Kalman-based nested hybrid filters for recursive inference in state-space 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*—We introduce a new sequential methodology to calibrate the fixed parameters and trace the stochastic dynamical variables of a state-space system. The proposed framework is based on the nested hybrid filters (NHF) of [1], that combine two layers of filters, one inside the other, to compute the joint posterior probability distribution of the static parameters and the state variables. In particular, we explore the use of deterministic sampling techniques in the first layer of the algorithm, instead of Monte Carlo methods, which reduces computational cost and so makes the algorithms potentially better-suited for high-dimensional state and parameter spaces. We present numerical results for a stochastic Lorenz 63 model.**

***Index Terms*—filtering, Kalman, Monte Carlo, Bayesian inference**

I. Introduction

We address the problem of tracking the time evolution of state-space dynamical systems, whose behaviour relies on a set of unknown fixed parameters. Then, both time-varying states and static parameters need to be estimated, and for this purpose long observation sequences are collected.

Generally, jointly carrying out Bayesian model calibration (parameter estimation) and filtering or data assimilation (state tracking) poses several practical and theoretical difficulties. Major breakthroughs have been attained in the last few years, including methods such as sequential Monte Carlo square (SMC^2) [2] or particle Markov chain Monte Carlo (PMCMC) [3]. They aim at computing the posterior probability distribution of all the unknown variables and parameters of the system. However, these are batch techniques, i.e., the whole sequence of observations may have to be processed every time a new observation arrives in order to obtain estimates. Nested particle filters (NPFs) [4] apply the same principles as SMC^2, a scheme of two interwined layers of Monte Carlo methods to estimate parameters and states, respectively, but NPFs are purely recursive. This methodology is better suited for long sequences of observations, although its computational cost is still prohibitive in high-dimensional problems as it uses Monte Carlo in both layers of filters. For these settings, nested hybrid filters (NHF's) [1] are more appealing, since they introduce Gaussian filtering techniques in the second layer of the algorithm, which reduces the computational cost.

In this paper, we introduce a further generalization of the NHF methodology by describing how to apply non-Monte Carlo methods in the first layer of the algorithm. Besides the additional reduction in computational cost, the new scheme allows the combination of virtually any type of Gaussian or particle filter in any of the two layers of the nested structure. To be specific, we show in detail how to obtain an NHF that employs a deterministic-sampling Gaussian approximation (such as the cubature Kalman filter (CKF) [5] or the unscented Kalman filter (UKF) [6]) in the first (parameter) layer with an extended Kalman filter (EKF) in the second (state) layer.

We state the problem to be addressed in Section II. In Section III, we describe the new methodology and how it can be made to work recursively. Some numerical results for the stochastic Lorenz 63 model are shown in Section IV and conclusions are drawn in Section V.

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).

II. Problem Statement

A. State space models

We are interested in Markov state-space dynamic systems that evolve in discrete-time and can be described as

$$\boldsymbol{X}_0 \sim p(\boldsymbol{x}_0), \tag{1}$$

$$\Theta \sim p(\boldsymbol{\theta}), \tag{2}$$

$$\boldsymbol{X}_t \sim p(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}, \boldsymbol{\theta}), \tag{3}$$

$$\boldsymbol{Y}_t \sim p(\boldsymbol{y}_t|\boldsymbol{x}_t, \boldsymbol{\theta}), \tag{4}$$

where

- $t \in \mathbb{N}$ denotes discrete time;
- $\boldsymbol{X}_t$ is the d_x-dimensional (random) state vector at time t, taking values in the state space $\mathcal{X} \subseteq \mathbb{R}^{d_x}$;
- Θ is the d_θ-dimensional vector of fixed parameters;
- $p(\boldsymbol{\theta})$ and $p(\boldsymbol{x}_0)$ are the a *priori* pdfs of the parameters and the state;
- $p(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}, \boldsymbol{\theta})$ is the conditional density of the state $\boldsymbol{X}_t$ given $\boldsymbol{X}_{t-1} = \boldsymbol{x}_{t-1}$ and the parameter vector $\Theta = \boldsymbol{\theta}$;
- $\boldsymbol{Y}_t$ is the d_y-dimensional observation vector at time t, taking values in the observation space $\mathcal{Y} \subseteq \mathbb{R}^{d_y}$. We assume $\boldsymbol{Y}_t$ is conditionally independent of all other observations given $\boldsymbol{X}_t$ and Θ;

- $p(\boldsymbol{y}_t|\boldsymbol{x}_t,\boldsymbol{\theta})$ is the conditional pdf of $\boldsymbol{Y}_t$ given $\boldsymbol{X}_t=\boldsymbol{x}_t$ and $\Theta=\boldsymbol{\theta}$.

A broad class of systems can be described by the model in (1)–(4), both linear and nonlinear, with Gaussian or non-Gaussian perturbations. We assume that the prior distributions of the state, $p(\boldsymbol{x}_0)$, and the parameters, $p(\boldsymbol{\theta})$, are known, and we aim at estimating both Θ and $\boldsymbol{X}_t$ recursively.

