A probabilistic scheme for joint parameter estimation and state prediction in complex dynamical systems

Sara Pérez-Vieites*

Department of Signal Theory & Communications, Universidad Carlos III de Madrid. Avenida de la Universidad 30, 28911 Leganés, Madrid (Spain).

Inés P. Mariño†

Department of Biology and Geology, Physics and Inorganic Chemistry, Universidad Rey Juan Carlos. C/ Tulipán s/n, 28933 Móstoles, Madrid, Spain.

Joaquín Míguez‡

Department of Signal Theory & Communications, Universidad Carlos III de Madrid. Avenida de la Universidad 30, 28911 Leganés, Madrid (Spain).

(Dated: August 15, 2017)

Abstract

Many problems in the geophysical sciences demand the ability to calibrate the parameters and predict the time evolution of complex dynamical models using sequentially-collected data. Here we introduce a general methodology for the joint estimation of the static parameters and the forecasting of the state variables of nonlinear, and possibly chaotic, dynamical models. The proposed scheme is essentially probabilistic. It aims at recursively computing the sequence of joint posterior probability distributions of the unknown model parameters and its (time varying) state variables conditional on the available observations. The latter are possibly partial and contaminated by noise. The new framework combines a Monte Carlo scheme to approximate the posterior distribution of the fixed parameters with filtering (or *data assimilation*) techniques to track and predict the distribution of the state variables. For this reason, we refer to the proposed methodology as *nested filtering*. In this paper we specifically explore the use of Gaussian filtering methods, but other approaches fit naturally within the new framework. As an illustrative example, we apply three different implementations of the methodology to the tracking of the state, and the estimation of the fixed parameters, of a stochastic two-scale Lorenz 96 system. This model is commonly used to assess data assimilation procedures in meteorology. For this example, we compare different nested filters and show estimation and forecasting results for a 4,000-dimensional system.

* spvieites@tsc.uc3m.es

† ines.perez@urjc.es

‡ joaquin.miguez@uc3m.es

A common feature to many problems in some of the most active fields of science is the need to calibrate (i.e., to estimate the parameters) and then forecast the time evolution of complex (often high-dimensional) dynamical systems using sequentially-collected observations. One can find obvious examples in meteorology, where current models for global weather forecasting involve the tracking of millions of time-varying state variables, as well as in oceanography or in climate modelling. Traditionally, model calibration and state tracking and forecasting have been addressed separately. The problem of state tracking is often termed data assimilation *in Geophysics, while it is referred as* stochastic *or* Bayesian *filtering by researchers in applied probability. Carrying out the two tasks jointly, parameter estimation and state forecasting, is a hard problem posing several practical and theoretical difficulties. Only in the last few years there have been advances leading to well-principled methods that solve this joint problem numerically with theoretical guarantees of performance. However, existing procedures are computationally too expensive to be applied in real-world applications involving more than a few tens of unknown variables and/or parameters.*

In this paper we introduce a general scheme for joint parameter estimation and state tracking and forecasting in partially observed dynamical systems. The methodology is probabilistic and it involves two layers of estimators, one for the static parameters and another one for the time-varying state variables. It can be interpreted that the state estimators are nested *within a main algorithm that tackles the estimation of the parameters. For this reason we refer to the overall scheme as a* nested filter*. The methodology is devised for systems where the number of static parameters to be estimated is moderate, while the number of state variables can be much larger. Different instances of nested filters can be constructed by choosing different estimators for the state variables, while we propose to implement parameter estimation via a sequential Monte Carlo procedure. We have obtained theoretical results for this general class of nested filters based on a generic assumption on the outputs of the state trackers. As an example, we have implemented three versions of the method to estimate the parameters and forecast the evolution of a partially-observed stochastic Lorenz 96 system with up to 4,000 state variables.*

I. INTRODUCTION

A common feature to many problems in some of the most active fields of science is the need to calibrate (i.e., estimate the parameters) and then forecast the time evolution of complex (often very high-dimensional) dynamical systems using sequentially-collected observations. One can find obvious examples in meteorology, where current models for global weather forecasting involve the tracking of millions of time-varying state variables [1], as well as in oceanography [2] or in climate modelling [3]. This problem is not constrained to Geophysics, though. In Biochemistry and Ecology it is often necessary to forecast the evolution of populations of interacting species (typically animal and/or vegetal species in Ecology and different types of reacting molecules in Biochemistry), which usually involves the estimation of the parameters that govern the interaction as well [4].

Traditionally, model calibration, i.e., the estimation or adjustment of the model static parameters, and the tracking and forecasting of the time-varying state variables have been addressed separately. The problem of state tracking is often termed *data assimilation* in Geophysics, while it is referred as *stochastic* or *Bayesian* filtering by researchers in computational statistics and applied probability. Carrying out both tasks jointly, parameter estimation and state forecasting, is a hard problem posing several practical and theoretical difficulties. A number of heuristic procedures have been suggested (see, e.g., [5, 6]) however they are subject to problems related to observability (i.e., ambiguities) and there are no performance guarantees. Only in the last few years there have been advances leading to well-principled probabilistic methods that solve the joint problem numerically and supported by rigorous performance analyses [7, 8].

Such procedures belong to the class of Bayesian methodologies. They aim at calculating the posterior probability distribution of *all the unknown variables and parameters* of the model. Every unknown in the system, either a static parameter or time-evolving state variable, is modelled as random and, therefore, it is possible in principle to compute (or at least approximate) its conditional probability distribution given the available data. These conditional, or *posterior*, distributions contain all the information relevant for the estimation task. From them, one can compute point estimates of the parameters and states but also quantify the estimation error. However, state-of-the-art methods for Bayesian parameter estimation and stochastic filtering are batch techniques, i.e., they process the whole set of

available observations repeatedly in order to produce numerical solutions. For this reason, they are not well suited to problems where observations are collected sequentially and have to be processed as they arrive (or, simply, when the sequence of observations is too long). The popular particle Markov chain Monte Carlo (pMCMC) [7] and the sequential Monte Carlo square (SMC^2) [8] schemes are examples of such batch methods. A common characteristic of these techniques is that they rely on Monte Carlo approximations in order to approximate the posterior probability distribution of the parameters and the states. Although there are some *recursive* schemes that enable the sequential processing of the observed data as they are collected [9], they do not yield full posterior distributions of the unknowns, but only point estimates. Therefore, it is not possible to quantify the uncertainty of the estimation or the forecast. Moreover, they are subject to various convergence (and complexity) issues, e.g., when the posterior probability distribution is multimodal, when it contains singularities or when the parameter likelihoods cannot be computed exactly.

In this paper we introduce a general probabilistic scheme to perform the joint task of parameter estimation and state tracking and forecasting. The methodology is Bayesian, i.e., it aims at the computation of the posterior probability distribution of the unknowns given the available data. It involves two layers of estimators, one for the static parameters and another one for the time-varying state variables. It can be interpreted that the state estimators and predictors are *nested* or inserted within a main algorithm that tackles the estimation of the parameters. For this reason we refer to the overall scheme as a *nested filter*. The estimation of the static parameters and the dynamic variables is carried out in a purely sequential and recursive manner. This property makes the proposed algorithm better suited for problems where long time series of data have to be handled. It can be shown that a particular case of the proposed scheme is the nested particle filter (NPF), a recursive version of the SMC^2 algorithm in [8], which has only recently been introduced in [10] and relies on a bank of particle filters [11, 12] to infer the posterior distribution of the variables and the parameters. However, in the general scheme that we propose here it is possible to replace the computationally heavy particle filters by simpler algorithms, easier to apply in practical problems. In particular, we propose a new class of nested hybrid filters, which use Gaussian filters such as the extended Kalman filter [13] or ensemble Kalman filter [14] for the forecasting of the state variables. An important reduction of running times in comparison with the NPF is achieved without a significant loss of accuracy.

The proposed methodology is devised for systems where the number of static parameters to be estimated is moderate, while the number of state variables can be much larger. Different instances of nested filters can be constructed by choosing different numerical techniques for state tracking and forecasting, while we propose to implement Bayesian parameter estimation via a sequential Monte Carlo procedure. We have obtained theoretical results for this general class of nested filters based on a generic assumption on the outputs of the method used to track the state variables.

To illustrate the performance of the novel method, we present the results of computer simulations with a stochastic two-scale Lorenz 96 model [15] with underlying chaotic dynamics. In meteorology, the two-scale Lorenz 96 model is commonly used as a benchmark system for data assimilation [16] and parameter estimation techniques [15] because it displays the basic physical features of atmospheric dynamics [17] (e.g., connection and sensitivity to perturbations). We have implemented three versions of the nested filtering method in order to estimate the parameters and forecast the evolution of up to 4,000 state variables.

The rest of the paper is organized as follows. In Section II the continuous-time and discrete-time state-space models are presented and we state the Bayesian inference problem to be solved. The nested filtering methods are introduced and explained in Section III, including the proposed nested hybrid filtering scheme. An asymptotic convergence theorem is stated and proved in Section IV. In Section V, the stochastic Lorenz 96 model which is used in the simulations is described and then, some illustrative numerical results are presented in Section VI. Finally, Section VII is devoted to the conclusions.

II. DYNAMICAL MODEL AND PROBLEM STATEMENT

A. From continuous-time to discrete-time dynamical systems

Let us consider a nonlinear and possibly chaotic dynamical system described by the multidimensional ordinary differential equation

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}(t), \boldsymbol{\theta}) \tag{1}$$

where t denotes continuous time, $\boldsymbol{x}(t) \in \mathbb{R}^{d_x}$ is the d_x-dimensional system state, $\boldsymbol{f}$ is a nonlinear function parametrized by a fixed $d_\theta \times 1$ vector of unknown parameters,

$\boldsymbol{\theta} = [\theta_1, \ldots, \theta_{d_\theta}]^\top \in \mathbb{R}^{d_\theta}$, and $\dot{\boldsymbol{x}}(t)$ denotes the vector of time derivatives $\dot{\boldsymbol{x}}(t) = [\dot{x}_1(t), \ldots, \dot{x}_{d_x}(t)]^\top$, with $\dot{x}_i(t) = \frac{dx_i}{dt}$.

The computational representation of the system in Eq. (1) requires a time-discretization scheme. We assume a grid $\{t_k\}_{k=0,1,\ldots}$ where $t_k = kh$, $h > 0$ is a time-discretization step and $k = 0, 1, \ldots$ is an integer index. If we let $\tilde{\boldsymbol{x}}_k$ denote the approximate value of $\boldsymbol{x}(t)$ at $t = t_k$ then most explicit numerical integration methods (such as the Euler or Runge-Kutta schemes) yield a d_x-dimensional difference equation of the form

$$\tilde{\boldsymbol{x}}_k = \tilde{\boldsymbol{x}}_{k-1} + h\bar{\boldsymbol{f}}(\tilde{\boldsymbol{x}}_{k-1}, \boldsymbol{\theta}) \tag{2}$$

where $\bar{\boldsymbol{f}}(\tilde{\boldsymbol{x}}_{k-1}, \boldsymbol{\theta})$ is an estimate of the vector of time derivatives $\dot{\boldsymbol{x}}(t_k) = [\dot{x}_1(t_k), \ldots, \dot{x}_{d_x}(t_k)]^\top$. Different discretization schemes adopt different estimates of $\dot{\boldsymbol{x}}(t_k)$, e.g., $\bar{\boldsymbol{f}} = \boldsymbol{f}$ for the Euler method, while Runge-Kutta yield more elaborate estimates.

Finally, we consider a stochastic version of Eq. (2) obtained by adding a statistically independent perturbation $\boldsymbol{v}_k = [v_{1,k}, \ldots, v_{d_x,k}]^\top \in \mathbb{R}^{d_x}$ at each time step. This yields the random sequence

$$\tilde{\boldsymbol{x}}_k = \tilde{\boldsymbol{x}}_{k-1} + h\bar{\boldsymbol{f}}(\tilde{\boldsymbol{x}}_{k-1}, \boldsymbol{\theta}) + \sigma\boldsymbol{v}_k, \tag{3}$$

where $\sigma \geq 0$ is a parameter that controls the power of the perturbations. For $\sigma = 0$ we recover Eq. (2). In general, we set $\sigma > 0$, though. The value of this parameter should be small enough to preserve the underlying dynamics of the system. The noise terms introduce additional degrees of freedom in the discrete-time model and enable a probabilistic analysis of the system and the characterization of the uncertainty of any resulting numerical estimates of the state $\boldsymbol{x}(t)$ or the unknown $\boldsymbol{\theta}$.

B. Observations

In this paper we address the problem of estimating the sequence of states, $\tilde{\boldsymbol{x}}_k$, and the vector of unknown parameters, $\boldsymbol{\theta}$, from a sequence of observation vectors, that we model as

$$\tilde{\boldsymbol{y}}_{kT} = \boldsymbol{g}(\tilde{\boldsymbol{x}}_{kT}, \boldsymbol{\theta}) + \sigma_o\tilde{\boldsymbol{u}}_{kT}, \quad k = 1, 2, \ldots, \quad T \geq 1, \tag{4}$$

where $\boldsymbol{g} : \mathbb{R}^{d_x} \to \mathbb{R}^{d_y}$ is a transformation that maps the state into a real vector of dimension d_y (with $d_y \leq d_x$), T is the discrete observation period[18] and $\tilde{\boldsymbol{u}}_k$ is a sequence of zero-mean independent vectors representing observational noise, whose power is scaled by a known

factor $\sigma_o > 0$. Note that the observation function of Eq. (4) is indexed by the same vector of unknown parameters, $\boldsymbol{\theta}$, as the dynamic Eq. (3), although not every parameter θ_i, $i \in \{1, \ldots, d_\theta\}$, necessarily appears in both equations.

C. Discrete-time state-space model

The sequences $\tilde{\boldsymbol{x}}_k$ and $\tilde{\boldsymbol{y}}_{kT}$ run on different time scales, with one observation vector $\tilde{\boldsymbol{y}}_{kT}$ collected for every subsequence $\tilde{\boldsymbol{x}}_{(k-1)T+1}, \ldots, \tilde{\boldsymbol{x}}_{kT}$ of T consecutive state vectors. Since any estimator of $\tilde{\boldsymbol{x}}_k$ and $\boldsymbol{\theta}$ actually depends on the available data, it is convenient to rewrite the dynamic model in the time scale of the observations. In particular, we hereafter work with the pair of random sequences $\boldsymbol{x}_n \coloneqq \tilde{\boldsymbol{x}}_{nT}$ and $\boldsymbol{y}_n \coloneqq \tilde{\boldsymbol{y}}_{nT}$. The observation equation on the new discrete-time scale follows trivially from Eq. (4), namely

$$\boldsymbol{y}_n = \boldsymbol{g}(\boldsymbol{x}_n, \boldsymbol{\theta}) + \sigma_o \boldsymbol{u}_n, \quad n = 1, 2, \ldots \tag{5}$$

where $\boldsymbol{u}_n \coloneqq \tilde{\boldsymbol{u}}_{nT}$.

