

This is the peer reviewed version of the following article:

Double Bayesian Smoothing as Message Passing / Di Viesti, Pasquale; Vitetta, Giorgio Matteo; Sirignano, Emilio. - In: IEEE TRANSACTIONS ON SIGNAL PROCESSING. - ISSN 1053-587X. - 67:21(2019), pp. 5495-5510. [10.1109/TSP.2019.2941064]

*Terms of use:*

The terms and conditions for the reuse of this version of the manuscript are specified in the publishing policy. For all terms of use and more information see the publisher's website.

18/12/2025 03:51

# Double Bayesian Smoothing as Message Passing

July 29, 2019

## Abstract

Recently, a novel method for developing filtering algorithms, based on the interconnection of two Bayesian filters and called double Bayesian filtering, has been proposed. In this manuscript we show that the same conceptual approach can be exploited to devise a new smoothing method, called double Bayesian smoothing. A double Bayesian smoother combines a double Bayesian filter, employed in its forward pass, with the interconnection of two backward information filters used in its backward pass. As a specific application of our general method, a detailed derivation of double Bayesian smoothing algorithms for conditionally linear Gaussian systems is illustrated. Numerical results for two specific dynamic systems evidence that these algorithms can achieve a better complexity-accuracy tradeoff and tracking capability than other smoothing techniques recently appeared in the literature.

Pasquale Di Viesti<sup>†</sup>      Giorgio M. Vitetta<sup>†</sup>      Emilio Sirignano<sup>†</sup>  
 pasquale.diviesti@unimore.it    giorgio.vitetta@unimore.it    emilio.sirignano@unimore.it

<sup>†</sup>Dept. of Engineering "Enzo Ferrari", University of Modena and Reggio Emilia.

**Keywords:** Hidden Markov Model, Smoothing, Factor Graph, Particle Filter, Kalman Filter, Sum-Product Algorithm.

## I. Introduction

The problem of *Bayesian smoothing* for a *state space model* (SSM) concerns the development of recursive algorithms able to estimate the *probability density function* (pdf) of the model state on a given observation interval, given a batch of noisy measurements acquired over it [1], [2]; the estimated pdf is known as a *smoothed* or *smoothing* pdf. Two general methods are available in the literature for recursively calculating smoothing densities; they are known as the *forward filtering-backward smoothing recursion* (e.g., see [3] and [4]) and the method based on the *two-filter smoothing formula* (e.g., see [5] and [6]). Both methods are based on the idea that the smoothing densities can be computed by combining the predicted and/or filtered densities generated by a Bayesian filtering method with the statistical information produced in the backward pass by a different filtering method; the latter method is *paired with the first one* and, in the case of the two-filter smoothing formula, is known as *backward information filtering* (BIF). Unluckily, closed form solutions for Bayesian smoothing are available for *linear Gaussian* and *linear Gaussian mixture* models only [1, 2, 7]. This has motivated the development of various methods based on approximating smoothing densities in different ways. For instance, the use of Gaussian approximations for the smoothing densities and of sigma points techniques for solving moment matching integrals has been investigated in [8–10]. Another class of methods (usually known as *particle smoothers*) is based on the exploitation of *sequential Monte Carlo* techniques, i.e. on approximating smoothing densities through a set of weighted particles (e.g., see [3, 5, 11–14] and references therein). Recently, substantial attention has been also paid to the development of smoothing algorithms for the class of *conditionally linear Gaussian* SSMs [15–19]. In this case, the above mentioned approximate methods can benefit from the so called *Rao-Blackwellization* technique, i.e. from the

marginalisation of the linear substructure of any conditionally linear Gaussian model; this can significantly reduce the overall computational complexity of both sigma-point based Gaussian smoothing [15] and particle smoothing [16–19] (that is usually known as *Rao-Blackwellized particle smoothing*, RBPS, in this case).

In this manuscript, we propose a novel general method for the development of *computationally efficient* particle smoothers. Our method exploits the same conceptual approach illustrated in [20] in the context of Bayesian filtering and dubbed *multiple Bayesian filtering*. That approach is based on the idea of developing new filtering algorithms by: a) interconnecting multiple heterogeneous Bayesian filters; b) representing the processing accomplished by each Bayesian filter and the exchange of statistical information among distinct filters as a message passing over a proper factor graph. In [20] the exploitation of this approach has been investigated in detail for the case in which two Bayesian filters are interconnected, i.e. *dual Bayesian filtering* (DBF) is employed. Moreover, it has been shown that accurate and computationally efficient DBF algorithms can be devised if the considered SSM is conditionally linear Gaussian. In this manuscript, we show that, if DBF is employed in the *forward* pass of a smoothing method, a BIF method, paired with DBF and based on the interconnection of two backward information filters can be devised by following some simple rules. Similarly as DBF, our derivation of such a BIF method, called *double backward information filtering* (DBIF), is based on a *graphical model*. Such a graphical model allows us to show that: a) the pdfs computed in DBIF can be represented as messages passed on it; b) all the expressions of the passed messages can be derived by applying the same rule, namely the so called *sum-product algorithm* (SPA) [21], [22], to it; c) iterative algorithms can be developed in a natural fashion once the cycles it contains have been identified and the order according to which messages are passed on them (i.e., the *message scheduling*) has been established; d) the statistical information generated by a DBIF algorithm in the backward pass can be easily merged with those produced by its paired DBF technique in the forward pass in order to evaluate the required smoothed pdfs. To exemplify the usefulness of the resulting smoothing method, based on the combination of DBF and DBIF, and called *double Bayesian smoothing* (DBS), the two DBF algorithms proposed in [20] for the class of conditionally linear Gaussian SSMs are taken into consideration, and the BITF algorithm paired with each of them and a simplified version of it are derived. This leads to the development of four new DBS algorithms, two generating an estimate of the joint smoothing density over the whole observation interval, the other two an estimate of the marginal smoothing densities over the same interval. Our computer simulations for two specific conditionally linear Gaussian SSMs evidence that, in the first case, the derived DBS algorithms perform very closely to the RBPS technique proposed in [18] and to the particle smoothers devised in [19], but at lower computational cost and time. In the second case, instead, two of the devised DBS techniques represent the only technically useful options, thanks to their good tracking capability. In fact, such techniques are able to operate reliably even when their competitors diverge in the forward pass.

It is worth stressing that the technical contribution provided by this manuscript represents a significant advancement with respect to the application of factor graph theory to particle smoothing illustrated in [19]. In fact, in that manuscript, we also focus on conditionally linear Gaussian models, but assume that the forward pass is accomplished by *marginalized particle filtering* (MPF; also known as *Rao-Blackwellized particle filtering*); in other words, Bayesian filtering is based on the interconnection of a particle filter with a *bank of Kalman filters*. In this manuscript, instead, the general method we propose applies to a couple of arbitrary interconnected Bayesian filters. Moreover, the specific smoothing algorithms we derive assume that the forward pass is carried out by a filtering algorithm based on the interconnection of a particle filter with a *single* extended Kalman filter.

The remaining part of this manuscript is organized as follows. In Section II., a general graphical model, on which the processing accomplished in DBF, DBIF, and DBS is based, is illustrated. In Section III., a specific instance of the graphical model illustrated in the previous section is developed under the assumptions that the filters employed in the forward pass are an extended Kalman filter and a particle filter, and that the considered SSM is conditionally linear Gaussian. Then, the scheduling and the computation of the messages passed over this model are analysed in detail and new DBS algorithms are devised. The differences and similarities between these algorithms and other known smoothing techniques are analysed in Section IV.. A comparison, in terms of accuracy, computational complexity, and execution time, between the proposed techniques and three smoothers recently appeared in the literature, is provided in Section V. for two conditionally linear Gaussian SSMs. Finally, some conclusions are offered in Section VI..

*Notations:* The same notation as refs. [19, 20] and [23] is adopted.

## II. Graphical Model for a Couple of Interconnected Bayesian Information Filters and Message Passing on it

In this manuscript, we consider a discrete-time SSM whose  $D$ -dimensional *hidden state* in the  $k$ -th interval is denoted  $\mathbf{x}_k \triangleq [x_{0,k}, x_{1,k}, \dots, x_{D-1,k}]^T$ , and whose *state update* and *measurement models* are expressed by

$$\mathbf{x}_{k+1} = \mathbf{f}_k(\mathbf{x}_k) + \mathbf{w}_k \quad (1)$$

and

$$\begin{aligned} \mathbf{y}_k &\triangleq [y_{0,k}, y_{1,k}, \dots, y_{P-1,k}]^T \\ &= \mathbf{h}_k(\mathbf{x}_k) + \mathbf{e}_k, \end{aligned} \quad (2)$$

respectively, with  $k = 1, 2, \dots, T$ . Here,  $\mathbf{f}_k(\mathbf{x}_k)$  ( $\mathbf{h}_k(\mathbf{x}_k)$ ) is a time-varying  $D$ -dimensional ( $P$ -dimensional) real function,  $T$  is the duration of the observation interval and  $\mathbf{w}_k$  ( $\mathbf{e}_k$ ) is the  $k$ -th element of the process (measurement) noise sequence  $\{\mathbf{w}_k\}$  ( $\{\mathbf{e}_k\}$ ); this sequence consists of  $D$ -dimensional ( $P$ -dimensional) *independent and identically distributed* (iid) Gaussian noise vectors, each characterized by a zero mean and a covariance matrix  $\mathbf{C}_w$  ( $\mathbf{C}_e$ ). Moreover, statistical independence between  $\{\mathbf{e}_k\}$  and  $\{\mathbf{w}_k\}$  is assumed.

From a statistical viewpoint, a complete statistical description of the considered SSM is provided by the pdf  $f(\mathbf{x}_1)$  of its initial state, its *Markov model*  $f(\mathbf{x}_{k+1}|\mathbf{x}_k)$  and its *observation model*  $f(\mathbf{y}_k|\mathbf{x}_k)$  for any  $k$ ; the first pdf is assumed to be known, whereas the last two pdfs can be easily derived from Eq. (1) and Eq. (2), respectively.

In the following, we focus on the problem of developing novel smoothing algorithms and, in particular, algorithms for the estimation of the *joint smoothed pdf*  $f(\mathbf{x}_{1:T}|\mathbf{y}_{1:T})$  (problem **P.1**) and the sequence of *marginal smoothed pdfs*  $\{f(\mathbf{x}_k|\mathbf{y}_{1:T}), k = 1, 2, \dots, T\}$  (problem **P.2**); here,  $\mathbf{y}_{1:T} \triangleq [\mathbf{y}_1^T, \mathbf{y}_2^T, \dots, \mathbf{y}_T^T]^T$  is a  $P \cdot T$ -dimensional vector. Note that, in principle, once problem **P.1** is solved, problem **P.2** can be easily tackled; in fact, if the joint pdf  $f(\mathbf{x}_{1:T}|\mathbf{y}_{1:T})$  is known, all the posterior pdfs  $\{f(\mathbf{x}_k|\mathbf{y}_{1:T})\}$  can be evaluated by *marginalization*.

The development of our smoothing algorithms is mainly based on the graphical approach illustrated in our previous manuscripts [19, Sec. III], [20, Sec. II] and [23, Sec. III] for Bayesian filtering and smoothing. This approach consists in the following steps:

1. The state vector  $\mathbf{x}_k$  is partitioned in two substates, denoted  $\mathbf{x}_k^{(1)}$  and  $\mathbf{x}_k^{(2)}$  and having sizes  $D_1$  and  $D_2 = D - D_1$ , respectively. Note that, if  $\bar{\mathbf{x}}_k^{(i)}$  represents the portion of  $\mathbf{x}_k$  not included in  $\mathbf{x}_k^{(i)}$  (with  $i = 1$  and 2), our assumptions entail that  $\bar{\mathbf{x}}_k^{(1)} = \mathbf{x}_k^{(2)}$  and  $\bar{\mathbf{x}}_k^{(2)} = \mathbf{x}_k^{(1)}$ .
2. A *sub-graph* that allows to represent both Bayesian filtering and BIF for the substate  $\mathbf{x}_k^{(i)}$  (with  $i = 1$  and 2) as message passing algorithms on it is developed, under the assumption that the complementary substate  $\bar{\mathbf{x}}_k^{(i)}$  is statistically known. This means that filtered and predicted densities of  $\mathbf{x}_k^{(i)}$  are represented as messages passed on the edges of this sub-graph and the rules for computing them result from the application of the SPA to it.
3. The two sub-graphs devised in the previous step (one referring to  $\mathbf{x}_k^{(1)}$ , the other one to  $\mathbf{x}_k^{(2)}$ ) are *interconnected*, so that a single graphical model referring to the whole state  $\mathbf{x}_k$  is obtained.
4. Algorithms for Bayesian filtering and BIF for the whole state  $\mathbf{x}_k$  are derived by applying the SPA to the graphical model obtained in the previous step.

Let us analyse now the steps 2.-4. in more detail. As far as **step 2.** is concerned, the sub-graph devised for the substate  $\mathbf{x}_k^{(i)}$  is based on the same principles illustrated in our manuscripts cited above (in particular, ref. [20]) and is illustrated in Fig. 1. The  $k$ -th recursion (with  $k = 1, 2, \dots, T$ ) of Bayesian filtering for the sub-state  $\mathbf{x}_k^{(i)}$  is represented as a *forward* message passing on this factor graph, that involves the Markov model  $f(\mathbf{x}_{k+1}^{(i)}|\mathbf{x}_k^{(i)}, \bar{\mathbf{x}}_k^{(i)})$  and the observation model  $f(\mathbf{y}_k|\mathbf{x}_k^{(i)}, \bar{\mathbf{x}}_k^{(i)})$ . This allows to compute the messages  $\vec{m}_{\text{fe1}}(\mathbf{x}_k^{(i)})$ ,  $\vec{m}_{\text{fe2}}(\mathbf{x}_k^{(i)})$  and  $\vec{m}_{\text{fp}}(\mathbf{x}_{k+1}^{(i)})$ , that convey the *first filtered* pdf of  $\mathbf{x}_k^{(i)}$ , the *second filtered* pdf of  $\mathbf{x}_k^{(i)}$  and the *predicted* pdf of  $\mathbf{x}_{k+1}^{(i)}$ , respectively, on the basis of the messages  $\vec{m}_{\text{fp}}(\mathbf{x}_k^{(i)})$ ,  $m_{\text{ms}}(\mathbf{x}_k^{(i)})$  and  $m_{\text{pm}}(\mathbf{x}_k^{(i)})$ ; the last three messages represent the predicted pdf of  $\mathbf{x}_k^{(i)}$  evaluated in the previous (i.e., in the

$(k-1)$ -th) recursion of Bayesian filtering, and the messages conveying the *measurement* and the *pseudo-measurement* information, respectively, available in the  $k$ -recursion. The considered filtering algorithm requires the availability of the messages  $m_{\text{pm}}(\mathbf{x}_k^{(i)})$ ,  $m_{\text{mg1}}(\bar{\mathbf{x}}_k^{(i)})$ ,  $m_{\text{mg2}}(\bar{\mathbf{x}}_k^{(i)})$ , that are computed on the basis of external statistical information. The presence of the messages  $m_{\text{mg1}}(\bar{\mathbf{x}}_k^{(i)})$  and  $m_{\text{mg2}}(\bar{\mathbf{x}}_k^{(i)})$  is due the fact that the substate  $\bar{\mathbf{x}}_k^{(i)}$  represents a *nuisance state* for the considered filtering algorithm; in fact, these messages convey filtered (or predicted) pdfs of  $\bar{\mathbf{x}}_k^{(i)}$  and are employed to integrate out the dependence of the pdfs  $f(\mathbf{y}_k|\mathbf{x}_k^{(i)}, \bar{\mathbf{x}}_k^{(i)})$  and  $f(\mathbf{x}_{k+1}^{(i)}|\mathbf{x}_k^{(i)}, \bar{\mathbf{x}}_k^{(i)})$ , respectively, on  $\bar{\mathbf{x}}_k^{(i)}$ . Note also that these two messages are not necessarily equal, since more refined information about  $\bar{\mathbf{x}}_k^{(i)}$  could become available after that the message  $m_{\text{ms}}(\mathbf{x}_k^{(i)})$  has been computed. On the other hand, the message  $m_{\text{pm}}(\mathbf{x}_k^{(i)})$  conveys the statistical information provided by a pseudo-measurement<sup>1</sup> about  $\mathbf{x}_k^{(i)}$ . In Fig. 1, following [20, Sec. II], it is assumed that the pseudo-measurement  $\mathbf{z}_k^{(i)}$  is available in the estimation of  $\mathbf{x}_k^{(i)}$  and that  $m_{\text{pm}}(\mathbf{x}_k^{(i)})$  represents the pdf of  $\mathbf{z}_k^{(i)}$  conditioned on  $\mathbf{x}_k^{(i)}$ , that is

$$m_{\text{pm}}(\mathbf{x}_k^{(i)}) \triangleq f(\mathbf{z}_k^{(i)}|\mathbf{x}_k^{(i)}). \quad (3)$$