B. Model inference

The key difficulty of this problem is the Bayesian estimation of the parameters in vector Θ, since their calibration is necessary to track the state variables and, predict the evolution of the system. From the viewpoint of Bayesian analysis, we want to compute the posterior pdf $p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t})$ as it contains all the relevant information for the estimation task at discrete time t. However, this pdf can be written as

$$p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t})=\int p(\boldsymbol{\theta},\boldsymbol{x}_t|\boldsymbol{y}_{1:t})d\boldsymbol{x}_t, \tag{5}$$

leading naturally to approximations for $p(\boldsymbol{\theta},\boldsymbol{x}_t|\boldsymbol{y}_{1:t})$ for each t. This means that we not only estimate the parameter vector Θ, but we also implicitly track the state dynamical variables. Then, the aim of this paper is to obtain a Gaussian approximation of $p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t})$ within a nested Gaussian filtering scheme, whose second layer of filters will provide, in addition, Gaussian approximations for $p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t},\boldsymbol{\theta})$.

III. Nested Gaussian filters

In this section, we introduce a class of NHF's with Gaussian filters in both layers. We outline the methodology used to obtain the Gaussian approximations of $p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t})$ and $p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t},\boldsymbol{\theta})$.

A. Sequential Gaussian approximation

Assuming $p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t-1})$ is Gaussian, we aim at computing expectations of the form $\int f(\boldsymbol{\theta})p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t})d\boldsymbol{\theta}$ for any function of the parameters, $f(\boldsymbol{\theta})$. Using Bayes' rule, we have

$$p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t})=\frac{p(\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1},\boldsymbol{\theta})}{p(\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1})}\times p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t-1}), \tag{6}$$

hence, we can rewrite the integral as

$$\int f(\boldsymbol{\theta})p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t})d\boldsymbol{\theta}=\int \psi(\boldsymbol{\theta})p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t-1})d\boldsymbol{\theta}, \tag{7}$$

where $\psi(\boldsymbol{\theta})$ can be expressed as

$$\psi(\boldsymbol{\theta})=\frac{f(\boldsymbol{\theta})p(\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1},\boldsymbol{\theta})}{p(\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1})}. \tag{8}$$

As we assume that $p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t-1})$ is Gaussian, we can describe (7) using cubature rules [5] or the unscented transform (UT) [6]. Then, $p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t-1})$ can be represented at time t by a set of reference points and weights, $\{\boldsymbol{\theta}_t^i,w_t^i\}$, $i=1,\dots,M$, that we can use to approximate the integral (7) as

$$\int \psi(\boldsymbol{\theta})p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t-1})d\boldsymbol{\theta}\simeq\sum_{i=1}^{M}\psi(\boldsymbol{\theta}_t^i)w_t^i. \tag{9}$$

On the other hand, the pdf in the denominator of expression (8), $p(\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1})$, can be written as

$$p(\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1})=\int p(\boldsymbol{y}_t,\boldsymbol{\theta}|\boldsymbol{y}_{1:t-1})d\boldsymbol{\theta}, \tag{10}$$

where the joint pdf of $\boldsymbol{Y}_t$ and Θ given all previous observations can be decomposed as

$$p(\boldsymbol{y}_t,\boldsymbol{\theta}|\boldsymbol{y}_{1:t-1})=p(\boldsymbol{y}_t|\boldsymbol{\theta},\boldsymbol{y}_{1:t-1})p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t-1}). \tag{11}$$

Then, the integral in (10) can also be approximated using the same reference points and weights as

$$p(\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1})\simeq\sum_{i=1}^{M}p(\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1},\boldsymbol{\theta}_t^i)w_t^i. \tag{12}$$

Finally, we can calculate the pdfs $p(\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1},\boldsymbol{\theta}_t^i)$, $i=1,\dots,M$, using a bank of M Gaussian filters placed in the second layer of the nested filter. Once they are computed, we can approximate $p(\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1})$ as in (12).

The argument above enables us to approximate any integral $\int f(\boldsymbol{\theta})p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t})d\boldsymbol{\theta}$. In particular, we can compute the mean vector and covariance matrix of $p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t})$ by taking $f(\boldsymbol{\theta})=\boldsymbol{\theta}$ and $f(\boldsymbol{\theta})=(\boldsymbol{\theta}-\hat{\boldsymbol{\theta}}_t)(\boldsymbol{\theta}-\hat{\boldsymbol{\theta}}_t)^\top$, respectively, where

$$\hat{\boldsymbol{\theta}}_t=\int \boldsymbol{\theta}p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t})d\boldsymbol{\theta}. \tag{13}$$

Specifically, we obtain the formulation for approximating the mean parameter vector, $\hat{\boldsymbol{\theta}}_t$, and its covariance matrix, $\hat{\boldsymbol{C}}_t^{\boldsymbol{\theta}}$, sequentially as

$$\hat{\boldsymbol{\theta}}_t\simeq\sum_{i=1}^{M}\boldsymbol{\theta}_t^i\frac{p(\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1},\boldsymbol{\theta}_t^i)}{p(\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1})}w_t^i \quad \text{and} \tag{14}$$

$$\hat{\boldsymbol{C}}_t^{\boldsymbol{\theta}}\simeq\sum_{i=1}^{M}(\boldsymbol{\theta}_t^i-\hat{\boldsymbol{\theta}}_t)(\boldsymbol{\theta}_t^i-\hat{\boldsymbol{\theta}}_t)^\top\frac{p(\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1},\boldsymbol{\theta}_t^i)}{p(\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1})}w_t^i. \tag{15}$$

