A nested hybrid filter for parameter estimation and state tracking in homogeneous multi-scale models

Sara Pérez-Vieites
Department of Signal Theory and Communications
Universidad Carlos III de Madrid
Madrid, Spain
spvieites@tsc.uc3m.es

Joaquín Míguez
Department of Signal Theory and Communications
Universidad Carlos III de Madrid
Madrid, Spain
joaquin.miguez@uc3m.es

***Abstract*—Multi-scale problems, where variables of interest evolve in different time-scales and live in different state-spaces, can be found in many fields of science. Here, we introduce a new recursive methodology for Bayesian inference that aims at estimating the static parameters and tracking the dynamic variables of these kind of systems. Although the proposed approach works in rather general multi-scale systems, for clarity we analyze the case of a homogeneous multi-scale model with 3 time-scales (static parameters, slow dynamic state variables and fast dynamic state variables). The proposed scheme, based on nested filtering methology of [1], combines three intertwined layers of filtering techniques that approximate recursively the joint posterior probability distribution of the parameters and both sets of dynamic state variables given a sequence of partial and noisy observations. We explore the use of sequential Monte Carlo schemes in the first and second layers while we use an unscented Kalman filter to obtain a Gaussian approximation of the posterior probability distribution of the fast variables in the third layer. Some numerical results are presented for a stochastic two-scale Lorenz 96 model with unknown parameters.**

I. Introduction

Multi-scale problems can be found in many fields of science, such as biology, chemistry, fluid dynamics and material science, and so their mathematical modeling and the computation of quantitative solutions are of broad interest [2]. However, the modeling of such systems is complex, since they are governed by processes at different time-scales that may be described by diverse laws. Moreover, the cross dependencies among these processes must also be modeled. Therefore, the problem of tracking the time evolution of a multi-scale dynamical system involves the prediction and estimation of several sets of variables that live in different state-spaces and evolve in different time scales. Moreover, the tracking of the variables of interest usually has to be performed from the observation of partial and noisy observations. Efficient recursive algorithms for this task are badly needed.

The simplest case of a multi-scale problem consists of a system with unknown static parameters and dynamic state variables, since the parameters may be considered as state variables that evolve at a greater time scale. Hence, it is a multi-scale problem with only two time scales. In general, carrying out both parameter estimation and state tracking at once implies several practical and theoretical difficulties. A few well-principled methods have been proposed in the last few years, though. Sequential Monte Carlo square (SMC^2) [3] or particle Markov chain Monte Carlo (PMCMC) [4] are examples of schemes that yield theoretically-guaranteed solutions to this problem. They aim at computing the joint posterior probability distribution of all the unknown variables and parameters of the system. Unfortunately, both SMC^2 and PMCMC are batch techniques. In other words, every time a new observation arrives, the whole sequence of observations has to be processed from scratch in order to update the estimates. Nested particle filters (NPFs) [5] apply the same principles as SMC^2 but in a recursive way. It is a scheme with two interwined layers of Monte Carlo methods, one inside the other, that estimate parameters and states, using the first layer for estimating the parameters and the second layer for tracking state variables. This methodology is better suited for long sequences of observations, however, the use of Monte Carlo in both layers of filters still makes its computational cost prohibitive in high-dimensional problems. Nested hybrid filters (NHFs) [1] introduce Gaussian filtering techniques in the second layer of the algorithm, reducing the computational cost considerably and making the methodology more appealing for these settings.

In this paper, we propose a generalization of the NHF methodology to tackle the problem of recursive Bayesian inference for a class of homogeneous multi-scale state-space models. We analyse the case of a dynamical system with three time scales (static parameters, slow dynamic state variables and fast dynamic state variables), but the methodology works in the same way for more general examples (namely, systems with n scales).

The new scheme is a three-layer nested filter that approximates, in a recursive manner, the posterior probability distributions of the parameters and the two sets of state variables given the sequence of available observations. In the first layer, we approximate the posterior probability distribution of the parameters by using sequential Monte Carlo (SMC) algorithm. This scheme is intertwined with a second layer of SMC techniques that approximate the posterior probability distribution

This research was partially supported by *Agencia Estatal de Investigación* of Spain (RTI2018-099655-B-I00 CLARA), the regional government of Madrid (project no. Y2018/TCS-4705 PRACTICO) and the Office of Naval Research (award no. N00014-19-1-2226).

of the slow state variables. Finally, in a third layer of filtering, we use an unscented Kalman filter (UKF) [6] in order to obtain a Gaussian approximation of the posterior probability distribution of the fast state variables. The computations on the second layer are conditional on the candidate parameter values generated on the first layer, while the calculations on the third layer are conditional on the candidates drawn at the first and second layers.

The rest of the paper is organised as follows. We state the problem to be addressed in Section II. In Section III, we describe the new methodology for multi-scale systems with static parameters and two sets of dynamic state variables. Numerical results for the stochastic two-scale Lorenz 96 model are shown in Section IV and conclusions are drawn in Section V.

II. Problem Statement

A. State space models

We are interested in systems that can be described by multidimensional stochastic differential equations (SDEs). In particular, we consider a general class of models that can be described by the pair of stochastic differential vector equations

$$d\boldsymbol{x} = f_{\boldsymbol{x}}(\boldsymbol{x}, \boldsymbol{\theta})dt + g_{\boldsymbol{x}}(\boldsymbol{z}, \boldsymbol{\theta})dt + \sigma_x d\boldsymbol{v}, \tag{1}$$

$$d\boldsymbol{z} = f_{\boldsymbol{z}}(\boldsymbol{x}, \boldsymbol{\theta})dt + g_{\boldsymbol{z}}(\boldsymbol{z}, \boldsymbol{\theta})dt + \sigma_z d\boldsymbol{w}, \tag{2}$$

where t denotes continuous time, $\boldsymbol{x}(t) \in \mathbb{R}^{d_x}$ and $\boldsymbol{z}(t) \in \mathbb{R}^{d_z}$ are the slow and fast states of the system, respectively, $f_{\boldsymbol{x}} \colon \mathbb{R}^{d_x} \times \mathbb{R}^{d_\theta} \to \mathbb{R}^{d_x}$, $g_{\boldsymbol{x}} \colon \mathbb{R}^{d_z} \times \mathbb{R}^{d_\theta} \to \mathbb{R}^{d_x}$, $f_{\boldsymbol{z}} \colon \mathbb{R}^{d_x} \times \mathbb{R}^{d_\theta} \to \mathbb{R}^{d_z}$ and $g_{\boldsymbol{z}} \colon \mathbb{R}^{d_z} \times \mathbb{R}^{d_\theta} \to \mathbb{R}^{d_z}$ are possibly non-linear functions parameterized by a fixed vector of unknown parameters, $\boldsymbol{\theta} \in \mathbb{R}^{d_\theta}$, $\sigma_x, \sigma_z > 0$ are known scale parameters that control the intensity of the stochastic perturbations, and $\boldsymbol{v}(t)$ and $\boldsymbol{w}(t)$ are vectors of independent standard Wiener processes with dimension d_x and d_z, respectively.