In order to obtain a dynamic equation for the sequence $\boldsymbol{x}_n$, however, we need to iterate Eq. (3) T times. To be specific, $\boldsymbol{x}_n \coloneqq \tilde{\boldsymbol{x}}_{nT}$ is generated from $\boldsymbol{x}_{n-1} \coloneqq \tilde{\boldsymbol{x}}_{(n-1)T}$ in T steps as

$$\begin{aligned}
\tilde{\boldsymbol{x}}_{(n-1)T} &= \boldsymbol{x}_{n-1} \\
\tilde{\boldsymbol{x}}_{(n+1)T+1} &= \tilde{\boldsymbol{x}}_{(n-1)T} + \bar{\boldsymbol{f}}(\tilde{\boldsymbol{x}}_{(n-1)T}, \boldsymbol{\theta}) + \sigma \tilde{\boldsymbol{v}}_{(n-1)T+1} \\
&\vdots \\
\tilde{\boldsymbol{x}}_{nT-1} &= \tilde{\boldsymbol{x}}_{nT-2} + \bar{\boldsymbol{f}}(\tilde{\boldsymbol{x}}_{nT-2}, \boldsymbol{\theta}) + \sigma \tilde{\boldsymbol{v}}_{nT-1} \\
\boldsymbol{x}_n &= \tilde{\boldsymbol{x}}_{nT-1} + \bar{\boldsymbol{f}}(\tilde{\boldsymbol{x}}_{nT-1}, \boldsymbol{\theta}) + \sigma \tilde{\boldsymbol{v}}_{nT}
\end{aligned} \tag{6}$$

and we concisely represent the transformation in Eq. (6) as

$$\boldsymbol{x}_n = \bar{\boldsymbol{F}}_{T,\sigma}(\boldsymbol{x}_{n-1}, \boldsymbol{\theta}, \boldsymbol{v}_n), \tag{7}$$

where $\bar{\boldsymbol{F}} : \mathbb{R}^{d_x} \times \mathbb{R}^{d_\theta} \times \mathbb{R}^{Td_x} \to \mathbb{R}^{d_x}$ and $\boldsymbol{v}_n = \left[\tilde{\boldsymbol{v}}_{(n-1)T+1}, \ldots, \tilde{\boldsymbol{v}}_{nT}\right]^\top \in \mathbb{R}^{Td_x}$ is the sequence of random perturbations in the original dynamical model of Eq. (3).

Eqs. (6) and (7), together with a probability distribution of the initial condition $\boldsymbol{x}_0$ (note that the first observation is collected at time $n = 1$, hence at $t = T$) yield a Markov state-space model in discrete time.

In this paper we aim at devising recursive methods to

- estimate the parameter vector, $\boldsymbol{\theta}$,
- estimate the state $\boldsymbol{x}_n = \tilde{\boldsymbol{x}}_{nT}$, and
- predict the sequence $\tilde{\boldsymbol{x}}_{nT+1}, \ldots, \tilde{\boldsymbol{x}}_{(n+1)T}$,

at each $n = 1, 2, ...$ given the observation record $\boldsymbol{y}_{1:n} = \{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_n\}$. Such prediction and estimation methods are better described using a probabilistic notation. Given a random vector $\boldsymbol{z}$, let $p(\boldsymbol{z})$ denote the probability density function (pdf) of $\boldsymbol{z}$. This is an argument-wise notation: if we have two random variables $\boldsymbol{z}_1$ and $\boldsymbol{z}_2$, then $p(\boldsymbol{z}_1)$ and $p(\boldsymbol{z}_2)$ denote the pdf's of $\boldsymbol{z}_1$ and $\boldsymbol{z}_2$ respectively, even if the two functions are different. Similarly, $p(\boldsymbol{z}_1, \boldsymbol{z}_2)$ denotes the joint pdf and $p(\boldsymbol{z}_1|\boldsymbol{z}_2)$ is the conditional pdf of $\boldsymbol{z_1}$ given $\boldsymbol{z}_2$. The expected value of a random vector $\boldsymbol{x}$ conditional on another random vector $\boldsymbol{y}$ is denoted $\mathrm{E}[\boldsymbol{x}|\boldsymbol{y}]$. This kind of notation is conventional in Bayesian analysis.

The state-space model comprising the dynamics of $\boldsymbol{x}_n$ and the observation $\boldsymbol{y}_n$ can be represented by the triplet

$$\boldsymbol{x}_0 \sim p(\boldsymbol{x}_0) \tag{8}$$

$$\boldsymbol{x}_n \sim p(\boldsymbol{x}_n|\boldsymbol{x}_{n-1}, \boldsymbol{\theta}) \tag{9}$$

$$\boldsymbol{y}_n \sim p(\boldsymbol{y}_n|\boldsymbol{x}_n, \boldsymbol{\theta}) \tag{10}$$

where $p(\boldsymbol{x}_0)$ is the a priori pdf of the state (i.e. a probabilistic characterization of the system initial condition), $p(\boldsymbol{x}_n|\boldsymbol{x}_{n-1}, \boldsymbol{\theta})$ is the conditional pdf of $\boldsymbol{x}_n$ given the state $\boldsymbol{x}_{n-1}$ and the parameters in $\boldsymbol{\theta}$, and $p(\boldsymbol{y}_n|\boldsymbol{x}_n, \boldsymbol{\theta})$ is the conditional pdf of the observation given the state and the parameters. Let us remark that:

- The a priori pdf $p(\boldsymbol{x}_0)$ can be replaced by a delta distribution if the initial condition of $\boldsymbol{x}_0$ is known.
- The pdf $p(\boldsymbol{x}_n|\boldsymbol{x}_{n-1}, \boldsymbol{\theta})$ may not have, in general, a closed-form expression. However, it is relatively straightforward to generate a Monte Carlo sample $\boldsymbol{x}_n$ given $\boldsymbol{x}_{n-1}$, and $\boldsymbol{\theta}$ using the multi-step transformation of Eq. (6) (and assuming that it is possible to draw samples from the noise pdf $p(\tilde{\boldsymbol{v}}_k)$).
- The observations are conditionally independent given the states and the parameter

vector $\boldsymbol{\theta}$. In particular, the joint pdf $p(\boldsymbol{y}_{1:n}|\boldsymbol{x}_{1:n}, \boldsymbol{\theta})$ is factorized as

$$p(\boldsymbol{y}_{1:n}|\boldsymbol{x}_{1:n}, \boldsymbol{\theta}) = \prod_{j=1}^{n} p(\boldsymbol{y}_j|\boldsymbol{x}_j, \boldsymbol{\theta}) \tag{11}$$

As we adopt a Bayesian approach to tackle the estimation of $\boldsymbol{x}_n$ and $\boldsymbol{\theta}$, we need to model the parameter vector $\boldsymbol{\theta}$ as random as well. Hence, we augment the state-space model (8)-(10) with a prior pdf for $\boldsymbol{\theta}$, denoted $p(\boldsymbol{\theta})$.

D. Problem statement

The main goal of this paper is to introduce new and efficient recursive Monte Carlo methods for the approximation of the sequence of probability distributions, described by the pdf's

$$p(\boldsymbol{x}_n, \boldsymbol{\theta}|\boldsymbol{y}_{1:n}), \quad n = 1, 2, \ldots \tag{12}$$

Given the approximate distributions, point-estimates of $\boldsymbol{\theta}$ and $\boldsymbol{x}_n$ can be computed at each discrete-time step n. We will also see that, as a by-product of the approximation of $p(\boldsymbol{x}_n, \boldsymbol{\theta}|\boldsymbol{y}_{1:n})$, the proposed methods also produce predictions of

$$\tilde{\boldsymbol{x}}_{nT+1} \approx \boldsymbol{x}(h(nT+1)), \ldots, \tilde{\boldsymbol{x}}_{(n+1)T-1} \approx \boldsymbol{x}(h(n+1)T - h) \tag{13}$$

at time $t = nTh$.

III. NESTED FILTERS FOR PARAMETER AND STATE ESTIMATION

In this section we introduce a general recursive scheme for the approximation of the sequence of probability measures

$$\pi_n(d\boldsymbol{\theta}, d\boldsymbol{x}_n) := p(\boldsymbol{\theta}, \boldsymbol{x}_n|\boldsymbol{y}_{1:n}) d\boldsymbol{\theta} d\boldsymbol{x}_n \quad n = 1, 2, \ldots \tag{14}$$

The key ingredient of the methodology is a sequentially-computed Monte Carlo approximation of the posterior probability measure of the unknown parameters,

$$\mu_n(d\boldsymbol{\theta}) := p(\boldsymbol{\theta}|\boldsymbol{y}_{1:n}) d\boldsymbol{\theta}. \tag{15}$$

At each step n, the approximate measure with N Monte Carlo samples has the form $\mu_n^N(d\boldsymbol{\theta}) = \frac{1}{N}\sum_{i=1}^{N} \delta_{\boldsymbol{\theta}_n^i}(d\boldsymbol{\theta})$, where $\boldsymbol{\theta}_n^i$ is the i-th sample ($1 \leq i \leq N$) and $\delta_{\boldsymbol{\theta}_n^i}$ denotes the unit Dirac delta measure located at $\boldsymbol{\theta}_n^i$.

We term the scheme "nested" because, for each Monte Carlo sample $\boldsymbol{\theta}_n^i$, we need to apply a Bayesian (probabilistic) filter to approximate the posterior probability measure of the state,

$$\phi_{n,\boldsymbol{\theta}_n^i}(d\boldsymbol{x}_n) \coloneqq p(\boldsymbol{x}_n|\boldsymbol{\theta}_n^i, \boldsymbol{y}_{1:n})d\boldsymbol{x}_n, \quad n = 1, 2, \ldots \tag{16}$$

conditional on the parameter vector $\boldsymbol{\theta}_n^i$. The combination of these two probability measures yields the joint distribution of the parameters and the dynamic states of the system, namely

$$\begin{aligned} \pi_n(d\boldsymbol{\theta} \times d\boldsymbol{x}_n) &= \phi_{n,\boldsymbol{\theta}}(d\boldsymbol{x}_n)\mu_n(d\boldsymbol{\theta}) \\ &= p(\boldsymbol{x}_n|\boldsymbol{\theta}, \boldsymbol{y}_{1:n})p(\boldsymbol{\theta}|\boldsymbol{y}_{1:n})d\boldsymbol{x}_n d\boldsymbol{\theta} \\ &= p(\boldsymbol{\theta}, \boldsymbol{x}_n|\boldsymbol{y}_{1:n})d\boldsymbol{x}_n d\boldsymbol{\theta} \end{aligned} \tag{17}$$

where the second equality follows from Eqs. (14) and (15) and the third identity is a consequence of the definition of conditional probability.

In Section III A we introduce a general scheme for the recursive Monte Carlo approximation of $\mu_n(d\boldsymbol{\theta})$, which yields the basic proposed methodology, and discuss the underlying recursive computations and approximations in Section III B. Then, in Section III C we propose two practical methods that rely on different Gaussian (Kalman-like) approximations of the conditional filter $\phi_{n,\boldsymbol{\theta}}(d\boldsymbol{x}_n)$.

A. Nested filtering

In order to build up the proposed methodology, let us focus on the computation of the posterior measure of the parameters $\mu_n(d\boldsymbol{\theta}) = p(\boldsymbol{\theta}|\boldsymbol{y}_{1:n})d\boldsymbol{\theta}$. From a Bayesian perspective, $\mu_n(d\boldsymbol{\theta})$ contains all the statistical information for the estimation of $\boldsymbol{\theta}$ at time n, however, it cannot be computed in closed-form in general.

We seek a Monte Carlo approximation of $\mu_n(d\boldsymbol{\theta})$ and one simple way of attaining this is to apply the importance sampling (IS) method [19] sequentially. Let $q_n(\boldsymbol{\theta})$ be a proposal, or importance, pdf. The following (naive) algorithm yields a weighted Monte Carlo approximation of $\mu_n(d\boldsymbol{\theta})$ at each time n:

1. Draw N i.i.d. samples $\boldsymbol{\theta}_n^i$, $i = 1, 2, \ldots, N$, from $q_n(\boldsymbol{\theta})$.

2. Compute importance weights,

$$\tilde{w}_n^i = \frac{p(\boldsymbol{y}_n|\boldsymbol{\theta}_n^i, \boldsymbol{y}_{1:n})p(\boldsymbol{\theta}_n^i|\boldsymbol{y}_{1_n-1})}{q_n(\boldsymbol{\theta}_n^i)}, \quad i = 1, \ldots, N, \tag{18}$$

and normalise them

$$w_n^i = \frac{\tilde{w}_n^i}{\sum_{j=1}^N \tilde{w}_n^j}, \quad i = 1, \ldots, N. \tag{19}$$

To be specific, after step 2 above we obtain the IS estimate $\mu_n^N(d\boldsymbol{\theta}) = \sum_{i=1}^N w_n^i \delta_{\boldsymbol{\theta}_n^i}(d\boldsymbol{\theta})$. Furthermore, if we choose $q_n(\boldsymbol{\theta}) = p(\boldsymbol{\theta}|\boldsymbol{y}_{1:n-1})$, then the naive sequential IS method becomes extremely simple:

1. Draw $\boldsymbol{\theta}_n^i \sim p(\boldsymbol{\theta}|\boldsymbol{y}_{1:n-1})$ i.i.d., for $i = 1, \ldots, N$.

2. Compute $w_n^i \propto u_n(\boldsymbol{\theta}_n^i)$, $i = 1, \ldots, N$.

The weight normalization is left implicit in step 2, where we have additionally introduced the notation

$$u_n(\boldsymbol{\theta}) \coloneqq p(\boldsymbol{y}_n|\boldsymbol{\theta}, \boldsymbol{y}_{1:n-1}) \tag{20}$$

for the *marginal likelihood function* of $\boldsymbol{\theta}$ at time n. The sequence of functions $u_n(\boldsymbol{\theta})$, $n = 1, 2, ...$, plays a key role in the rest of this paper.

Unfortunately, this method is not practical because

- it is not possible to draw from $p(\boldsymbol{\theta}|\boldsymbol{y}_{1:n})$, at least exactly, and

- the likelihood $u_n(\boldsymbol{\theta}_n^i)$ cannot be evaluated exactly either.

Specifically note that, given the state-space model (8)-(10), the function $u_n(\boldsymbol{\theta})$ can be written as the integral

$$u_n(\boldsymbol{\theta}) = \int p(\boldsymbol{y}_n|\boldsymbol{x}_n, \boldsymbol{\theta}) p(\boldsymbol{x}_n|\boldsymbol{\theta}, \boldsymbol{y}_{1:n-1}) d\boldsymbol{x}_n \tag{21}$$

which has no closed-form expression when the transformation $\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\theta})$ in Eq. (1) is nonlinear. Eq. (21), however, shows that we can obtain an estimate of $u_n(\boldsymbol{\theta})$ if we can first obtain a tractable approximation of the predictive measure

$$\xi_{n,\boldsymbol{\theta}}(d\boldsymbol{x}_n) \coloneqq p(\boldsymbol{x}_n|\boldsymbol{\theta}, \boldsymbol{y}_{1:n-1}) d\boldsymbol{x}_n, \tag{22}$$

i.e., an approximation for which the integral in Eq. (21) can be computed numerically.

The difficulty of drawing samples from $\mu_n(d\boldsymbol{\theta}) = p(\boldsymbol{\theta}|\boldsymbol{y}_{1:n-1}) d\boldsymbol{\theta}$ can be circumvented if we content ourselves with an approximate, or perturbed, sampling step. In particular, if we have computed a Monte Carlo approximation $\mu_{n-1}^N(d\boldsymbol{\theta}) = \frac{1}{N} \sum_{i=1}^N \delta_{\boldsymbol{\theta}_{n-1}^i}(d\boldsymbol{\theta})$ at time $n-1$

(assume all samples are equally weighted, $w_{n-1}^i = \frac{1}{N}$, for the sake of the argument) then we can draw $\boldsymbol{\theta}_n^i$, $i = 1, \ldots, N$, i.i.d. from the mixture distribution

$$\bar{\mu}_{n-1}^N(d\boldsymbol{\theta}) = \frac{1}{N}\sum_{i=1}^{N} \kappa_N(d\boldsymbol{\theta}|\boldsymbol{\theta}_{n-1}^i), \tag{23}$$

where $\kappa_N(d\boldsymbol{\theta}|\boldsymbol{\theta}')$ is a Markov kernel, i.e., a probability distribution for $\boldsymbol{\theta}$ conditioned on $\boldsymbol{\theta}'$. There are many possibilities for the choice of $\kappa_N(d\boldsymbol{\theta}|\boldsymbol{\theta}')$. A particularly simple one is the Gaussian kernel

$$\kappa_N(d\boldsymbol{\theta}|\boldsymbol{\theta}') = \mathcal{N}(\boldsymbol{\theta}|\boldsymbol{\theta}', \sigma_N^2\boldsymbol{\Sigma})d\boldsymbol{\theta}, \tag{24}$$

where $\mathcal{N}(\boldsymbol{\theta}|\boldsymbol{\theta}', \boldsymbol{\Sigma})$ denotes the Gaussian pdf with mean $\boldsymbol{\theta}'$ and covariance matrix $\boldsymbol{\Sigma}$, and we choose σ_N^2 such that $\lim_{N\to\infty} \sigma_N^2 = 0$. Most kernels $\kappa_N(d\boldsymbol{\theta}|\boldsymbol{\theta}')$ such that $\lim_{N\to\infty} \kappa_N(d\boldsymbol{\theta}|\boldsymbol{\theta}') = \delta_{\boldsymbol{\theta}'}(d\boldsymbol{\theta})$ will work in practice (intuitively, these are kernels that narrow down around the mean $\boldsymbol{\theta}'$ as N increases).

