. The complexity and execution time of these approaches make them unsuitable for real-time estimation or high-dimensional problem. Recent advancements spotlight the work of [15], which proposes three novel algorithms addressing the computational complexity of backward smoothing techniques. Notably, the CPF-AS algorithm offers a linear complexity (with respect to the number of particles), contrasting the typical quadratic complexity seen in conventional methods. Complementing these developments, the surveys by [28] and [41], review SMC methods for parameter estimation for SSMs, synthesizing key developments and methodologies in the field.

In this paper, we propose a novel approach for simultaneous online estimation of parameters and latent process dynamics. Our method combines CPF-AS with kernel smoothing technique, is termed KCPF-AS. This integration does not add any complexity compared to the standard CPF-AS. To enhance the effectiveness of KCPF-AS, we extend to CPF-AS the tuning rule proposed in [57] to select the kernel bandwidth. This method is based on an optimization of the empirical Kullback-Leibler divergence (KLD) between a proposal density and the optimal filtering density. KCPF-AS achieves performance comparable to existing methods while significantly reducing the computational burden. With a minimal number of particles, our approach outperforms the results from [57], which used a significantly higher number of particles, indicating its efficiency and potential for real-time applications in complex or high-dimensional models. We present several contributions in this paper:

- We propose an adaptive kernel CPF with AS, enabling efficient estimation of both the state and parameters of nonlinear and non-Gaussian SSMs, suitable for both standard and high-dimensional scenarios.
- We adapt an existing tuning rule to select kernel bandwidth for KCPF-AS, addressing variance inflation due to artificial parameter dynamics.
- We incorporate an adaptive optimization approach based on KLD, which mitigates the issue of filter degeneracy and divergence.
- We demonstrate the effectiveness of our method through numerical examples, comparing its performance with state-of-the-art algorithms for simultaneous state-parameter estimation and for tackling high-dimensional challenges.

The rest of the paper is organized as follows. In Section II, we provide a brief overview of sequential PFs and CPFs with AS, highlighting the differences between the two algorithms. Section III presents our proposed approach based on the CPF-AS and kernel regularization. The performance of our approach is evaluated in Section IV through numerical experiments, where it is compared with several competitive algorithms for simultaneous state-parameter estimation, including in high-dimensional challenging scenarios. All proofs are gathered in Appendix A.

II. Problem Statement and Related Works

Consider the following class of Markovian models

$$\begin{cases} y_t \sim p_\theta(y_t|x_1,\dots,x_t) \\ x_t \sim p_\theta(x_t|x_1,\dots,x_{t-1}), \end{cases} \tag{1}$$

where $x_t \in \mathbb{X} \subset \mathbb{R}^d$ and $y_t \in \mathbb{Y} \subset \mathbb{R}^m$ for $t = 1,\dots,T$ represent the state and the measurements process, respectively. The parameter $\theta \in \Theta \subset \mathbb{R}^r$ are the unknown model parameters. We refer to the index variable t as time, though it may not necessarily signify a temporal progression. In the case of non-Markovian models, both the transition density $p_\theta(x_t|x_1,\dots,x_{t-1})$ and the measurement density $p_\theta(y_t|x_1,\dots,x_t)$ can depend on the entire past trajectory of the latent process. A SSM model is a particular case of (1), where we assume conditional independence:

Assumption 1. *The observations $(y_1,\dots,y_t)$ are independent conditionally to the hidden states $(x_1,\dots,x_t)$.*

Under Assumption 1, conditional on the state process, the observations are mutually independent. Moreover, the observation y_t depends on the state process only through x_t. Even though it simplifies some steps of the PFs, the approach presented here yet works for non-Markovian models. When working with SSMs, a common challenge is the reconstruction of the latent process x_t at time t given a sequence of observations $y_{1:T}$. Filtering corresponds to the scenario where $T = t$, prediction to the case $T < t$, and smoothing refers to the case where $T > t$.

The primary objective of this study is to perform simultaneous estimation of the latent process $(x_t)_{t\geq 1}$ and the parameters θ from the available observations $y_{1:t}$. Although our focus is on filtering, the proposed methodology also extends to prediction and smoothing tasks.

Notations: For any sequences $\{u_r\}_{r\in\mathbb{Z}}$, we use the notation $u_{s:t} = \{u_s, \dots, u_t\}$. We also denote $[N] = 1, \dots, N$. For each $t \in [T]$, we represent by $\gamma_{\theta,t}(x_{1:t})$ the sequence of unnormalized densities on the measurable space $(\mathbb{X}^t, \mathcal{X}^t)$, parametrized by $\theta \in \Theta \subset \mathbb{R}^r$. The corresponding normalized probability densities are given by

$$\overline{\gamma}_{\theta,t}(x_{1:t}) = \frac{\gamma_{\theta,t}(x_{1:t})}{\tilde{\gamma}_{\theta,t}}, \text{ with } \tilde{\gamma}_{\theta,t} = \int \gamma_{\theta,t}(x_{1:t}) dx_{1:t}.$$

For SSMs, these densities relate to $\overline{\gamma}_{\theta,t}(x_{1:t}) = p_\theta(x_{1:t}|y_{1:t})$ and $\gamma_{\theta,t}(x_{1:t}) = p_\theta(x_{1:t}, y_{1:t})$ where

$$p_\theta(x_{1:t}, y_{1:t}) = p_\theta(x_1) \prod_{t=2}^{T} p_\theta(x_t|x_{t-1}) \prod_{t=1}^{T} p_\theta(y_t|x_t).$$

and $\tilde{\gamma}_{\theta,t} = p_\theta(y_{1:t})$. Note that if we are able to compute the joint filtering density $\overline{\gamma}_{\theta,t}(x_{1:t})$, then we can deduce the optimal filter denoted $\overline{\gamma}_{\theta,t}(x_t) = p_\theta(x_t|y_{1:t})$ by integrating over the variables $x_{1:t-1}$ and the prediction filter $\overline{\gamma}_{\theta,t^-}(x_t) = p_\theta(x_t|y_{1:t-1})$ by the Chapman-Kolmogorov equation:

$$p_\theta(x_t|y_{1:t-1}) = \int p_\theta(x_t|x_{t-1}) \overline{\gamma}_{\theta,t-1}(x_{t-1}) dx_{t-1}.$$

Throughout the paper, let $\mathcal{K}_h$ be a kernel with bandwidth parameter $h \in [0, 1]$, acting as a bounded map from $\mathbb{R}^r$ to $\mathbb{R}$, such that $\int \mathcal{K}_h d\lambda = 1$, where λ is the Lebesgue measure. We define a parametric multivariate kernel $\mathcal{K}_h(u, \psi)$ as the product of r univariate kernel $\mathcal{K}_{jh}(u, \psi)$. Hence, $\mathcal{K}_h(u, \psi) = h^{-r} \prod_{j=1}^{r} \mathcal{K}_{jh}(u, \psi)$, where ψ represents the parametric component.

A. Particle Filters

PFs, also called SMC methods, provide robust solutions to challenges posed by models frequently encountered in real-world applications. It is often not possible to calculate $\overline{\gamma}_{\theta,t}(x_{1:t})$ and $\overline{\gamma}_{\theta,t}(x_t)$ analytically. SMC methods, however, enable efficient approximations of these densities [18, 16, 19]. This section offers a concise overview of the SMC sampler, laying the foundation for the CPF-AS algorithm, which will be discussed in the next section. We refer the reader to [1, 38, 37] for more details.

Consider a weighted system of N particles, denoted as $\{x^i_{1:t-1}, w^i_{\theta,t-1}\}_{i\in[N]}$, aiming to approximate the posterior distribution at time $t-1$. *i.e.*, the weighted particles represent an empirical approximation of the target distribution $\overline{\gamma}_{\theta,t}(x_{1:t-1})$ given by

$$\widehat{\overline{\gamma}}^N_{\theta,t}(dx_{1:t-1}) = \sum_{i=1}^{N} \overline{w}^i_{\theta,t} \delta_{x^i_{1:t-1}}(x_{1:t-1}),$$

where $\delta_{x^i_{1:t-1}}(x_{1:t-1})$ is the Dirac delta function at $x^i_{1:t-1}$, and $\overline{w}^i_{\theta,t} = w^i_{\theta,t-1} / \sum_j w^j_{\theta,t-1}$ are the normalized weights. This PF is then propagated to time t by sampling $\{a^i_t, x^i_t\}_{i\in[N]}$ independently and conditionally to the particles generated up to time $t-1$ from the proposal kernel[1]

$$M_{\theta,t}(a_t, x_t) = \overline{w}^{a_t}_{\theta,t-1} q_\theta(x_t|x^{a_t}_{1:t-1}, y_t), \tag{2}$$

where $q_\theta(x_t|x^{a_t}_{1:t-1}, y_t)$ corresponds to the proposal sampling distribution used to propagate the particles from time $t-1$ to time t.

In formulation (2), the resampling step is implicit and corresponds to sampling the ancestor indices a_t. Each entire variable a_t belongs to $[N]$ and represents the index of the resampled ancestor particle for x^i_t. Hence, when we write $x^{a^i_t}_{1:t}$ we refer to the ancestral path of x^i_t. The particle trajectory is defined recursively as

$$x^i_{1:t} = (x^{a^i_t}_{1:t-1}, x^i_t).$$

Once we have sampled the N ancestor indices $(a^i_t)_{i\in[N]}$ and the particles x^i_t from the kernel (2) at time t, we assign weights to these particles according to the following weight function for $t \geq 2$

$$w_{\theta,t} \propto \overline{w}_{\theta,t-1} \frac{p_\theta(y_t|x_t) p_\theta(x_t|x_{t-1})}{q_\theta(x_t|x_{1:t-1}, y_t)}. \tag{3}$$

The weights (3) are derived under the Markov assumption of the latent states and Assumption 1 by successively applying Bayes' theorem (see, [18] for more details).