We outline the procedure for the sequential computation of (the Gaussian approximations of) $p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t})$, $t=1,2,\dots$, in Algorithm 1. The calculations done in the second layer of filters are summarized in step 2a. Notice that, at any time $t=n\geq 1$, we update the reference points $\boldsymbol{\theta}_t^i$, $i=1,\dots,M$, and, therefore, we need to run the M Gaussian filters in the second layer from scratch (i.e., from $t=0$ to $t=n$) in order to evaluate the densities $p(\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1},\boldsymbol{\theta}_t^i)$. Thus, Algorithm 1 is sequential but not recursive and, as a consequence, not well suited to handle long sequences of observations.

B. Recursive algorithm

For every new observation, the pdf's $p(\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1},\boldsymbol{\theta}_t^i)$ are computed by running the nested filters from time 0 until the current time t, which makes the computational cost increase with t^2.

However, the entries of the covariance matrix, $\hat{\boldsymbol{C}}_t^{\boldsymbol{\theta}}$, also tend to decrease over time, which makes the difference between consecutive reference points, $\boldsymbol{\theta}_t^i-\boldsymbol{\theta}_{t-1}^i$, decrease as well. If we also assume that the function $p(\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1},\boldsymbol{\theta})$ is continuous in $\boldsymbol{\theta}$, then we can make the computation recursive by assuming

Algorithm 1 Nested Gaussian filters.

Inputs:

- Prior pdfs $p(\boldsymbol{x}_0)$ and $p(\boldsymbol{\theta})$.

Procedure:

1) Initialization
   a) Generate M reference points, $\boldsymbol{\theta}_0^i$, from $p(\boldsymbol{\theta}) \sim \mathcal{N}(\boldsymbol{\theta}_0, \boldsymbol{C}_0^{\boldsymbol{\theta}})$ for $i = 1, \ldots, M$, with weights w_0^i.
2) Sequential step, $t \geq 1$.
   a) For each $i = 1, \ldots, M$, use a Gaussian filter to compute $p(\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1}, \boldsymbol{\theta}_t^i)$.
   b) Compute $\hat{\boldsymbol{\theta}}_t$ and $\hat{\boldsymbol{C}}_t^{\boldsymbol{\theta}}$ from (14) and (15).
   c) Generate new reference points $\boldsymbol{\theta}_{t+1}^i$, $i = 1, \ldots, M$, from $\hat{\boldsymbol{\theta}}_t$ and $\hat{\boldsymbol{C}}_t^{\boldsymbol{\theta}}$, and weights w_{t+1}^i.

Outputs: $\hat{\boldsymbol{\theta}}_t$ and $\hat{\boldsymbol{C}}_t^{\boldsymbol{\theta}}$.

that $p(\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1}, \boldsymbol{\theta}_t^i) \simeq p(\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1}, \boldsymbol{\theta}_{t-1}^i)$ when $\boldsymbol{\theta}_t^i \simeq \boldsymbol{\theta}_{t-1}^i$. For the sake of clarity we summarize the steps for computing $p(\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1}, \boldsymbol{\theta}_t^i)$ in Algorithm 2, relying on a bank of EKF's.

Algorithm 3 outlines a recursive nested Gaussian filter with a UKF/CKF in the first layer and EKF's in the second layer. It can be seen as a recursive and explicit implementation of Algorithm 1. The initialization remains the same (step 1a), computing M reference points $\boldsymbol{\theta}_0^i$ and weights w_0^i, $i = 1, \ldots, M$, from the prior $p(\boldsymbol{\theta}) \sim \mathcal{N}(\boldsymbol{\theta}_0, \boldsymbol{C}_0^{\boldsymbol{\theta}})$. Also, we initialize the state and its covariance matrix in every Gaussian filter of the second layer (step 1b), setting $\hat{\boldsymbol{x}}_0^i = \hat{\boldsymbol{x}}_0$ and $\hat{\boldsymbol{C}}_0^{\boldsymbol{x},i} = \boldsymbol{C}_0^{\boldsymbol{x}}$, $i = 1, \ldots, M$, from the prior $p(\boldsymbol{x}_0) = \mathcal{N}(\boldsymbol{x}_0|\hat{\boldsymbol{x}}_0, \boldsymbol{C}_0^{\boldsymbol{x}})$.