The computation of the messages  $\vec{m}_{\text{fe1}}(\mathbf{x}_k^{(i)})$ ,  $\vec{m}_{\text{fe2}}(\mathbf{x}_k^{(i)})$  and  $\vec{m}_{\text{fp}}(\mathbf{x}_{k+1}^{(i)})$  on the basis of the messages  $\vec{m}_{\text{fp}}(\mathbf{x}_k^{(i)})$ ,  $m_{\text{ms}}(\mathbf{x}_k^{(i)})$ ,  $m_{\text{pm}}(\mathbf{x}_k^{(i)})$ ,  $m_{\text{mg1}}(\bar{\mathbf{x}}_k^{(i)})$  and  $m_{\text{mg2}}(\bar{\mathbf{x}}_k^{(i)})$  is based on the two simple rules illustrated in [23, Figs. 8-a) and 8-b), p. 1535] and can be summarized as follows. The first and second filtered pdfs (i.e., the first and the second forward estimates) of  $\mathbf{x}_k^{(i)}$  are evaluated as

$$\vec{m}_{\text{fe1}}(\mathbf{x}_k^{(i)}) = \vec{m}_{\text{fp}}(\mathbf{x}_k^{(i)}) m_{\text{ms}}(\mathbf{x}_k^{(i)}), \quad (4)$$

and

$$\vec{m}_{\text{fe2}}(\mathbf{x}_k^{(i)}) = \vec{m}_{\text{fe1}}(\mathbf{x}_k^{(i)}) m_{\text{pm}}(\mathbf{x}_k^{(i)}), \quad (5)$$

respectively, where

$$m_{\text{ms}}(\mathbf{x}_k^{(i)}) \triangleq \int f(\mathbf{y}_k|\mathbf{x}_k^{(i)}, \bar{\mathbf{x}}_k^{(i)}) m_{\text{mg1}}(\bar{\mathbf{x}}_k^{(i)}) d\bar{\mathbf{x}}_k^{(i)} \quad (6)$$

and  $m_{\text{pm}}(\mathbf{x}_k^{(i)})$  is defined in Eq. (3). Equations (4)-(6) describe the processing accomplished in the *measurement update* of the considered recursion. This is followed by the *time update*, in which the new predicted pdf (i.e., the new forward prediction)

$$\begin{aligned} \vec{m}_{\text{fp}}(\mathbf{x}_{k+1}^{(i)}) &= \int \int f(\mathbf{x}_{k+1}^{(i)}|\mathbf{x}_k^{(i)}, \bar{\mathbf{x}}_k^{(i)}) \vec{m}_{\text{fe2}}(\mathbf{x}_k^{(i)}) \\ &\quad \cdot m_{\text{mg2}}(\bar{\mathbf{x}}_k^{(i)}) d\mathbf{x}_k^{(i)} d\bar{\mathbf{x}}_k^{(i)}, \end{aligned} \quad (7)$$

is computed. The message passing procedure described above is initialised by setting  $\vec{m}_{\text{fp}}(\mathbf{x}_1^{(i)}) = f(\mathbf{x}_1^{(i)})$  (where  $f(\mathbf{x}_1^{(i)})$  is the pdf resulting from the marginalization of  $f(\mathbf{x}_1)$  with respect to  $\bar{\mathbf{x}}_1^{(i)}$ ) in the first recursion and is run for  $k = 1, 2, \dots, T$ . Once this procedure is over, BIF is executed for the substate  $\mathbf{x}_k^{(i)}$ ; its  $(T-k)$ -th recursion (with  $k = T-1, T-2, \dots, 1$ ) can be represented as a *backward* message passing on the factor graph shown in Fig. 1. In this case, the messages  $\tilde{m}_{\text{bp}}(\mathbf{x}_k^{(i)})$ ,  $\tilde{m}_{\text{be1}}(\mathbf{x}_k^{(i)})$ ,  $\tilde{m}_{\text{be2}}(\mathbf{x}_k^{(i)}) = \tilde{m}_{\text{be}}(\mathbf{x}_k^{(i)})$ , that convey the *backward predicted* pdf of  $\mathbf{x}_k^{(i)}$ , the *first backward filtered* pdf of  $\mathbf{x}_k^{(i)}$  and the *second backward filtered* pdf of  $\mathbf{x}_k^{(i)}$ , respectively, are evaluated on the basis of the messages  $\tilde{m}_{\text{be}}(\mathbf{x}_{k+1}^{(i)})$ ,  $m_{\text{pm}}(\mathbf{x}_k^{(i)})$  and  $m_{\text{ms}}(\mathbf{x}_k^{(i)})$ , respectively; note that  $\tilde{m}_{\text{be}}(\mathbf{x}_{k+1}^{(i)})$  represents the backward filtered pdf of  $\mathbf{x}_k^{(i)}$  computed in the previous (i.e., in the  $(T-(k+1))$ -th) recursion of BIF. Moreover, the first and second backward filtered pdfs of  $\mathbf{x}_k^{(i)}$  are evaluated as (see Fig. 1)

$$\tilde{m}_{\text{be1}}(\mathbf{x}_k^{(i)}) = \tilde{m}_{\text{bp}}(\mathbf{x}_k^{(i)}) m_{\text{pm}}(\mathbf{x}_k^{(i)}), \quad (8)$$

<sup>1</sup>Generally speaking, a pseudo-measurement is a *fictional* measurement that is computed on the basis of statistical information provided by a filtering algorithm different from the one benefiting from it.

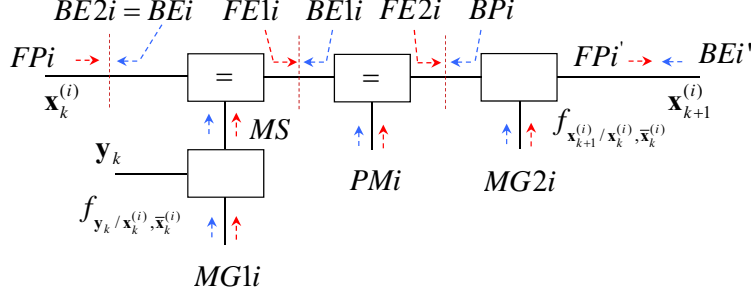


Figure 1: Factor graph involved in the  $k$ -th  $((T-k)$ -th) recursion of Bayesian filtering (BIF) for the substate  $\mathbf{x}_k^{(i)}$  and forward (backward) message passing on it. The flow of messages in the forward (backward) pass are indicated by red (blue) arrows, respectively; the brown vertical lines cutting each graph identify the partitioning associated with formulas (10) (left cut), (11) (central cut) and (12) (right cut). The messages  $\tilde{m}_{fp}(\mathbf{x}_k^{(i)})$ ,  $\tilde{m}_{bp}(\mathbf{x}_k^{(i)})$ ,  $\tilde{m}_{fp}(\mathbf{x}_{k+1}^{(i)})$ ,  $\tilde{m}_{be}(\mathbf{x}_{k+1}^{(i)})$ ,  $m_{ms}(\mathbf{x}_k^{(i)})$ ,  $m_{mgl}(\bar{\mathbf{x}}_k^{(i)})$ ,  $m_{pm}(\mathbf{x}_k^{(i)})$ ,  $\tilde{m}_{fel}(\mathbf{x}_k^{(i)})$  and  $\tilde{m}_{bel}(\mathbf{x}_k^{(i)})$  are denoted  $FPI$ ,  $BPI$ ,  $FPI'$ ,  $BEi$ ,  $MSi$ ,  $MGLi$ ,  $PMi$ ,  $FEli$  and  $BEli$  respectively, to ease reading.

and

$$\tilde{m}_{be2}(\mathbf{x}_k^{(i)}) = \tilde{m}_{be}(\mathbf{x}_k^{(i)}) = \tilde{m}_{be1}(\mathbf{x}_k^{(i)}) m_{ms}(\mathbf{x}_k^{(i)}), \quad (9)$$

respectively, where  $m_{pm}(\mathbf{x}_k^{(i)})$  and  $m_{ms}(\mathbf{x}_k^{(i)})$  are still expressed by Eq. (3) and Eq. (6), respectively. The BIF message passing is initialised by setting  $\tilde{m}_{be}(\mathbf{x}_T^{(i)}) = m_{fe}(\mathbf{x}_T^{(i)})$  in its first recursion and is run for  $k = T-1, T-2, \dots, 1$ . Once the backward pass is over, a solution to problem **P.2** becomes available for the substate  $\mathbf{x}_k^{(i)}$ , since the marginal smoothed pdf  $f(\mathbf{x}_k^{(i)}, \mathbf{y}_{1:T}, \mathbf{z}_{1:T}^{(i)})$  (where  $\mathbf{z}_{1:T}^{(i)}$  is the  $P \cdot T$ -dimensional vector resulting from the ordered concatenation of the all the observed pseudo-measurements  $\{\mathbf{z}_k^{(i)}\}$ ) can be evaluated as<sup>2</sup>

$$f(\mathbf{x}_k^{(i)}, \mathbf{y}_{1:T}, \mathbf{z}_{1:T}^{(i)}) = \tilde{m}_{fp}(\mathbf{x}_k^{(i)}) \tilde{m}_{be2}(\mathbf{x}_k^{(i)}) \quad (10)$$

$$= \tilde{m}_{fel}(\mathbf{x}_k^{(i)}) \tilde{m}_{be1}(\mathbf{x}_k^{(i)}) \quad (11)$$

$$= \tilde{m}_{fe2}(\mathbf{x}_k^{(i)}) \tilde{m}_{bp}(\mathbf{x}_k^{(i)}), \quad (12)$$

with  $k = 1, 2, \dots, T$ . Note that, from a graphical viewpoint, formulas (10)-(12) can be related with the three different partitionings of the graph shown in Fig. 1 (where a specific partitioning is identified by a brown dashed vertical line cutting the graph in two parts).

Given the graphical model represented in Fig. 1, **step 3.** can be accomplished by adopting the same conceptual approach as [19, Sec. III] and [20, Par. II-B], where the factor graphs on which smoothing and filtering, respectively, are based are obtained by merging two sub-graphs, each referring to a distinct substate. For this reason, in this case, the graphical model for the whole state  $\mathbf{x}_k$  is obtained by interconnecting two distinct factor graphs, each structured like the one shown in Fig. 1. In [20, Par. II-B], message passing on the resulting graph is described in detail for the case of *Bayesian filtering*. In this manuscript, instead, our analysis of message passing concerns BIF *and smoothing only*. The devised graph and the messages passed on it are shown in Fig. 2. Note that, in developing our graphical model, it has been assumed that the *smoothed* pdf referring to  $\mathbf{x}_k^{(i)}$  (and conveyed by the message  $m_{sm}(\mathbf{x}_k^{(i)})$ ) is computed on the basis of Eq. (10), i.e. by merging the messages  $\tilde{m}_{fp}(\mathbf{x}_k^{(i)})$  and  $\tilde{m}_{be}(\mathbf{x}_k^{(i)}) = \tilde{m}_{be2}(\mathbf{x}_k^{(i)})$ . Moreover, the following elements (identified by brown lines) have been added to its  $i$ -th sub-graph (with  $i = 1$  and  $2$ ): a) two equality nodes; b) the block  $BIF_i \rightarrow BIF_j$  for extracting useful information from the messages computed on the  $i$ -th sub-graph and delivered to the  $j$ -th one. The former elements allow the  $i$ -th backward information filter to generate copies of the messages  $\tilde{m}_{be}(\mathbf{x}_{k+1}^{(i)})$  and  $m_{sm}(\mathbf{x}_k^{(i)})$ , that are made available to the other sub-graphs.

<sup>2</sup>Note that, similarly as refs. [19] and [23], a *joint* smoothed pdf is considered here in place of the corresponding *posterior* pdf.

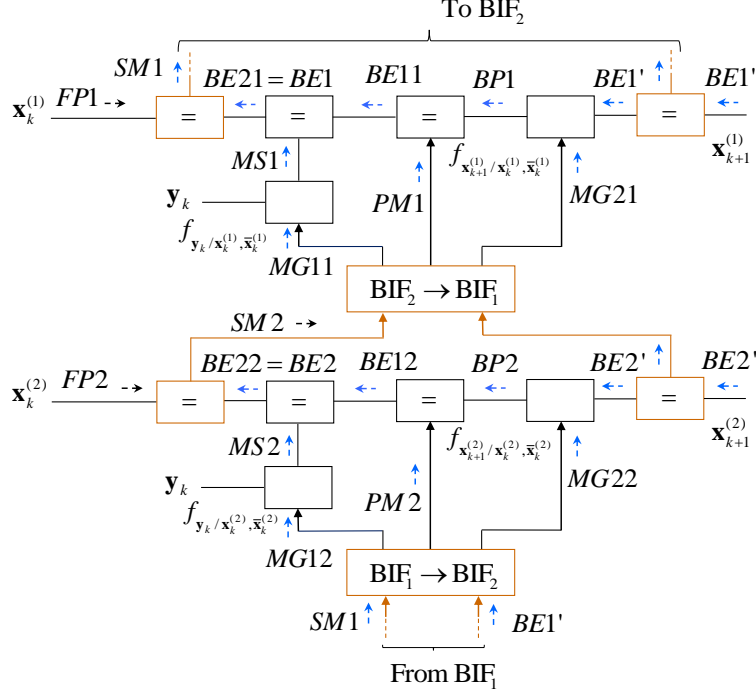


Figure 2: Graphical model based on the sub-graph shown in Fig. 1 and referring to the interconnection of two backward information filters. The message computed in the backward (forward) pass are identified by blue (black) arrows. The message  $m_{sm}(\mathbf{x}_k^{(i)})$  is denoted  $SMi$  to ease reading.

In the latter element, instead, the messages  $m_{pm}(\mathbf{x}_k^{(i)})$  (see Eq. (3)) and  $m_{mgq}(\bar{\mathbf{x}}_k^{(i)})$  (with  $q = 1$  and  $2$ ; see Eqs. (6) and (7)) are computed; note that this block is connected to *oriented* edges only, i.e. to edges on which the flow of messages is unidirectional.

Given the graphical model represented in Fig. 2, **step 4.** can be easily accomplished. In fact, recursive BIF and smoothing algorithms can be derived by systematically applying the SPA to it after that a proper *scheduling* has been established for message passing. In doing so, we must always keep in mind that:

- 1) Message passing on the  $i$ -th subgraph represents BIF/smoothing for the substate  $\mathbf{x}_k^{(i)}$ ; the exchange of messages between the sub-graphs, instead, allows us to represent the interaction of two interconnected BIF/smoothing algorithms in a effective and rigorous way.
- 2) Different approximations can be used for the predicted/filtered/smoothed pdfs computed in the message passing on each of the two sub-graphs and for the involved Markov/observation models. For this reason, generally speaking, the two interconnected filtering/BIF/smoothing algorithms are not required to be of the same type.
- 3) The  $k$ -th recursion of the overall BIF algorithm is fed by the backward estimates  $\tilde{m}_{be}(\mathbf{x}_{k+1}^{(1)})$  ( $BE1'$ ) and  $\tilde{m}_{be}(\mathbf{x}_{k+1}^{(2)})$  ( $BE2'$ ), and generates the new backward predictions  $\tilde{m}_{bp}(\mathbf{x}_k^{(1)})$  ( $BP1$ ) and  $\tilde{m}_{bp}(\mathbf{x}_k^{(2)})$  ( $BP2$ ), and the two couples of filtered densities  $\{(\tilde{m}_{be1}(\mathbf{x}_k^{(i)}), \tilde{m}_{be2}(\mathbf{x}_k^{(i)})), i = 1, 2\}$  ( $\{BE1i, BE2i, i = 1, 2\}$ ). Moreover, merging the predicted densities computed in the forward pass (i.e., the messages  $\{FPi\}$ ) with the second backward filtered densities (i.e., the messages  $\{BE2i = BEi\}$ ) allows us to generate the smoothed pdfs for each substate according to Eq. (10). However, a *joint* filtered/smoothed density for the whole state  $\mathbf{x}_k$  is *unavailable*.
- 4) Specific algorithms are employed to compute the pseudo-measurement and the nuisance substate pdfs in the  $BIF_i \rightarrow BIF_j$  blocks appearing in Fig. 2. These algorithms depend on the considered SSM and on the selected message scheduling; for this reason, a general description of their structure cannot be provided.
- 5) The graphical model shown in Fig. 2, unlike the one illustrated in Fig. 1, is *not cycle free*. The presence of cycles raises the problems of identifying all the messages that can be iteratively refined and establishing the order according to which they are computed. Generally speaking, iterative message passing on the

devised graphical model involves both the couple of measurement updates and the backward prediction accomplished in each of the interconnected backward information filters. In fact, this should allow each filter to progressively refine the nuisance substate density employed in its second measurement update and backward prediction, and improve the quality of the pseudo-measurements exploited in its first measurement update. For this reason, if  $n_i$  iterations are run, the overall computational complexity of each recursion is multiplied by  $n_i$ .

The final important issue about the graphical model devised for both Bayesian filtering and BIF concerns the possible presence of *redundancy*. In all the considerations illustrated above, *disjoint* substates  $\mathbf{x}_k^{(1)}$  and  $\mathbf{x}_k^{(2)}$  have been assumed. Actually, in ref. [20], it has been shown that our graphical approach can be also employed if the substates  $\mathbf{x}_k^{(1)}$  and  $\mathbf{x}_k^{(2)}$  cover  $\mathbf{x}_k$ , but do not necessarily form a partition of it. In other words, some overlapping between these two substates is admitted. When this occurs, the forward/backward filtering algorithm run over the whole graphical model contains a form of *redundancy*, since  $N_d \triangleq D_1 + D_2 - D$  elements of the state vector  $\mathbf{x}_k$  are independently estimated by the interconnected forward/backward filters. The parameter  $N_d$  can be considered as the *degree of redundancy* characterizing the filtering/smoothing algorithm. Moreover, in ref. [20], it has been shown that the presence of redundancy in a Bayesian filtering algorithm can significantly enhance its tracking capability (i.e., reduce its probability of divergence); however, this result is obtained at the price of an increased complexity with respect to the case in which the interconnected filters are run over disjoint substates.

### III. Double Backward Information Filtering and Smoothing Algorithms for Conditionally Linear Gaussian State Space Models

In this section we focus on the development of two new DBS algorithms for conditionally linear Gaussian models. We first describe the graphical models on which these algorithms are based; then, we provide a detailed description of the computed messages and their scheduling in a specific case.