Finally, the proposed nested filtering (NF) scheme that combines the approximations described above (for the computation of $u_n(\boldsymbol{\theta})$ and the sampling step) is outlined in Algorithm 1. The terminology NF is derived from the Bayesian jargon, where posterior probability measures like $\mu_n(d\boldsymbol{\theta}) = p(\boldsymbol{\theta}|\boldsymbol{y}_{1:n})d\boldsymbol{\theta}$ are often termed "filters". The filters are "nested" because for each sample $\boldsymbol{\theta}_n^i$ in the approximation $\mu_n(d\boldsymbol{\theta}) = \frac{1}{N}\sum_{i=1}^N \delta_{\boldsymbol{\theta}_n^i} d\boldsymbol{\theta}$ we need to approximate the predictive measure $\xi_{n,\boldsymbol{\theta}_n^i}(d\boldsymbol{x}_n)$, and the conditional filter $\phi_{n,\boldsymbol{\theta}_n^i}(d\boldsymbol{x}_n)$ in the state space.

Algorithm 1 *General nested filter.*

1. *Initialization*

   Draw $\boldsymbol{\theta}_0^{(i)}, i = 1, \ldots, N$, *i.i.d. samples from* $\mu_0(d\boldsymbol{\theta}) = p(\boldsymbol{\theta})d\boldsymbol{\theta}$.

2. *Recursive step*

   (a) *For* $i = 1, \ldots, N$*:*

   i. *Draw* $\bar{\boldsymbol{\theta}}_n^{(i)}$ *from* $\kappa_N(d\boldsymbol{\theta}|\boldsymbol{\theta}_{n-1}^i)$.

   ii. *Approximate* $\hat{\xi}_{n,\bar{\boldsymbol{\theta}}_n^i}(d\boldsymbol{x}_n) \approx p(\boldsymbol{x}_n|\bar{\boldsymbol{\theta}}_n^i, \boldsymbol{y}_{1:n-1})d\boldsymbol{x}_n$.

   iii. *Use* $\hat{\xi}_{n,\bar{\boldsymbol{\theta}}_n^i}(d\boldsymbol{x}_n)$ *to compute the estimate*

$$\hat{u}_n(\bar{\boldsymbol{\theta}}_n^i) = \int p(\boldsymbol{y}_n|\bar{\boldsymbol{\theta}}_n^i, \boldsymbol{x}_n)\hat{\xi}_{n,\bar{\boldsymbol{\theta}}_n^i}(d\boldsymbol{x}_n) \approx u_n(\bar{\boldsymbol{\theta}}_n^i). \tag{25}$$

   and let $w_n^i \propto \hat{u}_n(\bar{\boldsymbol{\theta}}_n^i)$ *be the normalized weight of* $\bar{\boldsymbol{\theta}}_n^i$.

(b) Resample the discrete distribution

$$\bar{\mu}_n^N(d\boldsymbol{\theta}) = \sum_{i=1}^{N} w_n^i \delta_{\bar{\boldsymbol{\theta}}_n^i}(d\boldsymbol{\theta}) \tag{26}$$

N times with replacement in order to obtain the set $\{\boldsymbol{\theta}_{i=1}^N\}$ and the approximation $\mu_n^N(d\boldsymbol{\theta}) = \frac{1}{N}\sum_{i=1}^N \delta_{\boldsymbol{\theta}_n^i}(d\boldsymbol{\theta})$.

The random probability measure $\mu_n^N(d\boldsymbol{\theta})$ can be easily used to compute estimates of the unknown parameters and to quantify estimation errors. For example, the posterior-mean estimator of $\boldsymbol{\theta}$ can be approximated as

$$\begin{aligned} \hat{\boldsymbol{\theta}}_n &= \int \boldsymbol{\theta}\mu_n(d\boldsymbol{\theta}) \\ &\approx \int \boldsymbol{\theta}\mu_n^N(d\boldsymbol{\theta}) = \frac{1}{N}\sum_{i=1}^{N} \boldsymbol{\theta}_n^i = \hat{\boldsymbol{\theta}}_n^N \end{aligned} \tag{27}$$

i.e., the integral with respect to (w.r.t.) the true posterior measure $\mu_n(d\boldsymbol{\theta})$ in Eq. (27) is approximated by the average of the samples $\{\boldsymbol{\theta}_n^i\}_{n=1}^N$. These samples are often termed *particles* in the computational statistics literature. One can also estimate, e.g., the mean square error (MSE) of $\hat{\boldsymbol{\theta}}_n$. Specifically,

$$\begin{aligned} MSE_n &= \int \| \boldsymbol{\theta} - \hat{\boldsymbol{\theta}}_n \|^2 \, \mu_n(d\boldsymbol{\theta}) \\ &\approx \int \| \boldsymbol{\theta} - \hat{\boldsymbol{\theta}}_n^N \|^2 \, \mu_n^N(d\boldsymbol{\theta}) = \frac{1}{N}\sum_{i=1}^{N} \| \boldsymbol{\theta}_n^i - \hat{\boldsymbol{\theta}}_n^N \|^2 . \end{aligned} \tag{28}$$

B. Estimation of the likelihood $u_n(\boldsymbol{\theta})$

In Section III A we have laid out a general methodology for the Monte Carlo approximation of the posterior distribution of the unknown parameters, $\mu_n(d\boldsymbol{\theta}) = p(\boldsymbol{\theta}|\boldsymbol{y}_{1:n})d\boldsymbol{\theta}$. The practical applicability of the method, however, depends on the ability to compute estimates of the predictive measure $\xi_{n,\boldsymbol{\theta}}(d\boldsymbol{\theta}) = p(\boldsymbol{x}_n|\boldsymbol{\theta}, \boldsymbol{y}_{1:n-1})d\boldsymbol{\theta}$ and the likelihood $u_n(\boldsymbol{\theta}) = \int p(\boldsymbol{y}_n|\boldsymbol{x}_n, \boldsymbol{\theta})\xi_{n,\boldsymbol{\theta}}(d\boldsymbol{\theta})$.

A conceptually simple way to tackle this problem is to compute Monte Carlo approximations for $\xi_{n,\boldsymbol{\theta}}$ and $u_n(\boldsymbol{\theta})$ as well. To be specific, it is possible to use a standard particle filter [11, 12] with M particles to produce estimates $\xi_{n,\boldsymbol{\theta}}^M(d\boldsymbol{\theta})$ and $u_n^M(\boldsymbol{\theta})$ of $\xi_{n,\boldsymbol{\theta}}(d\boldsymbol{\theta})$ and $u_n(\boldsymbol{\theta})$, respectively, that converge in a proper probabilistic sense when $M \to \infty$ [7, 10]. Depending on the way the particle filter is implemented, the resulting nested filter can reduce

to a version of the SMC^2 algorithm in [8], which is a batch method (i.e., non recursive) whose computational complexity increases with n^2, or yield the nested particle filter (NPF) in [10]. The latter is purely recursive, hence its computational cost increases linearly with n.

In practice, however, both the SMC^2 and the NPF are too costly to be applied to systems where the state dimension d_x is large. As an alternative, $\xi_{n,\boldsymbol{\theta}}(d\boldsymbol{\theta})$ and $u_n(d\boldsymbol{\theta})$ can be estimated using a variety of approximate Gaussian filters, including the extended Kalman filter (EKF) [13], the unscented Kalman filter (UKF) [20], the ensemble Kalman filter (EnKF) [14] or the 3DVAR filter [21].

In the sequel we assume that a Gaussian filtering algorithm is used for the approximation of $\xi_{n,\boldsymbol{\theta}}(d\boldsymbol{\theta})$ and $u_n(d\boldsymbol{\theta})$. Specific schemes that employ the EKF and the EnKF are presented in Section III C (including some numerical approximations to avoid the computation and inversion of large matrices).

Figure 1 provides a schematic representation of the recursive computations needed for the implementation of a nested filter. The procedure can be outlined as follows:

- At time $n-1$:

  Assume that we have the approximation

$$\hat{\xi}_{n-1,\bar{\boldsymbol{\theta}}^i_{n-1}}(d\boldsymbol{x}_n) = \mathcal{N}(\boldsymbol{x}_{n-1}|\hat{\boldsymbol{x}}^{(i)}_{n-1}, \hat{\boldsymbol{P}}^{(i)}_{n-1})d\boldsymbol{x}_n, \tag{29}$$

  where $\hat{\boldsymbol{x}}^{(i)}_{n-1} \approx E[\boldsymbol{x}_{n-1}|\boldsymbol{y}_{1:n-2}, \bar{\boldsymbol{\theta}}^i_{n-1}]$ is the approximate predictive mean of $\boldsymbol{x}_{n-1}$ and $\hat{\boldsymbol{P}}^{(i)}_{n-1} \approx E[(\boldsymbol{x}_n - \hat{\boldsymbol{x}}^{(i)}_{n-1})(\boldsymbol{x}_n - \hat{\boldsymbol{x}}^{(i)}_{n-1})^\top|\boldsymbol{y}_{1:n-2}, \bar{\boldsymbol{\theta}}^i_{n-1}]$ is the approximate predictive covariance matrix, both conditional on $\bar{\boldsymbol{\theta}}^i_{n-1}$.

  Given the observation $\boldsymbol{y}_{n-1}$, we perform an update step to compute the conditional filter

$$\begin{aligned}\hat{\phi}_{n-1,\bar{\boldsymbol{\theta}}^i_{n-1}}(d\boldsymbol{x}_{n-1}) &\propto p(\boldsymbol{y}_{n-1}|\bar{\boldsymbol{\theta}}^i_{n-1}, \boldsymbol{x}_{n-1})\hat{\xi}_{n-1,\bar{\boldsymbol{\theta}}^i_n}(d\boldsymbol{x}_{n-1}) \\ &\propto p(\boldsymbol{x}_{n-1}|\bar{\boldsymbol{\theta}}^i_{n-1}, \boldsymbol{y}_{1:n-1})d\boldsymbol{x}_{n-1}\end{aligned} \tag{30}$$

  (where the second proportionality is approximate) and, after resampling, we obtain the new set of particles $\{\boldsymbol{\theta}^i_{n-1}\}_{i=1}^N$ and

$$\hat{\phi}_{n-1,\boldsymbol{\theta}^i_{n-1}}(d\boldsymbol{x}_{n-1}) \approx p(\boldsymbol{x}_n|\boldsymbol{\theta}^i_{n-1}, \boldsymbol{y}_{1:n-1})d\boldsymbol{x}_{n-1} \tag{31}$$

  Recall that all approximate measures are Gaussian. In particular,

$$\hat{\phi}_{n-1,\boldsymbol{\theta}^i_n}(d\boldsymbol{x}_{n-1}) = \mathcal{N}(\boldsymbol{x}_{n-1}|\bar{\boldsymbol{x}}^i_{n-1}, \bar{\boldsymbol{P}}^i_{n-1})d\boldsymbol{x}_{n-1}, \tag{32}$$

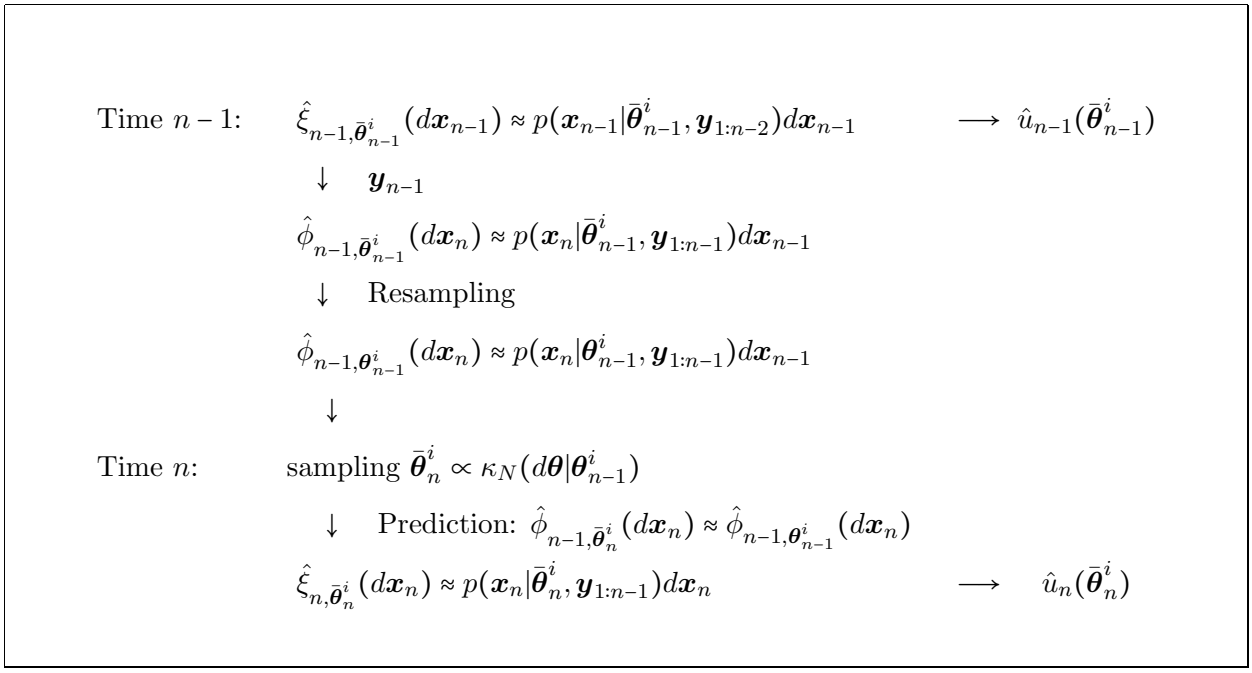


FIG. 1: Schematic representation of the recursive approximation of the posterior measures, $\xi_{n,\boldsymbol{\theta}}$ and $\phi_{n,\boldsymbol{\theta}}$, and the likelihood function $u_n(\boldsymbol{\theta})$.

where

$$\bar{\boldsymbol{x}}^i_{n-1} \approx \mathrm{E}[\boldsymbol{x}_{n-1}|\boldsymbol{\theta}^i_{n-1}, \boldsymbol{y}_{1:n-1}] \quad \text{and} \quad \bar{\boldsymbol{P}}^{(i)}_{n-1} \approx \mathrm{E}[(\boldsymbol{x}_n - \bar{\boldsymbol{x}}^{(i)}_{n-1})(\boldsymbol{x}_n - \bar{\boldsymbol{x}}^{(i)}_{n-1})^\top|\boldsymbol{y}_{1:n-1}, \boldsymbol{\theta}^i_{n-1}]. \quad (33)$$

- At time n:

  A new parameter vector $\bar{\boldsymbol{\theta}}^i_n \sim \kappa_N(d\boldsymbol{\theta}|\boldsymbol{\theta}^i_{n-1})$ is generated. Assuming that the measure $\hat{\phi}_{n-1,\boldsymbol{\theta}}$ is continuous in the parameter $\boldsymbol{\theta}$ (see [10] and [22] for a discussion), and $\bar{\boldsymbol{\theta}}^i_n$ is a (small enough) perturbation of $\boldsymbol{\theta}^i_{n-1}$, it is reasonable to approximate

  $$\hat{\phi}_{n-1,\bar{\boldsymbol{\theta}}^i_n}(d\boldsymbol{x}_n) \approx \hat{\phi}_{n-1,\boldsymbol{\theta}^i_{n-1}}(d\boldsymbol{x}_n). \quad (34)$$