The sequential procedure starts by approximating $p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t-1}, \boldsymbol{\theta}_t^i)$ with the second layer of Gaussian filters (step 2(a)i). This is done differently depending on whether we assume $\boldsymbol{\theta}_t^i \simeq \boldsymbol{\theta}_{t-1}^i$ or not. To be specific, we evaluate the maximum component of the vector $|\boldsymbol{\theta}_t^i - \boldsymbol{\theta}_{t-1}^i|$ and compare it against a prescribed threshold $\lambda > 0$ in order to determine whether the prediction and update steps in the second layer of filters can be recursive or not. Specifically:

- If $\max_{1 \leq j \leq d_\theta} |\boldsymbol{\theta}_{j,t}^i - \boldsymbol{\theta}_{j,t-1}^i| < \lambda$ is not satisfied for $\boldsymbol{\theta}_t^i$, the i-th filter runs from scratch following the scheme in Algorithm 2.
- When $\max_{1 \leq j \leq d_\theta} |\boldsymbol{\theta}_{j,t}^i - \boldsymbol{\theta}_{j,t-1}^i| < \lambda$ is satisfied for $\boldsymbol{\theta}_t^i$, only one prediction and update step (from time $t-1$ to time t) is needed. We approximate $p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t-1}, \boldsymbol{\theta}_t^i)$ from $p(\boldsymbol{x}_{t-1}|\boldsymbol{y}_{1:t-1}, \boldsymbol{\theta}_t^i) \approx p(\boldsymbol{x}_{t-1}|\boldsymbol{y}_{1:t-1}, \boldsymbol{\theta}_{t-1}^i)$.

Then, we can use $p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t-1}, \boldsymbol{\theta}_t^i)$ in order to compute $p(\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1}, \boldsymbol{\theta}_t^i)$ as in step 2b of Algorithm 2. Finally, we can compute the mean vector $\hat{\boldsymbol{\theta}}_t$ and the covariance matrix $\hat{\boldsymbol{C}}_t^{\boldsymbol{\theta},i}$ at time t in step 2b, by using (14) and (15). We prepare the new reference points $\boldsymbol{\theta}_{t+1}^i$ and their weights w_{t+1}^i from $\mathcal{N}(\hat{\boldsymbol{\theta}}_t, \hat{\boldsymbol{C}}_t^{\boldsymbol{\theta},i})$ for next time step.

We can take advantage of filters in the second layer in order to provide state estimates as well. Let us write the expectation

Algorithm 2 Extended Kalman filter conditional on $\boldsymbol{\theta}_t^i$, used in the second layer of the nested filter.

Inputs:

- Prior pdfs $p(\boldsymbol{x}_0)$.
- Known parameter vector $\boldsymbol{\theta}_t^i$.
- State-space model

$$\boldsymbol{x}_t = f(\boldsymbol{x}_{t-1}, \boldsymbol{\theta}) + \boldsymbol{v}_t, \qquad \boldsymbol{v}_t \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{V}), \tag{16}$$

$$\boldsymbol{y}_t = g(\boldsymbol{x}_t, \boldsymbol{\theta}) + \boldsymbol{r}_t, \qquad \boldsymbol{r}_t \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{R}). \tag{17}$$

Procedure:

1) Initialization
   a) Assume $p(\boldsymbol{x}_0) = \mathcal{N}(\hat{\boldsymbol{x}}_0, \hat{\boldsymbol{C}}_0^{\boldsymbol{x}})$.
2) Sequential step, $t \geq 1$.
   a) **Prediction step.** Compute

$$\tilde{\boldsymbol{x}}_{t,\boldsymbol{\theta}_t^i} = f(\hat{\boldsymbol{x}}_{t-1,\boldsymbol{\theta}_t^i}, \boldsymbol{\theta}_t^i), \tag{18}$$

$$\tilde{\boldsymbol{C}}_{t,\boldsymbol{\theta}_t^i}^{\boldsymbol{x}} = \boldsymbol{J}_{f,\hat{\boldsymbol{x}}_{t-1,\boldsymbol{\theta}_t^i}} \hat{\boldsymbol{C}}_{t-1,\boldsymbol{\theta}_t^i}^{\boldsymbol{x}} \boldsymbol{J}_{f,\hat{\boldsymbol{x}}_{t-1,\boldsymbol{\theta}_t^i}}^\top + \boldsymbol{V}, \tag{19}$$

   where $\boldsymbol{J}_{f,\boldsymbol{x}}$ is the Jacobian matrix of $f(\cdot)$ evaluated at $\hat{\boldsymbol{x}}_{t-1,\boldsymbol{\theta}_t^i}$.
   b) Use the unscented transform or a Gaussian cubature rule to compute

$$\begin{aligned} &p(\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1}, \boldsymbol{\theta}_t^i) \\ &= \int p(\boldsymbol{y}_t|\boldsymbol{x}_t, \boldsymbol{\theta}_t^i) p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t-1}, \boldsymbol{\theta}_t^i) d\boldsymbol{x}_t \\ &\simeq \int p(\boldsymbol{y}_t|\boldsymbol{x}_t, \boldsymbol{\theta}_t^i) \mathcal{N}(\boldsymbol{x}_t|\tilde{\boldsymbol{x}}_{t,\boldsymbol{\theta}_t^i}, \tilde{\boldsymbol{C}}_{t,\boldsymbol{\theta}_t^i}^{\boldsymbol{x}}) d\boldsymbol{x}_t. \end{aligned}$$