#### A. Graphical Modelling

In this paragraph, we focus on a specific instance of the graphical model illustrated in Fig. 2, since we make the same specific choices as ref. [20] for both the considered SSM and the two Bayesian filters employed in the forward pass. For this reason, we assume that: a) the SSM described by eqs. (1)-(2) is conditionally linear Gaussian [18], [23], [24], so that its state vector  $\mathbf{x}_k$  can be partitioned into its *linear component*  $\mathbf{x}_k^{(L)}$  and its *nonlinear component*  $\mathbf{x}_k^{(N)}$  (having sizes  $D_L$  and  $D_N$ , respectively, with  $D_N + D_L = D$ ); b) the dual Bayesian filter employed in the forward pass results from the interconnection of an *extended Kalman filter* with a *particle filter*<sup>3</sup> (these filters are denoted  $F_1$  and  $F_2$ , respectively), as described in detail in ref. [20]. As far as the last point is concerned, it is also worth mentioning that, on the one hand, filter  $F_2$  estimates the nonlinear state component only (so that  $\mathbf{x}_k^{(2)} = \mathbf{x}_k^{(N)}$  and  $\bar{\mathbf{x}}_k^{(2)} = \mathbf{x}_k^{(L)}$ ) and approximates the predicted/filtered densities of this component through a set of  $N_p$  weighted particles. On the other hand, filter  $F_1$  employs a Gaussian approximation of all its predicted/filtered densities, and works on the *whole system state* or on the *linear state component*. In the first case (denoted **C.1** in the following), we have that  $\mathbf{x}_k^{(1)} = \mathbf{x}_k$  and  $\bar{\mathbf{x}}_k^{(1)}$  is empty, so that both  $F_1$  and  $F_2$  estimate the nonlinear state component (for this reason, the corresponding degree of redundancy in the developed smoothing algorithm is  $N_d = D_N$ ); in the second case (denoted **C.2** in the following), instead,  $\mathbf{x}_k^{(1)} = \mathbf{x}_k^{(L)}$  and  $\bar{\mathbf{x}}_k^{(1)} = \mathbf{x}_k^{(N)}$ , so that filters  $F_1$  and  $F_2$  estimate *disjoint* substates (consequently,  $N_d = 0$ ).

Our selection of the forward filtering scheme has the following implications on the developed DBIF scheme. The first backward information filter (denoted  $BIF_1$ ) is the backward filter associated with an extended Kalman filter operating over on the *whole system state* (case **C.1**) or on the *linear state component* (case **C.2**). The second backward filter (denoted  $BIF_2$ ), instead, is a backward filter associated with a particle filter operating on the nonlinear state component only. In practice, following [17–19],  $BIF_2$  is employed to

<sup>3</sup>In particular, a *sequential importance resampling* filter is employed [25].



update the weights of all the elements of the particle set generated by filter  $F_2$  in the forward pass. Then, based on the graphical model shown in Fig. 2, the factor graph illustrated in Fig. 3 can be drawn for case **C.1**. It is important to point out that:

1) The first backward information filter (BIF<sub>1</sub>) is based on *linearised* (and, consequently, *approximate*) Markov/measurement models, whereas the second one (BIF<sub>2</sub>) relies on *exact* models, as explained in more detail below. These models are the same as those employed in ref. [20].

2) Since the nuisance substate  $\bar{\mathbf{x}}_k^{(1)}$  is empty, no marginalization is required in BIF<sub>1</sub>; for this reason, the messages  $\{m_{\text{mg}q}(\bar{\mathbf{x}}_k^{(1)}); q = 1, 2\}$  (i.e., *MG11* and *MG21*) visible in Fig. 2 do not appear in Fig. 3. Moreover, the message  $m_{\text{sm}}(\mathbf{x}_k^{(1)}) = m_{\text{sm}}(\mathbf{x}_k)$  is generated on the basis of Eq. (11), instead of Eq. (10).

3) The backward filtered pdf  $\tilde{m}_{\text{be}}(\mathbf{x}_{k+1}^{(2)}) = \tilde{m}_{\text{be}}(\mathbf{x}_{k+1}^{(N)})$  and the smoothed pdf  $m_{\text{sm}}(\mathbf{x}_k^{(2)}) = m_{\text{sm}}(\mathbf{x}_k^{(N)})$  (i.e., the messages *BE2'* and *SM2*, respectively) feed the BIF<sub>2</sub>→BIF<sub>1</sub> block, where they are processed jointly to generate the pseudo-measurement message  $m_{\text{pm}}(\mathbf{x}_k^{(1)}) = m_{\text{pm}}(\mathbf{x}_k)$  (*PM1*) feeding filter  $F_1$ . Similarly, the backward filtered pdf  $\tilde{m}_{\text{be}}(\mathbf{x}_{k+1}^{(1)}) = \tilde{m}_{\text{be}}(\mathbf{x}_{k+1})$  (*BE1'*) and the smoothed pdf  $m_{\text{sm}}(\mathbf{x}_k^{(1)}) = m_{\text{sm}}(\mathbf{x}_k)$  (*SM1*) feed the BIF<sub>1</sub>→BIF<sub>2</sub> block, where the pseudo-measurement message  $m_{\text{pm}}(\mathbf{x}_k^{(2)}) = m_{\text{pm}}(\mathbf{x}_k^{(N)})$  (*PM2*) and the messages  $\{m_{\text{mg}q}(\bar{\mathbf{x}}_k^{(2)}) = m_{\text{mg}q}(\bar{\mathbf{x}}_k^{(L)}); q = 1, 2\}$  (i.e., *MG12* and *MG22*) are evaluated.

In the remaining part of this paragraph, we first provide various details about the backward filters BIF<sub>1</sub> and BIF<sub>2</sub>, and the way pseudo-measurements are computed for each of them; then, we comment on how the factor graph shown in Fig. 3 should be modified if case **C.2** is considered.

BIF<sub>1</sub> - This backward filter is based on the *linearized* versions of Eqs. (1) and (2), i.e. on the models (e.g., see [1, pp. 194-195] and [20, Par. III-A])

$$\mathbf{x}_{k+1} = \mathbf{F}_k \mathbf{x}_k + \mathbf{u}_k + \mathbf{w}_k \quad (13)$$

and

$$\mathbf{y}_k = \mathbf{H}_k^T \mathbf{x}_k + \mathbf{v}_k + \mathbf{e}_k, \quad (14)$$

respectively; here,  $\mathbf{F}_k \triangleq [\partial \mathbf{f}_k(\mathbf{x}) / \partial \mathbf{x}]_{\mathbf{x}=\mathbf{x}_{\text{fe},k}}$ ,  $\mathbf{u}_k \triangleq \mathbf{f}_k(\mathbf{x}_{\text{fe},k}) - \mathbf{F}_k \mathbf{x}_{\text{fe},k}$ ,  $\mathbf{H}_k^T \triangleq [\partial \mathbf{h}_k(\mathbf{x}) / \partial \mathbf{x}]_{\mathbf{x}=\mathbf{x}_{\text{fp},k}}$ ,  $\mathbf{v}_k \triangleq \mathbf{h}_k(\mathbf{x}_{\text{fp},k}) - \mathbf{H}_k^T \mathbf{x}_{\text{fp},k}$  and  $\mathbf{x}_{\text{fp},k}(\mathbf{x}_{\text{fe},k})$  is the *forward prediction* (*forward estimate*) of  $\mathbf{x}_k$  computed by  $F_1$  in its  $(k-1)$ -th ( $k$ -th) recursion. Consequently, the approximate models

$$\tilde{f}(\mathbf{x}_{k+1} | \mathbf{x}_k) = \mathcal{N}(\mathbf{x}_k; \mathbf{F}_k \mathbf{x}_k + \mathbf{u}_k, \mathbf{C}_w) \quad (15)$$

and

$$\tilde{f}(\mathbf{y}_k | \mathbf{x}_k) = \mathcal{N}(\mathbf{x}_k; \mathbf{H}_k^T \mathbf{x}_k + \mathbf{v}_k, \mathbf{C}_e) \quad (16)$$

appear in the graphical model shown in Fig. 3.

BIF<sub>2</sub> - In developing this backward filter, the state vector  $\mathbf{x}_k$  is represented as the ordered concatenation of its linear component  $\mathbf{x}_k^{(L)} \triangleq [x_{0,k}^{(L)}, x_{1,k}^{(L)}, \dots, x_{D_L-1,k}^{(L)}]^T$  and its nonlinear component  $\mathbf{x}_k^{(N)} \triangleq [x_{0,k}^{(N)}, x_{1,k}^{(N)}, \dots, x_{D_N-1,k}^{(N)}]^T$ . Based on [23, eq. (3)], the Markov model

$$\mathbf{x}_{k+1}^{(N)} = \mathbf{A}_k^{(N)}(\mathbf{x}_k^{(N)}) \mathbf{x}_k^{(L)} + \mathbf{f}_k^{(N)}(\mathbf{x}_k^{(N)}) + \mathbf{w}_k^{(N)} \quad (17)$$

is adopted for the nonlinear state component (this model corresponds to the last  $D_N$  lines of Eq. (1)); here,  $\mathbf{f}_k^{(N)}(\mathbf{x}_k^{(N)})$  ( $\mathbf{A}_k^{(N)}(\mathbf{x}_k^{(N)})$ ) is a time-varying  $D_N$ -dimensional real function ( $D_N \times D_L$  real matrix) and  $\mathbf{w}_k^{(N)}$  consists of the last  $D_N$  elements of the noise term  $\mathbf{w}_k$  appearing in Eq. (1) (the covariance matrix of  $\mathbf{w}_k^{(N)}$  is denoted  $\mathbf{C}_w^{(N)}$ ). Moreover, it is assumed that the observation model (2) can be put in the form (see [20, eq. (31)] or [23, eq. (4)])

$$\mathbf{y}_k = \mathbf{g}_k(\mathbf{x}_k^{(N)}) + \mathbf{B}_k(\mathbf{x}_k^{(N)}) \mathbf{x}_k^{(L)} + \mathbf{e}_k, \quad (18)$$

where  $\mathbf{g}_k(\mathbf{x}_k^{(N)})$  ( $\mathbf{B}_k(\mathbf{x}_k^{(N)})$ ) is a time-varying  $P$ -dimensional real function ( $P \times D_L$  real matrix). Consequently, the considered backward filter is based on the *exact* pdfs

$$\begin{aligned} & f(\mathbf{x}_{k+1}^{(N)} | \mathbf{x}_k^{(N)}, \mathbf{x}_k^{(L)}) \\ &= \mathcal{N}(\mathbf{x}_k^{(N)}; \mathbf{A}_k^{(N)}(\mathbf{x}_k^{(N)}) \mathbf{x}_k^{(L)} + \mathbf{f}_k^{(N)}(\mathbf{x}_k^{(N)}), \mathbf{C}_w^{(N)}) \end{aligned} \quad (19)$$

and

$$\begin{aligned} & f\left(\mathbf{y}_k \mid \mathbf{x}_k^{(N)}, \mathbf{x}_k^{(L)}\right) \\ &= \mathcal{N}\left(\mathbf{x}_k; \mathbf{g}_k\left(\mathbf{x}_k^{(N)}\right) + \mathbf{B}_k\left(\mathbf{x}_k^{(N)}\right) \mathbf{x}_k^{(L)}, \mathbf{C}_e\right), \end{aligned} \quad (20)$$

both appearing in the graphical model drawn in Fig. 3.

*Computation of the pseudo-measurements for the first backward filter* - Filter BIF<sub>1</sub> is fed by pseudo-measurement information about the *whole state*  $\mathbf{x}_k$ . The method for computing these information is similar to the one illustrated in ref. [19, Sects. III-IV] and can be summarised as follows. The pseudo-measurements about the nonlinear state component are represented by the  $N_p$  particles conveyed by the smoothed pdf  $m_{\text{sm}}(\mathbf{x}_k^{(N)})$  (SM2). On the other hand,  $N_p$  pseudo-measurements about the linear state component are evaluated by means of the same method employed by *marginalized particle filtering* (MPF) for this task. This method is based on the idea that the random vector (see Eq. (17))

$$\mathbf{z}_k^{(L)} \triangleq \mathbf{x}_{k+1}^{(N)} - \mathbf{f}_k^{(N)}\left(\mathbf{x}_k^{(N)}\right), \quad (21)$$

depending on the *nonlinear state component only*, must equal the sum

$$\mathbf{A}_k^{(N)}\left(\mathbf{x}_k^{(N)}\right) \mathbf{x}_k^{(L)} + \mathbf{w}_k^{(N)}, \quad (22)$$

that depends on the *linear state component*. For this reason,  $N_p$  realizations of  $\mathbf{z}_k^{(L)}$  (21) are computed in the BIF<sub>2</sub>→BIF<sub>1</sub> block on the basis of the messages  $\tilde{m}_{\text{be}}(\mathbf{x}_{k+1}^{(N)})$  (BE2') and  $m_{\text{sm}}(\mathbf{x}_k^{(N)})$ , and are treated as measurements about  $\mathbf{x}_k^{(L)}$ .

*Computation of the pseudo-measurements for the second backward filter* - The messages  $\tilde{m}_{\text{be}}(\mathbf{x}_{k+1})$  (BE1') and  $m_{\text{sm}}(\mathbf{x}_k)$  (SM1) feeding the BIF<sub>1</sub>→BIF<sub>2</sub> block are employed for: a) generating the messages  $\{m_{\text{mg}q}(\mathbf{x}_k^{(L)}); q = 1, 2\}$  required to integrate out the dependence of the state update and measurement models (i.e., of the densities  $f(\mathbf{x}_{k+1}^{(N)} | \mathbf{x}_k^{(N)}, \mathbf{x}_k^{(L)})$  (19) and  $f(\mathbf{y}_k | \mathbf{x}_k^{(N)}, \mathbf{x}_k^{(L)})$  (20), respectively) on the substate  $\mathbf{x}_k^{(L)}$ ; b) generating pseudo-measurement information about  $\mathbf{x}_k^{(N)}$ . As far as the last point is concerned, the approach we adopt is the same as that developed for *dual marginalized particle filtering* (dual MPF) in ref. [23, Sec. V] and also adopted in particle smoothing [19, Sects. III-IV]. The approach relies on the Markov model

$$\mathbf{x}_{k+1}^{(L)} = \mathbf{A}_k^{(L)}\left(\mathbf{x}_k^{(N)}\right) \mathbf{x}_k^{(L)} + \mathbf{f}_k^{(L)}\left(\mathbf{x}_k^{(N)}\right) + \mathbf{w}_k^{(L)}, \quad (23)$$

referring to the *linear state component* (see [19, eq. (1)] or [23, eq. (3)]); in the last expression,  $\mathbf{f}_k^{(L)}(\mathbf{x}_k^{(N)})$  ( $\mathbf{A}_k^{(L)}(\mathbf{x}_k^{(N)})$ ) is a time-varying  $D_L$ -dimensional real function ( $D_L \times D_L$  real matrix), and  $\mathbf{w}_k^{(L)}$  consists of the first  $D_L$  elements of the noise term  $\mathbf{w}_k$  appearing in (1) (the covariance matrix of  $\mathbf{w}_k^{(L)}$  is denoted  $\mathbf{C}_w^{(L)}$ , and independence between  $\{\mathbf{w}_k^{(L)}\}$  and  $\{\mathbf{w}_k^{(N)}\}$  is assumed for simplicity). From Eq. (23) it is easily inferred that the random vector

$$\mathbf{z}_k^{(N)} \triangleq \mathbf{x}_{k+1}^{(L)} - \mathbf{A}_k^{(L)}\left(\mathbf{x}_k^{(N)}\right) \mathbf{x}_k^{(L)}, \quad (24)$$

must equal the sum

$$\mathbf{f}_k^{(L)}\left(\mathbf{x}_k^{(N)}\right) + \mathbf{w}_k^{(L)}, \quad (25)$$

that depends on  $\mathbf{x}_k^{(N)}$  *only*; for this reason,  $\mathbf{z}_k^{(N)}$  (24) can be interpreted as a pseudo-measurement about  $\mathbf{x}_k^{(N)}$ . In this case, the pseudo-measurement information is conveyed by the message  $m_{\text{pm}}(\mathbf{x}_k^{(N)})$  (PM2) that expresses the *correlation* between the pdf of the random vector  $\mathbf{z}_k^{(N)}$  (24) (computed on the basis of the statistical information about the linear state component made available by BIF<sub>1</sub>) and the pdf obtained for  $\mathbf{z}_k^{(N)}$  under the assumption that this vector is expressed by Eq. (25). The message  $m_{\text{pm}}(\mathbf{x}_k^{(N)})$  is evaluated for each of the particles representing  $\mathbf{x}_k^{(N)}$  in BIF<sub>2</sub>; this results in a set of  $N_p$  particle weights employed in the first measurement update of BIF<sub>2</sub> and different from those computed on the basis of  $\mathbf{y}_k$  (18) in its second measurement update.