The SMC procedure is initialized by drawing samples from the proposal density $x^i_1 \sim q_\theta(x_1)$. These samples are then weighted according to the ratio of the target to the proposal density $w_{\theta,1} = \gamma_{\theta,1}(x_1)/q_\theta(x_1)$. In practice, when available, the hidden Markov chain's transition density, $p_\theta(x_t|x_{t-1})$ serves as a suitable choice for q_θ, aligning with the Bootstrap filter's principles [27]. Algorithm 1 summarizes the standard SMC procedure.

Algorithm 1 SMC (each step is for $i \in [N]$)

Input: $N, y_{1:T}, q_\theta$.
Output: $\widehat{x}_{1:T}$.

1: Draw $x^i_1 \sim q_\theta(x_1)$ and set $w^i_{\theta,1} = \gamma_{\theta,1}(x^i_1)/q_\theta(x^i_1)$.
2: **for** $t = 2, \dots, T$ **do**
3: Draw $\{a^i_t, x^i_t\}$ using (2) and set $w^i_{\theta,t}$ according to (3).
4: Set $x^i_{1:t} = (x^{a^i_t}_{1:t-1}, x^i_t)$.
5: **end for**
6: **Output:** $\widehat{x}_t = \sum_{i=1}^{N} \overline{w}^i_{\theta,t} x^i_t$.

[1]Here the kernel $M_{\theta,t}$ depends on the entire particle system up to time $t-1$ but for notational convenience we omit the explicit dependence.

B. Conditional Particle Filters with Ancestor Sampling

The CPF with AS is a notable innovation presented in [38], offering an evolution of the conventional CPF algorithm introduced in [1]. The integration of an AS step addresses the inherent challenges posed by the mixing properties of the Markov kernel within the traditional CPF algorithm, as discussed in [38, 12]. Both strategies rely on the specification of a one particle trajectory *a priori*, referred to as $\tilde{x}_{1:T}$. This reference trajectory acts as a guiding path, steering particles towards state space regions of relevance. If chosen in accordance with an appropriate weight function, this trajectory ensures that particles explore regions with a high likelihood. In standard SMC, samples $\{a_t^i, x_t^i\}$ are drawn independently from the kernel (2). However, CPF modifies this by sampling conditionally to $\tilde{x}_{1:T}$ throughout the procedure. This means particles are only sampled from the kernel (2) for $i \in [N-1]$, with the last one set deterministically as $x_t^N = \tilde{x}_t$. A subsequent AS step reassigns the index variable a_t^N. CPF deterministically sets this value as $a_t^N = N$, while this one is drawn randomly for CPF-AS. This slight modification significantly improves the kernel's mixing properties in CPF-AS, yielding a PF that is less correlated to the reference trajectory, compared to CPF. This improvement is highlighted in [12, 38, 37] and we give an illustrative diagram in Fig. 1.

We define $\tilde{x}_{t:T}$ as the reference trajectory from time t up to T, and $(x_{1:t-1}^i)_{i\in[N]}$ as the set of N trajectory particles generated by the filter from the initial time up to $t-1$. The AS aims to assign ancestral paths to the reference path $\tilde{x}_{t:T}$. To achieve this, we introduce an ancestor index a_t, which encodes the ancestry of the particles. The connection between $\tilde{x}_{t:T}$ and one of the particles $(x_{1:t-1}^i)_{i\in[N]}$ is established by assigning a value to the random variable a_t^N. This random variable follows a multinomial distribution with values in $[N]$ and probabilities $\overline{w}_{\theta,t-1|T}^i$ given by

$$\overline{w}_{\theta,t-1|T}^i \propto w_{\theta,t-1}^i p_\theta(\tilde{x}_t | x_{t-1}^i), \tag{4}$$

with $w_{\theta,t-1}^i$ defined in (3). The formula above can be seen as an an application of Bayes' theorem, reminiscent of backward sampling weights in backward simulators as mentioned in [25, 38]. This similarity, however, does not necessitate an explicit backward pass in AS. Hence, the AS step corresponds to a one step of the backward simulation which explains the close relationship between the CPF with a backward simulation and the CPF-AS for SSMs. Nevertheless, for non Markovian models this equivalence between these two filters does not hold in general ([37]). The CPF-AS and CPF methods for SSMs are described in Algorithm 2 and illustrated in Fig. 1.

Conditioning the PF on a pre-specified set of particles may not seem obvious, but it brings an attractive property of invariance of the Markov chain for any number of particles N. This invariance property follows by the construction of the CPF-AS in [38], and the fact that the law of the particles resampled in the ancestor sampling step is independent of permutations of the particles indices. Therefore, the conditioned particles can be placed in any position. In practice, it has been observed that a moderate number of particles (typically between 5 to 20), is sufficient to achieve a rapidly mixing kernel (see, [1]).

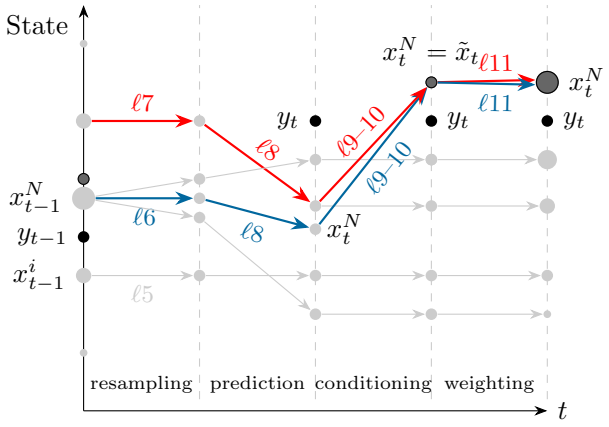

Fig. 1. Comparison of CPF and CPF-AS schemes using $N = 5$ particles (light gray points). The main difference, highlighted in red for CPF-AS, lies in the resampling step, depicted in blue for CPF. The conditioning particle $\tilde{x}_{t-1:t}$ is shown as a dark gray point. ℓj denotes the corresponding lines in Algorithm 2.

Algorithm 2 CPF with and without AS.

Input: $N, y_{1:T}, q_\theta, \tilde{x}_{1:T}[1]$.
Output: $\tilde{x}_{1:T}$.

for $k = 1, \ldots, K$ **do**
  Draw $x_1^i \sim q_\theta(x_1)$ for $i \in [N-1]$.
  Set $x_1^N = \tilde{x}_1[k]$ and $w_{\theta,1}^i = \gamma_{\theta,1}(x_1^i)/q_\theta(x_1^i)$.
  for $t = 2, \ldots, T$ **do**
    Draw a_t^i s.t. $\mathbb{P}(a_t^i = j) \propto w_{\theta,t-1}^j$ for $i \in [N-1]$
    Set $a_t^N = N$. ▷ CPF step
    Set a_t^N according to (4). ▷ CPF-AS step
    Draw $x_t^i \sim q_\theta(x_t | x_{1:t-1}^{a_t^i}, y_{1:t})$ for $i \in [N-1]$.
    Set $x_t^N = \tilde{x}_t[k]$.
    Set $x_{1:t}^i = (x_{1:t-1}^{a_t^i}, x_t^i)$ for $i \in [N]$.
    Set $w_{\theta,t}^i$ according to (3).
  end for
  Draw J with $\mathbb{P}(J = i) = \overline{w}_{\theta,T}^i$.
  Set $\tilde{x}_{1:T}[k+1] = x_{1:T}^J$.
end for

The efficiency of the CPF and CPF-AS algorithms is influenced by the initial choice of the reference trajectory $\tilde{x}_{1:T}$. A "good" conditioning particles must be "close" to the true state to guide the filter with reasonable computational costs. The distribution of $\tilde{x}_{1:T}$ must be chosen such that the output of the CPF is precisely the target distributions $\overline{\gamma}_{\theta,T}(x_{1:T})$. According to [1, Theorem 2.1], if $\tilde{x}_{1:T}$ is simulated according to $\overline{\gamma}_{\theta,T}(x_{1:T})$, then running the CPF with this reference trajectory provides other sequences distributed according to $\overline{\gamma}_{\theta,T}(x_{1:T})$. Another important result of their Theorem concerns a "bad" reference trajectory, which after a few number of iterations, has no impact on the convergence of the filter. In [8, Figure 6], the authors illustrate this observation on the Kitagawa example presented in Section IV-A. This initial trajectory selection is further examined in our high-dimensional example presented in Section IV-B.

By running one iteration of the CPF-AS algorithm, until the final time step T, we obtain a set of particles that define an

empirical distribution on $\mathbb{X}^T$ given by

$$\widehat{\overline{\gamma}}_{\theta,T}^N(x_{1:T}) = \sum_{i=1}^N \overline{w}_{\theta,T}^i \delta_{x_{1:T}^i}(x_{1:T}).$$

The convergence of $\widehat{\overline{\gamma}}_{\theta,T}^N$ towards $\overline{\gamma}_{\theta,T}$ happens for $N \to +\infty$ (see, [1, Theorem 2.1]). However, in practice with a finite N, the approximation tends to be rather poor (unless T is very small), as it typically suffers from path degeneracy. Iterative methods, such as those explored in [1, 56], offer promising solutions to achieve more robust approximations by coupling CPF-AS with MCMC procedures. In [56, Theorem 1], the authors provide convergence results under mild assumptions and show that the convergence is independent of the number of particles but depends on the number of MCMC iterations K. This result is particularly interesting in practice, as it suggests that a smaller number of MCMC iterations is needed compared to N.