Equations (1)–(2) do not have closed form solutions for general nonlinear functions $f_{\boldsymbol{x}}$ and $f_{\boldsymbol{z}}$ and they have to be discretized for their numerical integration. In order to handle the slow and fast time scales, we apply a macro-micro solver [7], [8] that runs an Euler-Maruyama scheme at each layer, albeit with different integration steps. To be specific, we use Δ_z as the integration step of $\boldsymbol{z}$ while $\Delta_x \gg \Delta_z$ is the integration step of $\boldsymbol{x}$. Then, we can simulate $\boldsymbol{x}$ and $\boldsymbol{z}$ using the pair of difference equations

$$\begin{aligned}\boldsymbol{x}_n = \boldsymbol{x}_{n-1} &+ \Delta_x(f_{\boldsymbol{x}}(\boldsymbol{x}_{n-1}, \boldsymbol{\theta}) + g_{\boldsymbol{x}}(\bar{\boldsymbol{z}}_n, \boldsymbol{\theta})) \\ &+ \sqrt{\Delta_x}\sigma_x \boldsymbol{v}_n,\end{aligned} \tag{3}$$

$$\begin{aligned}\boldsymbol{z}_k = \boldsymbol{z}_{k-1} &+ \Delta_z(f_{\boldsymbol{z}}(\boldsymbol{x}_{\lfloor \frac{k-1}{h} \rfloor}, \boldsymbol{\theta}) + g_{\boldsymbol{z}}(\boldsymbol{z}_{k-1}, \boldsymbol{\theta})) \\ &+ \sqrt{\Delta_z}\sigma_z \boldsymbol{w}_k,\end{aligned} \tag{4}$$

where $\boldsymbol{x}_n \simeq \boldsymbol{x}(n\Delta_x)$ and $\boldsymbol{z}_k \simeq \boldsymbol{z}(k\Delta_z)$, $n \in \mathbb{N}$ denotes discrete time in the time scale of the slow variables, $k \in \mathbb{N}$ denotes discrete time in the fast variables time scale, $h = \frac{\Delta_x}{\Delta_z} \in \mathbb{Z}^+$ is the ratio between the two time scales, $\boldsymbol{v}_n$ and $\boldsymbol{w}_k$ are Gaussian random variables of zero mean and covariance matrices $\boldsymbol{I}_{d_x}$ and $\boldsymbol{I}_{d_z}$ respectively, and $\bar{\boldsymbol{z}}_n$ is an average computed as

$$\bar{\boldsymbol{z}}_n = \frac{1}{h} \sum_{i=h(n-1)+1}^{hn} \boldsymbol{z}_i. \tag{5}$$

We assume that we observe both state variables $\boldsymbol{x}_n$ and $\boldsymbol{z}_k$, but only in the (slow) time scale of $\boldsymbol{x}$. Then, the n-th observation is a random variable, $\boldsymbol{y}_n \in \mathbb{R}^{d_y}$, of dimension d_y, which we model as

$$\boldsymbol{y}_n = l(\boldsymbol{z}_{hn}, \boldsymbol{x}_n, \boldsymbol{\theta}) + \boldsymbol{r}_n, \tag{6}$$

where $l \colon \mathbb{R}^{d_z} \times \mathbb{R}^{d_x} \times \mathbb{R}^{d_\theta} \to \mathbb{R}^{d_y}$ is a transformation that maps the states into the observation space, and $\boldsymbol{r}_n$ is a 0-mean observational-noise vector with covariance matrix $\boldsymbol{R}$.

B. Model inference

The key difficulty of this problem is the *joint* Bayesian estimation of the parameters, $\boldsymbol{\theta}$, and all states, $\boldsymbol{x}$ and $\boldsymbol{z}$, since they all depend on each other. This means that the estimation of the fixed parameters is necessary to track both sets of state variables and, at the same time, tracking the slow state variables is needed for predicting the time evolution of the fast states and vice versa. From the viewpoint of Bayesian analysis, we aim at approximating the joint posterior probability density function (pdf) $p(\boldsymbol{\theta}, \boldsymbol{x}_{0:n}, \boldsymbol{z}_{hn} | \boldsymbol{y}_{1:n})$. Using the chain rule, we can factorize this pdf as

$$\begin{aligned}p(\boldsymbol{z}_{hn}, \boldsymbol{x}_n, \boldsymbol{\theta} | \boldsymbol{y}_{1:n}) = &p(\boldsymbol{z}_{hn} | \boldsymbol{x}_{0:n}, \boldsymbol{y}_{1:n}, \boldsymbol{\theta}) \\ &\times p(\boldsymbol{x}_{0:n} | \boldsymbol{y}_{1:n}, \boldsymbol{\theta}) p(\boldsymbol{\theta} | \boldsymbol{y}_{1:n}),\end{aligned} \tag{7}$$

where we identify the three key distributions that we need to approximate. Each one of these pdf's can be handled in a different *layer* of computation. Hence, we aim at designing a nested filter (in the vein of [1] and [5]) with three layers. In the first one we compute an approximation of $p(\boldsymbol{\theta} | \boldsymbol{y}_{1:n})$. In the second one, we approximate $p(\boldsymbol{x}_{0:n} | \boldsymbol{y}_{1:n}, \boldsymbol{\theta})$, and in the third one we tackle $p(\boldsymbol{z}_{hn} | \boldsymbol{x}_{0:n}, \boldsymbol{y}_{1:n}, \boldsymbol{\theta})$. The general methodology is described in Section III-A and one possible implementation is obtained in Section III-B.

III. Multi-scale nested filter

In this section, we introduce a class of NHFs with three layers of computation. We outline the methodology used to obtain the Gaussian approximation of $p(\boldsymbol{z}_{hn} | \boldsymbol{x}_{0:n}, \boldsymbol{y}_{1:n}, \boldsymbol{\theta})$ and the Monte Carlo approximations of the distributions with pdf's $p(\boldsymbol{x}_{0:n} | \boldsymbol{y}_{1:n}, \boldsymbol{\theta})$ and $p(\boldsymbol{\theta} | \boldsymbol{y}_{1:n})$. Let us remark that, given a point-mass Monte Carlo estimate of the distribution with pdf $p(\boldsymbol{x}_{0:n} | \boldsymbol{y}_{1:n}, \boldsymbol{\theta})$, it is straightforward to integrate out the sequence $\boldsymbol{x}_{0:n-1}$ to obtain an estimate of the distribution with pdf $p(\boldsymbol{x}_n | \boldsymbol{y}_{1:n}, \boldsymbol{\theta})$. This is standard procedure, e.g., in particle filtering [9].

A. Nested conditional filtering in multiple layers

In order to compute $p(\boldsymbol{\theta}|\boldsymbol{y}_{1:n})$ in the first layer of the filter, we can recursively factorize this pdf as

$$p(\boldsymbol{\theta}|\boldsymbol{y}_{1:n}) \propto p(\boldsymbol{y}_n|\boldsymbol{\theta}, \boldsymbol{y}_{1:n-1}) p(\boldsymbol{\theta}|\boldsymbol{y}_{1:n-1}), \tag{8}$$

using Bayes' theorem, where $p(\boldsymbol{y}_n|\boldsymbol{\theta}, \boldsymbol{y}_{1:n-1})$ is the likelihood of $\boldsymbol{\theta}$ and $p(\boldsymbol{\theta}|\boldsymbol{y}_{1:n-1})$ is the posterior pdf at the previous time step. Then, the first layer is devoted to the calculation of $p(\boldsymbol{\theta}|\boldsymbol{y}_{1:n})$ and the key difficulty for the update of the latter pdf is the computation of the likelihood $p(\boldsymbol{y}_n|\boldsymbol{\theta}, \boldsymbol{y}_{1:n-1})$, which has to be evaluated in a second layer of computation.