  Then, we can obtain the (Gaussian) predictive measure

  $$\begin{aligned}\hat{\xi}_{n,\bar{\boldsymbol{\theta}}^i_n}(d\boldsymbol{x}_n) &= \int p(\boldsymbol{x}_n|\boldsymbol{x}_{n-1})\hat{\phi}_{n-1,\bar{\boldsymbol{\theta}}^i_n}(d\boldsymbol{x}_{n-1}) \\ &\approx p(\boldsymbol{x}_n|\bar{\boldsymbol{\theta}}^i_n, \boldsymbol{y}_{1:n-1}) \quad (35)\end{aligned}$$

  and, finally, we calculate the approximate likelihood

  $$\hat{u}(\bar{\boldsymbol{\theta}}^i_n) = \int p(\boldsymbol{y}_n|\boldsymbol{x}_n, \bar{\boldsymbol{\theta}}^i_n)\hat{\xi}_{n,\bar{\boldsymbol{\theta}}^i_n}(d\boldsymbol{x}_n) \approx u_n(\bar{\boldsymbol{\theta}}^i_n). \quad (36)$$

The normalized weight of $\bar{\boldsymbol{\theta}}_n^i$ is, therefore,

$$w_n^i = \frac{\hat{u}_n(\bar{\boldsymbol{\theta}}_n^i)}{\sum_{j=1}^N \hat{u}_n(\bar{\boldsymbol{\theta}}_n^j)}. \tag{37}$$

Note that, given the Gaussian approximate filters $\hat{\phi}_{n,\boldsymbol{\theta}_n^i}(d\boldsymbol{x}_n)$ and the weights $w_n^i \propto \hat{u}_n(\bar{\boldsymbol{\theta}}_n^i)$ it is possible to approximate the joint posterior distribution of $\boldsymbol{\theta}$ and $\boldsymbol{x}_n$, namely,

$$\pi_n(d\boldsymbol{\theta} \times d\boldsymbol{x}_n) \approx \pi_n^N(d\boldsymbol{\theta} \times d\boldsymbol{x}_n) = \sum_{i=1}^N w_n^i \delta_{\bar{\boldsymbol{\theta}}_n^i}(d\boldsymbol{\theta}) \hat{\phi}_{n,\bar{\boldsymbol{\theta}}_n^i}(d\boldsymbol{x}_n). \tag{38}$$

C. Nested Hybrid Filters

The recursive approximation scheme of Figure 1 can be implemented via different Gaussian filtering techniques. The resulting methods are termed nested hybrid filters (NHFs) because they combine the Monte Carlo approximation $\mu_n^N(d\boldsymbol{\theta})$ for the posterior distribution of the parameters with Gaussian approximations for the conditional filters $\hat{\phi}_{n,\boldsymbol{\theta}_n^i}$. We describe two specific techniques in detail, the extended Kalman filter (EKF) [13] and the ensemble Kalman filter (EnKF) [14]. The EKF can be applied when the nonlinearities $\bar{\boldsymbol{f}}(\tilde{\boldsymbol{x}}_k, \boldsymbol{\theta})$, in the state equation, and the observation function $\boldsymbol{g}(\tilde{\boldsymbol{x}}_k, \boldsymbol{\theta})$ are differentiable. However, it can be inefficient in high-dimensional state-spaces, which require storing and processing $d_x \times d_x$ covariance matrices with $d_x \gg 1$. To avoid this limitation the EnKF summarises the information of both the state-mean and the covariance matrix into a set of Monte Carlo samples. To be specific, the approximate filter $\hat{\phi}_{n,\boldsymbol{\theta}_n^i}(d\boldsymbol{x}_n)$ in the EnKF is represented by an ensemble of M Monte Carlo particles $\{\boldsymbol{x}_n^{i,j}\}_{j=1}^M$. Both approaches are described next. A numerical comparison is presented in Section VI.

1. *Extended Kalman filter*

Let us assume that the prior pdf of the state is Gaussian with known mean and covariance matrix, namely

$$p(\boldsymbol{x}_0) = \mathcal{N}(\boldsymbol{x}_0 | \bar{\boldsymbol{x}}_0, \bar{\boldsymbol{P}}_0) \tag{39}$$

The noise terms in the dynamic equation (3) and the observation equation (5) are also assumed Gaussian, with zero mean and known covariance matrices,

$$\boldsymbol{v}_k \sim \mathcal{N}(\boldsymbol{v}_k | \boldsymbol{0}, \boldsymbol{Q}) \quad \text{and} \quad \boldsymbol{u}_k \sim \mathcal{N}(\boldsymbol{u}_k | \boldsymbol{0}, \boldsymbol{R}). \tag{40}$$

The EKF algorithm can be used when the functions $\bar{\boldsymbol{f}}$ in Eq. (3) and $\boldsymbol{g}$ in Eq. (5) are either linear or differentiable. In general, we assume both functions are nonlinear and differentiable, with $\boldsymbol{J}_{\bar{\boldsymbol{f}},\boldsymbol{x},\boldsymbol{\theta}}$ and $\boldsymbol{J}_{\boldsymbol{g},\boldsymbol{x},\boldsymbol{\theta}}$ denoting their respective Jacobian matrices evaluated at the point $\boldsymbol{x}$ in the state-space and $\boldsymbol{\theta}$ in the parameter space. Note that $\boldsymbol{J}_{\bar{\boldsymbol{f}},\boldsymbol{x},\boldsymbol{\theta}}$ is $d_x \times d_x$ and $\boldsymbol{J}_{\boldsymbol{g},\boldsymbol{x},\boldsymbol{\theta}}$ is $d_y \times d_y$.

The NHF constructed around a bank of EKFs is outlined in Algorithm 2 below.

Algorithm 2 *NHF via EKF.*

1. *Initialization: draw N i.i.d. particles $\boldsymbol{\theta}_0^i \sim \mu_0(d\boldsymbol{\theta}), i = 1,\ldots,N$. Let $\bar{\boldsymbol{x}}_0^i = \bar{\boldsymbol{x}}_0$ and $\bar{\boldsymbol{P}}_0^i = \boldsymbol{P}_0$ for every i.*

2. *Recursive step: at time n, we have available $\mu_{n-1}^N(d\boldsymbol{\theta}) = \frac{1}{N}\sum_{i=1}^N \delta_{\bar{\boldsymbol{\theta}}_{n-1}^i}(d\boldsymbol{\theta})$ and, for each $i = 1,\ldots,N$, $\hat{\phi}_{n-1,\boldsymbol{\theta}_n^i}(d\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{x}_{n-1}|\bar{\boldsymbol{x}}_{n-1}^i, \bar{\boldsymbol{P}}_{n-1}^i)d\boldsymbol{x}_n$.*

   (a) *Prediction:*

      i. *Draw $\bar{\boldsymbol{\theta}}_n^i \sim \kappa_N(d\boldsymbol{\theta}|\boldsymbol{\theta}_{n-1}^i)$, $i = 1,\ldots,N$.*

      ii. *Let $\breve{\boldsymbol{x}}_0^i = \bar{\boldsymbol{x}}_{n-1}^i$ and $\breve{\boldsymbol{P}}_0^i = \bar{\boldsymbol{P}}_{n-1}^i$. Then, for each $i = 1,\ldots,N$ and $k = 1,\ldots,T$ compute*

$$\breve{\boldsymbol{x}}_k^i = \bar{\boldsymbol{f}}(\breve{\boldsymbol{x}}_{k-1}^i, \bar{\boldsymbol{\theta}}_n^i) \tag{41}$$

$$\breve{\boldsymbol{P}}_k^i = \boldsymbol{J}_{\bar{\boldsymbol{f}},\breve{\boldsymbol{x}}_{k-1}^i,\bar{\boldsymbol{\theta}}_n^i}\breve{\boldsymbol{P}}_{k-1}^i\boldsymbol{J}_{\bar{\boldsymbol{f}},\breve{\boldsymbol{x}}_{k-1}^i,\bar{\boldsymbol{\theta}}_n^i}^\top + \sigma^2\boldsymbol{Q} \tag{42}$$

      iii. *Set $\hat{\xi}_{n,\bar{\boldsymbol{\theta}}_n^i}(d\boldsymbol{x}_n) = \mathcal{N}(\boldsymbol{x}_n|\hat{\boldsymbol{x}}_n^i, \hat{\boldsymbol{P}}_n^i)d\boldsymbol{x}_n$ where $\hat{\boldsymbol{x}}_n^i = \breve{\boldsymbol{x}}_T^i$ and $\hat{\boldsymbol{P}}_n^i = \hat{\boldsymbol{P}}_T^i$.*

   (b) *Update:*

      i. *For $i = 1,\ldots,N$, compute*

$$\boldsymbol{S}_n^i = \boldsymbol{J}_{\boldsymbol{g},\hat{\boldsymbol{x}}_n^i,\bar{\boldsymbol{\theta}}_n^i}\hat{\boldsymbol{P}}_n^i\boldsymbol{J}_{\boldsymbol{g},\hat{\boldsymbol{x}}_n^i,\bar{\boldsymbol{\theta}}_n^i}^\top + \sigma_o^2\boldsymbol{R} \tag{43}$$

$$\boldsymbol{K}_n^i = \hat{\boldsymbol{P}}_n^i\boldsymbol{J}_{\boldsymbol{g},\hat{\boldsymbol{x}}_n^i,\bar{\boldsymbol{\theta}}_n^i}^\top(\boldsymbol{S}_n^i)^{-1} \tag{44}$$

$$\breve{\boldsymbol{x}}_n^i = \hat{\boldsymbol{x}}_n^i + \boldsymbol{K}_n^i(\boldsymbol{y}_n - \boldsymbol{g}(\hat{\boldsymbol{x}}_n^i, \bar{\boldsymbol{\theta}}_n^i)) \tag{45}$$

$$\breve{\boldsymbol{P}}_n^i = (\boldsymbol{I}_{d_x} - \boldsymbol{K}_n^i\boldsymbol{J}_{\boldsymbol{g},\hat{\boldsymbol{x}}_n^i,\bar{\boldsymbol{\theta}}_n^i})\hat{\boldsymbol{P}}_n^i \tag{46}$$

      ii. *Compute $\hat{u}(\bar{\boldsymbol{\theta}}_n^i) = \mathcal{N}(\boldsymbol{y}_n|\boldsymbol{g}(\hat{\boldsymbol{x}}_n^i, \bar{\boldsymbol{\theta}}_n^i), \boldsymbol{S}_n^i)$ and obtain the normalized weights,*

$$w_i = \frac{\hat{u}(\bar{\boldsymbol{\theta}}_n^i)}{\sum_{j=1}^N \hat{u}_n(\bar{\boldsymbol{\theta}}_n^j)}, \quad i = 1,\ldots,N. \tag{47}$$

iii. Set the filter approximation

$$\hat{\phi}_{n,\bar{\boldsymbol{\theta}}_n^i}(d\boldsymbol{x}_n) = \mathcal{N}(\boldsymbol{x}_n|\check{\boldsymbol{x}}_n^i, \check{\boldsymbol{P}}_n^i)d\boldsymbol{x}_n. \tag{48}$$

(c) Resampling: draw indices $j_1, \ldots, j_N$ from the multinomial distribution with probabilities $w_n^1, \ldots, w_n^N$, then set

$$\boldsymbol{\theta}_n^i = \bar{\boldsymbol{\theta}}_n^{j_i}, \quad \bar{\boldsymbol{x}}_n^i = \check{\boldsymbol{x}}_n^{j_i} \quad \textit{and} \quad \bar{\boldsymbol{P}}_n^i = \check{\boldsymbol{P}}_n^{j_i} \tag{49}$$

for $i = 1, \ldots, N$. Hence,

$$\hat{\phi}_{n,\boldsymbol{\theta}_n^i}(d\boldsymbol{x}_n) = \mathcal{N}(\boldsymbol{x}_n|\bar{\boldsymbol{x}}_n^i, \bar{\boldsymbol{P}}_n^i)d\boldsymbol{x}_n \quad \textit{and} \quad \mu_n^N(d\boldsymbol{\theta}) = \frac{1}{N}\sum_{i=1}^{N}\delta_{\boldsymbol{\theta}_n^i}(d\boldsymbol{\theta}).$$

The computationally most expensive steps in Algorithm 2 are the inversion of the observation covariance matrix $\boldsymbol{S}_n^i$ in step 2(b)i and the computation of the predictive state covariance matrices $\check{\boldsymbol{P}}_k^i$ in step 2(a)ii. The latter involves $\mathcal{O}(d_x^3)$ operations while the computation of $(\boldsymbol{S}_n^i)^{-1}$ is $\mathcal{O}(d_y^3)$. Therefore, these steps quickly become intractable when d_x and/or d_y increase beyond moderate values.

To mitigate this limitation we have implemented both the inversion of $\boldsymbol{S}_n^i$ and the computation of $\check{\boldsymbol{P}}_k^i$ in an approximate manner. The approximation schemes are both based on block decompositions of the target matrices in such a way that the computational effort can be controlled a priori. Complete details are provided in Appendices A and B. The block-approximate calculation of $\check{\boldsymbol{P}}_k^i$ depends on the form of the Jacobian matrix $\boldsymbol{J}_{\bar{\boldsymbol{f}},\boldsymbol{x},\boldsymbol{\theta}}$ and the procedure in Appendix B is described for the Jacobian resulting from the Lorenz 96 model presented in Section V. However the method can be readily extended to different examples.

2. Ensemble Kalman filter

The EKF method requires to store and process $d_x \times d_x$ covariance matrices. As an alternative, the EnKF algorithm summarises the information that is carried by the $\bar{\boldsymbol{x}}_n^i$'s and the $\bar{\boldsymbol{P}}_n^i$'s into samples, i.e., we represent the filters as ensembles of M realizations $\{\boldsymbol{x}_n^{i,j}\}_{j=1}^M$ each.

Each ensemble can be stored in a $d_x \times M$ matrix $\boldsymbol{X}_n^i = [\boldsymbol{x}_n^{i,1}, \boldsymbol{x}_n^{i,2}, \ldots, \boldsymbol{x}_n^{i,M}]$. The i-th mean

and the i-th covariance matrix can be computed as

$$\bar{\boldsymbol{x}}_n^i = \frac{1}{M}\boldsymbol{X}_n^i \mathbf{1} \tag{50}$$

$$\bar{\boldsymbol{P}}_n^i = \frac{1}{M}\tilde{\boldsymbol{X}}_n^i (\tilde{\boldsymbol{X}}_n^i)^\top \tag{51}$$

respectively, where $\mathbf{1} = [1, \ldots, 1]^\top$ is an M-dimensional column vector and $\tilde{\boldsymbol{X}}_n^i = \boldsymbol{X}_n^i - \bar{\boldsymbol{x}}_n^i \mathbf{1}^\top$ is an ensemble of deviations from $\bar{\boldsymbol{x}}_n^i$. We hence write $\mathcal{N}(\boldsymbol{x}_n|\boldsymbol{X}_n^i)$ as a shorthand for the pdf $\mathcal{N}(\boldsymbol{x}_n|\bar{\boldsymbol{x}}_n^i, \bar{\boldsymbol{P}}_n^i)$.

The NHF constructed around a bank of EnKFs is outlined in Algorithm 3 below.