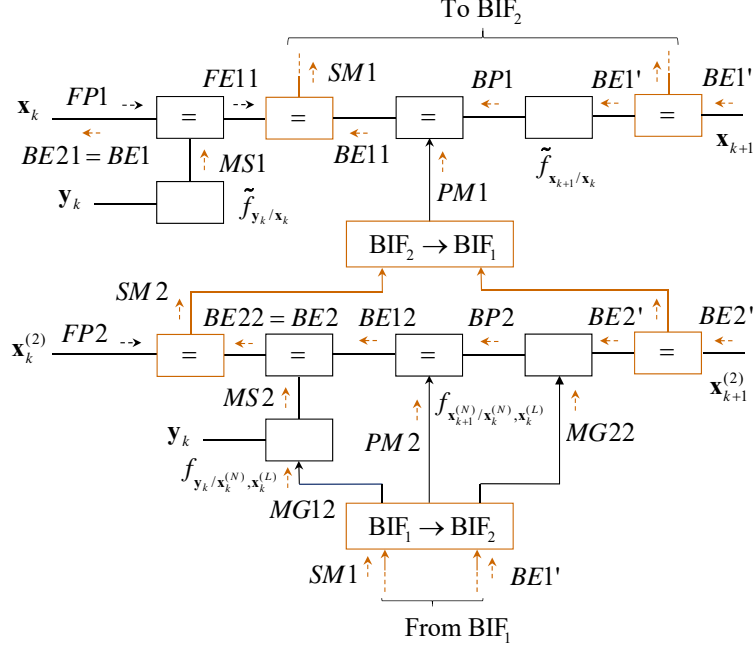


Figure 3: Graphical model referring to the interconnection of two backward information filters, one paired with an extended Kalman filter, the other one with a particle filter.

A graphical model similar to the one shown in Fig. 3 can be easily derived from the general model appearing in Fig. 2 for case **C.2** too. The relevant differences with respect to case **C.1** can be summarized as follows:

- 1) The backward filters  $\text{BIF}_1$  and  $\text{BIF}_2$  estimate  $\mathbf{x}_k^{(1)} = \mathbf{x}_k^{(L)}$  and  $\mathbf{x}_k^{(2)} = \mathbf{x}_k^{(N)}$ , respectively; consequently, their nuisance substates are  $\bar{\mathbf{x}}_k^{(1)} = \mathbf{x}_k^{(N)}$  and  $\bar{\mathbf{x}}_k^{(2)} = \mathbf{x}_k^{(L)}$ , respectively.
- 2) The  $\text{BIF}_2 \rightarrow \text{BIF}_1$  block is fed by the backward predicted/smoothed pdfs computed by  $\text{BIF}_2$ ; such pdfs are employed for: a) generating the messages  $m_{\text{mg}1}(\mathbf{x}_k^{(N)})$  ( $\text{MG11}$ ) and  $m_{\text{mg}2}(\mathbf{x}_k^{(N)})$  ( $\text{MG21}$ ) required to integrate out the dependence of the Markov model (see Eq. (23))

$$\begin{aligned}
 & f(\mathbf{x}_{k+1}^{(L)} | \mathbf{x}_k^{(N)}, \mathbf{x}_k^{(L)}) \\
 &= \mathcal{N}(\mathbf{x}_k^{(L)}; \mathbf{A}_k^{(L)}(\mathbf{x}_k^{(N)}) \mathbf{x}_k^{(L)} + \mathbf{f}_k^{(L)}(\mathbf{x}_k^{(N)}), \mathbf{C}_w^{(L)})
 \end{aligned} \tag{26}$$

and of the measurement model  $f(\mathbf{y}_k | \mathbf{x}_k^{(N)}, \mathbf{x}_k^{(L)})$  (20), respectively, on  $\mathbf{x}_k^{(N)}$ ; b) generating pseudo-measurement information about the substate  $\mathbf{x}_k^{(L)}$  only. As far as point a) is concerned, it is also important to point out that the model  $f(\mathbf{y}_k | \mathbf{x}_k^{(N)}, \mathbf{x}_k^{(L)})$  ( $f(\mathbf{x}_{k+1}^{(L)} | \mathbf{x}_k^{(N)}, \mathbf{x}_k^{(L)})$ ) on which  $\text{BIF}_1$  is based can be derived from Eq. (20) (Eq. (26)) after setting  $\mathbf{x}_k^{(N)} = \mathbf{x}_{\text{fp},k}^{(N)}$  ( $\mathbf{x}_k^{(N)} = \mathbf{x}_{\text{fe},k}^{(N)}$ ); here,  $\mathbf{x}_{\text{fp},k}^{(N)}$  ( $\mathbf{x}_{\text{fe},k}^{(N)}$ ) denotes the prediction (the estimate) of  $\mathbf{x}_k^{(N)}$  evaluated by the filter  $\text{F}_2$  in the forward pass (further details about this can be found in ref. [20, Par. III-A])

The derivation of specific DBS algorithms based on the graphical model illustrated in Fig. 3 requires defining the scheduling of the messages passed on it and deriving mathematical expressions for such messages. These issues are investigated in detail in the following paragraph.

## B. Message Scheduling and Computation

In this paragraph, the scheduling of a new *recursive smoothing algorithm*, called *double Bayesian smoothing algorithm* (DBSA) and based on the graphical model illustrated in Fig. 3, and a simplified version of it are

described. Moreover, the expression of the messages computed by the DBSA are illustrated.

The scheduling adopted in the DBSA mimics the one employed in ref. [19] (which, in turn, has been inspired by [17] and [18]). Moreover, in devising it, the presence of cycles in the underlying graphical model has been accounted for by allowing multiple passes of messages over the edges which such cycles consist of; this explains why an iterative procedure is embedded in each recursion of the DBSA. Our description of the devised scheduling is based on Fig. 4, that refers to the  $(T - k)$ -th recursion of the backward pass of the DBSA (with  $k = T - 1, T - 2, \dots, 1$ ) and to the  $n$ -th iteration accomplished within this recursion (with  $n = 1, 2, \dots, n_i$ , where  $n_i$  represents the overall number of iterations). Note that, in this figure, a simpler notation is adopted for most of the considered messages to ease reading; in particular, the symbols  $q$ ,  $q^{(n)}$ ,  $qL$ ,  $qL^{(n)}$ ,  $qN$  and  $qN^{(n)}$  are employed to represent the messages  $m_q(\mathbf{x}_k)$ ,  $m_q^{(n)}(\mathbf{x}_k)$ ,  $m_q(\mathbf{x}_k^{(L)})$ ,  $m_q^{(n)}(\mathbf{x}_k^{(L)})$ ,  $m_q(\mathbf{x}_k^{(N)})$ ,  $m_q^{(n)}(\mathbf{x}_k^{(N)})$ , respectively (independently of the presence of an arrow and of its direction in the considered message), and the presence of the superscript  $(n)$  in a given message means that such a message is computed in the  $n$ -th iteration. Moreover, each of the passed messages conveys a Gaussian pdf or a pdf in particle form. In the first case, the pdf of a state/substate  $\mathbf{x}$  is conveyed by the message

$$m_G(\mathbf{x}) = \mathcal{N}(\mathbf{x}; \eta, \mathbf{C}), \quad (27)$$

where  $\eta$  and  $\mathbf{C}$  denote the mean and the covariance of  $\mathbf{x}$ , respectively. In the second case, instead, its pdf is conveyed by the message

$$m_P(\mathbf{x}) = \sum_{j=1}^{N_p} m_{P,j}(\mathbf{x}), \quad (28)$$

where

$$m_{P,j}(\mathbf{x}) \triangleq w_j \delta(\mathbf{x} - \mathbf{x}_j) \quad (29)$$

is its  $j$ -th *component*; this represents the contribution of the  $j$ -th particle  $\mathbf{x}_j$  and its weight  $w_j$  to  $m_P(\mathbf{x})$  (28). The nature of each message can be easily inferred from Fig. 4, since Gaussian messages and messages in particle form are identified by blue and red arrows, respectively.

Before analysing the adopted scheduling, we need to define the *input* messages feeding the considered recursion of the DBSA and the outputs that such a recursion produces. In the considered recursion, the DBSA *input* messages originate from:

1) The  $k$ -th recursion of the forward pass. These messages have been generated by the DBF technique paired with the considered BIF scheme and, in particular, by the DBF algorithm derived in ref. [20, Par. III-B], and have been stored (so that they are made available to the backward pass).

2) The previous recursion (i.e., the  $(T - k - 1)$ -th recursion) of the DBSA itself.

As far as the input messages computed in the forward pass are concerned, BIF<sub>1</sub> is fed by the Gaussian messages (see Fig. 4)

$$\vec{m}_{\text{fp}}(\mathbf{x}_k) \triangleq \mathcal{N}(\mathbf{x}_k; \eta_{\text{fp},k}, \mathbf{C}_{\text{fp},k}). \quad (30)$$

and

$$\vec{m}_{\text{fe1}}(\mathbf{x}_k) \triangleq \mathcal{N}(\mathbf{x}_k; \eta_{\text{fe1},k}, \mathbf{C}_{\text{fe1},k}), \quad (31)$$

that convey the predicted pdf and the first filtered pdf, respectively, computed by F<sub>1</sub> (in its  $(k - 1)$ -th and in its  $k$ -th recursion, respectively). The covariance matrix  $\mathbf{C}_{\text{fe1},k}$  and the mean vector  $\eta_{\text{fe1},k}$  are evaluated on the basis of the associated precision matrix (see [19, Eqs. (14)-(17)])

$$\mathbf{W}_{\text{fe1},k} = \mathbf{H}_k \mathbf{W}_e \mathbf{H}_k^T + \mathbf{W}_{\text{fp},k} \quad (32)$$

and of the associated transformed mean vector

$$\mathbf{w}_{\text{fe1},k} = \mathbf{H}_k \mathbf{W}_e (\mathbf{y}_k - \mathbf{v}_k) + \mathbf{w}_{\text{fp},k}, \quad (33)$$

respectively; here,  $\mathbf{W}_e \triangleq \mathbf{C}_e^{-1}$ ,  $\mathbf{W}_{\text{fp},k} \triangleq (\mathbf{C}_{\text{fp},k})^{-1}$  and  $\mathbf{w}_{\text{fp},k} \triangleq \mathbf{W}_{\text{fp},k} \eta_{\text{fp},k}$ .

The backward filter BIF<sub>2</sub>, instead, is fed by the forward messages  $\vec{m}_{\text{fp}}(\mathbf{x}_k^{(N)})$  and  $\vec{m}_{\text{fe1}}(\mathbf{x}_k^{(N)})$ , both in particle form (see Fig. 4); their  $j$ -th components are represented by

$$\vec{m}_{\text{fp},j}(\mathbf{x}_k^{(N)}) \triangleq w_p \delta(\mathbf{x}_k^{(N)} - \mathbf{x}_{k,j}^{(N)}), \quad (34)$$

and

$$\tilde{m}_{\text{fe}1,j}(\mathbf{x}_k^{(N)}) \triangleq w_{\text{fe},k,j} \delta(\mathbf{x}_k^{(N)} - \mathbf{x}_{k,j}^{(N)}), \quad (35)$$

respectively, with  $j = 1, 2, \dots, N_p$ ; here,  $\mathbf{x}_{k,j}^{(N)}$  is the  $j$ -th particle predicted by  $F_2$  in the  $(k-1)$ -th recursion of the forward pass (i.e., the  $j$ -th element of the particle set  $S_k \triangleq \{\mathbf{x}_{k,1}^{(N)}, \mathbf{x}_{k,2}^{(N)}, \dots, \mathbf{x}_{k,N_p}^{(N)}\}$ ), whereas  $w_p \triangleq 1/N_p$  and  $w_{\text{fe},k,j}$  represent the (normalised) weights assigned to this particle in the messages  $\tilde{m}_{\text{fp}}(\mathbf{x}_k^{(N)})$  and  $\tilde{m}_{\text{fe}1}(\mathbf{x}_k^{(N)})$ , respectively.

On the other hand, the input messages originating from the previous recursion of the backward pass are the backward filtered Gaussian pdf

$$\tilde{m}_{\text{be}}(\mathbf{x}_{k+1}) \triangleq \mathcal{N}(\mathbf{x}_{k+1}; \eta_{\text{be},k+1}, \mathbf{C}_{\text{be},k+1}) \quad (36)$$

and the backward pdf

$$\tilde{m}_{\text{be}}(\mathbf{x}_{k+1}^{(N)}) \triangleq \delta(\mathbf{x}_{k+1}^{(N)} - \mathbf{x}_{\text{be},k+1}^{(N)}), \quad (37)$$

that represents  $\mathbf{x}_{k+1}^{(N)}$  through a single particle having unit weight; these are computed by  $\text{BIF}_1$  and  $\text{BIF}_2$ , respectively. Consequently, in the considered recursion of the backward pass, all the forward/backward input messages described above are processed to compute: 1) the new backward pdfs  $\tilde{m}_{\text{be}}(\mathbf{x}_k)$  and  $\tilde{m}_{\text{be}}(\mathbf{x}_k^{(N)})$ ; 2) the smoothed statistical information about  $\mathbf{x}_k$  ( $\mathbf{x}_k^{(N)}$ ) by properly merging forward and backward messages generated by  $F_1$  and  $\text{BIF}_1$  ( $F_2$  and  $\text{BIF}_2$ ). As far as the last point is concerned, the evaluation of smoothed information is based on the same conceptual approach as [17–19]. In fact, in our work, the *joint* smoothing pdf  $f(\mathbf{x}_{1:T}|\mathbf{y}_{1:T})$  is estimated by providing multiple (say,  $M$ ) *realizations* of it. A single realization (i.e., a single *smoothed* state trajectory) is computed in each backward pass; consequently, generating the whole output of the DBSA requires running a single forward pass and  $M$  distinct backward passes. Moreover, the evaluation of the smoothed information is based on the factorisation (10) or (11). In fact, these formulas are exploited to merge the statistical information emerging from the forward pass with that computed in any of the  $M$  backward passes.

The message passing on which the DBSA is based can be divided in the three consecutive phases listed below.

**Phase I** - In the first phase, the backward predicted pdf  $\tilde{m}_1(\mathbf{x}_k)$  (1) is computed on the basis of the backward filtered pdf  $\tilde{m}_{\text{be}}(\mathbf{x}_{k+1})$  ( $BE1'$ ).

**Phase II** - In this phase, an iterative procedure for computing and progressively refining the first backward filtered and the smoothed pdfs of the whole state ( $\text{BIF}_1$ ), and the second filtered and the smoothed pdfs of the nonlinear state component ( $\text{BIF}_2$ ) is carried out. More specifically, in the  $n$ -th iteration of this procedure (with  $n = 1, 2, \dots, n_i$ ), the ordered computation of the following messages is accomplished in eight consecutive steps (see Fig. 4): 1)  $m_2^{(n)}(\mathbf{x}_k)$  ( $2^{(n)}$ ; pdf conveying pseudo-measurement information about  $\mathbf{x}_k$ ); 2)  $\tilde{m}_3^{(n)}(\mathbf{x}_k)$  ( $3^{(n)}$ ; *first backward filtered pdf of  $\mathbf{x}_k$* ); 3)  $m_4^{(n)}(\mathbf{x}_k)$  ( $4^{(n)}$ ; *smoothed pdf of  $\mathbf{x}_k$* ); 4)  $m_1^{(n)}(\mathbf{x}_k^{(L)})$  ( $1L^{(n)}$ ; pdf for integrating out the dependence of  $f(\mathbf{x}_{k+1}^{(N)}|\mathbf{x}_k^{(N)}, \mathbf{x}_k^{(L)})$  and  $f(\mathbf{y}_k|\mathbf{x}_k^{(N)}, \mathbf{x}_k^{(L)})$  on  $\mathbf{x}_k^{(L)}$ ),  $\tilde{m}_3^{(n)}(\mathbf{x}_k^{(N)})$  ( $3N^{(n)}$ ; *backward predicted pdf of  $\mathbf{x}_k^{(N)}$* ); 5)  $m_2^{(n)}(\mathbf{x}_k^{(N)})$  ( $2N^{(n)}$ ; pdf conveying pseudo-measurement information about  $\mathbf{x}_k^{(N)}$ ); 6)  $m_4^{(n)}(\mathbf{x}_k^{(N)})$  ( $4N^{(n)}$ ; *first backward filtered pdf of  $\mathbf{x}_k^{(N)}$* ); 7)  $m_5^{(n)}(\mathbf{x}_k^{(N)})$  ( $5N^{(n)}$ ; message conveying measurement-based information about  $\mathbf{x}_k^{(N)}$ ); 8)  $m_6^{(n)}(\mathbf{x}_k^{(N)})$  ( $6N^{(n)}$ ; *second backward filtered pdf of  $\mathbf{x}_k^{(N)}$* ),  $m_1^{(n)}(\mathbf{x}_k^{(N)})$  ( $1N^{(n)}$ ; *smoothed pdf of  $\mathbf{x}_k^{(N)}$* ). Note that the message  $m_1^{(n)}(\mathbf{x}_k^{(N)})$  computed in the last step of the  $n$ -th iteration is stored in a memory cell (identified by the label ‘D’), so that it becomes available at the beginning of the next iteration.

**Phase III** - In the third phase, the final smoothed pdf  $m_1^{(n_i)}(\mathbf{x}_k^{(N)})$  is exploited to compute: a) the final backward pdf (i.e., the output message of  $\text{BIF}_2$ )  $\tilde{m}_{\text{be}}(\mathbf{x}_k^{(N)})$ ; b) the new pseudo-measurement message  $m_2^{(n_i+1)}(\mathbf{x}_k)$ , the final backward filtered pdf  $\tilde{m}_3^{(n_i+1)}(\mathbf{x}_k)$ , the final smoothed pdf  $m_4^{(n_i+1)}(\mathbf{x}_k)$  of  $\mathbf{x}_k$  and, finally, the final backward pdf (i.e., the output message of  $\text{BIF}_1$ )  $m_{\text{be}}(\mathbf{x}_k)$ .



**Phase II** - In the  $n$ -th iteration of this phase, the eight consecutive steps listed below are carried out; for each step, all the computed messages are described.