In order to identify the tasks to be performed in this second layer, we rewrite the likelihood as the integral

$$\begin{aligned} p(\boldsymbol{y}_n|\boldsymbol{\theta}, \boldsymbol{y}_{1:n}) = \int & p(\boldsymbol{y}_n|\boldsymbol{x}_{0:n}, \boldsymbol{y}_{1:n-1}, \boldsymbol{\theta}) \\ & \times p(\boldsymbol{x}_{0:n}|\boldsymbol{y}_{1:n-1}, \boldsymbol{\theta}) d\boldsymbol{x}_{0:n}, \end{aligned} \tag{9}$$

where $p(\boldsymbol{x}_{0:n}|\boldsymbol{y}_{1:n-1}, \boldsymbol{\theta})$ is the joint one-step-ahead predictive pdf of the slow state variables, $\boldsymbol{x}_{0:n}$, and $p(\boldsymbol{y}_n|\boldsymbol{x}_{0:n}, \boldsymbol{y}_{1:n-1}, \boldsymbol{\theta})$ is the likelihood of $\boldsymbol{\theta}$ *and* $\boldsymbol{x}_{0:n}$. In order to evaluate this integral, we need to track the sequence of posterior pdf's $p(\boldsymbol{x}_{0:n}|\boldsymbol{y}_{1:n-1}, \boldsymbol{\theta})$, which is precisely the task of the second layer of computation.

The likelihood $p(\boldsymbol{y}_n|\boldsymbol{x}_{0:n}, \boldsymbol{y}_{1:n-1}, \boldsymbol{\theta})$ in (9) is evaluated in the third layer. Again, we write it as an integral, namely,

$$\begin{aligned} p(\boldsymbol{y}_n|\boldsymbol{x}_{0:n}, & \boldsymbol{y}_{1:n-1}, \boldsymbol{\theta}) = \\ & = \int p(\boldsymbol{y}_n|\boldsymbol{z}_{hn}, \boldsymbol{x}_{0:n}, \boldsymbol{y}_{1:n-1}, \boldsymbol{\theta}) \\ & \qquad \times p(\boldsymbol{z}_{hn}|\boldsymbol{x}_{0:n}, \boldsymbol{y}_{1:n-1}, \boldsymbol{\theta}) d\boldsymbol{z}_{hn} \\ & = \int p(\boldsymbol{y}_n|\boldsymbol{z}_{hn}, \boldsymbol{x}_n, \boldsymbol{\theta}) p(\boldsymbol{z}_{hn}|\boldsymbol{x}_{0:n}, \boldsymbol{y}_{1:n-1}, \boldsymbol{\theta}) d\boldsymbol{z}_{hn}, \end{aligned} \tag{10}$$

where $p(\boldsymbol{z}_{hn}|\boldsymbol{x}_{0:n}, \boldsymbol{y}_{1:n-1}, \boldsymbol{\theta})$ is the one-step-ahead predictive pdf of the fast state variables (conditional on the slow ones and the parameters) and $p(\boldsymbol{y}_n|\boldsymbol{z}_{hn}, \boldsymbol{x}_n, \boldsymbol{y}_{1:n-1}, \boldsymbol{\theta})$ is the likelihood of $\boldsymbol{\theta}$ and both state variables, $\boldsymbol{x}_n$ and $\boldsymbol{z}_{hn}$. We point out the simplification of the likelihood in (10), which follows from the assumption of conditional independence of the observations (given the full states of the system) in the model of Section II. In order to compute the integral in (10), we need to track the posterior pdf $p(\boldsymbol{z}_{hn}|\boldsymbol{x}_{0:n}, \boldsymbol{y}_{1:n-1}, \boldsymbol{\theta})$ in the third layer of computation.

To summarize:

- The third layer tracks the fast variables, by computing the pdf $p(\boldsymbol{z}_{hn}|\boldsymbol{x}_{0:n}, \boldsymbol{y}_{1:n-1}, \boldsymbol{\theta})$, and evaluates the likelihood $p(\boldsymbol{y}_n|\boldsymbol{x}_{0:n}, \boldsymbol{y}_{1:n-1}, \boldsymbol{\theta})$.
- The second layer takes the likelihood $p(\boldsymbol{y}_n|\boldsymbol{x}_{0:n}, \boldsymbol{y}_{1:n-1}, \boldsymbol{\theta})$ and uses it to track the posterior pdf of the slow states, $p(\boldsymbol{x}_{0:n}|\boldsymbol{y}_{1:n-1}, \boldsymbol{\theta})$. It uses this posterior to evaluate the likelihood $p(\boldsymbol{y}_n|\boldsymbol{y}_{1:n-1}, \boldsymbol{\theta})$.
- The first layer takes $p(\boldsymbol{y}_n|\boldsymbol{y}_{1:n-1}, \boldsymbol{\theta})$ and uses this likelihood to track the posterior pdf of the parameters, $p(\boldsymbol{\theta}|\boldsymbol{y}_{1:n})$.

B. Hybrid numerical scheme

A particular implementation of the multi-scale nested filter is summarized in Algorithms 1–3. Specifically, we describe the use of SMC algorithms for the computations in the first and second layers of the filter in order to approximate $p(\boldsymbol{\theta}|\boldsymbol{y}_{1:n-1})$ and $p(\boldsymbol{x}_{0:n}|\boldsymbol{y}_{1:n-1}, \boldsymbol{\theta})$ respectively. In the third layer of computation we run a UKF [6] to obtain a Gaussian approximation of the form $p(\boldsymbol{z}_{hn}|\boldsymbol{x}_{0:n}, \boldsymbol{y}_{1:n-1}, \boldsymbol{\theta}) \approx \mathcal{N}(\boldsymbol{z}_{hn}|\hat{\boldsymbol{z}}_{hn}, \boldsymbol{C}_n(\boldsymbol{z}))$, where $\mathcal{N}(\boldsymbol{x}|\mu, \boldsymbol{C})$ denotes a Gaussian pdf with mean μ and covariance matrix $\boldsymbol{C}$. In the remaining of this section we outline the key steps of the three layers in the algorithm.

Algorithm 1 describes the first layer of the filter, which aims at the approximation of the posterior distribution of the parameters. It receives as inputs the prior distributions of the parameters, $p(\boldsymbol{\theta})$, and both state variables, $p(\boldsymbol{x}_0)$ and $p(\boldsymbol{z}_0)$. In the initialization step, they are used to generate the starting particles (for the SMC schemes) and sigma-points (for the UKF) needed at each layer. Specifically, we generate N parameter samples $\{\boldsymbol{\theta}_0^i\}_{1\leq i\leq N}$, J slow state particles per each parameter sample, $\{\boldsymbol{x}_0^{i,j}\}_{1\leq j\leq J}$, and $L+1$ ($L = 2d_z$) sigma-points of the fast state per each slow state sample, $\{\boldsymbol{z}_0^{i,j,l}\}_{0\leq l\leq L}$, to obtain a set of the form $\{\boldsymbol{\theta}_0^i, \{\boldsymbol{x}_0^{i,j}, \{\boldsymbol{z}_0^{i,j,l}\}_{0\leq l\leq L}\}_{1\leq j\leq J}\}_{1\leq i\leq N}$. Moreover, a Markov kernel $\kappa_N^{\boldsymbol{\theta}'}(d\boldsymbol{\theta})$ is needed for the jittering of parameter samples [5], i.e., to draw new particles, $\{\bar{\boldsymbol{\theta}}_n^i\}_{1\leq i\leq N}$, at each discrete-time step. Then, we compute the approximate likelihood $\hat{p}^{J,L}(\boldsymbol{y}_n|\boldsymbol{y}_{1:n-1}, \bar{\boldsymbol{\theta}}_n^i)$ in order to obtain the weights $\{\tilde{v}_n^i\}_{1\leq i\leq N}$. The states $\{\boldsymbol{x}_{n-1}^{i,j}, \{\boldsymbol{z}_{h(n-1)}^{i,j,l}\}_{0\leq l\leq L}\}_{1\leq j\leq J}$ are propagated to time n in the nested layers of filtering in step 2b. Finally, we normalize the weights in order to resample not only the parameter particles $\bar{\boldsymbol{\theta}}_n^i$, but also their associated sets of state variables.