   c) **Update step.** Compute

$$\hat{\boldsymbol{x}}_{t,\boldsymbol{\theta}_t^i} = \tilde{\boldsymbol{x}}_{t,\boldsymbol{\theta}_t^i} + \boldsymbol{K}_t(\boldsymbol{y}_t - g(\tilde{\boldsymbol{x}}_{t,\boldsymbol{\theta}_t^i}, \boldsymbol{\theta}_t^i)), \tag{20}$$

$$\hat{\boldsymbol{C}}_{t,\boldsymbol{\theta}_t^i}^{\boldsymbol{x}} = (\boldsymbol{I}_{d_x} - \boldsymbol{K}_t \boldsymbol{J}_g) \tilde{\boldsymbol{C}}_{t,\boldsymbol{\theta}_t^i}^{\boldsymbol{x}}, \tag{21}$$

$$\boldsymbol{K}_t = \tilde{\boldsymbol{C}}_{t,\boldsymbol{\theta}_t^i}^{\boldsymbol{x}} \boldsymbol{J}_{g,\tilde{\boldsymbol{x}}_{t,\boldsymbol{\theta}_t^i}}^\top (\boldsymbol{J}_{g,\tilde{\boldsymbol{x}}_{t,\boldsymbol{\theta}_t^i}} \tilde{\boldsymbol{C}}_{t,\boldsymbol{\theta}_t^i}^{\boldsymbol{x}} \boldsymbol{J}_{g,\tilde{\boldsymbol{x}}_{t,\boldsymbol{\theta}_t^i}}^\top + \boldsymbol{R}),$$

   where $\boldsymbol{J}_{g,\boldsymbol{x}}$ is the Jacobian matrix of $g(\cdot)$ evaluated at $\tilde{\boldsymbol{x}}_{t,\boldsymbol{\theta}_t^i}$. Approximate $p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t}, \boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{x}_t|\hat{\boldsymbol{x}}_{t,\boldsymbol{\theta}_t^i}, \hat{\boldsymbol{C}}_{t,\boldsymbol{\theta}_t^i}^{\boldsymbol{x}})$.

Outputs: $\hat{\boldsymbol{x}}_{t,\boldsymbol{\theta}_t^i}$, $\hat{\boldsymbol{C}}_{t,\boldsymbol{\theta}_t^i}^{\boldsymbol{x}}$ and $p(\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1}, \boldsymbol{\theta}_t^i)$.

of $\boldsymbol{X}_t$ as

$$\mathbb{E}[\boldsymbol{X}_t|\boldsymbol{y}_{1:t}] = \int_\Theta \Big[\int_\mathcal{X} \boldsymbol{x}_t p(\boldsymbol{x}_t|\boldsymbol{\theta}, \boldsymbol{y}_{1:t}) d\boldsymbol{x}_t \Big] p(\boldsymbol{\theta}|\boldsymbol{y}_{1:t}) d\boldsymbol{\theta}, \tag{22}$$

where the integral in square brackets can be approximated by the M Gaussian filters of the second layer. In this case, we assume they are the EKF's of Algorithm 2 conditional on $\Theta = \boldsymbol{\theta}_t^i$. This yields a Gaussian approximation of $p(\boldsymbol{x}_t|\boldsymbol{\theta}_t^i, \boldsymbol{y}_{1:t}) \simeq \mathcal{N}(\boldsymbol{x}_t|\hat{\boldsymbol{x}}_{t,\boldsymbol{\theta}_t^i}, \hat{\boldsymbol{C}}_{t,\boldsymbol{\theta}_t^i})$, where

$$\hat{\boldsymbol{x}}_{t,\boldsymbol{\theta}_t^i} \simeq \mathbb{E}[\boldsymbol{X}_t|\boldsymbol{\theta}_t^i, \boldsymbol{y}_{1:t}] \quad \text{and} \tag{23}$$

$$\hat{\boldsymbol{C}}_{t,\boldsymbol{\theta}_t^i}^{\boldsymbol{x}} \simeq \mathbb{E}[(\boldsymbol{X}_t - \hat{\boldsymbol{x}}_{t,\boldsymbol{\theta}_t^i})(\boldsymbol{X}_t - \hat{\boldsymbol{x}}_{t,\boldsymbol{\theta}_t^i})^\top | \boldsymbol{y}_{1:t}, \boldsymbol{\theta}_t^i]. \tag{24}$$

Then, a Gaussian approximation $p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t}) \approx \mathcal{N}(\boldsymbol{x}_t|\hat{\boldsymbol{x}}_t, \hat{\boldsymbol{C}}_t^{\boldsymbol{x}})$ can be constructed, where

$$\hat{\boldsymbol{x}}_t \simeq \sum_{i=1}^{M} \hat{\boldsymbol{x}}_{t,\boldsymbol{\theta}_t^i} \frac{p(\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1}, \boldsymbol{\theta}_t^i)}{p(\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1})} w_t^i \quad \text{and} \tag{25}$$

$$\hat{\boldsymbol{C}}_t^{\boldsymbol{x}} \simeq \sum_{i=1}^{M} (\hat{\boldsymbol{x}}_{t,\boldsymbol{\theta}_t^i} - \hat{\boldsymbol{x}}_t)(\hat{\boldsymbol{x}}_{t,\boldsymbol{\theta}_t^i} - \hat{\boldsymbol{x}}_t)^\top \frac{p(\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1}, \boldsymbol{\theta}_t^i)}{p(\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1})} w_t^i. \tag{26}$$

Algorithm 3 Recursive nested Gaussian filters.