After a complete pass of the CPF-AS, a trajectory $x_{1:T}^J$ is sampled from the particle trajectories (lines 13-14 of Algorithm 2). This resampled trajectory becomes the reference trajectory for the next iteration of the MCMC step, *i.e.*, $\tilde{x}_{1:T}[k+1] = x_{1:T}^J$. To draw K distinct realizations we just need to repeat K times the CPF-AS procedure. For sufficiently large K, the realizations of $\{\tilde{x}_{1:T}[1], \ldots, \tilde{x}_{1:T}[K]\}$ is sampled from the stationary distribution $\overline{\gamma}_{\theta,T}(x_{1:T})$.

The online nature of this methodology is valid for $K = 1$. For $K > 1$, maintaining online estimation involves the incorporation of a MCMC step at time t to rejuvenate particles. This introduction is conditional upon observing, through a degeneracy criterion, such as the effective sample, that degeneracy exceeds a specified threshold. This approach is supported by [10, 22, 23]. Advanced sequential MCMC schemes, address online filtering inference problems (*e.g.,* [3, 5, 26, 51]). However, for high-dimensional hidden Markov process, path degeneracy will occur as the dimension of the process increases. As a consequence, this degeneracy can be seen even in a single iteration of the algorithm (see, [4, 54]). In such contexts, MCMC moves are particularly effective in mitigating path degeneracy. For practical filtering problems, to manage computational costs while maintaining an online algorithm, Markov transition kernels at time t focus on a fixed time lag ℓ, updating only the variables $x_{t-\ell+1}, \ldots, x_t$. References that summarize these points and open up new perspectives for online MCMC are given in [53, 52].

C. State and Parameters Estimation

The task of state and parameter estimation is central to many applications and several particle-based MCMC approaches have been introduced, [1, 50, 39, 38, 35, 36, 37, 55]. Some authors proposed to augment the state vector by considering parameters as state variables with distinct dynamics, as described in [31, 40] and extended in [57]:

$$\theta_t = \theta_{t-1} + \eta_t^\theta, \tag{5}$$

where $\theta_t \in \Theta \subset \mathbb{R}^r$ and η_t^θ is an artificial centered noise with variance $\mathcal{W}_t$. This artificial noise mechanism was introduced to counter the filter's divergence when assuming constant parameter $\theta_t = \theta_{t-1}$. We are focused on estimating the hidden state x_t and unknown parameters θ_t, *i.e.,* $p(x_t, \theta_t|y_t)$ denoted as $\overline{\gamma}_{\theta,t}(x_t, \theta_t)$. According to Bayes' rule, one has

$$\overline{\gamma}_{\theta,t}(x_t, \theta_t) \propto \overline{\gamma}_{\theta,t}(x_t) \times \overline{\gamma}_{\theta,t}(\theta_t),$$

where $\overline{\gamma}_{\theta,t}(x_t)$ can be approximated by any PFs discussed in Sections II-A and II-B, and it remains to estimate $\overline{\gamma}_{\theta,t}(\theta_t) = p(\theta_t|y_{1:t})$.

The goal is now to generate N particles $\{x_t^i, \theta_t^i\}$ from the filtering distribution $\overline{\gamma}_{\theta,t}(x_t, \theta_t)$. This is achieved by sampling $\{x_t^i, \theta_t^i\}$ from the importance density $q(x_t, \theta_t|x_{1:t-1}^i, \theta_{t-1}^i, y_t)$. The importance weight for each particle is then calculated using the recursive formula

$$w_{\theta,t} = \overline{w}_{\theta,t-1} \frac{p(y_t|x_t, \theta_t) p(x_t|x_{t-1}, \theta_t) \delta_{\theta_{t-1}}(\theta_t)}{q(x_t, \theta_t|x_{1:t-1}, \theta_{t-1}, y_t)}. \tag{6}$$

Standard PF approximations lead to the following quantities:

$$\begin{aligned} \widehat{\overline{\gamma}}_t^N(\theta_t, x_t) &= \sum_{i=1}^N \overline{w}_{\theta,t}^i \delta_{(x_t^i, \theta_t^i)}(\theta_t, x_t) \\ \widehat{\overline{\gamma}}_t^N(\theta_t) &= \sum_{i=1}^N \overline{w}_{\theta,t}^i \delta_{\theta_t^i}(\theta_t). \end{aligned} \tag{7}$$

To mitigate the divergence caused by the artificial dynamics (5), the authors in [40, 57] replace each point mass $\delta_{\theta_t^i}$ in (7) by a Gaussian kernel regularization. This approach results in the modification of the weights in (6) and the definition of a new set of weights:

$$w_{\theta,t} \propto \overline{w}_{\theta,t-1} \frac{p(y_t|x_t, \theta_t) p(x_t|x_{t-1}, \theta_t) \mathcal{N}_h(\theta_t; m_t, \mathcal{V}_t)}{q(x_t, \theta_t|x_{1:t-1}, \theta_{t-1}, y_t)}, \tag{8}$$

where $\mathcal{N}_h(y; m, v)$ denotes the multivariate Gaussian kernel density evaluated at point y with mean m and variance v, depending on the bandwidth h. Thanks to this approximation the regularization kernel becomes the natural choice for the parameters proposal distribution. Thus, we sample $\{x_t^i, \theta_t^i\}$ according to:

$$q(x_t|x_{1:t-1}, \theta_t, y_t) \times \mathcal{N}_h(\theta_t; m_t, \mathcal{V}_t). \tag{9}$$

Consequently, the weight calculation simplifies to

$$w_{\theta,t} \propto \overline{w}_{\theta,t-1} \frac{p(y_t|x_t, \theta_t) p(x_t|x_{t-1}, \theta_t)}{q(x_t|x_{1:t-1}, \theta_t, y_t)}, \tag{10}$$

and we recognize the standard formula (3) but now for the augmented state. The resulting distribution becomes a continuous distribution w.r.t. θ_t and the approximation of the optimal filter $\overline{\gamma}_{\theta,t}(x_t, \theta_t)$ and of $\overline{\gamma}_{\theta,t}(\theta_t)$ is given by:

$$\widehat{\overline{\gamma}}_{\theta,t}^N(\theta_t) = \sum_{i=1}^N \overline{w}_{\theta,t}^i \mathcal{N}_h(\theta_t; m_t^i, \mathcal{V}_t^i), \tag{11}$$

$$\widehat{\overline{\gamma}}_{\theta,t}^N(\theta_t, x_t) = \sum_{i=1}^N \overline{w}_{\theta,t}^i \mathcal{N}_h(\theta_t; m_t^i, \mathcal{V}_t^i) \delta_{x_t^i}(x_t). \tag{12}$$

At time t, let us denote $\overline{\theta}_t$ and V_t the vector of posterior conditional mean and covariance matrix, respectively, obtained from the PF approximations (7). Typically, for the mean m_t^i,

standard methods assign θ_{t-1}^i for each particle $i = 1, \ldots, N$. However, in this case, the posterior mean under the distribution (11) is indeed equal to $\overline{\theta}_{t-1}$, but with a posterior variance equal to $(1 + h^2)V_{t-1}$ and larger than the posterior variance V_t under the distribution (7). This phenomenon is known as the variance inflation ($\mathcal{W}_t = h^2 V_{t-1}$) and to correct this dispersion, it is necessary to shrink the mean of the kernel. In [40], the authors introduce a shrinkage parameter a_h leading to avoid the variance inflation and keeping the correct posterior mean $\overline{\theta}_t$ and variance V_t of the distribution (11). Nevertheless, the choice of the shrinkage parameter corresponding to the bandwidth parameter of the kernel is important and Gaussian kernels may not suit all applications.

III. Kernel Smoothing Conditional Particle Filter with Ancestor Sampling

We propose to combine the CPF-AS with kernel regularization while allowing to correct the phenomenon of variance inflation. Furthermore, we adapt the result in [57] to our general framework. In the same way we propose for our approach a tuning rule for the choice of the bandwidth based on the minimization of the empirical KLD between an estimate of the propagation density $p(x_t, \theta_t | y_{1:t-1})$ denoted with our notation $\widehat{\overline{\gamma}}^N_{\theta,t^-}(x_t, \theta_t)$ and an estimate of the optimal filtering density $\widehat{\overline{\gamma}}^N_{\theta,t}(x_t, \theta_t)$. The use of the CPF-AS method allows us to consider a very small number of particles (a range between 5-20), making our approach very competitive in practice for real-time state and parameter estimation.

Let us specify that, in order for the Gaussian approximation of the posterior distribution (11) to make sense, the parameter θ_t has to be expressed in a form that is consistent with such a distribution. For instance, variances could be expressed in logarithmic terms, probabilities through logit transformations, or more general distribution functions, and so on.

For one-dimensional θ_t, an effective strategy involves employing a mixture of non-Gaussian kernels, chosen to match the support of the parameter's distribution. For instance, for variance parameters, a Gamma kernel can be used instead of a Gaussian kernel. Following the moment-matching approach outlined in [40], the following system of equations can be solved for each $i \in [N]$.

$$\begin{cases} \mu_{t,h}^i = \mu_t(\psi_t^i) = a_h \theta_{t-1}^i + a_h^c \overline{\theta}_{t-1}, \\ v_{t,h}^i = \sigma_t^2(\psi_t^i) = (1 - a_h^2) V_{t-1}, \end{cases} \tag{13}$$

where ψ_t^i denotes the parametric form of the chosen kernel (*e.g.*, for a Gamma kernel, ψ_t^i represent the shape and rate parameters), with $a_h = \sqrt{1 - h^2}$ and $a_h^c = (1 - a_h)$. The terms $\mu_{t,h}^i$ and $v_{t,h}^i$ are the mean and the variance of the considered kernel, respectively. By selecting the kernel locations $\mu_{t,h}^i$ as in (13), we address the issue of over-dispersion. The role of the shrinkage parameter a_h is to moderate this over-dispersion by guiding particles θ_t^i closer to their overall mean. This ensures that the kernel distribution consistently maintains $\overline{\theta}_t$ as the overall posterior conditional mean, while preserving the appropriate conditional variance V_t. We generalize this moment-matching condition to encompass general kernels, yielding a continuous kernel regularization that preserves the same initial two moments as the discrete particle approximation in (7).