Step 1) - In this step, the message

$$m_1^{(n-1)}(\mathbf{x}_k^{(N)}) = \sum_{j=1}^{N_p} W_{1,k,j}^{(n-1)} \delta(\mathbf{x}_k^{(N)} - \mathbf{x}_{k,j}^{(N)}), \quad (41)$$

computed in the previous iteration and conveying the smoothed pdf of  $\mathbf{x}_k^{(N)}$  generated by  $F_2$  and  $BIF_2$  (see step 8)) is processed jointly with  $\tilde{m}_{be}(\mathbf{x}_{k+1}^{(N)})$  (37) in the  $BIF_2 \rightarrow BIF_1$  block to generate the message

$$m_2^{(n)}(\mathbf{x}_k) = \mathcal{N}(\mathbf{x}_k; \eta_{2,k}^{(n)}, \mathbf{C}_{2,k}^{(n)}), \quad (42)$$

that conveys the pseudo-measurement information provided to  $BIF_1$ . The mean vector  $\eta_{2,k}^{(n)}$  and the covariance matrix  $\mathbf{C}_{2,k}^{(n)}$  are evaluated as

$$\eta_{2,k}^{(n)} = \left[ \left( \eta_{L,k}^{(n)} \right)^T, \left( \eta_{N,k}^{(n)} \right)^T \right]^T \quad (43)$$

and

$$\mathbf{C}_{2,k}^{(n)} = \begin{bmatrix} \mathbf{C}_{LL,k}^{(n)} & \mathbf{C}_{LN,k}^{(n)} \\ \left( \mathbf{C}_{LN,k}^{(n)} \right)^T & \mathbf{C}_{NN,k}^{(n)} \end{bmatrix}, \quad (44)$$

respectively, where

$$\eta_{X,k}^{(n)} \triangleq \sum_{j=1}^{N_p} W_{1,k,j}^{(n-1)} \eta_{X,k,j} \quad (45)$$

is a  $D_X$ -dimensional mean vector (with  $X = L$  and  $N$ ),

$$\mathbf{C}_{XY,k}^{(n)} \triangleq \sum_{j=1}^{N_p} W_{1,k,j}^{(n-1)} \mathbf{r}_{XY,k,j} - \eta_{X,k} \eta_{Y,k}^T \quad (46)$$

is a  $D_X \times D_Y$  covariance (or cross-covariance) matrix (with  $XY = LL, NN$  and  $LN$ ),  $\eta_{L,k,j} = \tilde{\eta}_{k,j}$ ,  $\eta_{N,k,j} = \mathbf{x}_{k,j}^{(N)}$ ,  $\mathbf{r}_{LL,k,j} \triangleq \tilde{\mathbf{C}}_{k,j} + \tilde{\eta}_{k,j}(\tilde{\eta}_{k,j})^T$ ,  $\mathbf{r}_{NN,k,j} \triangleq \mathbf{x}_{k,j}^{(N)}(\mathbf{x}_{k,j}^{(N)})^T$  and  $\mathbf{r}_{LN,k,j} \triangleq \tilde{\eta}_{k,j}(\mathbf{x}_{k,j}^{(N)})^T$ . The covariance matrix  $\tilde{\mathbf{C}}_{k,j}$  and the mean vector  $\tilde{\eta}_{k,j}$  are computed on the basis of the associated precision matrix

$$\tilde{\mathbf{W}}_{k,j} \triangleq \left( \tilde{\mathbf{C}}_{k,j} \right)^{-1} = \left( \mathbf{A}_{k,j}^{(N)} \right)^T \mathbf{W}_w^{(N)} \mathbf{A}_{k,j}^{(N)} \quad (47)$$

and of the associated transformed mean vector

$$\tilde{\mathbf{w}}_{k,j} \triangleq \tilde{\mathbf{W}}_{k,j} \tilde{\eta}_{k,j} = \left( \mathbf{A}_{k,j}^{(N)} \right)^T \mathbf{W}_w^{(N)} \mathbf{z}_{k,j}^{(L)}, \quad (48)$$

respectively; here,  $\mathbf{A}_{k,j}^{(N)} \triangleq \mathbf{A}_k^{(N)}(\mathbf{x}_{k,j}^{(N)})$ ,

$$\mathbf{z}_{k,j}^{(L)} \triangleq \mathbf{x}_{be,k+1}^{(N)} - \mathbf{f}_{k,j}^{(N)} \quad (49)$$

is an iteration-independent pseudo-measurement (see Eq. (21)) and  $\mathbf{f}_{k,j}^{(N)} \triangleq \mathbf{f}_k^{(N)}(\mathbf{x}_{k,j}^{(N)})$ . Note that, in the first iteration,

$$W_{1,k,j}^{(n-1)} = W_{1,k,j}^{(0)} = w_{fe,k,j}, \quad (50)$$

for any  $j$ , i.e.  $m_1^{(0)}(\mathbf{x}_k^{(N)}) = \tilde{m}_{fe1}(\mathbf{x}_k^{(N)})$  (see Eq. (35)) since the initial information available about the particle set are those originating from the forward pass. For this reason, the particles  $\{\mathbf{x}_{k,j}^{(N)}\}$  and their weights  $\{w_{fe,k,j}\}$  are stored in the memory cell at the beginning of the first iteration.

Step 2) - In this step, the first backward filtered pdf  $\tilde{m}_3^{(n)}(\mathbf{x}_k)$  of  $\mathbf{x}_k$  is computed as (see Fig. 4)

$$\tilde{m}_3^{(n)}(\mathbf{x}_k) = \tilde{m}_1(\mathbf{x}_k) m_2^{(n)}(\mathbf{x}_k) \quad (51)$$

$$= \mathcal{N}(\mathbf{x}_k; \eta_{3,k}^{(n)}, \mathbf{C}_{3,k}^{(n)}), \quad (52)$$

where the messages  $\tilde{m}_1(\mathbf{x}_k)$  and  $m_2^{(n)}(\mathbf{x}_k)$  are given by Eq. (38) and Eq. (42), respectively. The covariance matrix  $\mathbf{C}_{3,k}^{(n)}$  and the mean vector  $\eta_{3,k}^{(n)}$  are computed on the basis of the associated precision matrix

$$\mathbf{W}_{3,k}^{(n)} \triangleq (\mathbf{C}_{3,k}^{(n)})^{-1} = \mathbf{W}_{1,k} + \mathbf{W}_{2,k}^{(n)} \quad (53)$$

and the associated transformed mean vector

$$\mathbf{w}_{3,k}^{(n)} \triangleq \mathbf{W}_{3,k}^{(n)} \eta_{3,k}^{(n)} = \mathbf{w}_{1,k} + \mathbf{w}_{2,k}^{(n)}, \quad (54)$$

respectively; here,  $\mathbf{W}_{2,k}^{(n)} \triangleq (\mathbf{C}_{2,k}^{(n)})^{-1}$ ,  $\mathbf{w}_{2,k}^{(n)} \triangleq \mathbf{W}_{2,k}^{(n)} \eta_{2,k}^{(n)}$ , and  $\mathbf{W}_{1,k}$  and  $\mathbf{w}_{1,k}$  are given by Eqs. (39) and (40), respectively. From Eqs. (53)-(54) the expressions

$$\mathbf{C}_{3,k}^{(n)} = \mathbf{W}_k^{(n)} \mathbf{C}_{2,k}^{(n)} \quad (55)$$

and

$$\eta_{3,k}^{(n)} = \mathbf{W}_k^{(n)} [\mathbf{C}_{2,k}^{(n)} \mathbf{w}_{1,k} + \eta_{2,k}^{(n)}] \quad (56)$$

can be easily inferred; here,  $\mathbf{W}_k^{(n)} \triangleq [\mathbf{C}_{2,k}^{(n)} \mathbf{W}_{1,k} + \mathbf{I}_D]^{-1}$ .

Step 3) - In this step, the smoothed pdf  $m_4^{(n)}(\mathbf{x}_k)$  of  $\mathbf{x}_k$  is evaluated as (see Fig. 4)

$$m_4^{(n)}(\mathbf{x}_k) = \vec{m}_{\text{fe}1}(\mathbf{x}_k) \tilde{m}_3^{(n)}(\mathbf{x}_k) \quad (57)$$

$$= \mathcal{N}(\mathbf{x}_k; \eta_{4,k}^{(n)}, \mathbf{C}_{4,k}^{(n)}), \quad (58)$$

where the messages  $\vec{m}_{\text{fe}1}(\mathbf{x}_k)$  and  $\tilde{m}_3^{(n)}(\mathbf{x}_k)$  are given by Eqs. (31) and (52), respectively. The covariance matrix  $\mathbf{C}_{4,k}^{(n)}$  and the mean vector  $\eta_{4,k}^{(n)}$  are computed on the basis of the associated precision matrix

$$\mathbf{W}_{4,k}^{(n)} = \mathbf{W}_{\text{fe}1,k} + \mathbf{W}_{3,k}^{(n)} \quad (59)$$

and of the associated transformed mean vector

$$\mathbf{w}_{4,k}^{(n)} = \mathbf{w}_{\text{fe}1,k} + \mathbf{w}_{3,k}^{(n)}, \quad (60)$$

respectively. Note that Eq. (57) represents an instance of Eq. (11), since  $\vec{m}_{\text{fe}1}(\mathbf{x}_k)$  and  $\tilde{m}_3^{(n)}(\mathbf{x}_k)$  correspond to  $\vec{m}_{\text{fe}1}(\mathbf{x}_k^{(i)})$  and  $\tilde{m}_{\text{be}1}(\mathbf{x}_k^{(i)})$ , respectively ( $\mathbf{x}_k^{(i)} = \mathbf{x}_k^{(N)}$  in this case).

Step 4) - In this step, the message

$$m_1^{(n)}(\mathbf{x}_k^{(L)}) \triangleq \int m_4^{(n)}(\mathbf{x}_k) d\mathbf{x}_k^{(N)} = \mathcal{N}(\mathbf{x}_k^{(L)}; \tilde{\eta}_{1,k}^{(n)}, \tilde{\mathbf{C}}_{1,k}^{(n)}), \quad (61)$$

is computed in the BIF<sub>1</sub>→BIF<sub>2</sub> block. In practice, the mean  $\tilde{\eta}_{1,k}^{(n)}$  and the covariance matrix  $\tilde{\mathbf{C}}_{1,k}^{(n)}$  are extracted from the mean  $\eta_{4,k}^{(n)}$  and the covariance matrix  $\mathbf{C}_{4,k}^{(n)}$  of  $m_4^{(n)}(\mathbf{x}_k)$  (58), respectively, since  $\mathbf{x}_k^{(L)}$  consists of the first  $D_L$  elements of  $\mathbf{x}_k$ .

Then, the backward predicted pdf  $\tilde{m}_3^{(n)}(\mathbf{x}_k^{(N)})$  is evaluated as (see Fig. 4)

$$\begin{aligned} \tilde{m}_3^{(n)}(\mathbf{x}_k^{(N)}) &= \int \int f(\mathbf{x}_{k+1}^{(N)} | \mathbf{x}_k^{(N)}, \mathbf{x}_k^{(L)}) \tilde{m}_{\text{be}}(\mathbf{x}_{k+1}^{(N)}) \\ &\quad \cdot m_1^{(n)}(\mathbf{x}_k^{(L)}) d\mathbf{x}_k^{(N)} d\mathbf{x}_{k+1}^{(N)}. \end{aligned} \quad (62)$$



Actually, what is really needed in our computations is the value taken on by this message (and also by messages  $m_2^{(n)}(\mathbf{x}_k^{(N)})$  and  $m_5^{(n)}(\mathbf{x}_k^{(N)})$  evaluated in step 5) and in step 7), respectively) for  $\mathbf{x}_k^{(N)} = \mathbf{x}_{k,j}^{(N)}$  (see Eqs. (75), (87) and (92)); such a value, denoted  $w_{3,k,j}^{(n)}$ , is computed as

$$w_{3,k,j}^{(n)} = D_{3,k,j}^{(n)} \exp\left(-\frac{1}{2}Z_{3,k,j}^{(n)}\right), \quad (63)$$

where

$$D_{3,k,j}^{(n)} = (2\pi)^{-D_N/2} (\det(\mathbf{C}_{3,k,j}^{(N)}[n]))^{-1/2}, \quad (64)$$

$$Z_{3,k,j}^{(n)} \triangleq \left\| \mathbf{x}_{\text{be},k+1}^{(N)} - \eta_{3,k,j}^{(N)}[n] \right\|_{\mathbf{W}_{3,k,j}^{(N)}[n]}^2, \quad (65)$$

$\|\mathbf{x}\|_{\mathbf{W}}^2 \triangleq \mathbf{x}^T \mathbf{W} \mathbf{x}$  denotes the square of the norm of the vector  $\mathbf{x}$  with respect to the positive definite matrix  $\mathbf{W}$ ,

$$\eta_{3,k,j}^{(N)}[n] \triangleq \mathbf{A}_{k,j}^{(N)} \tilde{\eta}_{1,k}^{(n)} + \mathbf{f}_{k,j}^{(N)}, \quad (66)$$

$\mathbf{W}_{3,k,j}^{(N)}[n] \triangleq (\mathbf{C}_{3,k,j}^{(N)}[n])^{-1}$  and

$$\mathbf{C}_{3,k,j}^{(N)}[n] \triangleq \mathbf{A}_{k,j}^{(N)} \tilde{\mathbf{C}}_{1,k}^{(n)} \left( \mathbf{A}_{k,j}^{(N)} \right)^T + \mathbf{C}_w^{(N)}. \quad (67)$$

Step 5) - In this step, the message  $m_2^{(n)}(\mathbf{x}_k^{(N)})$ , conveying pseudo-measurement information about the nonlinear state component, is computed in the BIF<sub>1</sub>→BIF<sub>2</sub> block. The value  $w_{2,k,j}^{(n)}$  taken on by this message for  $\mathbf{x}_k^{(N)} = \mathbf{x}_{k,j}^{(N)}$  is evaluated as

$$w_{2,k,j}^{(n)} = D_{2,k,j}^{(n)} \exp\left(-\frac{1}{2}Z_{2,k,j}^{(n)}\right) \quad (68)$$

for any  $j$ ; here,

$$Z_{2,k,j}^{(n)} \triangleq \left\| \tilde{\eta}_{z,k,j}^{(n)} \right\|_{\tilde{\mathbf{W}}_{z,k,j}^{(n)}}^2 - \left\| \mathbf{f}_{k,j}^{(L)} \right\|_{\mathbf{W}_w^{(L)}}^2 - \left\| \tilde{\eta}_{2,k,j}^{(n)} \right\|_{\tilde{\mathbf{W}}_{2,k,j}^{(n)}}^2, \quad (69)$$

$\mathbf{W}_w^{(L)} \triangleq [\mathbf{C}_w^{(L)}]^{-1}$ ,  $\mathbf{f}_{k,j}^{(L)} \triangleq \mathbf{f}_k^{(L)}(\mathbf{x}_{k,j}^{(N)})$ ,  $\tilde{\mathbf{W}}_{z,k,j}^{(n)} \triangleq (\tilde{\mathbf{C}}_{z,k,j}^{(n)})^{-1}$ ,  $\tilde{\mathbf{w}}_{z,k,j}^{(n)} \triangleq \tilde{\mathbf{W}}_{z,k,j}^{(n)} \tilde{\eta}_{z,k,j}^{(n)}$ ,

$$\tilde{\eta}_{z,k,j}^{(n)} = \tilde{\eta}_{\text{be},k+1} - \mathbf{A}_{k,j}^{(L)} \tilde{\eta}_{1,k}^{(n)}, \quad (70)$$

$$\tilde{\mathbf{C}}_{z,k,j}^{(n)} = \tilde{\mathbf{C}}_{\text{be},k+1} - \mathbf{A}_{k,j}^{(L)} \tilde{\mathbf{C}}_{1,k}^{(n)} \left( \mathbf{A}_{k,j}^{(L)} \right)^T, \quad (71)$$

$$\tilde{\mathbf{W}}_{2,k,j}^{(n)} \triangleq \left( \tilde{\mathbf{C}}_{2,k,j}^{(n)} \right)^{-1} = \tilde{\mathbf{W}}_{z,k,j}^{(n)} + \mathbf{W}_w^{(L)}, \quad (72)$$

$$D_{2,k,j}^{(n)} \triangleq (2\pi)^{-D_L/2} [\det(\tilde{\mathbf{C}}_{k,j}^{(n)})]^{-1/2} \quad (73)$$

$\tilde{\mathbf{C}}_{k,j}^{(n)} \triangleq \tilde{\mathbf{C}}_{z,k,j}^{(n)} + \mathbf{C}_w^{(L)}$ ,  $\mathbf{A}_{k,j}^{(L)} \triangleq \mathbf{A}_k^{(L)}(\mathbf{x}_{k,j}^{(N)})$ ,  $\tilde{\eta}_{2,k,j}^{(n)}$  is evaluated on the basis of the associated transformed mean vector

$$\tilde{\mathbf{w}}_{2,k,j}^{(n)} \triangleq \tilde{\mathbf{W}}_{2,k,j}^{(n)} \tilde{\eta}_{2,k,j}^{(n)} = \tilde{\mathbf{w}}_{z,k,j}^{(n)} + \mathbf{W}_w^{(L)} \mathbf{f}_{k,j}^{(L)}, \quad (74)$$

and the mean  $\tilde{\eta}_{\text{be},k+1}$  and the covariance matrix  $\tilde{\mathbf{C}}_{\text{be},k+1}$  are extracted from the mean  $\eta_{\text{be},k+1}$  and the covariance matrix  $\mathbf{C}_{\text{be},k+1}$  of  $\tilde{m}_{\text{be}}(\mathbf{x}_{k+1})$  (36), since they refer to  $\mathbf{x}_k^{(L)}$  only.