Algorithm 2 describes the implementation of an SMC scheme in the second layer of the multi-scale nested filter, which works for a given sample $\bar{\boldsymbol{\theta}}_n^i$. In this layer we approximate the posterior pdf $p(\boldsymbol{x}_{0:n}|\boldsymbol{y}_{1:n}, \bar{\boldsymbol{\theta}}_n^i)$. The procedure is similar to the one in Algorithm 1, starting with the computation of the particle $\bar{\boldsymbol{x}}_n^{i,j}$, the approximate likelihood $\hat{p}^L(\boldsymbol{y}_n|\bar{\boldsymbol{x}}_n^{i,j}, \boldsymbol{x}_{0:n-1}^{i,j}, \boldsymbol{y}_{1:n-1}, \bar{\boldsymbol{\theta}}_n^i)$ and the weights $\{\tilde{u}_n^{i,j}\}_{1\leq j\leq J}$ in step 1a. In the third layer, the fast state variables are propagated and we obtain new samples $\{\boldsymbol{z}_{hn}^{i,j,l}\}_{0\leq l\leq L}$. Then, we can resample the set $\{\boldsymbol{x}_{0:n}^{i,j}, \{\boldsymbol{z}_{hn}^{i,j,l}\}_{0\leq l\leq L}\}_{1\leq j\leq J}$ with the normalized weights obtained in step 1c.

Algorithm 3 describes the UKF used in the third layer of the filter. In step 1, the algorithm generates new fast sigma-points $\tilde{\boldsymbol{z}}_q^{i,j,l}$, for $q = h(n-1)+1, \ldots, hn$ and $l = 0, \ldots, L$, conditional on the parameters $\bar{\boldsymbol{\theta}}_n^i$ and slow variables $\boldsymbol{x}_{n-1}^{i,j}$. Here, at each time step q, we compute the predictive mean and covariance matrix as

$$\check{\boldsymbol{z}}_q^{i,j} = \sum_{l=0}^{L} w^{i,j,l} \tilde{\boldsymbol{z}}_q^{i,j,l} \quad \text{and} \tag{17}$$

$$\check{\boldsymbol{C}}_q^{i,j}(\boldsymbol{z}) = \sum_{l=0}^{L} w^{i,j,l} (\tilde{\boldsymbol{z}}_q^{i,j,l} - \check{\boldsymbol{z}}_q^{i,j})(\tilde{\boldsymbol{z}}_q^{i,j,l} - \check{\boldsymbol{z}}_q^{i,j})^\top + \boldsymbol{W}, \tag{18}$$

Algorithm 1 SMC approximation of $p(\boldsymbol{\theta}|\boldsymbol{y}_{1:n})$.

Inputs

- Prior distributions $p(\boldsymbol{\theta})$, $p(\boldsymbol{x}_0)$ and $p(\boldsymbol{z}_0)$.
- A Markov kernel $\kappa_N^{\boldsymbol{\theta}'}(d\boldsymbol{\theta})$ which, given $\boldsymbol{\theta}'$, generates jittered parameters $\boldsymbol{\theta} \in \mathbb{R}^{d_\theta}$.

Initialization: this is a joint initialization for all three layers.

- Draw N i.i.d. sample $\boldsymbol{\theta}_0^i$, $i = 1, \ldots, N$ from the prior distribution $p(\boldsymbol{\theta})$.
- Draw J i.i.d. samples $\boldsymbol{x}_0^{i,j}$, $i = 1, \ldots, N$, $j = 1, \ldots, J$, from the prior distribution $p(\boldsymbol{x}_0)$.
- Draw $L+1$ sigma-points, $\boldsymbol{z}_0^{i,j,l}$, with their respective weights, $w^{i,j,l}$, $i = 1, \ldots, N$, $j = 1, \ldots, J$, $l = 0, \ldots, L$, from the prior distribution $p(\boldsymbol{z}_0|\hat{\boldsymbol{z}}_0, \boldsymbol{C}_0(\boldsymbol{z}))$ as

$$\begin{aligned} \boldsymbol{z}_0^{i,j,0} &= \hat{\boldsymbol{z}}_0, & w^{i,j,0} &= \frac{1}{1+d_z}, \\ \boldsymbol{z}_0^{i,j,l} &= \hat{\boldsymbol{z}}_0 + \boldsymbol{S}_l, & w^{i,j,l} &= \frac{1-w^{i,j,0}}{2d_z}, \\ \boldsymbol{z}_0^{i,j,l+d_z} &= \hat{\boldsymbol{z}}_0 - \boldsymbol{S}_l, & w^{i,j,l+d_z} &= \frac{1-w^{i,j,0}}{2d_z}, \end{aligned}$$

for $l = 1, \ldots, d_z$, being $\boldsymbol{S}_l$ the l-th row or column of the matrix square root of $\frac{d_z}{1-w^{i,j,0}}\boldsymbol{C}_0(\boldsymbol{z})$.

Procedure For $n \geq 0$:

1) Draw N i.i.d samples $\bar{\boldsymbol{\theta}}_n^i$ from $\kappa_N^{\boldsymbol{\theta}_{n-1}^i}(d\boldsymbol{\theta})$.
2) For $i = 1, \ldots, N$:
   a) Compute
   $$\tilde{v}_n^i = \hat{p}^{J,L}(\boldsymbol{y}_n|\boldsymbol{y}_{1:n-1}, \bar{\boldsymbol{\theta}}_n^i), \tag{11}$$
   where the approximate likelihood is evaluated at layer 2.
   b) Obtain new particles $\{\boldsymbol{x}_n^{i,j}, \{\boldsymbol{z}_{hn}^{i,j,l}\}_{0\leq l\leq L}\}_{1\leq j\leq J}$ at time n (from layers 2 and 3).
   c) Normalize the weights
   $$v_n^i = \frac{\tilde{v}_n^i}{\sum_{i=1}^N \tilde{v}_n^i}. \tag{12}$$
3) Resample: set for each $m = 1, \ldots, N$
   $$\begin{aligned} &\{\boldsymbol{\theta}_n^m, \{\boldsymbol{x}_n^{(m,j)}, \{\boldsymbol{z}_{hn}^{m,j,l}\}_{0\leq l\leq L}\}_{1\leq j\leq J}\} \\ &\quad = \{\bar{\boldsymbol{\theta}}_n^i, \{\boldsymbol{x}_n^{(i,j)}, \{\boldsymbol{z}_{hn}^{i,j,l}\}_{0\leq l\leq L}\}_{1\leq j\leq J}\} \end{aligned} \tag{13}$$
   with probability v_n^i.

Outputs: $\{\boldsymbol{\theta}_n^i, \{\boldsymbol{x}_n^{(i,j)}, \{\hat{\boldsymbol{z}}_{hn}^{i,j}, \boldsymbol{C}_n^{i,j}(\boldsymbol{z})\}\}_{1\leq j\leq J}\}_{1\leq i\leq N}$.

Algorithm 2 SMC approximation of $p(\boldsymbol{x}_{1:n}|\boldsymbol{y}_{1:n}, \boldsymbol{\theta})$.

Inputs

- Known parameter vector $\bar{\boldsymbol{\theta}}_n^i$ and known initial states, $\boldsymbol{x}_{0:n-1}^{i,j}$ and $\boldsymbol{z}_{0:h(n-1)}^{i,j,l}$, for $j = 1, \ldots, J$ and $l = 0, \ldots, L$.