Algorithm 3 *NHF via EnKF.*

1. *Initialization: draw N i.i.d. particles $\boldsymbol{\theta}_0^i \sim \mu_0(d\boldsymbol{\theta})$ and $\{\bar{\boldsymbol{x}}_0^{i,j}\} \sim p(\boldsymbol{x}_0), i = 1, \ldots, N$, $j = 1, \ldots, M$. Let $\boldsymbol{X}_0^i = [\boldsymbol{x}_0^{i,1}, \ldots, \boldsymbol{x}_0^{i,M}]$, $i = 1, \ldots, N$.*

2. *Recursive step: at time $n-1$, we have obtained $\mu_{n-1}^N(d\boldsymbol{\theta}) = \frac{1}{N}\sum_{i=1}^N \delta_{\bar{\boldsymbol{\theta}}_{n-1}^i}(d\boldsymbol{\theta})$ and, for each $i = 1, \ldots, N$, $\hat{\phi}_{n-1,\boldsymbol{\theta}_n^i}(d\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{x}_{n-1}|\bar{\boldsymbol{X}}_{n-1}^i)d\boldsymbol{x}_n$.*

   (a) *Prediction:*

      i. *Draw $\bar{\boldsymbol{\theta}}_n^i \sim \kappa_N(d\boldsymbol{\theta}|\boldsymbol{\theta}_{n-1}^i)$, $i = 1, \ldots, N$.*

      ii. *Let $\breve{\boldsymbol{X}}_0^i = \bar{\boldsymbol{X}}_{n-1}^i$. Then, for each $i = 1, \ldots, N$ and $k = 1, \ldots, T$ compute*

$$\breve{\boldsymbol{X}}_k^i = \bar{\boldsymbol{f}}(\breve{\boldsymbol{X}}_{k-1}^i, \bar{\boldsymbol{\theta}}_n^i) + \boldsymbol{V}_k^i \tag{52}$$

      where $\boldsymbol{V}_k^i = [\boldsymbol{v}_k^{i,1}, \ldots, \boldsymbol{v}_k^{i,M}]$, $i = 1, \ldots, N$, is a $d_x \times M$ matrix of Gaussian perturbations.

      iii. *Set $\hat{\xi}_{n,\bar{\boldsymbol{\theta}}_n^i}(d\boldsymbol{x}_n) = \mathcal{N}(\boldsymbol{x}_n|\hat{\boldsymbol{X}}_n^i)d\boldsymbol{x}_n$ where $\hat{\boldsymbol{X}}_n^i = \breve{\boldsymbol{X}}_T^i$.*

   (b) *Update:*

      i. *For $i = 1, \ldots, N$, compute*

$$\bar{\boldsymbol{M}}_n^i = \frac{1}{M}\tilde{\boldsymbol{X}}_n^i (\tilde{\boldsymbol{Z}}_n^i)^\top \tag{53}$$

$$\bar{\boldsymbol{S}}_n^i = \frac{1}{M}\tilde{\boldsymbol{Z}}_n^i (\tilde{\boldsymbol{Z}}_n^i)^\top + \boldsymbol{R} \tag{54}$$

$$\bar{\boldsymbol{K}}_n^i = \bar{\boldsymbol{M}}_n^i (\bar{\boldsymbol{S}}_n^i)^{-1} \tag{55}$$

$$\breve{\boldsymbol{X}}_n^i = \hat{\boldsymbol{X}}_n^i + \bar{\boldsymbol{K}}_n^i (\boldsymbol{y}_n \mathbf{1}^\top - \bar{\boldsymbol{Y}}_n^i) \tag{56}$$

where $\boldsymbol{R} = \sigma_o^2 \boldsymbol{I}_{d_y}$ *is the measurement noise covariance,* $\bar{\boldsymbol{y}}_n = \frac{1}{M}\bar{\boldsymbol{Y}}^i_n \mathbf{1}$ *and* $\bar{\boldsymbol{x}}^i_n = \frac{1}{M}\hat{\boldsymbol{X}}^i_n \mathbf{1}$*, with* $\bar{\boldsymbol{Y}}^i_n = \boldsymbol{g}(\hat{\boldsymbol{X}}^i_n, \boldsymbol{\theta}) + \boldsymbol{U}^i_n$ *and* $\boldsymbol{U}^i_n = [\boldsymbol{u}^1_n, \ldots, \boldsymbol{u}^M_n]$ *a matrix of Gaussian perturbations.* $\tilde{\boldsymbol{X}}^i_n$ *and* $\tilde{\boldsymbol{Z}}^i_n$ *are calculated as*

$$\tilde{\boldsymbol{X}}^i_n = \hat{\boldsymbol{X}}^i_n - \bar{\boldsymbol{x}}^i_n \mathbf{1}^\top \tag{57}$$

$$\tilde{\boldsymbol{Z}}^i_n = \frac{1}{M}\boldsymbol{g}(\hat{\boldsymbol{X}}^i_n, \boldsymbol{\theta}) - \bar{\boldsymbol{y}}^i_n \mathbf{1}^\top \tag{58}$$

ii. Compute $\hat{u}(\bar{\boldsymbol{\theta}}^i_n) = \mathcal{N}(\boldsymbol{y}_n | \boldsymbol{g}(\bar{\boldsymbol{x}}^i_n, \bar{\boldsymbol{\theta}}^i_n), \bar{\boldsymbol{S}}^i_n)$ *and obtain the normalized weights,*

$$w_i = \frac{\hat{u}(\bar{\boldsymbol{\theta}}^i_n)}{\sum_{j=1}^N \hat{u}_n(\bar{\boldsymbol{\theta}}^j_n)}, \quad i = 1, \ldots, N. \tag{59}$$

iii. Set the filter approximation

$$\hat{\phi}_{n,\bar{\boldsymbol{\theta}}^i_n}(d\boldsymbol{x}_n) = \mathcal{N}(\boldsymbol{x}_n | \check{\boldsymbol{X}}^i_n) d\boldsymbol{x}_n. \tag{60}$$

(c) Resampling: draw indices $j_1, \ldots, j_N$ *from the multinomial distribution with probabilities* $w^1_n, \ldots, w^N_n$*, then set*

$$\boldsymbol{\theta}^i_n = \bar{\boldsymbol{\theta}}^{j_i}_n, \quad \textit{and} \quad \bar{\boldsymbol{X}}^i_n = \check{\boldsymbol{X}}^{j_i}_n \tag{61}$$

for $i = 1, \ldots, N$*. Hence*

$$\hat{\phi}_{n,\boldsymbol{\theta}^i_n}(d\boldsymbol{x}_n) = \mathcal{N}(\boldsymbol{x}_n | \bar{\boldsymbol{X}}^i_n) d\boldsymbol{x}_n \quad \textit{and} \quad \mu^N_n(d\boldsymbol{\theta}) = \frac{1}{N}\sum_{i=1}^N \delta_{\boldsymbol{\theta}^i_n}(d\boldsymbol{\theta}).$$

As in Algorithm 2, a computationally expensive step is the inversion of the observation covariance matrix $\bar{\boldsymbol{S}}^i_n$ in step 2(b)i and we use the approximation described in Appendix A to alleviate the cost. However, in Algorithm 3 we avoid the computation of the predictive state covariance matrices.

IV. CONVERGENCE ANALYSIS

A. Preliminaries and notation

The nested filtering scheme of Section III A admits many implementations depending on how we choose to approximate the conditional measures $\xi_{n,\boldsymbol{\theta}}(d\boldsymbol{\theta})$ and $\phi_{n,\boldsymbol{\theta}}(d\boldsymbol{\theta})$ which, in turn, are needed to estimate the likelihood function $u_n(\boldsymbol{\theta})$ and, therefore, the importance weights

$$w^i_n \propto \hat{u}(\bar{\boldsymbol{\theta}}^i_n) \approx u(\bar{\boldsymbol{\theta}}^i_n), \quad i = 1, \ldots, N. \tag{62}$$

For each choice of approximation method the estimate $\hat{u}_n(\boldsymbol{\theta})$ may behave differently and yield different convergence properties. Here we assume that $\hat{u}_n(\boldsymbol{\theta})$ is a random variable with finite mean $\bar{u}_n(\boldsymbol{\theta}) = \mathrm{E}[\hat{u}_n(\boldsymbol{\theta})] < \infty$ and finite moments up to some prescribed order $p \geq 1$. Specifically, we make following assumption.

A. 1 *The estimator $\hat{u}_n(\boldsymbol{\theta})$ is random and can be written as*

$$\hat{u}_n(\boldsymbol{\theta}) = \bar{u}_n(\boldsymbol{\theta}) + m_n(\boldsymbol{\theta}), \tag{63}$$

where $m_n(\boldsymbol{\theta})$ is a zero-mean random vector satisfying $E[m_n(\boldsymbol{\theta})^p] \leq \sigma^p < \infty$ for some prescribed $p \geq 1$. Furthermore, the mean $\bar{u}_n(\boldsymbol{\theta}) = E[\hat{u}_n(\boldsymbol{\theta})]$ has the form

$$\bar{u}_n(\boldsymbol{\theta}) = u_n(\boldsymbol{\theta}) + b_n(\boldsymbol{\theta}), \tag{64}$$

where $b_n(\boldsymbol{\theta})$ is a deterministic and bounded bias function.

In the sequel we use $D \subseteq \mathbb{R}^{d_\theta}$ to denote the support set of the parameter vector $\boldsymbol{\theta}$ and the notation $\|a\|_\infty := \sup_{\boldsymbol{\theta} \in D} |a(\boldsymbol{\theta})|$ to indicate the absolute supremum of a real function $a : D \to \mathbb{R}$. The set of such functions is denoted $B(D)$, i.e., $B(D) := \{(a : D \to \mathbb{R}) : \|a\|_\infty < \infty\}$. For our analysis we assume that $u_n \in B(D)$ and, since we have also assumed the bias function b_n to be bounded, we have $\bar{u}_n \in B(D)$, i.e., $\|\bar{u}_n\|_\infty < \infty$.

We shall prove that, because of the bias $b_n(\boldsymbol{\theta})$, the approximation μ_n^N converges to the perturbed probability measure $\bar{\mu}_n$ induced by the mean function $\bar{u}_n$, instead of the true posterior probability measure μ_n induced by model (8)-(10) (and, therefore, by the true likelihood function u_n).

To be specific, the sequence of posterior measures μ_n, $n \geq 1$, can be constructed recursively, starting from a prior $\mu_0(d\boldsymbol{\theta})$, by means of the projective product operation [23]

$$\mu_n = u_n \star \mu_{n-1}.$$

When $u(\boldsymbol{\theta})$ is a positive and bounded function and α is a probability measure, the new measure $u \star \alpha$ is defined in terms of its integrals. in particular, if $a \in B(D)$ then

$$\int a(\boldsymbol{\theta})(u \star \alpha) d\boldsymbol{\theta} := \frac{\int a(\boldsymbol{\theta}) u(\boldsymbol{\theta}) \alpha(d\boldsymbol{\theta})}{\int u(\boldsymbol{\theta}) \alpha(d\boldsymbol{\theta})}.$$

For conciseness, hereafter we use the shorthand

$$(a, \alpha) := \int a(\boldsymbol{\theta}) \alpha(d\boldsymbol{\theta})$$

for the integral of a function $a(\boldsymbol{\theta})$ w.r.t. a measure α. With this notation, we can write

$$(a, \mu_n) = (a, u_n \star \mu_{n-1}) = \frac{(au_n, \mu_{n-1})}{(u_n, \mu_{n-1})}. \tag{65}$$

If, instead of the true likelihood u_n, we use the *biased* function $\bar{u}_n = u_n + b_n$ to update the posterior probability measure associated to the parameter vector $\boldsymbol{\theta}$ at each time n then we obtain the new sequence of measures

$$\bar{\mu}_0 = \mu_0, \quad \bar{\mu}_n = \bar{u}_n \star \bar{\mu}_{n-1}, \quad n = 1, 2, ...,$$

where, according to the definition of the projective product,

$$(a, \bar{\mu}_n) = \frac{(a\bar{u}_n, \mu_{n-1})}{(\bar{u}_n, \bar{\mu}_{n-1})}$$

for any integrable function $a(\boldsymbol{\theta})$. Note that the two sequence, μ_n and $\bar{\mu}_n$, start from the same prior μ_0. Obviously, we recover the original sequence, i.e, $\bar{\mu}_n \to \mu_n$, when the bias vanishes, $b_n \to \mathbf{0}$.

In this section we prove that the approximation μ_n^N generated by a generic nested filter that satisfies assumption A.1 converges to $\bar{\mu}_n$ in L_p, for each $n = 1, 2, ...,$ under regularity conditions. We split the analysis of the nested filter in three steps: jittering, weight computation and resampling. The approximation μ_{n-1}^N of $\bar{\mu}_{n-1}$ is available at the beginning of the n-th time step. After the jittering we obtain a new approximation,

$$\check{\mu}_{n-1}^N = \frac{1}{N} \sum_{i=1}^{N} \delta_{\bar{\boldsymbol{\theta}}_n^i}, \tag{66}$$

that can be proved to converge to μ_{n-1} using an auxiliary result from [10]. After the computation of the weights, the measure

$$\tilde{\mu}_n^N = \sum_{i=1}^{N} w_n^i \delta_{\bar{\boldsymbol{\theta}}_n^i} \tag{67}$$

is obtained and its convergence towards $\bar{\mu}_n$ must be established. Finally, after the resampling step, a standard piece of analysis proves the convergence of

$$\mu_n^N = \frac{1}{N} \sum_{i=1}^{N} \delta_{\boldsymbol{\theta}_n^i} \tag{68}$$

to $\bar{\mu}_n$.

B. Jittering

In the jittering step, a new cloud of particles $\{\bar{\boldsymbol{\theta}}_n^i\}_{i=1}^N$ is generated by propagating the existing samples across the kernels $\kappa_N(d\boldsymbol{\theta}|\boldsymbol{\theta}_{n-1}^i)$, $i = 1, \ldots, N$. This step has been analyzed in [10] in the context of the NPF. Several types of kernels can be used. In general, there is a trade-off between the number of particles that are changed using this kernel and the "amount of perturbation" that can be applied to each particle. For this reason, we let the jittering kernel κ_N depend explicitly on N. For our analysis, we make the assumption A.2 below.

A. 2 *The kernel κ_N used in the jittering step satisfies the inequality*

$$\sup_{\boldsymbol{\theta}' \in D} \int |h(\boldsymbol{\theta}) - h(\boldsymbol{\theta}')| \kappa_N(d\boldsymbol{\theta}|\boldsymbol{\theta}') \leq \frac{c_\kappa \|h\|_\infty}{\sqrt{N}} \tag{69}$$

for any $h \in B(D)$ and some constant $c_\kappa < \infty$ independent of N.

A simple kernel that satisfies A.2 is [10]

$$\kappa_N(d\boldsymbol{\theta}|\boldsymbol{\theta}') = (1 - \epsilon_N)\delta_{\boldsymbol{\theta}'}(d\boldsymbol{\theta}) + \epsilon_N \kappa(d\boldsymbol{\theta}|\boldsymbol{\theta}'),$$

where $0 < \epsilon_N \leq \frac{1}{\sqrt{N}}$ and $\kappa(d\boldsymbol{\theta}|\boldsymbol{\theta}')$ is an arbitrary Markov kernel with mean $\boldsymbol{\theta}'$ and finite variance, for example $\kappa(d\boldsymbol{\theta}|\boldsymbol{\theta}') = \mathcal{N}(\boldsymbol{\theta}|\boldsymbol{\theta}', \tilde{\sigma}^2 \boldsymbol{I})$, where $\tilde{\sigma}^2 < \infty$ and $\boldsymbol{I}$ is the identity matrix. Intuitively, this kind of kernel changes each particle with probability ϵ_N and leaves it unmodified with probability $1 - \epsilon_N$.

The convergence results to be given in this section are presented in terms of upper bounds for the L_p norms of the approximation errors. For a random vector $\boldsymbol{z}$, its L_p norm is $\|z\|_p = \mathrm{E}\left[|\boldsymbol{z}|^p\right]^{\frac{1}{p}}$. The approximate measures generated by the nested filter, e.g., μ_n^N, are measured-valued random variables. Therefore, integrals of the form (h, μ_n^N), for some $h \in B(D)$, are real random variables and it makes sense to evaluate the L_p norm of the random error $(h, \mu_n^N) - (h, \bar{\mu}_n)$. We start with the approximation $\check{\mu}_{n-1}^N$ produced after the jittering step at time n.