Inputs:

- Prior pdfs $p(\boldsymbol{x}_0)$ and $p(\boldsymbol{\theta})$.
- A fixed threshold $\lambda > 0$.

Procedure:

1) Initialization
   a) Generate M reference points, $\boldsymbol{\theta}_0^i$, for $p(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}_0, \boldsymbol{C}_0^{\boldsymbol{\theta}})$, $i = 1, \ldots, M$, with weights w_0^i.
   b) If $p(\boldsymbol{x}_0) = \mathcal{N}(\boldsymbol{x}_0|\hat{\boldsymbol{x}}_0, \boldsymbol{C}_0^{\boldsymbol{x}})$, then set $\hat{\boldsymbol{x}}_0^i = \hat{\boldsymbol{x}}_0$ and $\hat{\boldsymbol{C}}_0^{\boldsymbol{x},i} = \boldsymbol{C}_0^{\boldsymbol{x}}$ for $i = 1, \ldots, M$.
2) Sequential step, $t \geq 1$.
   a) For $i = 1, \ldots, M$:
      i) If $\max_{1 \leq j \leq d_\theta} |\boldsymbol{\theta}_{j,t}^i - \boldsymbol{\theta}_{j,t-1}^i| < \lambda$, then approximate $p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t-1}, \boldsymbol{\theta}_t^i)$ from $p(\boldsymbol{x}_{t-1}|\boldsymbol{y}_{1:t-1}, \boldsymbol{\theta}_t^i) \approx p(\boldsymbol{x}_{t-1}|\boldsymbol{y}_{1:t-1}, \boldsymbol{\theta}_{t-1}^i)$, where $p(\boldsymbol{x}_{t-1}|\boldsymbol{y}_{1:t-1}, \boldsymbol{\theta}_{t-1}^i) = \mathcal{N}(\boldsymbol{x}_{t-1}|\hat{\boldsymbol{x}}_{t-1,\boldsymbol{\theta}_{t-1}^i}, \hat{\boldsymbol{C}}_{t-1,\boldsymbol{\theta}_{t-1}^i}^{\boldsymbol{x}})$.
      Else, approximate $p(\boldsymbol{x}_t|\boldsymbol{y}_{1:t-1}, \boldsymbol{\theta}_t^i)$ from the prior $p(\boldsymbol{x}_0)$.
      ii) Use this approximation to compute $p(\boldsymbol{y}_t|\boldsymbol{y}_{1:t-1}, \boldsymbol{\theta}_t^i)$.
   b) Compute $\hat{\boldsymbol{\theta}}_t$, $\hat{\boldsymbol{C}}_t^{\boldsymbol{\theta}}$, $\hat{\boldsymbol{x}}_t$ and $\hat{\boldsymbol{C}}_t^{\boldsymbol{x}}$ from (14), (15), (25) and (26), respectively.
   c) Generate reference points $\boldsymbol{\theta}_{t+1}^i$ from $\hat{\boldsymbol{\theta}}_t$ and $\hat{\boldsymbol{C}}_t^{\boldsymbol{\theta}}$ for $i = 1, \ldots, M$.

Outputs: $\hat{\boldsymbol{x}}_t$, $\hat{\boldsymbol{\theta}}_t$, $\hat{\boldsymbol{C}}_t^{\boldsymbol{x}}$ and $\hat{\boldsymbol{C}}_t^{\boldsymbol{\theta}}$.

IV. Example

A. Stochastic Lorenz 63 model

Consider the 3-dimensional stochastic process $\boldsymbol{x}(\tau) = [x_1(\tau), x_2(\tau), x_3(\tau)]^\top$, for $\tau \in (0, \infty)$, taking values on $\mathbb{R}^3$, whose dynamics are described by the system of stochastic differential equations (SDEs)

$$dx_1 = -S(x_1 - x_2) + \sigma dv_1, \tag{27}$$

$$dx_2 = Rx_1 - x_2 - x_1 x_3 + \sigma dv_2, \tag{28}$$

$$dx_3 = x_1 x_2 - Bx_3 + \sigma dv_3, \tag{29}$$

where the v_i's are independent 1-dimensional Wiener processes, $\sigma > 0$ is a known scale parameter and $(S, R, B) \in \mathbb{R}$ are unknown static model parameters. Using Euler-Maruyama in order to integrate the SDEs (27)–(29), it is straightforward to write the state-space model as

$$\boldsymbol{x}_{t+1} = f_\Delta(\boldsymbol{x}_t, \boldsymbol{\theta}) + \sqrt{\Delta}\boldsymbol{v}_t, \quad t = 1, 2, \ldots \tag{30}$$