Building upon the notation established in Section II-C, let $\overline{\theta}_t \in \mathbb{R}^r$ represent the vector of the posterior mean at time t, and let $V_t \in \mathcal{M}_{r \times r}(\mathbb{R}_+^*)$ stand for the posterior variance matrix at time t. Then for $i \in [N]$ and $j \in [r]$,

$$\begin{aligned} \mu_{t,h}^i(\psi_{tj}^i) &= \int \theta_{tj} \mathcal{K}_{jh}(\theta_{tj}, \psi_{tj}^i) d\theta_{tj} \\ &= a_h \theta_{(t-1)j}^i + a_h^c \overline{\theta}_{(t-1)j}, \\ v_{tj,h}^i(\psi_{tj}^i) &= \int (\theta_{tj} - \mu_{tj,h}^i)^2 \mathcal{K}_{jh}(\theta_{tj}, \psi_{tj}^i) d\theta_{tj} \\ &= (1 - a_h^2) V_{(t-1)j}. \end{aligned} \tag{14}$$

The parameter ψ_{tj} should be selected to ensure that the kernel $\mathcal{K}_{jh}$ has the same marginal conditional mean and conditional variance as those obtained with the distribution (7). By solving (14) for $(\psi_{tj})_{j \in [r]}$, we generalize this moment-matching condition to embrace general kernels, resulting in a continuous kernel regularization that preserves the initial two moments as the discrete particle approximation (7).

Proposition III.1. *Let $h \in [0, 1]$. The kernel smoothing distribution defined as*

$$\widehat{\overline{\gamma}}^N_{\theta,t}(\theta_t) = \sum_{i=1}^N \overline{w}_{\theta,t}^i \mathcal{K}_h(\theta_t; \psi_t^i), \tag{15}$$

has the same marginal conditional mean and variance of the posterior distribution (7)*.*

Proof. See Section A of Appendix A. □

Proposition III.1 states that for vector parameters θ_t, the kernel distribution (15) provides a more general continuous approximation of (7) that matches first and second marginal moments. Consequently, the kernel regularization PF approximation of $\overline{\gamma}_{\theta,t}(x_t, \theta_t)$ is given by:

$$\widehat{\overline{\gamma}}^N_{\theta,t}(x_t, \theta_t) = \sum_{i=1}^N \overline{w}_{\theta,t}^i \mathcal{K}_h(\theta_t; \psi_t^i) \delta_{x_t^i}(x_t). \tag{16}$$

The KCPF-AS algorithm is described in Algorithm 3.

A. Optimal Tuning of the Kernel Bandwidth

For traditional PFs, a natural choice for the bandwidth consists in taking it as a decreasing function w.r.t. the number of particles N. Hence, for large N, the distribution $\overline{\gamma}_{\theta,t}(\theta_t)$ is more concentrated around the correct mean. However, in the context of KCPF-AS, this conventional approach is not optimal. This is because our technique is designed to operate efficiently even with a limited number of particles, N, irrespective of the sampling time t. In [40], the authors consider a Gaussian kernel with a fixed bandwidth equals to $(1 - C_\delta)^{1/2}$, with $C_\delta = (3\delta - 1)/2\delta$ and δ a discount factor living in $]0, 1[$ (typically close to one).

Proposition III.2 presents a procedure for selecting the bandwidth parameter for the KCPF-AS distribution (15). This method is based on the minimization of the empirical KLD between an estimate of the optimal filtering distribution

Algorithm 3 KCPF-AS

Input: $N, y_{1:T}, q_\theta, \tilde{x}_{1:T}[1], L, \widehat{h}_0^{0,1}$.
Output: $\widehat{x}_{1:T}, \widehat{\theta}_{1:T}$

for $k = 1, \ldots, K$ **do**
  Draw $\theta_1^i \sim p(\theta_1)$ for $i \in [N]$.
  Draw x_1^i from $q(x_1|\theta_1^i)$ for $i \in [N-1]$.
  Set $x_1^N = \tilde{x}_1[k]$ and $w_{\theta,1}^i = \gamma_{\theta_1^i}(x_1^i)/q(x_1^i|\theta_1^i)$.
  for $t = 2, \ldots, T$ **do**
    Draw a_t^i s.t. $\mathbb{P}(a_t^i = j) \propto w_{\theta,t-1}^j$ for $i \in [N-1]$.
    Set a_t^N according to (4).
    Draw $\theta_t^i \sim \mathcal{K}_{\widehat{h}_{t-1}^{L,k}}(\theta_t^i, \psi_t^i)$ with ψ_t^i sol. of (14) for $i \in [N]$.
    Draw x_t^i from $q_\theta(x_t|x_{1:t-1}^{a_t^i}, y_{1:t})$ for $i \in [N-1]$.
    Set $x_t^N = \tilde{x}_t[k]$.
    Set $w_{\theta,t}^i$ according to (3).
    Set $x_{1:t}^i = (x_{1:t-1}^{a_t^i}, x_t^i)$ for $i \in [N]$.
    for $\ell = 1, \ldots, L$ **do**
      Compute $\widehat{h}_t^{\ell,k}$ according to (17).
    end for
  end for
  Draw J with $\mathbb{P}(J = i) = \overline{w}_{\theta,T}^i$.
  Set $\tilde{x}_{1:T}[k+1] = x_{1:T}^J$.
end for

$\overline{\gamma}_{\theta,t}(x_t, \theta_t)$ and an estimate of the prediction distribution $\widehat{\overline{\gamma}}_{\theta,t^-}^N(x_t, \theta_t)$. A similar idea has been previously explored in [13, 57] for sequential importance resampling PFs.

The aim of this optimization is twofold: on the one hand to control the kernel bandwidth and correct the variance inflation problem and, on the other hand, to make the particles approximation (16) visiting regions of the state space with high posterior density. The kernel bandwidth is adaptive and thus allows to take into account the observations in the exploration step of the particles related to the parameters. As demonstrated in our numerical experiments in Section IV, this adaptive strategy successfully mitigates the divergence issue.

Several discrepancy measures, such as the KLD, were originally proposed for PF to determine the sample size N. This parameter significantly impacts the quality of PF estimates (see, [13, 43, 21]). The KLD between an optimal importance sampling probability density function $g^\star$ and a probability density function g is defined as

$$\begin{aligned}\mathfrak{D}_{KL}(g^\star, g) &= \int \log\left(\frac{g(x)}{g^\star(x)}\right) g^\star(x) dx \\ &= \mathbb{E}\left[\log\left(\frac{g(X)}{g^\star(X)}\right)\right], \qquad X \sim g^\star,\end{aligned}$$

where, if X can be directly sampled from $g^\star$, an estimator for $\mathfrak{D}_{KL}$ is obtained through the empirical mean of the logarithmic ratio of densities. However, since $g^\star$ is typically unknown, this estimation is facilitated by PFs. This approach is central to our methodology for kernel bandwidth selection, leveraging pre-existing PF algorithms tailored for state and parameter estimation. Accordingly, it is essential to establish appropriate regularity conditions for the SSMs (1).

Assumption 2.

1) *The measurement density $p(y_t|x_t, \theta_t)$ is continuous and differentiable w.r.t. x_t for all θ_t.*
2) *The transition density $p(x_t|x_{t-1}, \theta_t)$ is continuous and differentiable w.r.t. x_{t-1} for all θ_t.*
3) *The kernel function $\mathcal{K}_h(\theta_t, \psi_t)$ is differentiable with respect to ψ_t.*
4) *The proposal density $q(x_t|x_{1:t-1}, \theta_t, y_t)$ is given by the transition density $p(x_t|x_{t-1}, \theta_t)$ in* (9)*.*

Proposition III.2. *Under Assumption 2, an estimate for the optimal tuning rule for the bandwidth h_t of the kernel smoothing distribution* (16) *exists and is such that*

$$\widehat{h}_t = \underset{h_t \in [0,1]}{\operatorname{argmin}} \left\{ -\sum_{i=1}^N w_{\theta,t}^{i,-} \log w_{\theta,t}^i \right\}, \tag{17}$$

where $w_{\theta,t}^{i,-} = \overline{w}_{\theta,t-1}^i \mathcal{K}_{h_{t-1}}(\theta_t; \psi_t^i) p(x_t|x_{t-1}^i, \theta_t^i)$ and corresponds to the filter approximation of $\widehat{\overline{\gamma}}_{\theta,t^-}^N(x_t, \theta_t)$ at time t.

Proof. The proof follows the lines of [57, Proposition 1] and is postponed in Appendix A Section B. □

A distinction between our results and those presented in [57] is worth noting. Although our optimization problem may initially appear more complex, this complexity is primarily a matter of notation. In their study, the minimized function is represented as $\sum_i w_{t|t-1}^i \log w_t^i$. Here, $w_{t|t-1}^i$ indicates the weights corresponding to the prediction step, related to the distribution $\overline{\gamma}_{\theta,t^-}(x_t, \theta_t)$. Embedded in these weights are the proposal density $q(x_t|x_{1:t-1}, \theta_t, y_t)$ and the Gaussian kernel from their sampling approach. For computational efficiency, we adopt the transition density $p(x_t|x_{t-1}, \theta_t)$ as our choice for q (as specified in Assumption 2). This choice significantly simplifies the weight computations in practice.