Lemma 1 *Let the sequence of observations $y_{1:n}$ be arbitrary but fixed. If $h \in B(D)$, A.2 holds and*

$$\|(h, \mu_{n-1}^N) - (h, \bar{\mu}_{n-1})\|_p \leq \frac{c_{n-1} \|h\|_\infty}{\sqrt{N}} \tag{70}$$

for some $p \geq 1$ and a constant $c_{n-1} < \infty$ independent of N, then

$$\|(h, \check{\mu}_{n-1}^N) - (h, \bar{\mu}_{n-1})\|_p \leq \frac{c_{1,n}\|h\|_\infty}{\sqrt{N}}, \tag{71}$$

where the constant $c_{1,n} < \infty$ is also independent of N.

Proof: The proof of this Lemma is identical to the proof of [10, Lemma 3]. □

C. Computation of the weights

In order to analyze the errors at the weight computation step we need to incorporate some regularity assumptions on the likelihoods $\bar{u}_n(\boldsymbol{\theta})$, $n \geq 1$.

A. 3 *Given a fixed sequence of observations $y_{1:n}$, the family of functions $\{\bar{u}_n(\boldsymbol{\theta}), \boldsymbol{\theta} \in D\}$ satisfies the following inequalities for each $n = 1, 2, ...$:*

1. $\|\bar{u}_n\| < \infty$, and

2. $\bar{u}_n(\boldsymbol{\theta}) > 0$ for any $\boldsymbol{\theta} \in D$.

Let us note that if we assume $\|u_n\|_\infty < \infty$ then A.3.1 follows from assumption A.1. Similarly, if we choose D such that $u_n(\boldsymbol{\theta}) > 0$ for all $\boldsymbol{\theta} \in D$ then A.3.2 is a rather natural assumption, since $\hat{u}(\boldsymbol{\theta})$ is an estimator of a positive magnitude.

An upper bound for the error in the weight computation step is established next.

Lemma 2 *Let the sequence of observations $y_{1:n}$ be arbitrary but fixed, choose any $h \in B(D)$ and some $p \geq 1$. If assumptions A.1 and A.3 hold, and*

$$\|(h, \check{\mu}_{n-1}^N) - (h, \bar{\mu}_{n-1})\|_p \leq \frac{c_{1,n}\|h\|_\infty}{\sqrt{N}} \tag{72}$$

for some constant $c_{1,n} < \infty$ independent of N, then

$$\|(h, \tilde{\mu}_n^N) - (h, \bar{\mu}_n)\|_p \leq \frac{c_{2,n}\|h\|_\infty}{\sqrt{N}}, \tag{73}$$

where the constant $c_{2,n} < \infty$ is independent of N.

Proof: We address the characterization of the weights and, therefore, of the approximate measure $\tilde{\mu}_n^N = \sum_{i=1}^N w_n^i \delta_{\bar{\theta}_n^i}$. From the definition of the projective product in (65), the integrals of h w.r.t. $\bar{\mu}_n$ and $\tilde{\mu}_n^N$ can be written as

$$(h, \bar{\mu}_n) = \frac{(\bar{u}_n h, \bar{\mu}_{n-1})}{(\bar{u}_n, \bar{\mu}_{n-1})}, \quad \text{and} \quad (h, \tilde{\mu}_n^N) = \frac{(\hat{u}_n h, \check{\mu}_{n-1}^N)}{(\hat{u}_n, \check{\mu}_{n-1}^N)}, \tag{74}$$

respectively. From (74) one can write the difference $(h, \tilde{\mu}_n^N) - (h, \bar{\mu}_n)$ as

$$(h, \tilde{\mu}_n^N) - (h, \bar{\mu}_n) = \frac{(h\hat{u}_n, \check{\mu}_{n-1}^N) - (h\bar{u}_n, \bar{\mu}_{n-1})}{(\bar{u}_n, \bar{\mu}_{n-1})} + (h, \tilde{\mu}_n^N)\frac{(\bar{u}_n, \bar{\mu}_{n-1}) - (\hat{u}_n, \check{\mu}_{n-1}^N)}{(\bar{u}_n, \bar{\mu}_{n-1})},$$

which readily yields the inequality

$$|(h, \tilde{\mu}_n^N) - (h, \bar{\mu}_{n-1})| \leq \frac{|(h\hat{u}_n, \check{\mu}_{n-1}^N) - (h\bar{u}_n, \bar{\mu}_{n-1})|}{(\bar{u}_n, \bar{\mu}_{n-1})} + \frac{\|h\|_\infty |(\hat{u}_n, \check{\mu}_{n-1}^N) - (\bar{u}_n, \bar{\mu}_{n-1})|}{(\bar{u}_n, \bar{\mu}_{n-1})} \tag{75}$$

by simply noting that $|(h, \tilde{\mu}_n^N)| \leq \|h\|_\infty$, since $\tilde{\mu}_n^N$ is a probability measure. From (75) and Minkowski's inequality we easily obtain the bound

$$\begin{aligned} \|(h, \tilde{\mu}_n^N) - (h, \bar{\mu}_{n-1})\|_p \leq \frac{1}{(\bar{u}_n, \bar{\mu}_{n-1})} \big[&\|h\|_\infty \|(\hat{u}_n, \check{\mu}_{n-1}^N) - (\bar{u}_n, \bar{\mu}_{n-1})\|_p \\ &+ \|(h\hat{u}_n, \check{\mu}_{n-1}^N) - (h\bar{u}_n, \bar{\mu}_{n-1})\|_p, \big] \end{aligned} \tag{76}$$

where $(\bar{u}_n, \bar{\mu}_{n-1}) > 0$ from assumption A.3.2.

We need to find upper bounds for the two terms on the right hand side of (76). Consider first the term $\|(h\hat{u}_n, \check{\mu}_{n-1}^N) - (h\bar{u}_n, \bar{\mu}_{n-1})\|_p$. A simple triangle inequality yields

$$\|(h\hat{u}_n, \check{\mu}_{n-1}^N) - (h\bar{u}_n, \bar{\mu}_{n-1})\|_p \leq \|(h\hat{u}_n, \check{\mu}_{n-1}^N) - (h\bar{u}_n, \check{\mu}_{n-1}^N)\|_p + \|(h\bar{u}_n, \check{\mu}_{n-1}^N) - (h\bar{u}_t, \bar{\mu}_{n-1})\|_p. \tag{77}$$

On one hand, since $\sup_{\boldsymbol{\theta} \in D} |h(\boldsymbol{\theta})\bar{u}_n(\boldsymbol{\theta})| \leq \|h\|_\infty \|\bar{u}_n\|_\infty < \infty$ (see A.3.1), it follows from the assumption in Eq. (72) that

$$\|(h\bar{u}_n, \check{\mu}_{n-1}^N) - (h\bar{u}_n, \bar{\mu}_{n-1})\|_p \leq \frac{c_{1,n}\|h\|_\infty \|\bar{u}_n\|_\infty}{\sqrt{N}}, \tag{78}$$

where $c_{1,n} < \infty$ is a constant independent of N.

On the other hand, we may note that

$$|(h\hat{u}_n, \check{\mu}_{n-1}^N) - (h\bar{u}_n, \check{\mu}_{n-1}^N)|^p = \left| \frac{1}{N} \sum_{i=1}^N \Big(h(\bar{\boldsymbol{\theta}}_n^i)\hat{u}_n(\bar{\boldsymbol{\theta}}_n^i) - h(\bar{\boldsymbol{\theta}}_n^i)\bar{u}_n(\bar{\boldsymbol{\theta}}_n^i) \Big) \right|^p. \tag{79}$$

Let $\mathcal{G}_n$ be the σ-algebra generated by the random particles $\{\bar{\boldsymbol{\theta}}_{1:n-1}^i, \boldsymbol{\theta}_{0:n-1}^i\}_{1 \leq i \leq N}$ and assume that p is even. Then we can apply conditional expectations on both sides of (79) to obtain

$$\mathrm{E}\Big[\big| (h\hat{u}_n, \check{\mu}_{n-1}^N) - (h\bar{u}_n, \check{\mu}_{n-1}^N) \big|^p \Big| \mathcal{G}_n \Big] = \mathrm{E}\left[\left(\frac{1}{N} \sum_{i=1}^N h(\bar{\boldsymbol{\theta}}_n^i) m_n(\bar{\boldsymbol{\theta}}_n^i) \right)^p \Big| \mathcal{G}_n \right]$$

where the expression on the right hand side has been simplified by using the assumption $\hat{u}_n(\boldsymbol{\theta}) = \bar{u}_n(\boldsymbol{\theta}) + m_n(\boldsymbol{\theta})$ in A.1. Also from assumption A.1, the random variables $m_n(\bar{\boldsymbol{\theta}}_n^i)$ are conditionally independent (given $\mathcal{G}_n$), have zero mean and finite moments of order p, $E[m_n(\bar{\boldsymbol{\theta}}_n^i)^p] \leq \sigma^p < \infty$. If we realise that

$$\mathrm{E}[h(\bar{\boldsymbol{\theta}}_n^i)m_n(\bar{\boldsymbol{\theta}}_n^i)|\mathcal{G}_n] = h(\bar{\boldsymbol{\theta}}_n^i)\mathrm{E}[m_n(\bar{\boldsymbol{\theta}}_n^i)|\mathcal{G}_n] = 0$$

and bear in mind the conditional independence of the $m_n(\bar{\boldsymbol{\theta}}_n^i)$'s, then it is an exercise in combinatorics to show that the number of non-zero terms in

$$\mathrm{E}\left[\left(\frac{1}{N}\sum_{i=1}^{N} h(\bar{\boldsymbol{\theta}}_n^i)m_n(\bar{\boldsymbol{\theta}}_n^i)\right)^p \middle| \mathcal{G}_n\right] = \sum_{i_1}\ldots\sum_{i_p}\mathrm{E}\left[h(\bar{\boldsymbol{\theta}}_n^{i_1})m_n(\bar{\boldsymbol{\theta}}_n^{i_1})\ldots h(\bar{\boldsymbol{\theta}}_n^{i_p})m_n(\bar{\boldsymbol{\theta}}_n^{i_p})\middle|\mathcal{G}_n\right]$$

is at most $\tilde{c}^p N^{\frac{p}{2}}$, for some constant $\tilde{c}^p < \infty$ independent of N and h. Since each of the non-zero terms is upper bounded by $\mathrm{E}\left[(h(\bar{\boldsymbol{\theta}}_n^i)m_n(\bar{\boldsymbol{\theta}}_n^i))^p\middle|\mathcal{G}_n\right] \leq \|h\|_\infty^p\sigma^p < \infty$ (using A.1 again), then it follows that

$$\mathrm{E}\left[\left|(h\hat{u}_n, \check{\mu}_{n-1}^N) - (h\bar{u}_n, \check{\mu}_{n-1}^N)\right|^p\right] = \mathrm{E}\left[\left(\frac{1}{N}\sum_{i=1}^{N} h(\bar{\boldsymbol{\theta}}_n^i)m_n(\bar{\boldsymbol{\theta}}_n^i)\right)^p \middle| \mathcal{G}_n\right] \leq \frac{\tilde{c}^p\sigma^p\|h\|_\infty^p}{N^{\frac{p}{2}}} \tag{80}$$

for even p. Given (80), it is straightforward to show that the same result holds for every $p \geq 1$ using Jensen's inequality. Finally, since the bound on the right hand side of (80) is independent of $\mathcal{G}_n$, we can take expectations on both sides of the inequality and obtain that

$$\|(h\hat{u}_n, \check{\mu}_{n-1}^N) - (h\bar{u}_n, \check{\mu}_{n-1}^N)\|_p \leq \frac{\tilde{c}\sigma\|h\|_\infty}{\sqrt{N}}. \tag{81}$$

Substituting (81) and (78) into (77) yields

$$\|(h\hat{u}_n, \check{\mu}_{n-1}^N) - (h\bar{u}_n, \bar{\mu}_{n-1})\|_p \leq \frac{c_n'\|h\|_\infty^p\|\bar{u}_n\|_\infty}{\sqrt{N}}, \tag{82}$$

where $c_n' = c_{1,n} + \tilde{c}\sigma$ is a constant independent of N.

The same argument leading to the bound in (82) can be repeated, step by step, on the norm $\|(\hat{u}_n, \check{\mu}_{n-1}^N) - (\bar{u}_n, \bar{\mu}_{n-1})\|_p$ (simply taking $h(\boldsymbol{\theta}) = 1$), to arrive at

$$\|(\hat{u}_n, \check{\mu}_{n-1}^N) - (\bar{u}_n, \bar{\mu}_{n-1})\|_p \leq \frac{c_n'\|\bar{u}_n\|_\infty}{\sqrt{N}}. \tag{83}$$

To complete the proof, we substitute (82) and (83) back into (76) and so obtain

$$\|(h, \tilde{\mu}_n^N) - (h, \bar{\mu}_{n-1})\|_p \leq \frac{c_{2,n}\|h\|_\infty}{\sqrt{N}},$$

where the constant $c_{2,n} = \|\bar{u}_n\|_\infty (2c_n')/(\bar{u}_t, \bar{\mu}_{t-1}) < \infty$ is independent of N. □

D. Resampling

The quantification of the error in the resampling step of the nested filter is a standard piece of analysis, well known from the particle filtering literature (see, e.g., [23]). We can state the following result.

Lemma 3 *Let the sequence of observations $y_{1:n}$ be arbitrary but fixed. If $h \in B(D)$ and*

$$\|(h, \tilde{\mu}_n^N) - (h, \bar{\mu}_n)\|_p \leq \frac{c_{2,n}\|h\|_\infty}{\sqrt{N}} \tag{84}$$

for a constant $c_{2,n} < \infty$ independent of N, then

$$\|(h, \mu_n^N) - (h, \bar{\mu}_n)\|_p \leq \frac{c_{3,n}\|h\|_\infty}{\sqrt{N}},$$

where the constant $c_{3,n} < \infty$ is independent of N as well.

Proof: See, e.g., the proof of [24, Lemma 1]. □

E. Convergence theorem

Finally, we can put Lemmas 1, 2 and 3 together in order to prove the following statement on the convergence of a generic nested filter.

Theorem 1 *Let the sequence of observations $y_{1:n_o}$ be arbitrary but fixed, with $n_o < \infty$, and choose an arbitrary function $h \in B(D)$. If the assumptions A.1-A.3 hold, then*

$$\|(h, \mu_n^N) - (h, \bar{\mu}_n)\|_p \leq \frac{c_n\|h\|_\infty}{\sqrt{N}}, \quad \text{for } n = 0, 1, \ldots, n_o, \tag{85}$$

where $\{c_n\}_{0\leq n\leq n_o}$ is a sequence of constants independent of N.

Proof: We prove that (85) holds by induction in n. At time $n = 0$, we draw $\boldsymbol{\theta}_0^i$, $i = 1, \ldots, N$, independently from the prior μ_0 ant it is straightforward to show that $\|(h, \mu_0^N) - (h, \bar{\mu}_0)\|_p \leq \frac{c_0\|h\|_\infty}{\sqrt{N}}$, where c_0 does not depend on N (recall that $\mu_0 = \bar{\mu}_0$).