Step 6) - In this step, the message  $m_4^{(n)}(\mathbf{x}_k^{(N)})$ , conveying the first backward filtered pdf of  $\mathbf{x}_k^{(N)}$ , is computed as (see Fig. 4)

$$\tilde{m}_4^{(n)}(\mathbf{x}_k^{(N)}) = \tilde{m}_3^{(n)}(\mathbf{x}_k^{(N)}) m_2^{(n)}(\mathbf{x}_k^{(N)}). \quad (75)$$

The value  $w_{4,k,j}^{(n)}$  taken on by this message for  $\mathbf{x}_k^{(N)} = \mathbf{x}_{k,j}^{(N)}$  is given by (see Eqs. (63) and (68))

$$w_{4,k,j}^{(n)} \triangleq w_{2,k,j}^{(n)} w_{3,k,j}^{(n)} \quad (76)$$

for any  $j$ .

Step 7) - In this step, the message conveying measurement-based information about  $\mathbf{x}_k^{(N)}$  is computed as (see Fig. 4)

$$m_5^{(n)}(\mathbf{x}_k^{(N)}) = \int f(\mathbf{y}_k | \mathbf{x}_k^{(N)}, \mathbf{x}_k^{(L)}) m_1^{(n)}(\mathbf{x}_k^{(L)}) d\mathbf{x}_k^{(L)} \quad (77)$$

$$= \mathcal{N}\left(\mathbf{y}_k; \bar{\eta}_{5,k}^{(n)}\left(\mathbf{x}_k^{(N)}\right), \bar{\mathbf{C}}_{5,k}^{(n)}\left(\mathbf{x}_k^{(N)}\right)\right) \quad (78)$$

where

$$\bar{\eta}_{5,k}^{(n)}\left(\mathbf{x}_k^{(N)}\right) \triangleq \mathbf{B}_k\left(\mathbf{x}_k^{(N)}\right) \tilde{\eta}_{1,k}^{(n)} + \mathbf{g}_k\left(\mathbf{x}_k^{(N)}\right) \quad (79)$$

and

$$\bar{\mathbf{C}}_{5,k}^{(n)}\left(\mathbf{x}_k^{(N)}\right) \triangleq \mathbf{B}_k\left(\mathbf{x}_k^{(N)}\right) \tilde{\mathbf{C}}_{1,k}^{(n)} \mathbf{B}_k^T\left(\mathbf{x}_k^{(N)}\right) + \mathbf{C}_e. \quad (80)$$

Consequently, the value taken on by  $m_5^{(n)}(\mathbf{x}_k^{(N)})$  for  $\mathbf{x}_k^{(N)} = \mathbf{x}_{k,j}^{(N)}$  is

$$w_{5,k,j}^{(n)} = \mathcal{N}\left(\mathbf{y}_k; \bar{\eta}_{5,k,j}^{(n)}, \bar{\mathbf{C}}_{5,k,j}^{(n)}\right) \quad (81)$$

$$= D_{5,k,j}^{(n)} \exp\left(-\frac{1}{2} Z_{5,k,j}^{(n)}\right), \quad (82)$$

where

$$\bar{\eta}_{5,k,j}^{(n)}\left(\mathbf{x}_k^{(N)}\right) \triangleq \bar{\eta}_{5,k}^{(n)}\left(\mathbf{x}_{k,j}^{(N)}\right) = \mathbf{B}_{k,j} \tilde{\eta}_{1,k}^{(n)} + \mathbf{g}_{k,j}, \quad (83)$$

$$\bar{\mathbf{C}}_{5,k,j}^{(n)} \triangleq \bar{\mathbf{C}}_{5,k}^{(n)}\left(\mathbf{x}_{k,j}^{(N)}\right) = \mathbf{B}_{k,j} \tilde{\mathbf{C}}_{1,k}^{(n)} \mathbf{B}_{k,j}^T + \mathbf{C}_e, \quad (84)$$

$$\mathbf{B}_{k,j} \triangleq \mathbf{B}_k(\mathbf{x}_{k,j}^{(N)}), \mathbf{g}_{k,j} \triangleq \mathbf{g}_k(\mathbf{x}_{k,j}^{(N)}),$$

$$D_{5,k,j}^{(n)} \triangleq (2\pi)^{-P/2} [\det(\bar{\mathbf{C}}_{5,k,j}^{(n)})]^{-1/2} \quad (85)$$

$$Z_{5,k,j}^{(n)} \triangleq \left\| \mathbf{y}_k - \bar{\eta}_{5,k,j}^{(n)} \right\|_{\bar{\mathbf{W}}_{5,k,j}^{(n)}}^2 \quad (86)$$

and  $\bar{\mathbf{W}}_{5,k,j}^{(n)} \triangleq (\bar{\mathbf{C}}_{5,k,j}^{(n)})^{-1}$ . Then, the message  $\tilde{m}_6^{(n)}(\mathbf{x}_k^{(N)})$  is evaluated as (see Fig. 4)

$$\tilde{m}_6^{(n)}\left(\mathbf{x}_k^{(N)}\right) = \tilde{m}_4^{(n)}\left(\mathbf{x}_k^{(N)}\right) m_5^{(n)}\left(\mathbf{x}_k^{(N)}\right). \quad (87)$$

Its value for  $\mathbf{x}_k^{(N)} = \mathbf{x}_{k,j}^{(N)}$  is given by (see Eqs. (63), (68) and (82))

$$w_{6,k,j}^{(n)} = w_{4,k,j}^{(n)} w_{5,k,j}^{(n)} = w_{2,k,j}^{(n)} w_{3,k,j}^{(n)} w_{5,k,j}^{(n)} \quad (88)$$

$$= D_{6,k,j}^{(n)} \exp\left(-\frac{1}{2} Z_{6,k,j}^{(n)}\right) \quad (89)$$

where

$$D_{6,k,j}^{(n)} \triangleq D_{2,k,j}^{(n)} D_{3,k,j}^{(n)} D_{5,k,j}^{(n)} \quad (90)$$

and

$$Z_{6,k,j}^{(n)} \triangleq Z_{2,k,j}^{(n)} + Z_{3,k,j}^{(n)} + Z_{5,k,j}^{(n)}. \quad (91)$$

Note that the weight  $w_{6,k,j}^{(n)}$  conveys the information provided by the backward state transition  $(w_{3,k,j}^{(n)})$ , the pseudo-measurements  $(w_{2,k,j}^{(n)})$  and the measurements  $(w_{5,k,j}^{(n)})$ .

Step 8) - In this step, the message  $m_1^{(n)}(\mathbf{x}_k^{(N)})$ , conveying the smoothed pdf of  $\mathbf{x}_k^{(N)}$  evaluated in the  $n$ -th iteration, is computed as (see Fig. 4)

$$m_1^{(n)}(\mathbf{x}_k^{(N)}) = \vec{m}_{\text{fp}}(\mathbf{x}_k^{(N)}) \tilde{m}_6^{(n)}(\mathbf{x}_k^{(N)}); \quad (92)$$

this formula represents an instance of Eq. (10), since  $\vec{m}_{\text{fp1}}(\mathbf{x}_k^{(N)})$  and  $\tilde{m}_6^{(n)}(\mathbf{x}_k^{(N)})$  correspond to  $\vec{m}_{\text{fp}}(\mathbf{x}_k^{(i)})$  and  $\tilde{m}_{\text{be2}}(\mathbf{x}_k^{(i)})$ , respectively ( $\mathbf{x}_k^{(i)} = \mathbf{x}_k^{(N)}$  in this case). The  $j$ -th component of  $m_1^{(n)}(\mathbf{x}_k^{(N)})$  is evaluated as (see Eqs. (34) and (88))

$$m_{1,j}^{(n)}(\mathbf{x}_k^{(N)}) = \vec{m}_{\text{fp},j}(\mathbf{x}_k^{(N)}) w_{6,k,j}^{(n)} \quad (93)$$

$$= w_{1,k,j}^{(n)} \delta(\mathbf{x}_k^{(N)} - \mathbf{x}_{k,j}^{(N)}), \quad (94)$$

where

$$w_{1,k,j}^{(n)} \triangleq w_p w_{6,k,j}^{(n)}. \quad (95)$$

Then, the weights  $\{w_{1,k,j}^{(n)}\}$  are normalized; the  $j$ -th normalised weight is computed as

$$W_{1,k,j}^{(n)} \triangleq C_k^{(n)} w_{1,k,j}^{(n)}, \quad (96)$$

with  $j = 1, 2, \dots, N_p$ , where  $C_k^{(n)} \triangleq 1 / \sum_{j=0}^{N_p-1} w_{1,k,j}^{(n)}$ . Moreover, the weights  $\{W_{1,k,j}^{(n)}\}$  are stored for the next iteration. This concludes the  $n$ -th iteration. Then, the index  $n$  is increased by one, and a new iteration is started by going back to step 1) if  $n < n_i + 1$ ; otherwise (i.e., if  $n = n_i + 1$ ), we proceed with the next phase.

**Phase III** - In this phase,  $\tilde{m}_{\text{be}}(\mathbf{x}_k^{(N)})$  (i.e., the BIF<sub>2</sub> output message) is computed first; then, steps 1) and 2) of phase II are accomplished in order to compute all the statistical information required for the evaluation of the backward estimate  $\tilde{m}_{\text{be}}(\mathbf{x}_k)$  (i.e., the BIF<sub>1</sub> output message). More specifically, we first sample the set  $S_k$  once on the basis of the particle weights  $\{W_{1,k,j}^{(n_i)}\}$  computed in the last iteration; if the  $j_k$ -th particle (i.e.,  $\mathbf{x}_{k,j_k}^{(N)}$ ) is selected, we set

$$\mathbf{x}_{\text{be},k}^{(N)} = \mathbf{x}_{k,j_k}^{(N)}, \quad (97)$$

so that the message (see Eq. (37))

$$\tilde{m}_{\text{be}}(\mathbf{x}_k^{(N)}) \triangleq \delta(\mathbf{x}_k^{(N)} - \mathbf{x}_{\text{be},k}^{(N)}), \quad (98)$$

can be made available at the output of BIF<sub>2</sub>. On the other hand, the evaluation of the message  $\tilde{m}_{\text{be}}(\mathbf{x}_k)$  is accomplished as follows. The messages  $m_2^{(n_i+1)}(\mathbf{x}_k)$  and  $\tilde{m}_3^{(n_i+1)}(\mathbf{x}_k)$  are computed first (see Eqs. (42)-(49) and Eqs. (51)-(54), respectively). Then, the message  $\tilde{m}_{\text{be}}(\mathbf{x}_k)$  is computed as (see Fig. 4)

$$\tilde{m}_{\text{be}}(\mathbf{x}_k) = \tilde{m}_{\text{be2}}(\mathbf{x}_k) = \tilde{m}_{\text{be1}}^{(n_i+1)}(\mathbf{x}_k) m_{\text{ms}}(\mathbf{x}_k) \quad (99)$$

$$= \mathcal{N}(\mathbf{x}_k; \eta_{\text{be2},k}, \mathbf{C}_{\text{be2},k}), \quad (100)$$

where

$$m_{\text{ms}}(\mathbf{x}_k) = \mathcal{N}(\mathbf{x}_k; \eta_{\text{ms},k}, \mathbf{C}_{\text{ms},k}) \quad (101)$$

is the message conveying measurement information.

Moreover, the covariance matrices  $\mathbf{C}_{\text{ms},k}$  and  $\mathbf{C}_{\text{be2},k}$ , and the mean vectors  $\eta_{\text{ms},k}$  and  $\eta_{\text{be2},k}$  are evaluated on the basis of the associated precision matrices

$$\mathbf{W}_{\text{ms},k} \triangleq (\mathbf{C}_{\text{ms},k})^{-1} = \mathbf{H}_k \mathbf{W}_e \mathbf{H}_k^T, \quad (102)$$

$$\mathbf{W}_{\text{be}2,k} \triangleq (\mathbf{C}_{\text{be}2,k})^{-1} = \mathbf{W}_{\text{ms},k} + \mathbf{W}_{\text{be}1,k}^{(n_i+1)}, \quad (103)$$

and of the transformed mean vectors

$$\mathbf{w}_{\text{ms},k} \triangleq \mathbf{W}_{\text{ms},k} \eta_{\text{ms},k} = \mathbf{H}_k \mathbf{W}_e (\mathbf{y}_k - \mathbf{v}_k), \quad (104)$$

$$\mathbf{w}_{\text{be}2,k} \triangleq \mathbf{W}_{\text{be}2,k} \eta_{\text{be}2,k} = \mathbf{w}_{\text{ms},k} + \mathbf{w}_{\text{be}1,k}^{(n_i+1)}, \quad (105)$$

respectively. The  $k$ -th recursion is now over.

In the DBSA, the *first recursion* of the backward pass (corresponding to  $k = T-1$ ) requires the knowledge of the input messages  $\tilde{m}_{\text{be}}(\mathbf{x}_T)$  and  $\tilde{m}_{\text{be}}(\mathbf{x}_T^{(N)})$ . Similarly as any BIF algorithm, the evaluation of these messages in DBIF is based on the statistical information generated in the last recursion of the forward pass. In particular, the above mentioned messages are still expressed by Eqs. (36) and (37) (with  $k = T-1$  in both formulas), respectively. However, the vector  $\mathbf{x}_{\text{be},T}^{(N)}$  is generated by sampling the particle set  $S_T$  on the basis of the forward weights  $\{w_{\text{fe},T,j}\}$ , since backward predictions are unavailable at the final instant  $k = T$ . Therefore, if the  $j_T$ -th particle of  $S_T$  is selected, we set

$$\mathbf{x}_{\text{be},T}^{(N)} = \mathbf{x}_{\text{fe},T,j_T}^{(N)} \quad (106)$$

in the message  $\tilde{m}_{\text{be}}(\mathbf{x}_T^{(N)})$  entering the BIF<sub>2</sub> in the first recursion (see Eq. (37)). As far as BIF<sub>1</sub> is concerned, following [19], we choose

$$\mathbf{W}_{\text{be},T} = \mathbf{W}_{\text{fe}1,T} \quad (107)$$

and

$$\mathbf{w}_{\text{be},T} = \mathbf{w}_{\text{fe}1,T} \quad (108)$$

for the message  $\tilde{m}_{\text{be}}(\mathbf{x}_T)$ .

The DBSA is summarized in Algorithm 1. It generates all the statistical information required to solve problems **P.1** and **P.2**. Let us now discuss how this can be done in detail. As far as problem **P.1** is concerned, it is useful to point out that the DBSA produces a trajectory  $\{\mathbf{x}_{\text{be},k}^{(N)}, k = 1, 2, \dots, T\}$  for the *nonlinear* component (see Eq. (97)). Another trajectory, representing the time evolution of the *linear* state component only and denoted  $\{\mathbf{x}_{\text{be},k}^{(L)}, k = 1, 2, \dots, T\}$ , can be evaluated by sampling the message  $m_1^{(n_i)}(\mathbf{x}_k^{(L)})$  (see Eq. (61)) or by simply setting  $\mathbf{x}_{\text{be},k}^{(L)} = \tilde{\eta}_{1,k}^{(n_i)}$  (this task can be accomplished in phase III, after sampling the particle set  $S_k$ ; see also the task g- in phase III of Algorithm 1).

Since the DBSA solves problem **P.1**, it also solves problem **P.2**; in fact, once it has been run, an approximation of the marginal smoothed pdf at any instant can be simply obtained by marginalization. Unluckily, the last result is achieved at the price of a significant computational cost, since  $M$  backward passes are required. However, if we are interested in solving problem **P.2** only, a simpler particle smoother can be developed following the approach illustrated in ref. [19], so that a single backward pass has to be run. In this pass, the evaluation of the message  $\tilde{m}_{\text{be}}(\mathbf{x}_k^{(N)})$  (i.e., of the particle  $\mathbf{x}_{\text{be},k}^{(N)}$ ) involves the whole particle set  $S_k$  and their weights  $\{W_{1,k,j}^{(n_i)}\}$  (see Eq. (96)) evaluated in the last phase of the  $(T-k)$ -th recursion. More specifically, a new smoother is obtained by employing a different method for evaluating  $\mathbf{x}_{\text{be},k}^{(N)}$  (see phase III-BIF<sub>2</sub>); it consists in computing the smoothed estimate

$$\mathbf{x}_{\text{sm},k}^{(N)} = \sum_{j=1}^{N_p} W_{1,k,j}^{(n_i)} \mathbf{x}_{k,j}^{(N)} \quad (109)$$

of  $\mathbf{x}_k^{(N)}$  and, then, setting

$$\mathbf{x}_{\text{be},k}^{(N)} = \mathbf{x}_{\text{sm},k}^{(N)} \quad (110)$$

The resulting smoother is called *simplified* DBSA (SDBSA) in the following.