While the leave-one-out method offers a practical bandwidth estimation, our preference leans towards the KLD, which inherently offers a better exploration of the state space during the sampling step. Prioritizing this optimization early on is crucial, anticipating convergence as MCMC iterations increases.

The computational complexity of these optimization problems is in general model dependent and depends on how the optimization procedure is implemented. Typically, the optimization (17) can be done recursively by stochastic gradient methods, as highlighted in [29], *i.e.,* the algorithm defines a sequence $h_t^\ell, \ell = 0, 1, \ldots$, of parameters, where ℓ is the iteration number. For each iteration, we update h_t^ℓ based on a computed direction $d_t^{\ell+1}$ and a step size $\delta_t^{\ell+1}$, formalized as

$$h_t^{\ell+1} = h_t^\ell + \delta_t^{\ell+1} d_t^{\ell+1}.$$

The search direction (possibly biased) is obtained from the following Monte Carlo approximation

$$d_t \approx \sum_{i=1}^N w_{\theta,t}^{i,-} \partial_{h_t} \log w_{\theta,t}^i,$$

and its computation involves the differentiation of the weights w.r.t. h whose distribution is parametrized by h. This task can be carried through using the reparameterization trick of [30, Section 2.4] and the gradient can be computed explicitly

or approximated through methods like the finite difference method.

B. Convergence Properties

The theoretical guarantee of convergence of the algorithm is beyond the scope of this work, which is primarily intended to introduce our novel approach and make a comparison with several existing methods. Nevertheless, we discuss in this paragraph a way to obtain it and establish some of the practical issues affecting the convergence.

To study the convergence analysis of our approach, one can refer to existing studies such as [47, 33, 49, 6]. These works employ a similar kernel regularization technique to address the issue of PF degeneracy, particularly in scenarios with noise-free or very low-noise hidden dynamical systems, as well as situations where observation noise has minimal variance. In our approach, we apply kernel regularization specifically to the parameter dynamics to correct artificially introduced noise, without regularizing the observations or the hidden dynamics. The convergence in this context relies on several key assumptions, including the regularity of the kernel and its bandwidth, as well as the boundness of the measurement functions $p(y_t|x_t, \theta_t)$.

Combining these assumptions with the uniform ergodicity results obtained in [37, Theorem 1] for CPF-AS and established MCMC convergence results as presented in [34, Theorem 1] is a viable approach to studying the convergence of our methodology. Our empirical findings provide support for this perspective. Notably, the convergence primarily hinges on the choice of h_t and the number of MCMC iterations, with the number of particles having a lesser impact.

IV. Numerical Experiments

A. Kitagawa's Nonlinear Model

In this study, we explore the standard nonlinear time series model as a challenging example, which has been considered in numerous numerical experiments [7, 50, 57]. The model's complexity is attributed to the nonlinearity present in both the state and observation equations. Moreover, accurately approximating the filtering distribution is challenging due to the transition density of the state potentially being bimodal, a result of the cosine term. The model is defined as follows

$$\begin{cases} x_{t+1} = 0.5x_t + u_t + 25\frac{x_t}{1+x_t^2} + 8\cos(1.2t) + w_t \\ y_t = 0.05x_t^2 + e_t, \qquad \text{for } t = 1, \ldots, 100, \end{cases}$$

where $w_t \sim \mathcal{N}(0, Q_t)$ and $e_t \sim \mathcal{N}(0, R_t)$ are independent white Gaussian sequences. We adopt the parameter values for $\theta_t = (Q_t, R_t)$ as in [57] and [37], specifically, $\theta_t^\star = (Q_t^\star, R_t^\star) = (0.1, 1)$, with the initial condition for the hidden state x_t set to $x_1 = 5$. Our methodology significantly reduces both the particle count, N_1, and computational time compared to the setups in [57].

Firstly, we analyze the performances of our approach and we compare it with the standard Gaussian Smoothing Particle Filter (GSPF) proposed in [40, 57] for different particle numbers, $N = 20$ up to $N = 1000$, across different iterations $K = 1$ up to $K = 155$. It should be noted that $K = 1$ denotes the KCPF-AS algorithm without the MCMC smoother and corresponding to the online estimation. The prior density $p(\theta_1)$ for the two approaches is selected as a mutually independent multivariate normal distribution with mean $(0.5, 0.5)$ and variance $(1, 1)$, ensuring that $\theta_1^\star$ is included in the support of $p(\theta_1)$. Furthermore, since our approach adapts the kernel to the parameter support, a Gamma kernel was chosen. The results are represented in Table I. The computational cost for each approach is indicated in parentheses (measured in CPU seconds) and reported for one replication. KCPF-AS gives substantially better results than the standard GSPF, whatever the number of iterations K or even with fewer particles N. The decrease in MSE with increasing K aligns with theoretical expectations of improved Markov chain mixing for larger K. However, the performance gain from increasing particle count diminishes for sufficiently large K but the performance remains even better for the online ($K = 1$) estimation for our approach.

TABLE I
MSE and CPU comparison for the KCPF-AS and the GSPF for different particles N and different iterations K with $M = 100$ runs. CPU times are in seconds and reported for $M = 1$.

		$N = 20$		$N = 50$		$N = 1000$	
Method	K	MSE	CPU	MSE	CPU	MSE	CPU
KCPF-AS	1	0.055	0.78	0.053	0.87	0.043	2.19
KCPF-AS	10	0.053	0.83	0.051	0.90	0.042	2.50
KCPF-AS	45	0.051	0.86	0.049	0.97	0.040	2.85
KCPF-AS	155	0.047	1.42	0.042	1.56	0.034	53.8
GSPF	—	0.078	0.76	0.072	0.89	0.052	2.23

Secondly, for comparative analysis, we executed CPF-SAEM [37], PGAS and PSEM [50], with PSEM utilizing a FFBS smoother with N_2 forward filter particles and a set number of backward trajectories n_{MCMC}. A PMMH [1] was also assessed with identical parameters. Note that the latter can be interpreted as a Multiple Try Metropolis scheme, where the different candidates in the MCMC iterations are generated and weighted by the use of a PF. [44, 42] provide a comprehensive analysis of the synergy between these methods. To reduce the number of parameters to be calibrated for the different approaches, we take N_2 to the same order as N_1 and aligned the number of iterations in the EM step of PSEM and CPF-SAEM with the MCMC step count, *i.e.*, $n_{\text{iter}} = n_{\text{MCMC}} = K$. We experiment different particle counts N and iteration numbers K, as these parameters significantly influence the numerical results.

Results, depicted in Figures 2 and 3, display mean square error (MSE) for each method against different K and N_1, highlighting their convergence trajectories. PMMH and PSEM are the least reliable and stable methods. Their reliance on a substantial particle count, N_2, becomes apparent when N_2 values are comparable to N_1. Both methods demand a vast number of particles to generate satisfactory outcomes. However, the subpar performance of PSEM can also be attributed to its iteration number in the EM step, set equal to the MCMC iteration count. As K increases, PSEM surpasses PMMH in efficacy, but it still trails behind its competitor (even for $K = 50$), CPF-SAEM. Our approach proves competitive both

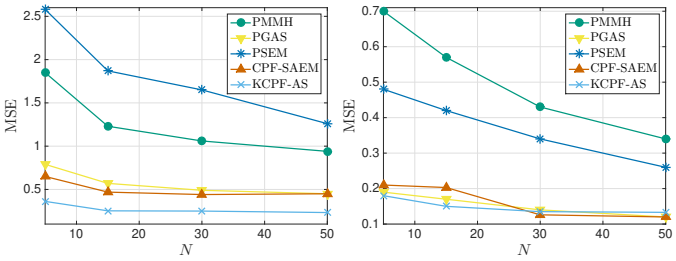

Fig. 2. MSE comparison for $K = 10$ (left) and $K = 50$ (right).

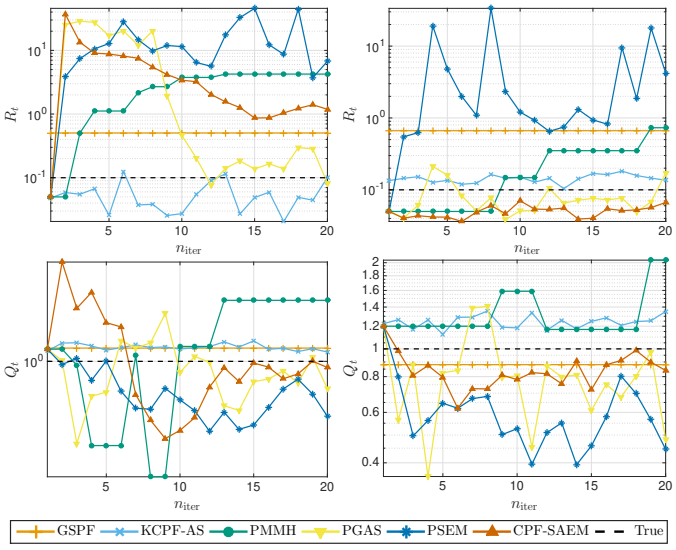

Fig. 3. Parameter convergence comparison with respect to the number of iterations $K = 1$ up to 20, $N_1 = 5$ (left) and $N_1 = 50$ (right).

in estimation results and real-time computational efficiency. For smaller K values, our method outperforms both CPF-SAEM and PGAS–explained by the insufficiency of iterations for the SAEM method's EM step and PGAS's MCMC. However, with $K = 50$, our results are similar to CPF-SAEM and PGAS, albeit with reduced computation time. For instance, our method, with $K = 50$ and $N_1 = 50$, demands 0.02 seconds, compared to CPF-SAEM's 0.19 seconds and PGAS's 0.09 seconds. This comparative analysis emphasizes that for larger K and a moderate particle count ($N_1 \approx 30$), the three best methods–KCPF-AS, CPF-SAEM, and PGAS–deliver commendable results. However, for smaller K values, our approach is consistently better, regardless of particle count (see Fig. 2).