Assume that, at time $n-1$,

$$\|(h, \mu_{n-1}^N) - (h, \bar{\mu}_{n-1})\|_p \leq \frac{c_{n-1}\|h\|_\infty}{\sqrt{N}}$$

where $c_{n-1} < \infty$ is independent of N. Then, we simply apply Lemmas 1, 2 and 3 in sequence to obtain

$$\|(h, \mu_n^N) - (h, \bar{\mu}_n)\|_p \leq \frac{c_n\|h\|_\infty}{\sqrt{N}}$$

for a constant $c_n = c_{3,n} < \infty$ independent of N. □

V. A STOCHASTIC LORENZ 96 MODEL

In order to assess the proposed methods numerically, we have applied them to a stochastic, discrete-time version of the two-scale Lorenz 96 model [17]. The latter is a deterministic system of nonlinear differential equations that displays some key features of atmosphere dynamics (including chaotic behavior) in a relatively simple model of arbitrary dimension (the number d_x of dynamic variables can be scaled as needed). The model consists of two sets of dynamic variables, $\boldsymbol{x}(t)$ and $\boldsymbol{z}(t)$. The system of differential equations takes the form

$$\begin{aligned} \dot{\boldsymbol{x}}(t) &= \boldsymbol{f}_1(\boldsymbol{x}(t), \boldsymbol{z}(t), \boldsymbol{\alpha}) \\ \dot{\boldsymbol{z}}(t) &= \boldsymbol{f}_2(\boldsymbol{z}(t), \boldsymbol{\alpha}) \end{aligned} \tag{86}$$

where $\boldsymbol{x}(t)$ and $\boldsymbol{z}(t)$ represent the *slow* and *fast* variables, respectively, and $\boldsymbol{\alpha}$ is a 4×1 parameter vector. Let us assume there are d_x slow variables, x_j, $j = 0, \ldots, d_x - 1$, and L fast variables per slow variable, i.e., z_l, $l = 0, ..., d_x L - 1$, overall. The maps, $\boldsymbol{f}_1$ and $\boldsymbol{f}_2$ are $\mathbb{R}^{d_x} \times \mathbb{R}^L \times \mathbb{R}^n \to \mathbb{R}^{d_x}$ and $\mathbb{R}^L \times \mathbb{R}^n \to \mathbb{R}^L$ functions, respectively, that can be written (skipping the time index t) as

$$\boldsymbol{f}_1 = [f_{1,0}, \ldots, f_{1,d_x-1}]^\top \quad \text{and} \quad f_{1,j}(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{\alpha}) = -x_{j-1}(x_{j-2} - x_{j+1}) - x_j + F - \frac{HC}{B} \sum_{l=(j-1)L}^{Lj-1} z_l,$$

$$\boldsymbol{f}_2 = [f_{2,0}, \ldots, f_{2,d_x L-1}]^\top \quad \text{and} \quad f_{2,l}(\boldsymbol{z}, \boldsymbol{\alpha}) = -CBz_{l+1}(z_{l+2} - z_{l-1}) - Cz_l + \frac{CF}{B} + \frac{HC}{B} z_{\lfloor \frac{l-1}{L} \rfloor},$$

where $j = 0, ..., d_x - 1$, $l = 0, ..., d_x L - 1$, and $\boldsymbol{\alpha}$ contains the parameters F,C,H and B. F is a forcing parameter that controls the turbulence of the chaotic flow, C determines the time scale of the fast variables $\{z_l\}_{l \geq 0}$, H controls the strength of the coupling between the fast and slow variables and B determines the amplitude of the fast variables [17]. The dynamic variables are assumed to be arranged on a circular structure, hence the operations on the j indices are modulo d_x and operations on the l indices are modulo L. This means that for any integer k, $j + k \equiv (j + k) \bmod d_x$ and $l + k \equiv (l + k) \bmod L$. Notation $\lfloor a \rfloor$ indicates the truncation of a positive real number a to the closest integer smaller than a.

We apply the 4th order Runge-Kutta (RK4) method to obtain a discrete-time version of the two-scale Lorenz 96 model. To be specific, we numerically integrate Eq. (86) by means of the difference equations

$$\bar{\boldsymbol{x}}_k = \bar{\boldsymbol{x}}_{k-1} + h\bar{\boldsymbol{f}}_1(\bar{\boldsymbol{x}}_{k-1}, \bar{\boldsymbol{z}}_{k-1}, \boldsymbol{\alpha}) + \sigma \boldsymbol{v}_k, \tag{87}$$

$$\bar{\boldsymbol{z}}_k = \bar{\boldsymbol{z}}_{k-1} + h\bar{\boldsymbol{f}}_2(\bar{\boldsymbol{z}}_{k-1}, \boldsymbol{\alpha}) + \bar{\sigma}\bar{\boldsymbol{v}}_k \tag{88}$$

where $h > 0$ is the integration step-size, $\boldsymbol{v}_k$ and $\bar{\boldsymbol{v}}$ are sequences of independent and identically distributed (i.i.d.) standard Gaussian random vectors, and $\sigma, \bar{\sigma} > 0$ are scale parameters. The vectors of slopes $\bar{\boldsymbol{f}}_1$ and $\bar{\boldsymbol{f}}_2$ in Eqs. (87) and (88) are calculated as

$$
\begin{aligned}
k_1 &= \boldsymbol{f}_1(\bar{\boldsymbol{x}}_{n-1}, \bar{\boldsymbol{z}}_{n-1}, \boldsymbol{\alpha}) \quad \text{and} \quad m_1 = \boldsymbol{f}_2(\bar{\boldsymbol{z}}_{n-1}, \boldsymbol{\alpha}),\\
k_2 &= \boldsymbol{f}_1(\bar{\boldsymbol{x}}_{n-1} + \frac{1}{2}hk_1, \bar{\boldsymbol{z}}_{n-1} + \frac{1}{2}hm_1, \boldsymbol{\alpha}) \quad \text{and} \quad m_2 = \boldsymbol{f}_2(\bar{\boldsymbol{z}}_{n-1} + \frac{1}{2}hm_1, \boldsymbol{\alpha}),\\
k_3 &= \boldsymbol{f}_1(\bar{\boldsymbol{x}}_{n-1} + \frac{1}{2}hk_2, \bar{\boldsymbol{z}}_{n-1} + \frac{1}{2}hm_2, \boldsymbol{\alpha}) \quad \text{and} \quad m_3 = \boldsymbol{f}_2(\bar{\boldsymbol{z}}_{n-1} + \frac{1}{2}hm_2, \boldsymbol{\alpha}),\\
k_4 &= \boldsymbol{f}_1(\bar{\boldsymbol{x}}_{n-1} + hk_3, \bar{\boldsymbol{z}}_{n-1} + hm_3, \boldsymbol{\alpha}) \quad \text{and} \quad m_4 = \boldsymbol{f}_2(\bar{\boldsymbol{z}}_{n-1} + hm_3, \boldsymbol{\alpha}),\\
\bar{\boldsymbol{f}}_1(\bar{\boldsymbol{x}}_{n-1}, \bar{\boldsymbol{z}}_{n-1}, \boldsymbol{\alpha}) &= \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4), \qquad (89)\\
\bar{\boldsymbol{f}}_2(\bar{\boldsymbol{z}}_{n-1}, \boldsymbol{\alpha}) &= \frac{1}{6}(m_1 + 2m_2 + 2m_3 + m_4) \qquad (90)
\end{aligned}
$$

where $j = 0, ..., d_x - 1$ and $l = 0, ..., d_x L - 1$.

We assume that the observations are linear but can only be collected from this system once every T discrete time steps. Moreover, only 1 out of K slow variables can be observed. Therefore, the observation process has the form

$$
\boldsymbol{y}_n = \begin{bmatrix} x_{K,nT} \\ x_{2K,nT} \\ \vdots \\ x_{d_y K,nT} \end{bmatrix} + \boldsymbol{u}_n, \qquad (91)
$$

where $n = 1, 2, ...$ and $\boldsymbol{u}_n$ is a sequence of i.i.d. random variables with common pdf $\mathcal{N}(\boldsymbol{u}_n | 0, \sigma_y^2 \mathbf{I}_{d_y})$.

In our computer experiments, system (87) is often employed to generate both ground-truth values for the slow variables $\{\boldsymbol{x}_n\}_{n\geq 0}$ and synthetic observations, $\{\boldsymbol{y}_n\}_{n\geq 1}$. As a forecast model for the slow variables it is common [15] to use the differential equation

$$
\dot{x}_j = f_j(\boldsymbol{x}, \boldsymbol{\theta}) = -x_{j-1}(x_{j-2} - x_{j+1}) - x_j + F - \ell(x_j, \mathsf{a}), \quad j = 0, ..., d_x - 1, \qquad (92)
$$

where $\mathsf{a} = [\mathsf{a}_1, \mathsf{a}_2]^\intercal$ is a (constant) parameter vector, $\boldsymbol{\theta} = [F, \mathsf{a}^\intercal]^\intercal$ contains all the parameters and function $\ell(x_{j,n-1}, \mathsf{a}) \in \mathbb{R}$ is a polynomial ansatz for the coupling term $\frac{HC}{B} \sum_{l=(j-1)L}^{Lj-1} \bar{x}_l$ in (87). Then, Eqs. (87) and (88) can be replaced by

$$
\boldsymbol{x}_k = \boldsymbol{x}_{k-1} + h\bar{\boldsymbol{f}}(\boldsymbol{x}_{k-1}, \boldsymbol{\theta}) + \sigma \boldsymbol{v}_k \qquad (93)
$$

where $\bar{\boldsymbol{f}}$ is the RK4 approximation of the function $\boldsymbol{f} = [f_0, \ldots, f_{d_x - 1}]^\intercal$ in Eq. (92).

In this paper we assume that $\ell(x_j, \mathsf{a})$ is a polynomial in x_j of degree 2, characterized by the roots a_1 and a_2. Assuming $\boldsymbol{y}_n$ is Gaussian distributed and $\boldsymbol{u}_n$ is a sequence of independent and identically distributed noise terms with Gaussian probability distribution, $p(\boldsymbol{u}) = \mathcal{N}(\boldsymbol{u}|0, \sigma_o^2 \boldsymbol{I}_{d_y})$, then

$$p(\boldsymbol{y}_n|\boldsymbol{x}_n, \boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{y}_n|\boldsymbol{x}_n, \sigma_o^2 \boldsymbol{I}_{d_y}) \tag{94}$$

which denotes a d_y-dimensional Gaussian density with zero mean and covariance matrix $\sigma_o^2 \boldsymbol{I}_{d_y}$, where $\boldsymbol{I}_{d_y}$ is the $d_y \times d_y$ identity matrix.

In Appendix B we provide a simplified numerical scheme for the approximate computation of the $d_x \times d_x$ predictive covariance matrices of the EKF algorithm when the Jacobian matrix $\boldsymbol{J}_{\bar{\boldsymbol{f}}, \boldsymbol{x}, \boldsymbol{\theta}}$ corresponds to the Lorenz 96 model just described.

VI. NUMERICAL RESULTS

We have conducted computer simulations to illustrate the performance of the proposed nested hybrid filtering methods. In particular, we have carried out computer experiments for three different schemes: the NPF of [10] and the two NHFs described in Section III C that rely on the EKF and the EnKF, respectively. The simulation setup is described below, followed by the discussion of our numerical results in Section VI B.

A. Simulation setup

For our computer experiments we have used the two-scale Lorenz 96 model of Eq. (86), in order to generate

- reference signals $\tilde{\boldsymbol{x}}_k$, $k = 0, 1, \ldots$, used as ground truth for the assessment of the estimators, and
- sequences of observations, $\boldsymbol{y}_n$, $n = 1, 2, \ldots$.

The model is integrated using the RK4 method with Gaussian perturbations (as outlined in Eqs. (87) and (88)). The integration step is set to $h = 10^{-3}$ continuous-time units through all experiments and the fixed model parameters are $F = 8$, $H = 0.75$, $C = 10$ and $B = 15$. For all experiments, we assume that there are $L = 10$ fast variables per slow variable, hence the

total dimension of the model is $10d_x$ (with different values of d_x for different experiments). The noise scaling factors are $\sigma = \frac{h}{4} = 0.25\times10^{-3}$ and $\sigma_o = 4$, both assumed known. We assume that half of the slow variables are observed in Gaussian noise, i.e., $K = 2$.

We assess the accuracy of the estimation algorithms in terms of the mean square error (MSE) of the predictors of the dynamic variables. For the NHFs, these estimators take the form

$$\hat{\breve{\boldsymbol{x}}}_k = \sum_{i=1}^{N} w_n^i \breve{\boldsymbol{x}}_k^i, \tag{95}$$

and the estimator of the error is

$$\tilde{\mathcal{E}}_k = \sum_{i=1}^{N} w_n^i \mathrm{trace}\{\breve{\boldsymbol{P}}_k^i\}. \tag{96}$$

In the plots, however, we show the empirical MSE per dimension resulting directly from the simulations,

$$MSE_k = \frac{1}{d_x} \parallel \breve{\boldsymbol{x}}_k - \tilde{\boldsymbol{x}}_k \parallel^2 . \tag{97}$$

averaged over several independent simulation runs.

The simulations presented below include running times for the different methods. They have been obtained with an iMac computer with 32 GB of DRAM and equipped with an Intel Core i7 processor.

B. Results

Table I shows a comparison of the performance of the NPF and the two NHFs, based on the EKF and the EnKF schemes as described in Section III, in terms of their running times and the MSE of the state estimators (averaged over time and dimensions). We have carried out this computer simulation for a model with dimension $d_x = 50$ and a gap between observations of $hT = 0.05$ continuous-time units. All algorithms work with $N = 200$ particles for the approximation of the posterior distributions of the fixed parameters. The second-layer particle filters in the NPF use $M = 200$ particles each and the EnKFs run with $M = 50$ samples each. It can be seen that the least error is achieved by the NHF-EnKF method, with a running time similar to the NPF, which achieves an MSE that is one order of magnitude higher. The NHF-EKF attains a moderate MSE (considerably better than the NPF) with just a fraction of the running time. In order to improve the performance of the NPF, the

numbers of particles M and N would have to be considerably increased, but this would increase the running times correspondingly (the complexity of the NPF is $\mathcal{O}(NM)$ [10]).

Algorithm	Running time (minutes)	MSE
NPF	9.872	6.062
NHF + EKF	1.196	1.653
NHF + EnKF	11.674	0.472

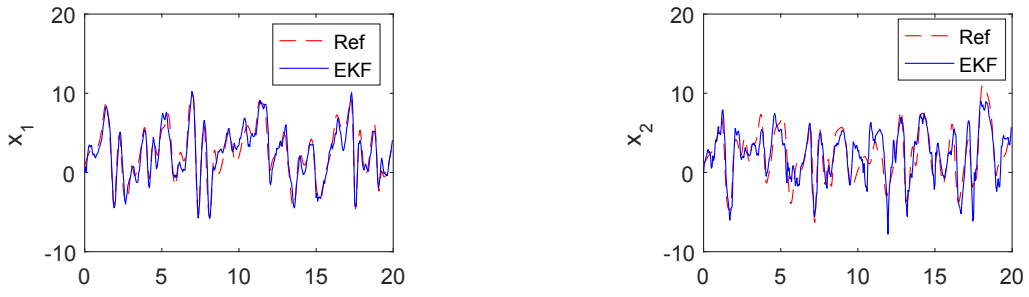
TABLE I: Running times and average MSE (over time and state dimensions) for the NPF and two NHFs, based on the EKF and the EnKF, respectively.

Next, we show results for a computer experiment in which we have used the NHF-EKF method to estimate the parameters F and a and track the state variables of the two-scale Lorenz system with dimension $d_x = 4,000$ and a gap between consecutive observations of $hT = 0.05$ continuous-time units. As in the rest of computer simulations, the number of particles used to approximate the sequence of parameter posterior distributions is $N = 200$.

Figure 2 shows the true state trajectories, together with their estimates, for the first two state variables of the two-scale Lorenz 96 model. We note that the first variable, $x_1(t)$, is observed in Gaussian noise (with $\sigma_o = 4$) while the second variable, $x_2(t)$, is not observed. The accuracy of the estimation is similar, though, over the 20 continuous-time units of the simulation run (corresponding to 20×10^3 discrete time steps).