where $f_\Delta : \mathbb{R}^{d_x} \times \mathbb{R}^{d_\theta} \to \mathbb{R}^{d_x}$ can be expressed as

$$\begin{aligned} f_{1,\Delta}(\boldsymbol{x}_t, \boldsymbol{\theta}) &= x_{1,t} - \Delta S(x_{1,t} - x_{2,t}), \\ f_{2,\Delta}(\boldsymbol{x}_t, \boldsymbol{\theta}) &= x_{2,t} + \Delta[(R - x_{3,t})x_{1,t} - x_{2,t}], \\ f_{3,\Delta}(\boldsymbol{x}_t, \boldsymbol{\theta}) &= x_{3,t} + \Delta(x_{1,t}x_{2,t} - Bx_{3,t}), \end{aligned}$$

where Δ is the integration step, $\boldsymbol{\theta} = (S, R, B)^\top$ and $\boldsymbol{v}_t$ is a sequence of 3-dimensional Gaussian independent random vectors with zero mean and covariance matrix $\sigma_x^2 \boldsymbol{I}_3$. Hence, $p(\boldsymbol{x}_t|\boldsymbol{x}_{t-1}, \boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{x}_t|f_\Delta(\boldsymbol{x}_{t-1}, \boldsymbol{\theta}), \sigma^2 \Delta \boldsymbol{I}_{d_x})$.

We assume the observation equation

$$\boldsymbol{y}_t = [x_{1,t}, x_{3,t}]^\top + \boldsymbol{r}_t, \tag{31}$$

where $\boldsymbol{r}_t$ is a 2-dimensional Gaussian random variable with covariance matrix $\sigma_y^2 \boldsymbol{I}_2$.

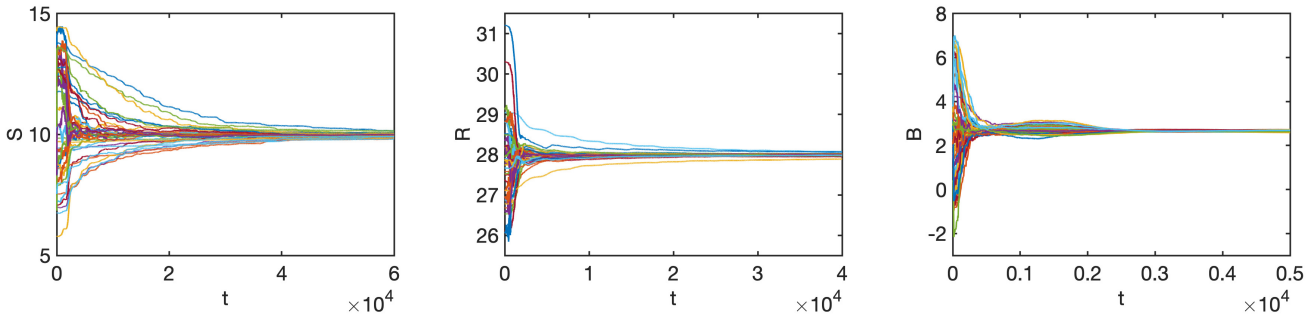
B. Numerical results

For our computer experiments we have used the stochastic Lorenz 63 model outlined in (30) and (31) in order to generate signals $\boldsymbol{x}_t$ and $\boldsymbol{y}_t, t = \{0, 1, \ldots\}$, used as the ground truth for the assessment of the algorithm. We integrate the model using Euler-Maruyama with step $\Delta = 2 \times 10^{-4}$ continuous-time units. We assume the fixed parameters are $S = 10$, $R = 28$ and $B = \frac{8}{3}$ (which yield underlying chaotic dynamics); and the initial state is $\hat{\boldsymbol{x}}_0 = [-6, -5.5, -24.5]^\top$. The noise scaling factors, $\sigma^2 = 0.1$ and $\sigma_y^2 = 1$, are known.

For the estimation task we use Algorithm 3. The initialization of the state for each simulation is generated with $p(\boldsymbol{x}_0) = \mathcal{N}(\boldsymbol{x}_0|\hat{\boldsymbol{x}}_0, \boldsymbol{I}_3)$. In the same way, $\boldsymbol{\theta}_0$ is initialized by $\mathcal{N}(\boldsymbol{\mu}_\theta, \boldsymbol{I}_{d_\theta})$, where $\boldsymbol{\mu}_\theta = \mathcal{U}(\boldsymbol{\theta}_\star - \boldsymbol{\epsilon}, \boldsymbol{\theta}_\star + \boldsymbol{\epsilon})$, being $\boldsymbol{\theta}_\star$ the true parameters and $\boldsymbol{\epsilon} = [3, 1, 0.5]^\top$. The algorithm does not collect an observation at every time step, but every 5 discrete-time steps (10^{-3} continuous-time units). Hence, the prediction step of the state variables at the second layer of nested filter corresponds to 5 discrete-time steps of the Euler scheme until a new observation arrives. Then, both the state and parameter distributions are updated. We assume a threshold $\lambda = 5 \times 10^{-3}$. Every simulation runs until $t = T = 10^5$.