The computational complexity of the DBSA and the SDBSA can be reduced by reusing the forward weights  $\{w_{\text{fe},k,j}\}$  in all the iterations of phase II, so that step 7) can be skipped; this means that, for any  $n$ ,

---

**Algorithm 1:** Double Bayesian Smoothing

---

- 1 **Forward filtering:** For  $k = 1$  to  $T$ : Run the DBF, and store  $\mathbf{W}_{\text{fe}1,k}$  (32),  $\mathbf{w}_{\text{fe}1,k}$  (33),  $S_k = \{\mathbf{x}_{k,j}^{(N)}\}$  and  $\{w_{\text{fe},k,j}\}_{j=1}^{N_p}$ .
  - 2 **Initialisation of backward filtering:** compute  $\mathbf{x}_{\text{be},T}^{(N)}$  (106),  $\mathbf{W}_{\text{be},T}$  (107) and  $\mathbf{w}_{\text{be},T}$  (108); then, compute  $\mathbf{C}_{\text{be},T} = (\mathbf{W}_{\text{be},T})^{-1}$ ,  $\eta_{\text{be},T} = \mathbf{C}_{\text{be},T} \mathbf{w}_{\text{be},T}$ .
  - 3 **Backward filtering and smoothing:**  
 for  $k = T - 1$  to 1 do  
   **a- Phase I:**  
   - *Backward prediction in BIF<sub>1</sub>:* compute  $\mathbf{W}_{1,k}$  (39) and  $\mathbf{w}_{1,k}$  (40).  
   - *Computation of iteration-independent information required in task b:* For  $j = 1$  to  $N_p$ : compute  $\mathbf{z}_{k,j}^{(L)}$  (49),  $\tilde{\mathbf{W}}_{k,j}$  (47),  $\tilde{\mathbf{w}}_{k,j}$  (48),  $\tilde{\mathbf{C}}_{k,j} = (\tilde{\mathbf{W}}_{k,j})^{-1}$  and  $\tilde{\eta}_{k,j} = \tilde{\mathbf{C}}_{k,j} \tilde{\mathbf{w}}_{k,j}$ .  
   - *Initialisation of particle weights:* Set  $W_{1,k,j}^{(0)} = w_{\text{fe},k,j}$ .  
   **Phase II:**  
   for  $n = 1$  to  $n_i$  do  
     **b-** Compute  $\eta_{2,k}^{(n)}$  (43) and  $\mathbf{C}_{2,k}^{(n)}$  (44).  
     **c-** Compute  $\mathbf{C}_{3,k}^{(n)}$  (55),  $\eta_{3,k}^{(n)}$  (56),  $\mathbf{W}_{3,k}^{(n)} = (\mathbf{C}_{3,k}^{(n)})^{-1}$ ,  $\mathbf{w}_{3,k}^{(n)} = \mathbf{W}_{3,k}^{(n)} \eta_{3,k}^{(n)}$ ,  $\mathbf{W}_{4,k}^{(n)}$  (59),  $\mathbf{w}_{4,k}^{(n)}$  (60),  $\mathbf{C}_{4,k}^{(n)} = (\mathbf{W}_{4,k}^{(n)})^{-1}$  and  $\eta_{4,k}^{(n)} = \mathbf{C}_{4,k}^{(n)} \mathbf{w}_{4,k}^{(n)}$ . Then, extract  $\tilde{\eta}_{1,k}^{(n)}$  ( $\tilde{\mathbf{C}}_{1,k}^{(n)}$ ) from  $\eta_{4,k}^{(n)}$  ( $\mathbf{C}_{4,k}^{(n)}$ ).  
     **d-** For  $j = 1$  to  $N_p$ : compute  $\eta_{3,k,j}^{(N)}[n]$  (66) and  $\mathbf{C}_{3,k,j}^{(N)}[n]$  (67). Then, compute  $D_{3,k,j}^{(n)}$  (64) and  $Z_{3,k,j}^{(n)}$  (65).  
     **e-** For  $j = 1$  to  $N_p$ : compute  $\tilde{\eta}_{z,k,j}^{(n)}$  (70),  $\tilde{\mathbf{C}}_{z,k,j}^{(n)}$  (71),  $\tilde{\mathbf{W}}_{z,k,j}^{(n)} = (\tilde{\mathbf{C}}_{z,k,j}^{(n)})^{-1}$ ,  $\tilde{\mathbf{w}}_{z,k,j}^{(n)} = \tilde{\mathbf{W}}_{z,k,j}^{(n)} \tilde{\eta}_{z,k,j}^{(n)}$ ,  $\tilde{\mathbf{W}}_{2,k,j}^{(n)}$  (72),  $\tilde{\mathbf{w}}_{2,k,j}^{(n)}$  (74). Then, compute  $D_{2,k,j}^{(n)}$  (73) and  $Z_{2,k,j}^{(n)}$  (69).  
     **f-** For  $j = 1$  to  $N_p$ : Compute  $\tilde{\eta}_{5,k,j}^{(n)}$  (83),  $\tilde{\mathbf{C}}_{5,k,j}^{(n)}$  (84),  $\tilde{\mathbf{W}}_{5,k,j}^{(n)} = (\tilde{\mathbf{C}}_{5,k,j}^{(n)})^{-1}$ ,  $D_{5,k,j}^{(n)}$  (85) and  $Z_{5,k,j}^{(n)}$  (86). Then, compute  $D_{6,k,j}^{(n)}$  (90),  $Z_{6,k,j}^{(n)}$  (91),  $w_{6,k,j}^{(n)}$  (89),  $w_{1,k,j}^{(n)}$  (95) and  $W_{1,k,j}^{(n)}$  (96).  
     Store the weights  $\{W_{1,k,j}^{(n)}\}$  for the next iteration.  
   end  
   **g- Phase III - BIF<sub>2</sub>:** Select the  $j_k$ -th particle  $\mathbf{x}_{k,j_k}^{(N)}$  by sampling the set  $S_k$  on the basis of the weights  $\{W_{1,k,j}^{(n_i)}\}$ , set  $\mathbf{x}_{\text{be},k}^{(N)} = \mathbf{x}_{k,j_k}^{(N)}$  and store  $\mathbf{x}_{\text{be},k}^{(N)}$  for the next recursion.  
   **h- Phase III - BIF<sub>1</sub>:** Compute  $\eta_{2,k}^{(n_i+1)}$ ,  $\mathbf{C}_{2,k}^{(n_i+1)}$ ,  $\mathbf{W}_{3,k}^{(n_i+1)}$  and  $\mathbf{w}_{3,k}^{(n_i+1)}$  (see steps 1) and 2)).  
   Then, compute  $\mathbf{W}_{\text{ms},k}$  (102),  $\mathbf{w}_{\text{ms},k}$  (104),  $\mathbf{W}_{\text{be}2,k}$  (103),  $\mathbf{w}_{\text{be}2,k}$  (105),  $\mathbf{C}_{\text{be},k} = (\mathbf{W}_{\text{be}2,k})^{-1}$  and  $\eta_{\text{be},k} = \mathbf{C}_{\text{be},k} \mathbf{w}_{\text{be}2,k}$ , and store  $\mathbf{C}_{\text{be},k}$  and  $\eta_{\text{be},k}$  for the next recursion.  
 end
-

we set  $w_{5,k,j}^{(n)} = w_{fe,k,j}$  in the evaluation of the  $j$ -th particle weight  $w_{6,k,j}^{(n)}$  according to Eq. (88) in step 8) of phase II. Our simulation results have evidenced that, at least for the SSMs considered in Section V., this modification does not affect the estimation accuracy of the derived algorithms; for this reason, it is always employed in our simulations.

The DBSA and the SDBSA refer to case **C.1**, i.e. to the case in which the substates estimated by the interconnected forward/backward filters share the substate  $\mathbf{x}_k^{(N)}$ . Let us focus now on case **C.2**, i.e. on the case on which the filters are run on disjoint substates. A filtering technique, called *simplified* DBF (SDBF), and based on the interconnection of a particle filter ( $F_2$ ) with a single Kalman filter ( $F_1$ ), is developed for this case in ref. [20, Par. III-B]. The BIF algorithm paired with it can be easily derived following the approach illustrated above for the DBSA; the resulting smoothing algorithm is dubbed *disjoint* DBSA (DDBSA) in the following. It is important to mention that, in deriving the DBSA, the following relevant changes are made with respect to the DBSA (see Fig. 2):

1) The iterative procedure embedded in the  $(T - k)$ -th recursion of the backward pass involves both the computation of the backward predicted pdf ( $BP1$ ) and of the message  $MS1$  in BIF<sub>1</sub>; for this reason, it requires marginalizing the pdfs  $f(\mathbf{x}_{k+1}^{(N)} | \mathbf{x}_k^{(N)}, \mathbf{x}_k^{(L)})$  and  $f(\mathbf{y}_k | \mathbf{x}_k^{(N)}, \mathbf{x}_k^{(L)})$ , respectively, with respect to  $\mathbf{x}_k^{(L)}$ . This result is achieved in the first iteration by setting  $\mathbf{x}_k^{(N)} = \mathbf{x}_{fe,k}^{(N)}$  in both these pdfs, where  $\mathbf{x}_{fe,k}^{(N)}$  denotes the estimate of  $\mathbf{x}_k^{(N)}$  computed by  $F_2$  in the forward pass. In the following iterations, we set  $\mathbf{x}_k^{(N)} = \mathbf{x}_{sm,k}^{(N)}$ , where  $\mathbf{x}_{sm,k}^{(N)}$  represents the estimate of  $\mathbf{x}_k^{(N)}$  evaluated on the basis of the statistical information provided by BIF<sub>2</sub> (through the message  $SM2$ ).

2) The pseudo-measurement message  $PM1$  (corresponding to  $m_2^{(n)}(\mathbf{x}_k)$  (42) in the DBSA) conveys information about  $\mathbf{x}_k^{(L)}$  only. Moreover, it is a Gaussian message, and its mean and covariance matrix are given by  $\eta_{L,k}^{(n)}$  and  $\mathbf{C}_{LL,k}^{(n)}$  (see Eqs. (45) and (46), respectively).

Finally, it is worth mentioning that a *simplified* version of the DDBSA (called SDDBSA) can be easily developed by making the same modifications as those adopted in deriving the SDBSA from the DBSA.

## IV. Comparison of the Developed Double Smoothing Algorithms with Related Techniques

The DBSA and the DDBSA developed in the previous Section are conceptually related to the Rao-Blackwellised particle smoothers proposed by Fong *et al.* [17] and by Lindsten *et al.* [18] (these algorithms are denoted Alg-F and Alg-L respectively, in the following) and to the RBSS algorithm devised by Vitetta *et al.* in ref. [19]. In fact, all these techniques share with the DBSA and the DDBSA the following important features: 1) all of them estimate the *joint* smoothing density over the whole observation interval by generating multiple *realizations* from it; 2) they accomplish a single forward pass and as many backward passes as the overall number of realizations; 3) they combine Kalman filtering with particle filtering. However, Alg-F, Alg-L and the RBSS algorithm employ, in both their forward and backward passes, as many Kalman filters as the number of particles ( $N_p$ ) to generate a particle-dependent estimate of the linear state component only. On the contrary, the DBSA (DDBSA) employs a *single* extended Kalman filter (a *single* Kalman filter), that estimates the whole system state (the linear state component only); this substantially reduces the memory requirements of particle smoothing and, consequently, the overall number of memory accesses accomplished on the hardware platform on which smoothing is run. As far as the last point is concerned, the memory requirements of a smoothing algorithm can be roughly assessed by estimating the overall number of real quantities that need to be stored in both its forward pass and its backward pass. It can be shown that overall number of real quantities to be stored by MPF, DBF and SDBF in the forward pass of the considered smoothing algorithms is of order  $\mathcal{O}(M_{MPF})$ ,  $\mathcal{O}(M_{DBF})$ , and  $\mathcal{O}(M_{SDBF})$ , respectively, with<sup>4</sup>

$$M_{MPF} = N_p T (2D_L^2 + 2D_L + D_N + 1), \quad (111)$$

$$M_{DBF} = T (2D^2 + 2D + N_p D_N + N_p) \quad (112)$$

<sup>4</sup>Note that the expressions (111)-(113) also account for the contributions due to measurement-based information (see Eqs. (102) and (104)).

and

$$M_{SDBF} = T(2D_L^2 + 2D_L + N_p D_N + N_p). \quad (113)$$

Moreover, the overall number of real quantities to be stored by Alg-L, RBSS, the DBSA and the DDBSA is approximately of order  $\mathcal{O}(M_{Alg-L})$ ,  $\mathcal{O}(M_{RBSS})$ ,  $\mathcal{O}(M_{DBSA})$  and  $\mathcal{O}(M_{DDBSA})$ , respectively, with

$$M_{Alg-L} = M_{MPF} + D_L^2 + D, \quad (114)$$

$$M_{RBSS} = M_{MPF} + D_L^2 + D, \quad (115)$$

$$M_{DBSA} = M_{DBF} + N_p + D^2 + D + D_N \quad (116)$$

and

$$M_{DDBSA} = M_{SDBF} + N_p + D_L^2 + D. \quad (117)$$

The memory requirements of the SDBSA and the SDDBSA (the SPS algorithm) are the same as those of the DBSA and the DDBSA (the RBSS algorithm), respectively. Note also that the quantities  $M_{DBSA}$  (116) and  $M_{DDBSA}$  (117) are smaller than  $M_{Alg-L}$  (114) and  $M_{RBSS}$  (115), since  $M_{MPF}$  is larger than  $M_{DBF}$  and  $M_{SDBF}$  because of its dependence on  $N_p$ .

The differences in the overall execution time measured for the simulated smoothing algorithms are related not only to their requirements in terms of memory resources, but also to their computational complexity. In our work, the computational cost of the smoothing algorithms derived in the previous section has been carefully assessed in terms of number of *floating point operations* (flops) to be executed over the whole observation interval. The general criteria adopted in estimating the computational cost of an algorithm are the same as those illustrated in [26, App. A, p. 5420] and are not repeated here for space limitations. A detailed analysis of the cost required by each of the tasks accomplished by our smoothing algorithms is provided in Appendix B. Our analysis leads to the conclusion that the overall computational cost of the DBSA and of the DDBSA is approximately of order  $\mathcal{O}(N_{DBSA})$  and  $\mathcal{O}(N_{DDBSA})$ , respectively, with

$$N_{DBSA} = T \{ N_{DBF} + M [38D^3/3 + 20D_N^3/3 + n_i N_p (2D_L^2 D_N + 2D_L D_N^2 + D_N^3/3 + 5D_L^3) + 6n_i D^3] \}, \quad (118)$$

and

$$N_{DDBSA} = T \{ N_{SDBF} + M [38D_L^3/3 + 20D_N^3/3 + n_i N_p (2D_L^2 D_N + 2D_L D_N^2 + D_N^3/3 + 5D_L^3) + 6n_i D_L^3] \}; \quad (119)$$

here,  $N_{DBF}$  and  $N_{SDBF}$  represent the computational complexity of a single recursion of the DBF and SDBF, respectively (see [20, Eqs. (97) and (98)]). Each of the expressions (118)-(119) has been derived as follows. First, the costs of all the tasks identified in Appendix B have been summed; then, the resulting expression has been simplified, keeping only the dominant contributions due to matrix inversions, matrix products and Cholesky decompositions, and discarding all the contributions that originate from the evaluation of the matrices  $\mathbf{A}_k^{(Z)}(\mathbf{x}_k^{(N)})$  (with  $Z = L$  and  $N$ ),  $\mathbf{F}_k$ ,  $\mathbf{H}_k$  and  $\mathbf{B}_k$  and the functions  $\mathbf{f}_k^{(Z)}(\mathbf{x}_k^{(N)})$  (with  $Z = L$  and  $N$ ),  $\mathbf{f}_k(\mathbf{x}_k)$  and  $\mathbf{g}_k(\mathbf{x}_k^{(N)})$ . Moreover, the sampling of the particle set in each recursion of the backward pass has been ignored.

From Eqs. (118)-(119) it is easily inferred that the computational complexities of the DBSA and the DDBSA are approximately of order  $\mathcal{O}(n_i M N_p D_L^3 T)$ . A similar approach can be followed for Alg-L and the RBSS algorithm; this leads to the conclusion that their complexities are approximately of order  $\mathcal{O}(M N_p D_L^3 T)$ , i.e. of the same order of the complexities of the DBSA and of the DDBSA if  $n_i = 1$  is assumed.

On the other hand, the SDBSA and the SDDBSA are conceptually related to the SPS algorithm devised by Vitetta *et al.* in ref. [19]. In fact, all these algorithms aim at solving problem **P.2** only (consequently, they are unable to generate the joint smoothed pdf  $f(\mathbf{x}_{1:T}|\mathbf{y}_{1:T})$ ) and carry out a *single backward pass*. This property makes them much faster than Alg-L, the RBSS algorithm, the DBSA and the DDBSA in the

computation of marginal smoothed densities. Finally, note that, similarly as the DBSA and the DDBSA techniques, the use of the SDBSA and the SDDBSA requires a substantially smaller number of memory accesses than the SPS algorithm, since the last algorithm employs MPF in its forward pass. Moreover, the computational cost of the SDBSA and the SDDBSA is approximately of order  $\mathcal{O}(n_i N_p D_L^3 T)$ , whereas that of the SPS algorithm is approximately of order  $\mathcal{O}(N_p D_L^3 T)$ ; consequently, they are all of the same order if  $n_i = 1$  is assumed.