Procedure For $n \geq 0$:

1) For $j = 1, \ldots, J$:
   a) Compute
   $$\tilde{u}_n^{i,j} = \hat{p}^L(\boldsymbol{y}_n|\bar{\boldsymbol{x}}_n^{i,j}, \boldsymbol{x}_{0:n-1}^{i,j}, \boldsymbol{y}_{1:n-1}, \bar{\boldsymbol{\theta}}_n^i), \tag{14}$$
   where the new particle $\bar{\boldsymbol{x}}_n^{i,j}$ is generated, and the approximate likelihood is evaluated, at layer 3.
   b) Obtain new particles $\{\boldsymbol{z}_{hn}^{i,j,l}\}_{0\leq l\leq L}$ at time n, from layer 3.
   c) Normalize the weights
   $$u_n^{i,j} = \frac{\tilde{u}_n^{i,j}}{\sum_{j=1}^J \tilde{u}_n^{i,j}}. \tag{15}$$
2) Resample: set
   $$\{\boldsymbol{x}_{0:n}^{i,m}, \{\boldsymbol{z}_{hn}^{i,m,l}\}_{0\leq l\leq L}\} = \{\bar{\boldsymbol{x}}_n^{i,j}, \boldsymbol{x}_{0:n-1}^{i,j}, \{\boldsymbol{z}_{hn}^{i,j,l}\}_{0\leq l\leq L}\} \tag{16}$$
   with probability $u_n^{i,j}$ for each $m = 1, \ldots, J$.

Outputs: $\{\boldsymbol{x}_{0:n}^{i,j}, \{\boldsymbol{z}_{hn}^{i,j,l}\}_{0\leq l\leq L}\}_{1\leq j\leq J}$ and $\{\tilde{u}_n^{i,j}\}_{1\leq j\leq J}$.

where $\boldsymbol{W} = \sigma_x^2 \boldsymbol{I}_{d_x}$ is the covariance matrix of the noise in the fast state equation. Then, we generate new sigma-points $\boldsymbol{z}_q^{i,j,l}$ from the normal pdf $\mathcal{N}(\check{\boldsymbol{z}}_q^{i,j}, \check{\boldsymbol{C}}_q^{i,j}(\boldsymbol{z}))$. Once we reach the time step $q = hn$, we use the new fast sigma-points to generate new slow particles for time n at step 2. To be specific, we project $\tilde{\boldsymbol{z}}_{hn}^{i,j,l}$ in the state equation of the slow state variables to obtain reference points $\{\tilde{\boldsymbol{x}}_n^{i,j,l}\}_{0\leq l\leq L}$. Then, we can generate a new slow particle from the normal distribution $\mathcal{N}(\check{\boldsymbol{x}}_n^{i,j}, \check{\boldsymbol{C}}_n^{i,j}(\boldsymbol{x}))$, where the mean and covariance matrix can be expressed as

$$\check{\boldsymbol{x}}_n^{i,j} = \sum_{l=0}^L w^{i,j,l} \tilde{\boldsymbol{x}}_n^{i,j,l} \quad \text{and} \tag{19}$$

$$\check{\boldsymbol{C}}_n^{i,j}(\boldsymbol{x}) = \sum_{l=0}^L w^{i,j,l} (\tilde{\boldsymbol{x}}_n^{i,j,l} - \check{\boldsymbol{x}}_n^{i,j})(\tilde{\boldsymbol{x}}_n^{i,j,l} - \check{\boldsymbol{x}}_n^{i,j})^\top + \boldsymbol{V}, \tag{20}$$

and $\boldsymbol{V} = \sigma_x^2 \boldsymbol{I}_{d_x}$ is the covariance matrix of the noise in the slow state equation. Next, we can compute exactly the (non-normalized) weight $\tilde{w}_n^{i,j,l} = p(\boldsymbol{y}_n|\tilde{\boldsymbol{z}}_{hn}^{i,j,l}, \bar{\boldsymbol{x}}_n^{i,j}, \bar{\boldsymbol{\theta}}_n^i)$ in step 3 and propagate the fast sigma-points and the particle $\bar{\boldsymbol{x}}_n^{i,j}$ through the observation function to obtain projected sigma-points (on the observation space) $\{\tilde{\boldsymbol{y}}_n^{i,j,l}\}_{0\leq l\leq L}$. We compute

the observational mean and covariance matrix as

$$\hat{\boldsymbol{y}}_n^{i,j} = \sum_{l=0}^{L} w^{i,j,l} \tilde{\boldsymbol{y}}_n^{i,j,l} \quad \text{and} \tag{21}$$

$$\boldsymbol{C}_n^{i,j}(\boldsymbol{y}) = \sum_{l=0}^{L} w^{i,j,l} (\tilde{\boldsymbol{y}}_n^{i,j,l} - \hat{\boldsymbol{y}}_n^{i,j})(\tilde{\boldsymbol{y}}_n^{i,j,l} - \hat{\boldsymbol{y}}_n^{i,j})^\top + \boldsymbol{R}, \tag{22}$$

where $\boldsymbol{R}$ is the covariance matrix of the noise in the observation equation. Finally, in step 4, we compute the Kalman gain using the observational covariance matrix of (22) and the cross-covariance matrix

$$\boldsymbol{C}_n^{i,j}(\boldsymbol{z}, \boldsymbol{y}) = \sum_{l=0}^{L} w^{i,j,l} (\tilde{\boldsymbol{z}}_q^{i,j,l} - \check{\boldsymbol{z}}_q^{i,j})(\tilde{\boldsymbol{y}}_n^{i,j,l} - \hat{\boldsymbol{y}}_n^{i,j})^\top, \tag{23}$$

and update the mean, $\hat{\boldsymbol{z}}_{hn}^{i,j}$, and covariance matrix, $\boldsymbol{C}_{hn}^{i,j}(\boldsymbol{z})$ of the fast variables. Finally, we generate new sigma-points in step 5.

Given the weights $\tilde{w}_n^{i,j,l}$, $l = 0, \ldots, L$ in step 3 of Algorithm 3, we can approximate the different likelihoods we need in the other layers. In Algorithm 2 we approximate $\tilde{u}_n^{i,j} = \hat{p}^L(\boldsymbol{y}_n | \boldsymbol{x}_{0:n}^{i,j}, \boldsymbol{y}_{1:n-1}, \bar{\boldsymbol{\theta}}_n^i)$ as

$$\tilde{u}_n^{i,j} = \sum_{l=0}^{L} w^{i,j,l} \tilde{w}_n^{i,j,l}. \tag{24}$$

Similarly, the approximation of $\tilde{v}_n^i = \hat{p}^{J,L}(\boldsymbol{y}_n | \boldsymbol{y}_{1:n-1}, \bar{\boldsymbol{\theta}}_n^i)$ is computed as

$$\tilde{v}_n^i = \frac{1}{J} \sum_{j=1}^{J} \tilde{u}_n^{i,j}. \tag{25}$$

IV. Example

A. Stochastic two-scale Lorenz 96 model

In order to illustrate the application of the method described in Section III, we consider a stochastic version of the two-scale Lorenz 96 model [10], which depends on a set of fixed parameters, a set of fast variables and a set of slow variables. The slow variables are represented by a d_x-dimensional vector, $\boldsymbol{x}$, while the fast variables, $\boldsymbol{z}$, are d_z-dimensional and $d_z > d_x$. The system is described, in continuous-time t, by the stochastic differential equations (SDEs)

$$dx_j = \Big[-x_{j-1}(x_{j-2} - x_{j+1}) - x_j + F - \frac{HC}{B} \sum_{l=(j-1)L}^{Lj-1} z_l \Big] dt + \sigma_x dv_j, \tag{32}$$

$$dz_l = \Big[-CBz_{l+1}(z_{l+2} - z_{l-1}) - Cz_l + \frac{CF}{B} + \frac{HC}{B} x_{\lfloor (l-1)L \rfloor} \Big] dt + \sigma_z dw_l, \tag{33}$$

where $j = 1, \ldots, d_x$, $l = 1, \ldots, d_z$; $\boldsymbol{v}$ and $\boldsymbol{w}$ are, respectively, d_x and d_z-dimensional vectors of independent standard Wiener processes; $\sigma_x > 0$ and $\sigma_z > 0$ are known scale parameters and

Algorithm 3 UKF approximation of $p(\boldsymbol{z}_{hn} | \boldsymbol{x}_{1:n}, \boldsymbol{y}_{1:n}, \boldsymbol{\theta})$.