Finally, Fig. 4 depicted the trajectory of the latent process, considering both the presence and absence of the optimization routine. At each time instance t, the KLD, given in (17), was calculated with $N_1 = 20$ and $K = 50$, halting the optimization post 20 iterations. For this, we use the Monte Carlo finite-difference methods to approximate the derivative of the weights, as detailed in Section III-A with the reparameterization trick of [30] accomplished for the Gamma kernel by the inverse cumulative function of the Gamma distribution using the Matlab function *gaminv* (see [32] for more details).The accuracy of the latent space reconstruction is strongly influenced by the values of the parameters θ_t. Incorrect parameter values

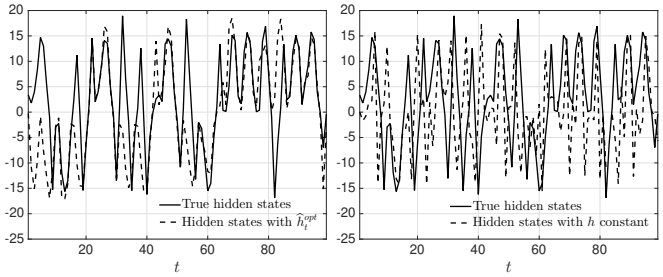

Fig. 4. Reconstruction of the latent process with (left) and without (right) optimal kernel smoothing.

can lead to inaccurate reconstructions, as demonstrated in [20], hence the importance of selecting the appropriate kernel bandwidth. We can note here that the KLD strategy surpasses the constant bandwidth approach, dividing the MSE in two ($\text{MSE}(\widehat{h}^{\text{opt}}) = 1.3 \times 10^{-3}$ vs. $\text{MSE}(\widehat{h}^{\text{const}}) = 2.6 \times 10^{-3}$). Our primary goal is the concurrent estimation of latent states and parameters, making the optimized criterion especially pertinent. Observations indicate that this leads to a more precise estimation of the latent process, effectively rectifying biases in the estimation error.

B. Large Spatial Sensor Network Example

This section evaluates our method's empirical performance on a high-dimensional problem explored in [52], focusing on estimating spatial dynamic physical phenomena through noisy data from large sensor networks. We consider a network comprising d sensors uniformly arranged on a two-dimensional grid $\{1, 2, \ldots, \sqrt{d}\} \times \{1, 2, \ldots, \sqrt{d}\}$, with each tasked to independently measure the state of a physical phenomenon at their respective locations. The state at the c-th sensor's position at time t is denoted by $x_{c,t} \in \mathbb{R}$, and its measurement is $y_{c,t} \in \mathbb{R}$. Thus, the full state across all sensor positions at time t is represented as $x_t = (x_{1,t}, \ldots, x_{d,t})^\top \in \mathbb{R}^d$, and all measurements form the vector $y_t = (y_{1,t}, \ldots, y_{d,t})^\top \in \mathbb{R}^d$.

The dynamic and measurement models are expressed as follows:

$$\begin{cases} x_t = \alpha x_{t-1} + v_t \\ y_t = x_t + e_t, \end{cases}$$

where $|\alpha| < 1$ assumed unknown, $e_t \in \mathbb{R}^d$ is a zero-mean Gaussian random vector with covariance $\Sigma_y = \sigma_y^2 I_{d\times d}$. The noise $v_t \in \mathbb{R}^d$ is also a zero-mean Gaussian random vector, with its covariance matrix Σ_v, where the (i, j)-th entry of Σ_v is given by

$$(\Sigma_v)_{i,j} = \eta \exp\left(-\frac{\|\mathcal{S}_i - \mathcal{S}_j\|_2^2}{2\beta}\right) + \gamma\delta_{i,j},$$

$\|\cdot\|_2$ denotes the Euclidean norm, $\mathcal{S}_i \in \mathbb{R}^2$ represents the physical position of sensor i, and $\delta_{i,j}$ is the Kronecker delta symbol. Following [52], we set $\alpha = 0.9, \eta = 3, \gamma = 0.01, \beta = 20$ and $\sigma_y^2 = 2$. This example is particularly valuable for evaluating our method since the posterior distribution can be derived analytically using the Kalman filter. We leveraged the authors' published results and Matlab implementation to

TABLE II
AVERAGE MSE COMPARISON ACROSS INCREASING STATE DIMENSIONS d.

Method	N	K	Dimension (d) 16	64	121	256	400
KCPF-AS	10	22	0.34	0.39	0.56	0.69	0.80
SMCMC	10	22	0.42	0.49	0.46	0.60	0.74
PF	10	—	3.95	4.34	7.40	7.94	8.91
KCPF-AS	50	22	0.20	0.24	0.34	0.61	0.74
SMCMC	50	22	0.21	0.26	0.41	0.55	0.65
PF	50	—	1.45	1.34	2.92	4.19	5.17
KCPF-AS	10	82	0.18	0.17	0.30	0.39	0.46
SMCMC	10	82	0.22	0.24	0.36	0.45	0.50
PF	10	—	3.44	4.08	7.50	7.80	9.03

TABLE III
CPU COMPARISON ACROSS INCREASING STATE DIMENSIONS d.

Method	N	K	Dimension (d) 16	64	121	256	400
KCPF-AS	10	22	0.06	0.12	0.22	0.51	0.89
SMCMC	10	22	0.19	0.29	0.55	1.43	2.75
PF	10	—	0.005	0.006	0.01	0.02	0.04
KCPF-AS	50	22	4.43	6.26	11.32	27.23	58.13
SMCMC	50	22	3.26	5.26	9.75	25.34	52.33
PF	50	—	0.05	0.60	0.07	0.16	0.32
KCPF-AS	10	82	0.45	0.83	1.34	3.46	5.50
SMCMC	10	82	1.30	2.09	3.84	9.61	18.80
PF	10	—	3.68	4.36	7.78	8.03	9.25

ensure a fair assessment of our approach. Specifically, for this example, the authors are interested in tracking a time-varying spatial physical phenomenon from the sequence y_t of noisy observations coming from the network, the parameters are assumed to be known. We relax this assumption for α and we integrate their Hamiltonian MCMC based kernel with our KCPF-AS framework without the MCMC, *i.e.,* for $K = 1$ in Algorithm 3 since this step is already integrated in their MCMC. This adaptation of their methodology to our context is termed Sequential MCMC (SMCMC). In this experiment, we compare the performance of SMCMC with that of KCPF-AS and a simple PF.

For the regularization of the unknown parameter α, we use a Gaussian kernel at each time step t, with mean $m_t = a_h\alpha_{t-1} + (1-a_h)\overline{\alpha}_{t-1}$ and a variance $\mathcal{V}_t = h_t^2 V_{t-1}$, where $a_h = \sqrt{1-h_t}$. Here $\overline{\alpha}_{t-1}$ and V_{t-1} denote to the posterior mean and variance at time $t-1$, calculated from the filter as:

$$\overline{\alpha}_{t-1} = \sum_{i=1}^{N} w_{t-1}^i \alpha_{t-1}^i \text{ and } V_{t-1} = \sum_{i=1}^{N} w_{t-1}^i (\alpha_{t-1}^i - \overline{\alpha}_{t-1})^2.$$

Hence, the adaptive kernel, derived as the solution of (17) can be easily accomplished using the standard parameterization trick for Gaussian distribution and the Gradient can be obtained from the explicit expressions of the derivatives of the Gaussian distribution. Note that the approach proposed in [52] also requires for the Hamiltonian dynamic the computation of the Gradient and the Hessian of these distributions (except for the kernel function).

Table II presents the average MSE comparison computed between PF, SMCMC, and KCPF-AS, and the optimal posterior mean obtained by the Kalman filter, across increasing state dimensions d, varying K and N. The associated CPU times for each case are detailed in Table III. The performance of the KCPF-AS remains very close to the SMCMC and becomes less efficient as the dimension becomes very large. Increasing the number of particles and keeping the same number of MCMC iterations reduces the MSE of SMCMC and KCPF-AS, but does not overcome the problem of degeneracy, which increases with the state dimension. On the other hand, when we increase the number of MCMC iterations, the KCPF-AS filter becomes more efficient. These results align with the findings of [52], where the authors note that incorporating MCMC moves within a standard PF yield superior performances compared to using a standard PF alone. In terms of CPU, KCPF-AS is faster at fixed K and for a small number of particles N. When we increase N, the two approaches are comparable with a slight gain for the SMCMC which is due in particular to the way in which the PF is nested in the MCMC core. Nevertheless, KCPF-AS gives better performance for this scenario.

TABLE IV
AVERAGE MSE COMPARISON ACROSS INCREASING STATE DIMENSIONS d FOR CASE A ("GOOD CONDITIONING") AND B ("BAD" CONDITIONING) FOR THE KSCPF-AS.

Method	N	K	Dimension (d) 16	64	121	256	400
Case A	10	22	0.25	0.32	0.46	0.65	0.72
Case B	10	22	0.39	0.40	0.48	0.68	0.74
Case A	10	82	0.14	0.18	0.33	0.38	0.45
Case B	10	82	0.20	0.24	0.35	0.39	0.46

This experiment highlight, that degeneracy issues become more significant in high-dimensional settings. While certain PFs mentioned in literature offer some mitigation of this degeneracy, this example illustrates that mere conditioning is inadequate for managing complexities in larger dimensions. Integrating MCMC methods–through the development of an efficient MCMC kernel as [52], or executing multiple MCMC moves–emerges as an effective strategy for high-dimensional challenges. The empirical evidence supports our proposed approach, addressing the complexities inherent in high-dimensional scenarios.