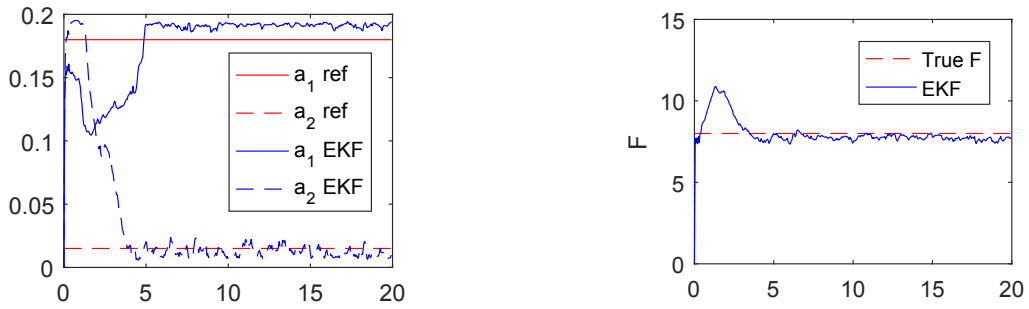
In Figure 3 we observe the estimates of the fixed parameters F, a_1 and a_2, together with the reference values. Note that the value $F = 8$ is ground truth, but the values of a_1 and a_2 are genie-aided least squares estimates obtained by observing directly the fast variables of the two-scale model. There is a similar time-to-convergence for the three parameters: after 5 continuous time units, the algorithm yields reliable estimates of F, a_1 and a_2.

In the next set of computer experiments we compare the NHF-EKF and the NHF-EnKF methods in terms of their average MSE and their running times for different values of the state dimension d_x and the gap between consecutive observations T (in discrete time steps). For each combination of d_x and T we have carried out 20 independent simulation runs. The number of particles in the parameter space is fixed, $N = 200$, for all simulations, but the size of the ensemble in the EnKFs is adjusted to the dimension, in particular, we set $M = d_x$.



(a) (b)

FIG. 2: Sequences of state values (dashed red line) and estimates (blue line) in x_1 (a) and x_2 (b) over time. Variable x_1 is observed (in Gaussian noise), while x_2 is unobserved.



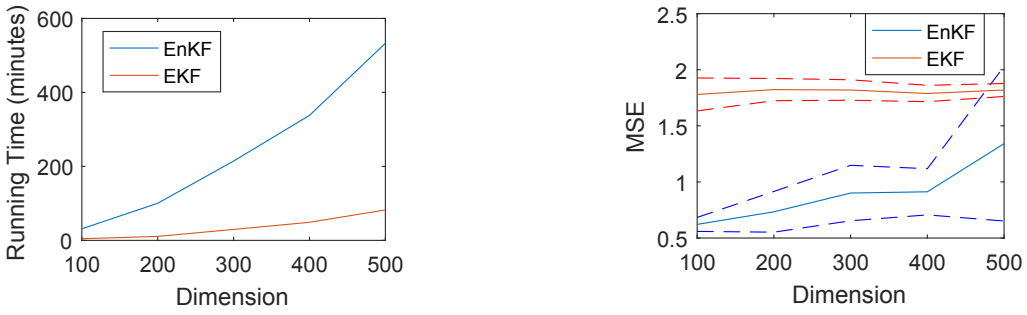
(a) (b)

FIG. 3: Estimates of the parameters $\mathsf{a} = [\mathsf{a}_1, \mathsf{a}_2]^\top$ and F in a 4,000-dimensional Lorenz 96 model. The reference values are represented in red dashed lines.

Figure 4 shows the running times and the average MSE attained by the two NHFs when the state dimension d_x ranges from 100 to 500. The gap between observations is fixed to $T = 50$ (i.e., 0.05 time units). We observe that the NHF-EKF method attains significantly lower running times (by a factor of ~ 6, compared to the NHF-EnKF), which increase only moderately with the dimension d_x. The NHF-EnKF scheme yields smaller values of MSE, however they appear to grow relatively quickly with d_x.

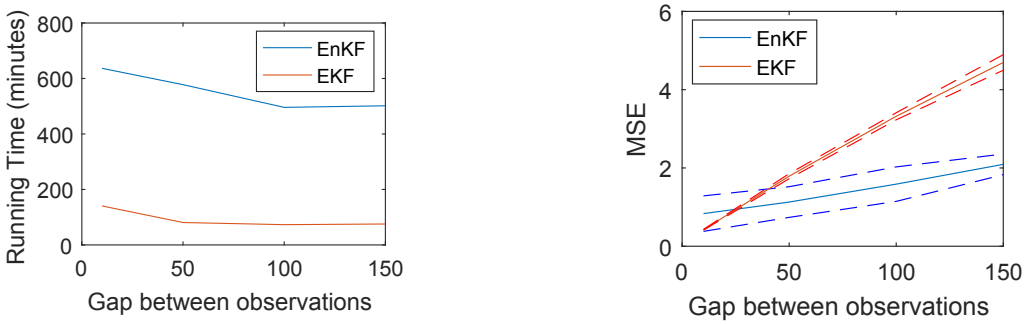
Finally, Figure 5 displays the running times and the average MSEs attained by the two NHFs as we increase the gap between observations from $T = 10$ to $T = 150$ (hence, from $hT = 0.01$ to $hT = 0.15$ continuous time units). The dimension of dynamic variables for this experiment is fixed to $d_x = 500$. Note that, as the gap T increases, less data points are effectively available for the estimation of both the parameters and the states. We observe, again, that the NHF-EnKF is computationally more costly than the NHF-EKF, however

it attains a consistently smaller MSE. Moreover, in this simulation we observe that the estimation errors of the NHF-EnKF increase at a lower rate (compared to the NHF-EKF) as less observations are collected, suggesting that it may be a more efficient algorithm in data-poor scenarios.



(a) (b)

FIG. 4: Comparison of the NHF-EKF (red lines) and NHF-EnKF (blue lines) in terms of their running time (a) and their MSE (b) as the state dimension d_x increases, with a fixed gap between observations of $T = 50$ discrete time steps. The dashed lines in (b) indicate the one standard deviation w.r.t. the mean.



(a) (b)

FIG. 5: Comparison of the NHF-EKF (red lines) and NHF-EnKF (blue lines) in terms of their running time (a) and their MSE (b) as the gap between observations T increases, with fixed state dimension $d_x = 500$. The dashed lines in (b) indicate the one standard deviation w.r.t. the mean.

VII. CONCLUSIONS

We have introduced a nested filtering methodology to recursively estimate the static parameters and the dynamic variables of nonlinear, possibly chaotic, dynamical systems. The proposed framework combines a recursive Monte Carlo approximation method to compute the posterior probability distribution of the static parameters with a variety of filtering techniques to estimate the posterior distribution of the state variables of the system. In particular, we have investigated the use of Gaussian filters, as they admit fast implementations that can be well suited to high dimensional systems. As a result, we have proposed two nested hybrid filters that combine a sequential importance sampling scheme for the (moderate dimensional) unknown static parameters of the dynamical system with either extended Kalman filtering or ensemble Kalman filtering for the (higher dimensional) time-varying states. We have presented numerical results for a two-scale stochastic Lorenz 96 system, a model commonly used for the assessment of data assimilation methods in the Geophysics. We illustrate the average performance of the methods in terms of estimation errors and running times, and show numerical results for a 4,000-dimensional system. This has been achieved with a relatively inefficient implementation of the method running on a desktop computer, hence we expect that the method can be applied to much larger scale systems using adequate hardware and software.

ACKNOWLEDGMENTS

This research has been partially supported by the Spanish Ministry of Economy and Competitiveness (project TEC2015-69868-C2-1-R ADVENTURE) and the Office of Naval Research (ONR) Global (Grant Award no. N62909-15-1-2011).

Appendix A: Simplification of the inverse $(\boldsymbol{S}^i)^{-1}$

The predictive covariance of the observation vector $\boldsymbol{y}_n$ is a $d_y \times d_y$ matrix $\boldsymbol{S}_n$. Inverting $\boldsymbol{S}_n$ has a cost $\mathcal{O}(d_y^3)$, which can become intractable. Assuming that variables located "far away" in the circumference of the Lorenz 96 model have small correlation we can approximate $\boldsymbol{S}_n$

as a block diagonal matrix, namely, $\hat{\boldsymbol{S}}_n = \boldsymbol{S}_n \odot \boldsymbol{M}$, where $\odot$ denotes element-wise product,

$$\boldsymbol{M} = \begin{bmatrix} \mathbf{1} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{1} & \dots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \dots & \mathbf{1} \end{bmatrix} \tag{A1}$$

is a mask matrix and $\mathbf{0}$ and $\mathbf{1}$ are, respectively, matrices of zeros and ones of dimension $d_q \times d_q$. There are Q blocks in the diagonal of $\boldsymbol{M}$, hence $d_y = Qd_q$. The original matrix could contain some non-zero values where the zero blocks of $\boldsymbol{M}$ are placed, however their values are assumed close to zero. The resulting matrix,

$$\hat{\boldsymbol{S}} = \begin{bmatrix} \bar{\boldsymbol{S}}_1 & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \bar{\boldsymbol{S}}_2 & \dots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \dots & \bar{\boldsymbol{S}}_Q \end{bmatrix}, \quad \text{is easily inverted as} \quad \hat{\boldsymbol{S}}_n^{-1} = \begin{bmatrix} \bar{\boldsymbol{S}}_1^{-1} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \bar{\boldsymbol{S}}_2^{-1} & \dots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \dots & \bar{\boldsymbol{S}}_Q^{-1} \end{bmatrix}$$

with a computational cost $\mathcal{O}(Qd_q^3) = \mathcal{O}(\frac{d_y^3}{Q^2})$.

Appendix B: Jacobian product simplifications for the Lorenz 96 model

Applying the Lorenz 96 model, the Jacobian matrix $\boldsymbol{J}_{\bar{\boldsymbol{f}},\boldsymbol{x},\boldsymbol{\theta}}$ in Algorithm 2 can be written as

$$\boldsymbol{J}_{\bar{\boldsymbol{f}},\boldsymbol{x},\boldsymbol{\theta}} = \begin{bmatrix} c_{1,1} & c_{1,2} & 0 & 0 & 0 & \dots & 0 & 0 & c_{1,d_x-1} & c_{1,d_x} \\ c_{2,1} & c_{2,2} & c_{2,3} & 0 & 0 & \dots & 0 & 0 & 0 & c_{2,d_x} \\ c_{3,1} & c_{3,2} & c_{3,3} & c_{3,4} & 0 & \dots & 0 & 0 & 0 & 0 \\ 0 & c_{4,2} & c_{4,3} & c_{4,4} & c_{4,5} & \dots & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & \dots & c_{d_x-2,d_x-3} & c_{d_x-2,d_x-2} & c_{d_x-2,d_x-1} & 0 \\ 0 & 0 & 0 & 0 & 0 & \dots & c_{d_x-1,d_x-3} & c_{d_x-1,d_x-2} & c_{d_x-1,d_x-1} & c_{d_x-1,d_x} \\ c_{d_x,1} & 0 & 0 & 0 & 0 & \dots & 0 & c_{d_x,d_x-2} & c_{d_x,d_x-1} & c_{d_x,d_x} \end{bmatrix}$$

where $\boldsymbol{J}(\bar{\boldsymbol{f}})$ is a $d_x \times d_x$ matrix. The $c_{i,j}$'s denote the non-zero entries of the matrix. The division by blocks of this matrix is quite similar to the one made in Eq. (A1). However, there are some additional restrictions. We can name the q-th block as $\boldsymbol{J}^{(q)}$, where $q = 1, 2, \dots, Q$ and Q is the number of blocks used. Then, if we adopt a Matlab notation to indicate the

rows and columns, we obtain $\boldsymbol{J}^{(q)}_{a:b,c:d}$, where a and b (respectively c and d) are the indices of the initial and final rows (columns) of the q-th block and they are computed as

$$a = (q-1) \times \frac{d_x}{Q} + 1, \tag{B1}$$

$$b = a + \frac{d_x}{Q} - 1, \tag{B2}$$

$$c = a - 2, \tag{B3}$$

$$d = b + 1. \tag{B4}$$

Moreover, the covariance matrix $\check{\boldsymbol{P}}_{k-1}$ can be simplified applying the division by blocks as in Eq. (A1), taking R blocks. After that, the first part of the product can be computed as

$$\boldsymbol{A}_{a:b,\lfloor \frac{c}{R} \rfloor \times R+1:\lceil \frac{d}{R} \rceil \times R} = \boldsymbol{J}_{a:b,c:d} \check{\boldsymbol{P}}_{k-1,c:d,\lfloor \frac{c}{R} \rfloor \times R+1:\lceil \frac{d}{R} \rceil \times R} \tag{B5}$$

where the matrix $\boldsymbol{A}$ is auxiliary to calculate the final product as

$$\check{\boldsymbol{P}}_k = \boldsymbol{A}_{:,a:b} \times \boldsymbol{J}^{-1}_{a:b,c:d} + \boldsymbol{Q} \tag{B6}$$

We have particular cases for the first and the last blocks, being necessary to rewrite Eqs. (B5) and (B6). The indices used to indicate the initial and final columns of the block, $(c:d)$, are replaced in the first case by $(a:d)$ in addition to the last two columns of the whole matrix. In the same way, for the last block we replace the columns by $(c:b)$, adding the first column of the matrix as well.

[1] A. M. Clayton, A. Lorenc, and D. M. Barker, Quarterly Journal of the Royal Meteorological Society **139**, 1445 (2013).

[2] P. J. V. Leeuwen, Monthly Weather Review **131**, 2071 (2003).

[3] D. P. Dee, S. M. Uppala, A. J. Simmons, P. Berrisford, P. Poli, S. Kobayashi, U. Andrae, M. A. Balmaseda, G. Balsamo, and P. Bauer, Quarterly Journal of the royal meteorological society **137**, 553 (2011).

[4] A. Golightly and D. J. Wilkinson, Interface Focus **1**, 807 (2011).

[5] A. Aksoy, F. Zhang, and J. W. Nielsen-Gammon, Monthly Weather Review **134**, 2951 (2006).

[6] G. Evensen, IEEE Control Systems **29** (2009).

[7] C. Andrieu, A. Doucet, and R. Holenstein, Journal of the Royal Statistical Society B **72**, 269 (2010).

[8] N. Chopin, P. E. Jacob, and O. Papaspiliopoulos, Journal of the Royal Statistical Society: Series B (Statistical Methodology) (2012).

[9] C. Andrieu, A. Doucet, S. S. Singh, and V. B. Tadić, Proceedings of the IEEE **92**, 423 (2004).

[10] D. Crisan and J. Miguez, to appear in Bernoulli **arXiv:1308.1883v3 [stat.CO]** (2016).

[11] N. Gordon, D. Salmond, and A. F. M. Smith, IEE Proceedings-F **140**, 107 (1993).

[12] A. Doucet, S. Godsill, and C. Andrieu, Statistics and Computing **10**, 197 (2000).

[13] B. D. O. Anderson and J. B. Moore, *Optimal Filtering* (Englewood Cliffs, 1979).

[14] G. Evensen, Ocean dynamics **53**, 343 (2003).

[15] J. Hakkarainen, A. Ilin, A. Solonen, M. Laine, H. Haario, J. Tamminen, E. Oja, and H. Järvinen, Nonlinear Proc. Geoph. **19**, 127 (2012).

[16] P. J. V. Leeuwen, Quarterly Journal of the Royal Meteorological Society **136**, 1991 (2010).

[17] H. M. Arnold, *Stochastic parametrisation and model uncertainty*, Ph.D. thesis, University of Oxford (2013).

[18] Specifically, one observation vector is collected every T discrete-time steps or, equivalently, every hT continuous-time units.

[19] C. P. Robert and G. Casella, *Monte Carlo Statistical Methods* (Springer, 2004).

[20] S. J. Julier and J. Uhlmann, Proceedings of the IEEE **92**, 401 (2004).

[21] K. Law, A. Stuart, and K. Zygalakis, *Data Assimilation* (Springer, 2015).

[22] A. Papavasiliou, Journal of Applied Probability **42**, 1053 (2005).

[23] A. Bain and D. Crisan, *Fundamentals of Stochastic Filtering* (Springer, 2008).

[24] J. Míguez, D. Crisan, and P. M. Djurić, Statistics and Computing **23**, 91 (2013).