Inputs

- Known parameter vector $\bar{\boldsymbol{\theta}}_n^i$ and initial states, $\boldsymbol{x}_{0:n-1}^{i,j}$ and $\boldsymbol{z}_{h(n-1)}^{i,j,l}$, for $l = 0, \ldots, L$.
- Integration steps Δ_x, Δ_z and time scale ratio $h = \frac{\Delta_x}{\Delta_z} \in \mathbb{Z}^+$.

Procedure For $n \geq 0$:

1) For $l = 0, \ldots, L$ and for $q = h(n-1)+1, ..., hn$,
   a) Integrate with step Δ_z
   $$\tilde{\boldsymbol{z}}_q^{i,j,l} = \boldsymbol{z}_{q-1}^{i,j,l} + \Delta_z (f_{\boldsymbol{z}}(\boldsymbol{z}_{q-1}^{i,j,l}, \bar{\boldsymbol{\theta}}_n^i) + g_{\boldsymbol{z}}(\boldsymbol{x}_{n-1}^{i,j}, \bar{\boldsymbol{\theta}}_n^i)), \tag{26}$$
   and compute the predictive mean, $\check{\boldsymbol{z}}_q^{i,j}$, and the predictive covariance matrix, $\check{\boldsymbol{C}}_q^{i,j}(\boldsymbol{z})$, of (17) and (18).
   b) Compute $L+1$ new sigma points $\{\boldsymbol{z}_q^{i,j,l}\}_{0 \leq l \leq L}$ from the predictive pdf $p(\boldsymbol{z}_q | \boldsymbol{x}_{0:n-1}, \boldsymbol{y}_{1:n-1}, \bar{\boldsymbol{\theta}}_n^i) \approx \mathcal{N}(\check{\boldsymbol{z}}_q^{i,j}, \check{\boldsymbol{C}}_q^{i,j}(\boldsymbol{z}))$.
2) In the space of the slow state variables:
   a) For $l = 0, \ldots, L$, propagate the sigma-points $\tilde{\boldsymbol{z}}_{h(n-1):hn}^{i,j,l}$, integrating with step Δ_x,
   $$\tilde{\boldsymbol{x}}_n^{i,j,l} = \boldsymbol{x}_{n-1}^{i,j} + \Delta_x (f_{\boldsymbol{x}}(\boldsymbol{x}_{n-1}^{i,j}, \bar{\boldsymbol{\theta}}_n^i) + g_{\boldsymbol{x}}(\bar{\boldsymbol{z}}_n^{i,j,l}, \bar{\boldsymbol{\theta}}_n^i)), \tag{27}$$
   where $\bar{\boldsymbol{z}}_n^{i,j,l} = \frac{1}{h} \sum_{q=h(n-1)}^{hn} \tilde{\boldsymbol{z}}_q^{i,j,l}$. Then, compute the mean $\check{\boldsymbol{x}}_n^{i,j}$ and the covariance matrix $\check{\boldsymbol{C}}_n^{i,j}(\boldsymbol{x})$, of (19) and (20).
   b) Sample $\bar{\boldsymbol{x}}_n^{i,j} \sim \mathcal{N}(\check{\boldsymbol{x}}_n^{i,j}, \check{\boldsymbol{C}}_n^{i,j}(\boldsymbol{x}))$.
3) Once we get a new observation $\boldsymbol{y}_n$,
   a) For $l = 0, \ldots, L$, propagate the sigma-points through the observation function
   $$\tilde{\boldsymbol{y}}_n^{i,j,l} = l(\tilde{\boldsymbol{z}}_{hn}^{i,j,l}, \bar{\boldsymbol{x}}_n^{i,j}, \bar{\boldsymbol{\theta}}_n^i), \tag{28}$$
   and compute the mean, $\hat{\boldsymbol{y}}_{hn}^{i,j}$, and the covariance matrix, $\boldsymbol{C}_n^{i,j}(\boldsymbol{y})$, of (21) and (22).
   b) Compute $\tilde{w}_n^{i,j,l} = p(\boldsymbol{y}_n | \tilde{\boldsymbol{z}}_{hn}^{i,j,l}, \bar{\boldsymbol{x}}_n^{i,j}, \bar{\boldsymbol{\theta}}_n^i)$.
4) Update the mean and the covariance matrix of the fast variables
   $$\boldsymbol{K}_n = \boldsymbol{C}_n^{i,j}(\boldsymbol{z}, \boldsymbol{y}) \big(\boldsymbol{C}_n^{i,j}(\boldsymbol{y}) \big)^{-1}, \tag{29}$$
   $$\hat{\boldsymbol{z}}_{hn}^{i,j} = \check{\boldsymbol{z}}_{hn}^{i,j} + \boldsymbol{K}_n (\hat{\boldsymbol{y}}_n^{i,j} - \boldsymbol{y}_n) \quad \text{and} \tag{30}$$
   $$\boldsymbol{C}_{hn}^{i,j}(\boldsymbol{z}) = \check{\boldsymbol{C}}_{hn}^{i,j}(\boldsymbol{z}) + \boldsymbol{K}_n \boldsymbol{C}_n^{i,j}(\boldsymbol{y}) \boldsymbol{K}_n^\top, \tag{31}$$
   where $\boldsymbol{C}_n^{i,j}(\boldsymbol{z}, \boldsymbol{y})$ is the cross-covariance matrix between $\tilde{\boldsymbol{z}}_{hn}^{i,j,l}$ and $\tilde{\boldsymbol{y}}_{hn}^{i,j,l}$, $l = 0, \ldots, L$, in (23).
5) From the new pdf $p(\boldsymbol{z}_{hn} | \boldsymbol{x}_{0:n}, \boldsymbol{y}_{1:n}, \bar{\boldsymbol{\theta}}^i) = \mathcal{N}(\boldsymbol{z}_{hn} | \hat{\boldsymbol{z}}_{hn}^{i,j}, \boldsymbol{C}_n^{i,j}(\boldsymbol{z}))$, generate $2d_z + 1$ sigma-points and weights $\{\boldsymbol{z}_{hn}^{i,j,l}, w^{i,j,l}\}_{0 \leq l \leq L}$.

Outputs: $\{\boldsymbol{z}_{hn}^{i,j,l}\}_{0 \leq l \leq L}$, $\bar{\boldsymbol{x}}_n^{i,j}$ and $\{\tilde{w}_n^{i,j,l}\}_{0 \leq l \leq L}$.