In Table IV, we explore the influence of the reference trajectory $\tilde{x}_{1:T}$ on the KCPF-AS performance on this high-dimensional example. We differentiate between "good", case A and "bad", Case B initial conditions for $\tilde{x}_{1:T}$, with "good" being very close to the actual signal $x_{1:T}$ and "bad" significantly diverging from it. This distinction is quantified by adding a Gaussian noise with means equal to 5% (resp. 30%) error of the signal for Case A (resp. Case B) and with a fixed variance of 0.01 for both scenarios. We note that beyond a certain dimension the conditioning no longer contributes anything to the performance of the filter compared to MCMC. In particular, even conditioning very close to the true trajectory does not provide more than bad conditioning. We therefore suspect that the degeneracy is so strong in large dimensions that conditioning and AS are not sufficient. It therefore becomes inevitable to combine them with MCMC

by considering a sufficient number K of iterations.

V. Conclusion and Perspectives

These study's numerical findings underscore the efficacy of integrating CPF-AS with kernel regularization for online estimation of both unknown parameters and the state in nonlinear SSMs. By employing AS, we not only enhance the mixing of the Markov kernel but also circumvent parameter divergence. This is achieved with a minimal particle count, typically ranging from 5 to 20 particles. This efficiency is particularly compelling for real-time computational applications. This outcome holds significance, especially in the context where introducing an artificial dynamic on parameters can inadvertently lead to variance inflation, potentially triggering filter divergence. The proposed approach relies on the choice of a tuning parameter and we propose a data-driven procedure based on the minimization of the empirical KLD between the prediction and optimal filter estimates. Furthermore, for high-dimensional problems where the degeneracy problem is very pronounced, we show the interest of combining this approach with MCMC methods. The method proposed in this paper still remains very competitive in this context and the scope of our methodology extends beyond SSMs. It offers promising potential for adaptation to models presenting intricate dependencies, such as those that are non-Markovian, nonparametric, or graphical models.

Appendix

A. Proof of Proposition III.1

Proof. Let $\mathcal{I}$ denote a latent variable that takes values in $\{1,\dots,N\}$ with $\mathbb{P}(\mathcal{I}=i)=w^i_{\theta,t}$ for $i\in[N]$ and recall that $a_h=\sqrt{1-h^2}$. For $j\in[r]$

$$\begin{aligned}\mathbb{E}[\theta_{tj}]=\mathbb{E}[\mathbb{E}[\theta_{tj}|\mathcal{I}]]&=\mathbb{E}[\mu^{\mathcal{I}}_{tj}]\\&=\mathbb{E}[a_h\theta^{\mathcal{I}}_{(t-1)j}+a^c_h\overline{\theta}_{(t-1)j}]\text{ by (14).}\\&=a_h\mathbb{E}[\theta^{\mathcal{I}}_{tj}]+a^c_h\overline{\theta}_{tj}\\&=a_h\overline{\theta}_{(t-1)j}+a^c_h\overline{\theta}_{(t-1)j}\\&=\overline{\theta}_{(t-1)j}.\end{aligned}$$

In the same way, for the variance we have

$$\begin{aligned}\mathbb{V}[\theta_{tj}]&=\mathbb{E}[\mathbb{V}[\theta_{tj}|\mathcal{I}]]+\mathbb{V}[\mathbb{E}[\theta_{tj}|\mathcal{I}]]\\&=\mathbb{E}[v^{\mathcal{I}}_{tj}]+\mathbb{V}[\mu^{\mathcal{I}}_{tj}]\\&=v^{\mathcal{I}}_{tj}+\mathbb{V}[a_h\theta^{\mathcal{I}}_{(t-1)j}+a^c_h\overline{\theta}_{(t-1)j}]\text{ by (14).}\\&=(1-a_h^2)V_{t-1}+a_h^2V_{t-1}\\&=V_{t-1}.\end{aligned}$$

□

B. Proof of Proposition III.2

Proof. Using the notation introduced previously, the KLD between the propagating densities $\overline{\gamma}_{\theta,t^-}(x_t,\theta_t)$ and the optimal filtering $\overline{\gamma}_{\theta,t}(x_t,\theta_t)$ at time t is given by

$$\mathfrak{D}_{KL}(t)=\int_{\mathcal{Z}}\log\left(\frac{\overline{\gamma}_{\theta,t^-}(x_t,\theta_t)}{\overline{\gamma}_{\theta,t}(x_t,\theta_t)}\right)\overline{\gamma}_{\theta,t^-}(x_t,\theta_t)dx_td\theta_t,$$

where $\mathcal{Z}$ is the augmented state space $\mathbb{X}\times\Theta$.

Using Bayes's theorem and the Markov property of the hidden states, we obtain

$$\overline{\gamma}_{\theta,t}(x_t,\theta_t)=\frac{p(y_t|x_t,\theta_t)\overline{\gamma}_{\theta,t^-}(x_t,\theta_t)}{p(y_t|y_{1:t-1},\theta_t)}.$$

Substituting this into the expression for $\mathfrak{D}_{KL}(t)$, we have

$$\begin{aligned}\mathfrak{D}_{KL}(t)=\int_{\mathcal{Z}}\log\left(\frac{\int_{\mathcal{X}}p(y_t|x_t,\theta_t)\overline{\gamma}_{\theta,t^-}(x_t,\theta_t)dx_t}{p(y_t|x_t,\theta_t)}\right)\\\times\overline{\gamma}_{\theta,t^-}(x_t,\theta_t)dx_td\theta_t.\end{aligned}$$

For nonlinear and non-Gaussian state space models, this integral is intractable in most cases, the key idea is to approximate it iteratively using an approximation of $\overline{\gamma}_{\theta,t}(dx_t,d\theta_t)$ as follows

$$\widehat{\overline{\gamma}}^N_{\theta,t}(x_t,\theta_t)=\sum_{i=1}^N\overline{w}^i_{\theta,t}\mathcal{K}_{h_{t-1}}(\theta_t;\psi^i_t)\delta_{x^i_t}(x_t).$$

Note that $\widehat{\overline{\gamma}}^N_{\theta,t}(\theta_t,x_t)$ is a discrete approximation in x_t but is continuous in θ_t. Furthermore, by the Chapman-Kolmogorov propagation (2) and taking the transition density $p(x_t|x_{t-1},\theta_t)$ for the proposal density q_θ we obtain for the approximation of $\overline{\gamma}_{\theta,t^-}(x_t,\theta_t)$

$$\widehat{\overline{\gamma}}^N_{\theta,t^-}(\theta_t,x_t)=\sum_{i=1}^N w^i_{\theta,t-1}p(x_t|x^i_{t-1},\theta^i_t)\mathcal{K}_{h_{t-1}}(\theta_t;\psi^i_t)\delta_{x^i_t}(x_t)$$

Hence, we work with an empirical version of $\mathfrak{D}_{KL}$ given by

$$\begin{aligned}\widehat{\mathfrak{D}}^N_{KL}(t)&=\int_{\mathcal{Z}}\log\left(\frac{\int_{\mathcal{X}}p(y_t|x_t,\theta_t)\sum_{i=1}^N A^i_{t-1}\delta_{x^i_t}(x_t)dx_t}{p(y_t|x_t,\theta_t)}\right)\\&\qquad\times\sum_{i=1}^N A^i_t\delta_{x^i_t}(x_t)\\&=\sum_{i=1}^N A^i_{t-1}\log\left(\frac{\sum_{i=1}^N w^i_{\theta,t-1}B^i_{t-1}}{p(y_t|x^i_t,\theta^i_t)}\right),\end{aligned}$$

where,

$$A^i_{t-1}=w^i_{\theta,t-1}p(x_t|x^i_{t-1},\theta^i_t)\mathcal{K}_{h_{t-1}}(\theta_t;\psi^i_t),$$

and

$$B^i_{t-1}=p(y_t|x^i_t,\theta^i_t)p(x_t|x^i_{t-1},\theta^i_t)\mathcal{K}_{h_{t-1}}(\theta_t;\psi^i_t).$$

Now using the recursivity of the weights (8) with kernel $\mathcal{K}_{h_{t-1}}$ and after normalization, we get

$$\widehat{\mathfrak{D}}^N_{KL}(t)=\sum_{i=1}^N A^i_{t-1}\log\left(\frac{A^i_{t-1}}{w^i_{\theta,t}}\right).$$

Hence, the optimal bandwidth $\widehat{h}_t$ is given as

$$\begin{aligned}\widehat{h}_t&=\underset{h_t\in[0,1]}{\operatorname{argmin}}\left\{-\sum_{i=1}^N A^i_{t-1}\log\left(\frac{w^i_{\theta,t}}{A^i_{t-1}}\right)\right\}\\&=\underset{h_t\in[0,1]}{\operatorname{argmin}}\left\{-\sum_{i=1}^N A^i_{t-1}\log w^i_{\theta,t}\right\}.\end{aligned}$$

where the last line follows from the fact that: $\sum_{i=1}^N A^i_{t-1}\log\left(A^i_{t-1}\right)$, is independent of h_t. □

REFERENCES

[1] C. Andrieu, A. Doucet, and R. Holenstein. Particle Markov chain Monte Carlo methods. *J. R. Stat. Soc. Ser. B Methodol.*, 72(3):269–342, 2010.

[2] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *IEEE Trans. Signal Process.*, 50(2):174–188, 2002.

[3] C. Berzuini, N. G. Best, W. R. Gilks, and C. Larizza. Dynamic conditional independence models and markov chain monte carlo methods. *Journal of the American Statistical Association*, 92(440):1403–1412, 1997.