Figure 1 shows the parameter estimates obtained by running 40 independent simulations. The three dimensions of $\hat{\boldsymbol{\theta}}_t$ are displayed over time (1a–1c) in order to illustrate how they converge as observations are collected. Although the length of the simulations is $T = 10^5$ continuous-time units, we have plotted just the sections of time where the estimates converge. The interval varies from one plot to another because the time of convergence is not the same for all parameters (having shorter times for B and longer times for S). In spite of that, this figure shows how all parameters converge to the true values for different initializations.

We have also assessed the accuracy of algorithm in terms of the mean square error (MSE) of the predictor of the state.



(a) Estimates $\hat{\boldsymbol{\theta}}_{1,t} = \hat{S}_t$. (b) Estimates $\hat{\boldsymbol{\theta}}_{2,t} = \hat{R}_t$. (c) Estimates $\hat{\boldsymbol{\theta}}_{3,t} = \hat{B}_t$.

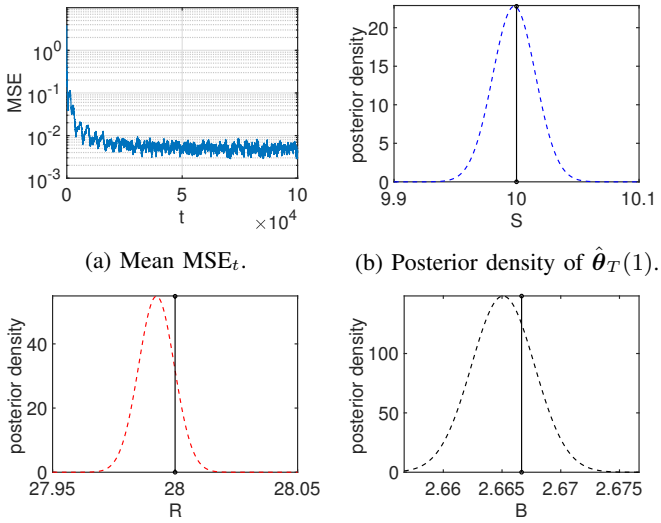
Fig. 1: Sequences of posterior-mean estimates, $\hat{\boldsymbol{\theta}}_t$, over time obtained from 40 independent simulation runs.

We show the empirical MSE per dimension resulting directly from the simulations,

$$\text{MSE}_t = \frac{1}{d_x}\|\boldsymbol{x}_t - \hat{\boldsymbol{x}}_t\|^2. \tag{32}$$

Figure 2a, on the other hand, illustrates the accuracy of state estimates, $\hat{\boldsymbol{x}}_t$, by averaging the MSE_t (32) of the previous set of 40 simulation runs. The error decreases with time as parameter estimates get closer to their true values, achieving its minimum around $t = 2.5 \times 10^4$. By that time, all parameter estimates in Figure 1 have already converged and, consequently, the state estimates as well.

In Figures 2b, 2c and 2d, the probability densities of each dimension of $\hat{\boldsymbol{\theta}}_t$ at time $t = T$ are plotted for a typical simulation run. They not only show the point estimates of the parameters, but they also display information related to their uncertainty. The mean of these Gaussian pdfs is close to the true parameters, in agreement with results seen in Figure 1. In addition, the variances are small, being all the probability distributions tightly packed around the ground truth.



(a) Mean MSE_t. (b) Posterior density of $\hat{\boldsymbol{\theta}}_T(1)$.

(c) Posterior density of $\hat{\boldsymbol{\theta}}_T(2)$. (d) Posterior density of $\hat{\boldsymbol{\theta}}_T(3)$.

Fig. 2: The mean MSE of 20 simulation runs over time is plotted in 2a. Figures 2b, 2c and 2d show the posterior density of parameters (dashed lines) at time $t = T$ and their true values (black vertical lines).

V. Conclusions

We have introduced a generalization of the NHF methodology of [1] that, using long sequences of observations collected over time, calibrates the static parameters and estimates the stochastic dynamical variables of a state space model. This scheme combines two layers of filters, one inside the other, in order to compute the joint posterior probability distribution of the parameters and the states. In this generalization of the methodology, we introduce the use of deterministic sampling techniques in the first layer of the algorithm (the cubature Kalman filter (CKF) or the unscented Kalman filter (UKF)), instead of Monte Carlo methods, describing in detail how the algorithms can work sequentially and recursively. The use of Gaussian filters in the two layers of the algorithm enables a significant reduction in computational complexity compared to Monte Carlo-based implementations. We have presented numerical results for a stochastic Lorenz 63 model, using an scheme with an UKF for the parameters in the first layer, and EKF's for the time-varying state variables in the second layer.

References

[1] Pérez-Vieites, Sara, Ins 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] 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.

[3] 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.

[4] 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.

[5] Arasaratnam, Ienkaran, and Simon Haykin. "Cubature Kalman filters." IEEE Transactions on Automatic Control 54.6 (2009): 1254-1269.

[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.