$\boldsymbol{\alpha} = (F, H, C, B)^\top \in \mathbb{R}$ are static model parameters. Using a micro-macro solver [7], [8] that runs an Euler-Maruyama scheme at each time-scale to integrate (32)–(33), the discrete-time state equation can be written as

$$\begin{aligned}\boldsymbol{x}_{n+1,j} =& \boldsymbol{x}_{n,j} + \Delta_x(f_{\boldsymbol{x},j}(\boldsymbol{x}_n, \boldsymbol{\alpha}) + g_{\boldsymbol{x},j}(\bar{\boldsymbol{z}}_{n+1}, \boldsymbol{\alpha})) \\ &+ \sqrt{\Delta_x}\sigma_x \boldsymbol{v}_{n+1,j}, \qquad (34)\\ \boldsymbol{z}_{k+1,l} =& \boldsymbol{z}_{k,l} + \Delta_z(f_{\boldsymbol{z},l}(\boldsymbol{x}_{\lfloor \frac{k}{h} \rfloor}, \boldsymbol{\alpha}) + g_{\boldsymbol{z},l}(\boldsymbol{z}_k, \boldsymbol{\alpha})) \\ &+ \sqrt{\Delta_z}\sigma_z \boldsymbol{w}_{k+1,l}, \qquad (35)\end{aligned}$$

where functions $f_{\boldsymbol{x},j} : \mathbb{R}^{d_x} \times \mathbb{R}^{d_\alpha} \to \mathbb{R}^{d_x}$, $g_{\boldsymbol{x},j} : \mathbb{R}^{d_z} \times \mathbb{R}^{d_\alpha} \to \mathbb{R}^{d_x}$, $f_{\boldsymbol{z},l} : \mathbb{R}^{d_x} \times \mathbb{R}^{d_\alpha} \to \mathbb{R}^{d_z}$ and $g_{\boldsymbol{z},l} : \mathbb{R}^{d_z} \times \mathbb{R}^{d_\alpha} \to \mathbb{R}^{d_z}$ can be expressed as

$$\begin{aligned} f_{\boldsymbol{x},j}(\boldsymbol{x}, \boldsymbol{\alpha}) &= -x_{j-1}(x_{j-2} - x_{j+1}) - x_j + F, \\ g_{\boldsymbol{x},j}(\boldsymbol{z}, \boldsymbol{\alpha}) &= -\frac{HC}{B} \sum_{l=(j-1)L}^{Lj-1} z_l, \\ f_{\boldsymbol{z},l}(\boldsymbol{x}, \boldsymbol{\alpha}) &= \frac{HC}{B} x_{\lfloor (l-1)L \rfloor} \quad \text{and} \\ g_{\boldsymbol{z},l}(\boldsymbol{z}, \boldsymbol{\alpha}) &= -CBz_{l+1}(z_{l+2} - z_{l-1}) - Cz_l + \frac{CF}{B}; \end{aligned}$$

$\boldsymbol{v}_n$ and $\boldsymbol{w}_n$ are sequences of independent Gaussian random vectors of dimension d_x and d_z, with zero mean and covariance matrices $\sigma_x^2 \boldsymbol{I}_{d_x}$ and $\sigma_z^2 \boldsymbol{I}_{d_z}$, respectively.

We assume a linear observation equation of the form

$$\boldsymbol{y}_n = \begin{bmatrix} \boldsymbol{x}_n \\ \boldsymbol{z}_{hn} \end{bmatrix} + \boldsymbol{r}_n, \qquad (36)$$

where $\boldsymbol{r}_n$ is a d_y-dimensional Gaussian random vector with known covariance matrix

$$\boldsymbol{R} = \begin{bmatrix} \sigma_{\boldsymbol{y},\boldsymbol{x}} \boldsymbol{I}_{d_{y,x}} & \boldsymbol{0} \\ \boldsymbol{0} & \sigma_{\boldsymbol{y},\boldsymbol{z}} \boldsymbol{I}_{d_{y,z}} \end{bmatrix}, \qquad (37)$$

where $\sigma_{\boldsymbol{y},\boldsymbol{x}}, \sigma_{\boldsymbol{y},\boldsymbol{z}} > 0$.

B. Numerical results

We have run simulations for the two-scale Lorenz 96 model of Section IV-A, with dimensions $d_x = 10$ and $d_z = 50$. The time steps for the Euler-Maruyama integrators are $\Delta_x = 10^{-3}$ and $\Delta_z = 10^{-4}$ continuous-time units. We set the fixed parameters as $F = 8$, $H = 0.75$, $C = 10$ and $B = 15$. In order to obtain the inital states $\boldsymbol{x}_0$ and $\boldsymbol{z}_0$, we simulate a deterministic version of (34)–(35) ($\sigma_x = \sigma_z = 0$) for 20 continuous-time units. We set the initial states as the values of variables $\boldsymbol{x}$ and $\boldsymbol{z}$ at the last time step of this simulation. This initialization is used in all simulations of this computer experiment in order to generate both "ground truth" sequences of $\boldsymbol{x}_n$ and $\boldsymbol{z}_k$ and the associated sequences of observations $\boldsymbol{y}_n$.

We assume that parameters H and B are known, while we need to estimate $\boldsymbol{\theta} = [F, C]^\top$, with dimension $d_\theta = 2$. The prior for the unknown parameters is uniform, namely $p(\boldsymbol{\theta}) = \mathcal{U}([2, 20]^2)$, while the priors used in the filtering algorithm for both unknown state variables are Gaussian, namely $p(\boldsymbol{x}_0) = \mathcal{N}(\boldsymbol{x}_0, 0.1\boldsymbol{I}_{d_x})$ and $p(\boldsymbol{z}_0) = \mathcal{N}(\boldsymbol{z}_0, 10\boldsymbol{I}_{d_z})$. The noise scaling factors, $\sigma_x = \frac{1}{2}$, $\sigma_z = \frac{1}{16}$, $\sigma_{y,x} = 10^{-1}$ and $\sigma_{y,z} = 10^{-3}$, are known. We run the multi-scale hybrid filter with $N = 40$ and $J = 100$ particles in the SMC schemes of the first and second layers of the algorithm. In the third layer, we have $2d_z + 1 = 101$ sigma-points for the UKF. The jittering kernel is $\kappa_N^{\boldsymbol{\theta}'}(d\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}|\boldsymbol{\theta}', \tilde{\sigma}^2 \boldsymbol{I}_{d_\theta})$, where $\tilde{\sigma}^2 = \frac{0.05}{\sqrt{N^3}}$.

We assess the accuracy of the algorithm in terms of the normalized mean square error (NMSE) of the estimators of the parameters, the slow state variables and the fast state variables. In the plots, we show the NMSE computed at time n,

$$\text{NMSE}_{\theta,n} = \frac{\| \boldsymbol{\theta}_n - \hat{\boldsymbol{\theta}}_n \|^2}{\| \boldsymbol{\theta}_n \|^2}, \qquad (38)$$

$$\text{NMSE}_{\boldsymbol{x},n} = \frac{\| \boldsymbol{x}_n - \hat{\boldsymbol{x}}_n \|^2}{\| \boldsymbol{x}_n \|^2}, \qquad (39)$$

$$\text{NMSE}_{\boldsymbol{z},n} = \frac{\| \boldsymbol{z}_{hn} - \hat{\boldsymbol{z}}_{hn} \|^2}{\| \boldsymbol{z}_{hn} \|^2}, \qquad (40)$$

averaged over 50 independent simulation runs of 20 continuous-time units each, where the estimators take the form

$$\hat{\boldsymbol{\theta}}_n = \sum_{i=1}^N v_n^i \bar{\boldsymbol{\theta}}^i, \qquad (41)$$

$$\hat{\boldsymbol{x}}_n = \sum_{i=1}^N \sum_{j=1}^J v_n^i u_n^{i,j} \bar{\boldsymbol{x}}_n^{i,j} \quad \text{and} \qquad (42)$$

$$\hat{\boldsymbol{z}}_{hn} = \sum_{i=1}^N \sum_{j=1}^J v_n^i u_n^{i,j} \hat{\boldsymbol{z}}_{hn}^{i,j}. \qquad (43)$$

Figure 1 shows the NMSE over time for the parameters and both state variables. NMSE_θ typically stabilizes after 5 continuous-time units of simulation, and it attains values of the order of 10^{-3}. On the other side, $\text{NMSE}_{\boldsymbol{x}}$ and $\text{NMSE}_{\boldsymbol{z}}$ converge more quickly and they yield values around 10^{-3} and 10^{-2} respectively.