[4] P. Bickel, B. Li, and T. Bengtsson. Sharp failure rates for the bootstrap particle filter in high dimensions. In *Pushing the limits of contemporary statistics: Contributions in honor of Jayanta K. Ghosh*, volume 3, pages 318–330. Institute of Mathematical Statistics, 2008.

[5] A. Brockwell, P. Del Moral, and A. Doucet. Sequentially interacting markov chain monte carlo methods. 2010.

[6] F. Campillo and V. Rossi. Convolution particle filter for parameter estimation in general state-space models. *IEEE Trans. Aerosp. Electron. Syst.*, 45(3):1063–1072, 2009.

[7] O. Cappe. Online sequential monte carlo em algorithm. In *2009 IEEE/SP 15th Workshop on Statistical Signal Processing*, pages 37–40, 2009.

[8] T. T. T. Chau, P. Ailliot, V. Monbet, and P. Tandeo. Comparison of simulation-based algorithms for parameter estimation and state reconstruction in nonlinear state-space models. *Discrete Contin. Dyn. Syst. - S*, 2022.

[9] S. B. Chitralekha, J. Prakash, H. Raghavan, R. Gopaluni, and S. L. Shah. A comparison of simultaneous state and parameter estimation schemes for a continuous fermentor reactor. *J. Process Control*, 20(8):934–943, 2010.

[10] N. Chopin. A sequential particle filter method for static models. *Biometrika*, 89(3):539–552, 2002.

[11] N. Chopin, P. E. Jacob, and O. Papaspiliopoulos. Smc2: an efficient algorithm for sequential analysis of state space models. *J. R. Stat. Soc., B: Stat. Methodol.*, 75(3):397–426, 2013.

[12] N. Chopin and Singh. On particle Gibbs sampling. *Bernoulli*, 21:1855–1883, 2015.

[13] J. Cornebise, É. Moulines, and J. Olsson. Adaptive methods for sequential importance sampling with application to state space models. *Stat. Comput.*, 18:461–480, 2008.

[14] D. Crisan and J. Míguez. Nested particle filters for online parameter estimation in discrete-time state-space Markov models. *Bernoulli*, 24(4A):3039 – 3086, 2018.

[15] H.-D. Dau and N. Chopin. On the complexity of backward smoothing algorithms. *arXiv preprint arXiv:2207.00976*, 2022.

[16] P. Del Moral, A. Doucet, and A. Jasra. Sequential monte carlo samplers. *J. R. Stat. Soc., B: Stat. Methodol.*, 68(3):411–436, 2006.

[17] B. Delyon, M. Lavielle, E. Moulines, et al. Convergence of a stochastic approximation version of the em algorithm. *Ann. Stat.*, 27(1):94–128, 1999.

[18] A. Doucet, S. Godsill, and C. Andrieu. On sequential monte carlo sampling methods for bayesian filtering. *Stat. Comput.*, 10(3):197–208, 2000.

[19] A. Doucet and A. M. Johansen. A tutorial on particle filtering and smoothing: Fifteen years later. *Handbook of nonlinear filtering*, 12(656-704):3, 2009.

[20] S. El Kolei and F. Patras. Analysis, detection and correction of misspecified discrete time state space models. *J. Comput. Appl. Math.*, 333:200–214, 2018.

[21] V. Elvira, J. Miguez, and P. M. Djurić. On the performance of particle filters with adaptive number of particles. *Statistics and Computing*, 31:1–18, 2021.

[22] P. Fearnhead. Markov chain monte carlo, sufficient statistics, and particle filters. *Journal of Computational and Graphical Statistics*, 11(4):848–862, 2002.

[23] W. R. Gilks and C. Berzuini. Following a moving target—monte carlo inference for dynamic bayesian models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(1):127–146, 2001.

[24] S. Godsill and T. Clapp. Improvement strategies for monte carlo particle filters. In *Sequential Monte Carlo methods in practice*, pages 139–158. Springer, 2001.

[25] S. J. Godsill, A. Doucet, and M. West. Monte carlo smoothing for nonlinear time series. *J. Am. Stat. Assoc.*, 99(465):156–168, 2004.

[26] A. Golightly and D. J. Wilkinson. Bayesian sequential inference for nonlinear multivariate diffusions. *Statistics and Computing*, 16:323–338, 2006.

[27] N. J. Gordon, D. J. Salmond, and A. F. Smith. Novel approach to nonlinear/non-gaussian bayesian state estimation. In *IEE proceedings F (radar and signal processing)*, volume 140, pages 107–113. IET, 1993.

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

[29] N. Ketkar. Stochastic gradient descent. *Deep learning with Python: A hands-on introduction*, pages 113–132, 2017.

[30] D. P. Kingma and M. Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2022.

[31] G. Kitagawa. A self-organizing state-space model. *J. Am. Stat. Assoc.*, pages 1203–1215, 1998.

[32] D. A. Knowles. Stochastic gradient variational bayes for gamma approximating distributions. arxiv e-prints, page. *arXiv preprint arXiv:1509.01631*, 2015.

[33] F. Le Gland and N. Oudjane. Stability and uniform approximation of nonlinear filters using the hilbert metric and application to particle filters. *Ann. Appl. Probab.*, 14(1):144–187, 2004.

[34] F. Liang, C. Liu, and R. Carroll. *Advanced Markov chain Monte Carlo methods: learning from past samples*. John Wiley & Sons, 2011.

[35] F. Lindsten. An efficient stochastic approximation em algorithm using conditional particle filters. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 6274–6278. IEEE, 2013.

[36] F. Lindsten. *Particle filters and Markov chains for learning of dynamical systems*. PhD thesis, Linköping

University Electronic Press, 2013.

[37] F. Lindsten, M. I. Jordan, and T. B. Schon. Particle gibbs with ancestor sampling. *J. Mach. Learn. Res.*, 15:2145–2184, 2014.

[38] F. Lindsten, T. Schön, and M. I. Jordan. Ancestor sampling for particle gibbs. In *Advances in Neural Information Processing Systems*, pages 2591–2599, 2012.

[39] F. Lindsten and T. B. Schön. On the use of backward simulation in the particle gibbs sampler. In *2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3845–3848. IEEE, 2012.

[40] J. Liu and M. West. Combined parameter and state estimation in simulation-based filtering. In *Sequential Monte Carlo methods in practice*, pages 197–223. Springer, 2001.

[41] D. Luengo, L. Martino, M. Bugallo, V. Elvira, and S. Särkkä. A survey of monte carlo methods for parameter estimation. *EURASIP Journal on Advances in Signal Processing*, 2020(1):25, 2020.

[42] L. Martino. A review of multiple try mcmc algorithms for signal processing. *Digital Signal Processing*, 75:134–152, 2018.

[43] L. Martino, V. Elvira, and F. Louzada. Effective sample size for importance sampling based on discrepancy measures. *Signal Processing*, 131:386–401, 2017.

[44] L. Martino, F. Leisen, and J. Corander. On multiple try schemes and the particle metropolis-hastings algorithm. *arXiv preprint arXiv:1409.0051*, 2014.

[45] C. Musso, N. Oudjane, and F. Le Gland. Improving regularised particle filters. *Sequential Monte Carlo methods in practice*, pages 247–271, 2001.

[46] J. Olsson, O. Cappé, R. Douc, and E. Moulines. Sequential monte carlo smoothing with application to parameter estimation in nonlinear state space models. *Bernoulli*, 14(1):155–179, 2008.

[47] N. Oudjane and C. Musso. Progressive correction for regularized particle filters. In *Proceedings of the Third International Conference on Information Fusion*, volume 2, pages THB2–10. IEEE, 2000.

[48] P. H. Rangegowda, J. Valluru, S. C. Patwardhan, and S. Mukhopadhyay. Simultaneous and sequential state and parameter estimation using receding-horizon nonlinear kalman filter. *J. Process Control*, 109:13–31, 2022.

[49] V. Rossi and J.-P. Vila. Nonlinear filtering in discrete time: A particle convolution approach. In *Annales de l'ISUP*, volume 50, pages 71–102, 2006.

[50] T. B. Schön, A. Wills, and B. Ninness. System identification of nonlinear state-space models. *Automatica*, 47(1):39–49, 2011.

[51] F. Septier, S. K. Pang, A. Carmi, and S. Godsill. On mcmc-based particle methods for bayesian filtering: Application to multitarget tracking. In *2009 3rd IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)*, pages 360–363. IEEE, 2009.

[52] F. Septier and G. W. Peters. Langevin and hamiltonian based sequential mcmc for efficient bayesian filtering in high-dimensional spaces. *IEEE Journal of selected topics in signal processing*, 10(2):312–327, 2015.

[53] F. Septier and G. W. Peters. An overview of recent advances in monte-carlo methods for bayesian filtering in high-dimensional spaces. *Theoretical aspects of spatial-temporal modeling*, pages 31–61, 2015.

[54] C. Snyder, T. Bengtsson, P. Bickel, and J. Anderson. Obstacles to high-dimensional particle filtering. *Monthly Weather Review*, 136(12):4629–4640, 2008.

[55] A. Svensson and F. Lindsten. Learning dynamical systems with particle stochastic approximation em. *arXiv preprint arXiv:1806.09548*, 2018.

[56] A. Svensson, T. B. Schön, and M. Kok. Nonlinear state space smoothing using the conditional particle filter. *IFAC-PapersOnLine*, 48(28):975–980, 2015.

[57] A. Tulsyan, B. Huang, R. B. Gopaluni, and J. F. Forbes. On simultaneous on-line state and parameter estimation in non-linear state-space models. *J. Process Control*, 23(4):516–526, 2013.

[58] G. C. Wei and M. A. Tanner. A monte carlo implementation of the em algorithm and the poor man's data augmentation algorithms. *J. Am. Stat. Assoc.*, 85(411):699–704, 1990.