Figure 2 displays the true values of $\boldsymbol{x}_{1,n}$ and $\boldsymbol{z}_{1,hn}$ for a typical simulation run, together with their estimators, during the last two continuous-time unit of the simulation for better visibility (although, the accuracy obtained is similar during the whole experiment).

Figure 3 shows the estimated posterior pdf of the parameter F (plot 3a), the parameter C (plot 3b), the first dimension of $\boldsymbol{x}$ (plot 3c) and the first dimension of $\boldsymbol{z}$ (plot 3d) at the last discrete-time step of a single simulation run, together with their true values. We can see in all these figures that the actual values of the parameters and signals are placed in a high-probability region of the density functions computed by the multi-scale nested filter.

V. Conclusions

We have introduced a new methodology for tracking the time evolution and evaluate any static parameters of multi-scale systems. It is a generalization of NHF [1], where the joint probability distribution of the parameters and state dynamic variables (i.e., a multi-scale system of 2 time scales) is approximated using two layers of nested filters given a sequence of partial and noisy observations.. Although this recursive scheme

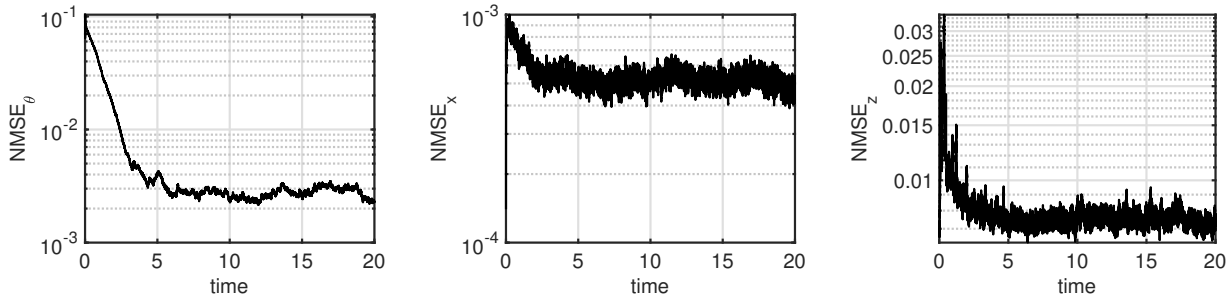
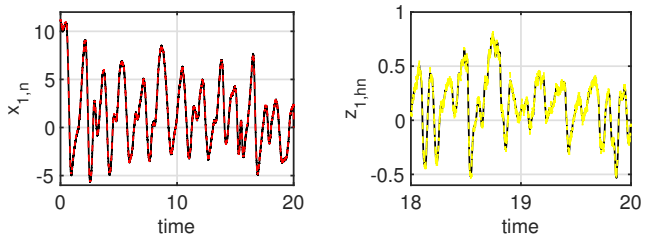
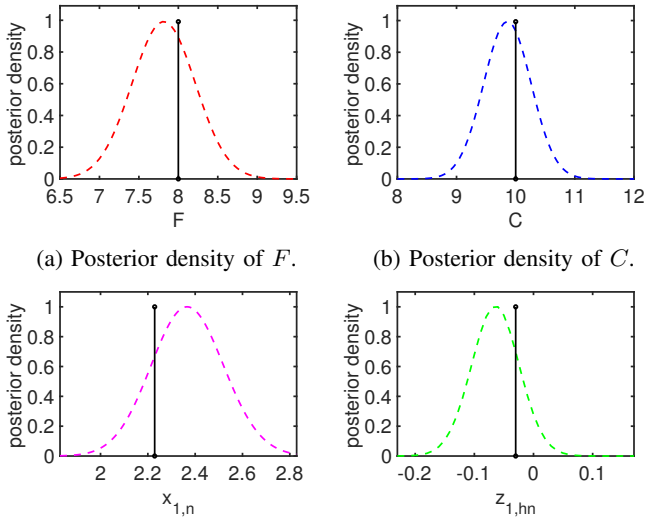


Fig. 1: NMSE of the parameters and both the slow and fast state variables over time.



(a) $\boldsymbol{x}_{1,n}$ over time. (b) $\boldsymbol{z}_{1,hn}$ over time.

Fig. 2: Sequences of the first dimension of the true values of $\boldsymbol{x}$ (2a) and $\boldsymbol{z}$ (2b) over time in color black, and their estimates in red and yellow, respectively.



(a) Posterior density of F. (b) Posterior density of C.

(c) Posterior density of $\boldsymbol{x}_{1,n}$. (d) Posterior density of $\boldsymbol{z}_{1,hn}$.

Fig. 3: Approximate posterior density functions of $\boldsymbol{\theta}$, $\boldsymbol{x}_{1,n}$ and $\boldsymbol{z}_{1,hn}$ at the last discrete-time step of a simulation. The pdf's for F, C, $\boldsymbol{x}_{1,n}$ and $\boldsymbol{z}_{1:hn}$ are kernel density estimators obtained from the Monte Carlo samples generated by the nested filters.

can tackle Bayesian inference for general homogeneous multi-scale systems (n different time scales), we have analyzed a dynamical system of 3 time-scales (static parameters, slow dynamic state variables and fast dynamic state variables). We have explored the use of SMC schemes in both first and second layers in order to approximate respectively the posterior probability distribution of the parameters and the posterior probability distribution of the slow state variables. In the third layer, we obtain a Gaussian approximation of the posterior probability distribution of the fast variables using UKF. Therefore, the computational cost of the algorithm increases with NJd_z. With this scheme, we have presented results for a stochastic two-scale Lorenz 96 model.

REFERENCES

[1] Pérez-Vieites, Sara, Inés P. Mariño, and Joaquín Míguez. "Probabilistic scheme for joint parameter estimation and state prediction in complex dynamical systems." Physical Review E 98.6 (2018): 063305.

[2] Weinan, E., and Bjorn Engquist. "Multiscale modeling and computation." Notices of the AMS 50.9 (2003): 1062-1070.

[3] Chopin, Nicolas, Pierre E. Jacob, and Omiros Papaspiliopoulos. "SMC2: an efficient algorithm for sequential analysis of state space models." Journal of the Royal Statistical Society: Series B (Statistical Methodology) 75.3 (2013): 397-426.

[4] Andrieu, Christophe, Arnaud Doucet, and Roman Holenstein. "Particle Markov chain Monte Carlo methods." Journal of the Royal Statistical Society: Series B (Statistical Methodology) 72.3 (2010): 269-342.

[5] Crisan, Dan, and Joaquín Míguez. "Nested particle filters for online parameter estimation in discrete-time state-space Markov models." Bernoulli 24.4A (2018): 3039-3086.

[6] Julier, Simon, Jeffrey Uhlmann, and Hugh F. Durrant-Whyte. "A new method for the nonlinear transformation of means and covariances in filters and estimators." IEEE Transactions on Automatic Control 45.3 (2000): 477-482.

[7] Vanden-Eijnden, Eric. "Fast communications: Numerical techniques for multi-scale dynamical systems with stochastic effects." Communications in Mathematical Sciences 1.2 (2003): 385-391.

[8] Weinan, E., Di Liu, and Eric VandenEijnden. "Analysis of multiscale methods for stochastic differential equations." Communications on Pure and Applied Mathematics 58.11 (2005): 1544-1585.

[9] Míguez, Joaquín, Dan Crisan, and Petar M. Djurić. "On the convergence of two sequential Monte Carlo methods for maximum a posteriori sequence estimation and stochastic global optimization." Statistics and Computing 23.1 (2013): 91-107.

[10] Arnold, Hannah Mary. Stochastic Parametrisation and Model Uncertainty. Diss. University of Oxford, 2013.