## V. Numerical Results

In this section we first compare, in terms of accuracy and execution time, the DBSA, the SDBSA, the DDBSA and the SDDBSA with Alg-L, the RBSS algorithm, and the SPS algorithm for a specific conditionally linear Gaussian SSM. The considered SSM is the same as the SSM#2 defined in [19] and describes the bidimensional motion of an agent. Its state vector in the  $k$ -th observation interval is defined as  $\mathbf{x}_k \triangleq [\mathbf{v}_k^T, \mathbf{p}_k^T]^T$ , where  $\mathbf{v}_k \triangleq [v_{x,k}, v_{y,k}]^T$  and  $\mathbf{p}_k \triangleq [p_{x,k}, p_{y,k}]^T$  (corresponding to  $\mathbf{x}_k^{(L)}$  and  $\mathbf{x}_k^{(N)}$ , respectively) represent the agent velocity and position, respectively (their components are expressed in m/s and in m, respectively). The state update equations are

$$\mathbf{v}_{k+1} = \rho \mathbf{v}_k + T_s \mathbf{a}_k(\mathbf{p}_k) + (1 - \rho) \mathbf{n}_{v,k} \quad (120)$$

and

$$\mathbf{p}_{k+1} = \mathbf{p}_k + T_s \mathbf{v}_k + (T_s^2/2) \mathbf{a}_k(\mathbf{p}_k) + \mathbf{n}_{p,k}, \quad (121)$$

where  $\rho$  is a forgetting factor (with  $0 < \rho < 1$ ),  $T_s$  is the sampling interval,  $\mathbf{n}_{v,k}$  is an *additive Gaussian noise* (AGN) vector characterized by the covariance matrix  $\mathbf{I}_2$ ,

$$\mathbf{a}_k(\mathbf{p}_k) = -a_0 \frac{\mathbf{p}_k}{\|\mathbf{p}_k\|} \frac{1}{1 + (\|\mathbf{p}_k\|/d_0)^2} \quad (122)$$

is the acceleration due to a force applied to the agent (and pointing towards the origin of our reference system),  $a_0$  is a scale factor (expressed in  $\text{m/s}^2$ ),  $d_0$  is a *reference distance* (expressed in m), and  $\mathbf{n}_{p,k}$  is an AGN vector characterized by the covariance matrix  $\sigma_p^2 \mathbf{I}_2$  and accounting for model inaccuracy. The measurement vector available in the  $k$ -th interval for state estimation is

$$\mathbf{y}_k = \mathbf{x}_k + \mathbf{e}_k, \quad (123)$$

where  $\mathbf{e}_k \triangleq [\mathbf{e}_{v,k}^T, \mathbf{e}_{p,k}^T]^T$  and  $\mathbf{e}_{v,k}$  ( $\mathbf{e}_{p,k}$ ) is an AGN vector characterized by the covariance matrix  $\sigma_{ev}^2 \mathbf{I}_2$  ( $\sigma_{ep}^2 \mathbf{I}_2$ ).

In our computer simulations, following [19] and [23], the estimation accuracy of the considered smoothing techniques has been assessed by evaluating two *root mean square errors* (RMSEs), one for the linear state component, the other for the nonlinear one, over an observation interval lasting  $T = 200 T_s$ ; these are denoted  $RMSE_L(\text{alg})$  and  $RMSE_N(\text{alg})$ , respectively, where ‘alg’ is the acronym of the algorithm these parameters refer to. Our assessment of the computational requirements is based, instead, on evaluating the average *computation time* required for processing a single *block* of measurements (this quantity is denoted CTB(alg) in the following). Moreover, the following values have been selected for the parameters of the considered SSM:  $\rho = 0.995$ ,  $T_s = 0.01$  s,  $\sigma_p = 5 \cdot 10^{-3}$  m,  $\sigma_{e,p} = 2 \cdot 10^{-2}$  m,  $\sigma_{e,v} = 2 \cdot 10^{-2}$  m/s,  $a_0 = 0.5$   $\text{m/s}^2$ ,  $d_0 = 5 \cdot 10^{-3}$  m and  $v_0 = 1$  m/s (the initial position  $\mathbf{p}_0 \triangleq [p_{x,0}, p_{y,0}]^T$  and the initial velocity  $\mathbf{v}_0 \triangleq [v_{x,0}, v_{y,0}]^T$  have been set to  $[0.01 \text{ m}, 0.01 \text{ m}]^T$  and  $[0.01 \text{ m/s}, 0.01 \text{ m/s}]^T$ , respectively).

Some numerical results showing the dependence of  $RMSE_L$  and  $RMSE_N$  on the number of particles ( $N_p$ ) for some of the considered smoothing algorithms are illustrated in Figs. 5 and 6, respectively (simulation results are indicated by markers, whereas continuous lines are drawn to fit them, so facilitating the interpretation of the available data). In this case,  $n_i = 1$  has been selected for all the derived particle smoothers,  $M = N_p$  has been chosen for all the smoothing algorithms generating multiple trajectories and the range  $[10, 150]$  has been considered for  $N_p$  (since no real improvement is found for  $N_p \gtrsim 150$ ). Moreover,  $RMSE_L$  and  $RMSE_N$  results are also provided for MPF and DBF, since these filtering techniques are employed in the forward pass of Alg-L, the RBSS algorithm and the SPS algorithm, and the DBSA and



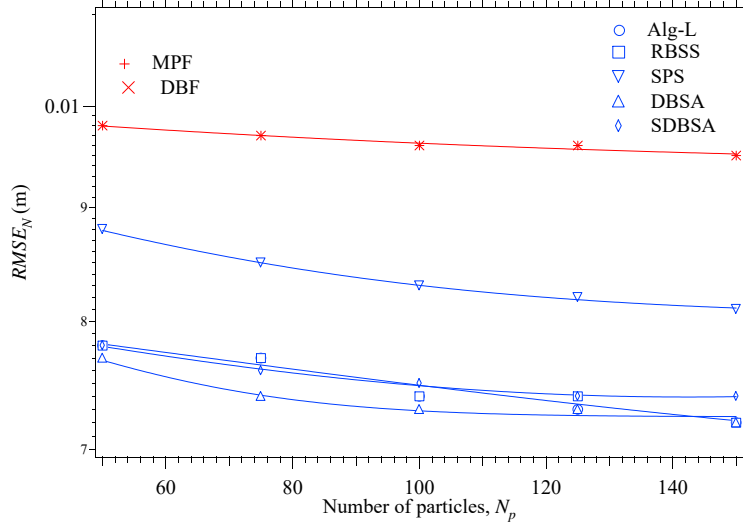


Figure 5: RMSE performance versus  $N_p$  for the nonlinear component ( $RMSE_N$ ) of the state of SSM #1; five smoothing algorithms (Alg-L, the DBSA, the SDBSA, the RBSS algorithm and the SPS algorithm) and two filtering techniques (MPF and DBF) are considered.

the SDBSA, respectively; this allows us to assess the improvement in estimation accuracy provided by the backward pass with respect to the forward pass for each smoothing algorithm. These results show that:

- 1) The DBSA, the SDBSA, Alg-L and the RBSS algorithm achieve similar accuracies in the estimation of both the linear and nonlinear state components.
- 2) The SPS algorithm is slightly outperformed by the other smoothing algorithms in terms of  $RMSE_N$  only; for instance,  $RMSE_N(\text{SPS})$  is about 1.11 times larger than  $RMSE_N(\text{SDBSA})$  for  $N_p = 100$ .
- 3) Even if the RBSS algorithm and the DBSA provide by far richer statistical information than their simplified counterparts (i.e., than the SPS algorithm and the SDBSA, respectively), they do not provide a significant improvement in the accuracy of state estimation; for instance,  $RMSE_N(\text{SPS})$  ( $RMSE_N(\text{SDBSA})$ ) is about 1.12 (1.03) time larger than  $RMSE_N(\text{RBSS})$  ( $RMSE_N(\text{DBSA})$ ) for  $N_p = 100$ .
- 4) The accuracy improvement in terms of  $RMSE_L$  ( $RMSE_N$ ) provided by all the smoothing algorithms except the SPS (by Alg-L, the RBSS algorithm, the DBSA and the SDBSA) is about 24% (about 23%) with respect to MPF and DBF, for  $N_p = 100$ . Moreover, the accuracy improvement in terms of  $RMSE_L$  ( $RMSE_N$ ) achieved by the SPS algorithm is about 24% (about 14%) with respect to the MPF for  $N_p = 100$ .
- 5) In the considered scenario, DBF is slightly outperformed by (perform similarly as) MPF in the estimation of the linear (nonlinear) state component; a similar result is reported in [20] for a different SSM.

Our simulations have also evidenced that the DBSA and the SDBSA perform similarly as the DDBSA and the SDDBSA; for this reason, RSME results referring to the last two algorithms are not shown in Figs. 5 and 6. This leads to the conclusion that, in the considered scenario, the presence of redundancy in double Bayesian smoothing does not provide any improvement with respect to the case in which the two interconnected filters operate on disjoint substates in the forward and in the backward passes. Note that the same conclusion had been reached in ref. [20, Sec. IV] for DBF only.

Despite their similar accuracies, the considered smoothing algorithms require different computational efforts; this is easily inferred from the numerical results appearing in Fig. 7 and illustrating the dependence of the CTB on  $N_p$  for all the above mentioned filtering and smoothing algorithms. In fact, these results show that  $\text{CTB}(\text{DBSA})$  is approximately 0.85 (0.48) times smaller than  $\text{CTB}(\text{Alg-L})$  ( $\text{CTB}(\text{RBSS})$ ); this is in agreement with the mathematical results illustrated in Section IV. about the complexity of Alg-L, the RBSS algorithms and the DBSA, i.e. with the fact the complexities of all these smoothers are approximately of order  $\mathcal{O}(M N_p D_L^3 T)$  (provided that  $n_i = 1$  is selected for the DBSA). Moreover, we have found that a 5.5% reduction in CTB is obtained if the DDBSA is employed in place of the DBSA (i.e., if double Bayesian smoothing is not redundant). Similar considerations hold for the SDBSA, the SDDBSA and the SPS algorithm. In fact,  $\text{CTB}(\text{SDBSA})$  is approximately 0.57 times smaller than  $\text{CTB}(\text{SPS})$ ; moreover, the

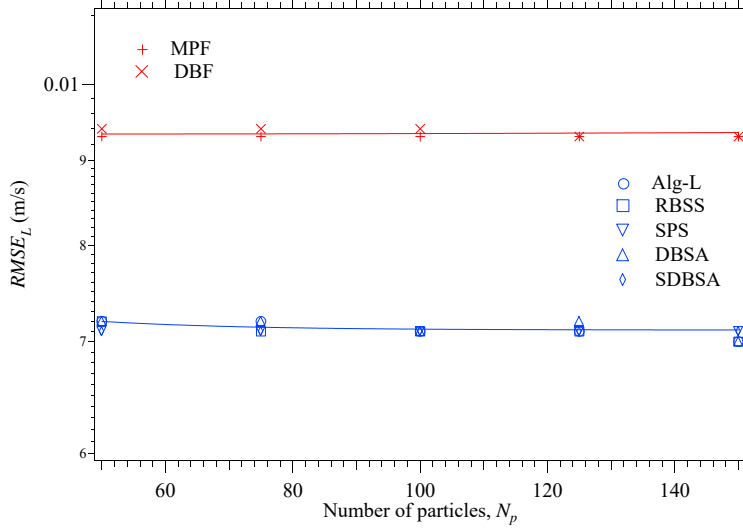


Figure 6: RMSE performance versus  $N_p$  for the linear component ( $RMSE_L$ ) of the state of SSM #1; five smoothing algorithms (Alg-L, the DBSA, the SDBSA, the RBSS algorithm and the SPS algorithm) and two filtering techniques (MPF and DBF) are considered.

CTB is reduced by 6.8% if the SDDBSA is employed in place of the SDBSA. It is also interesting to note that CTB(DBF) is approximately 0.55 times smaller than CTB(MPF) for the same value of  $N_p$ ; once again, this result is in agreement with the results shown in [20] for a different SSM.

All the numerical results illustrated above lead to the conclusion that, in the considered scenario, the DDBSA and the SDDBSA achieve the best accuracy-complexity tradeoff in their categories of smoothing techniques.

The second SSM we considered is the same as the second SSM illustrated in [20, Sec. IV] and refers to a sensor network employing  $P$  sensors placed on the vertices of a square grid (partitioning a square area having side equal to  $l$  m); these sensors receive the reference signals radiated, at the same power level and at the same frequency, by  $N$  independent targets moving on a plane. Each target evolves according to the motion model described by Eqs. (120)-(121) with  $\mathbf{a}_k(\mathbf{p}_k) = \mathbf{0}$  for any  $k$ . In this case, the considered SSM (denoted SSM#2 in the following) refers to the whole set of targets and its state vector  $\mathbf{x}_k$  results from the ordered concatenation of the vectors  $\{\mathbf{x}_k^{(i)}; i = 1, 2, \dots, N\}$ , where  $\mathbf{x}_k^{(i)} \triangleq [(\mathbf{v}_k^{(i)})^T, (\mathbf{p}_k^{(i)})^T]^T$ , and  $\mathbf{v}_k^{(i)}$  and  $\mathbf{p}_k^{(i)}$  represent the  $i$ -th target *velocity* and the *position*, respectively. Moreover, the following additional assumptions have been made about this SSM: 1) the process noises  $\mathbf{n}_{p,k}^{(i)}$  and  $\mathbf{n}_{v,k}^{(i)}$ , affecting the  $i$ -th target position and velocity, respectively, are given by  $\mathbf{n}_{p,k}^{(i)} = (T_s^2/2) \mathbf{n}_{a,k}^{(i)}$  and  $\mathbf{n}_{v,k}^{(i)} = T_s \mathbf{n}_{a,k}^{(i)}$ , where  $\{\mathbf{n}_{a,k}^{(i)}\}$  is two-dimensional AWGN, representing a random acceleration and having covariance matrix  $\sigma_a^2 \mathbf{I}_2$  (with  $i = 1, 2, \dots, N$ ); 2) the measurement acquired by the  $q$ -th sensor (with  $q = 1, 2, \dots, P$ ) in the  $k$ -th observation interval is given by

$$y_{q,k} = 10 \log_{10} \left( \Psi \sum_{i=1}^N \frac{d_0^2}{\|\mathbf{s}_q - \mathbf{p}_k^{(i)}\|^2} \right) + e_k, \quad (124)$$

where the measurement noise  $\{e_k\}$  is AWGN with variance  $\sigma_e^2$ ,  $\Psi$  denotes the normalised power received by each sensor from any target at a distance  $d_0$  from the sensor itself and  $\mathbf{s}_q$  is the position of the considered sensor; 3) the overall measurement vector  $\mathbf{y}_k$  results from the ordered concatenation of the measurements  $\{y_{q,k}; q = 1, 2, \dots, P\}$  and, consequently, provides information about the position only; 4) the initial position  $\mathbf{p}_0^{(i)} \triangleq [p_{x,0}^{(i)}, p_{y,0}^{(i)}]^T$  and the initial velocity  $\mathbf{v}_0^{(i)} \triangleq [v_{x,0}^{(i)}, v_{y,0}^{(i)}]^T$  of the  $i$ -th target are randomly selected (with  $i = 1, 2, \dots, N$ ). As far as the last point is concerned, it is important to mention that, in our computer simulations, distinct targets are placed in different squares of the partitioned area in a random fashion; moreover, the initial velocity of each target is randomly selected within the interval  $(v_{\min}, v_{\max})$  in order

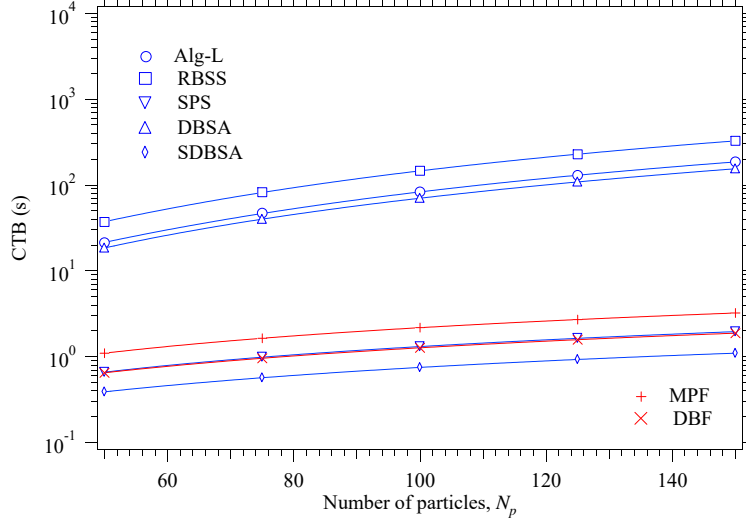


Figure 7: CTB versus  $N_p$  for five smoothing algorithms (Alg-L, DBSA, SDBSA, the RBSS algorithm and the SPS algorithm) and two filtering techniques (MPF and DBF); SSM #1 is considered.

to ensure that the trajectories of distinct targets do not cross each other in the observation interval. The following values have been selected for the parameters of SSM#2:  $P = 25$ ,  $l = 10^3$  m,  $T_s = 1$  s,  $\rho = 1$ ,  $\sigma_a^2 = 0.1$  m/s<sup>2</sup>,  $\sigma_e^2 = -35$  dB,  $\Psi = 1$ ,  $d_0 = 1$  m,  $v_{\min} = 0$  m/s and  $v_{\max} = 0.1$  m/s. Moreover,  $N = 3$  targets have been observed over a time interval lasting  $T = 60 T_s$  s. Our computer simulations have aimed at evaluating the accuracy achieved by the considered smoothing algorithms in tracking the position of all the targets. In practice, such an accuracy has been assessed by estimating the average RMSE referring to the estimates of the whole set  $\{\mathbf{p}_k^{(i)}; i = 1, 2, 3\}$ ; note that, if the  $i$ -th target is considered, its position  $\mathbf{p}_k^{(i)}$  represents the *nonlinear* component of the associated substate  $\mathbf{x}_k^{(i)}$ , because of the nonlinear dependence of  $\mathbf{y}_k$  on it (see Eq. (124)). Our computer simulations have evidenced that, in the considered scenario, the MPF and the SDBF techniques diverge frequently in the observation interval (some numerical results about the probability of divergence area available in [20, Sec. IV]); unluckily, when this occurs, all the smoothing algorithms that employ these techniques in their forward pass (namely, Alg-L, the RBSS algorithm, the SPS algorithm, the DBSA and the SDBSA) are unable to recover from this event and, consequently, are useless. The DBF technique, instead, thanks to its inner redundancy, is still able to track all the targets. Moreover, the two smoothing algorithms employing this technique in their forward pass (namely, the DBSA and the SDBSA), are able to improve the accuracy of position estimates in their backward pass; this is evidenced by Fig. 8, that shows the dependence of  $RMSE_N$  on the overall number of particles ( $N_p$ ) for the DBF technique, the DBSA and the SDBSA (the range  $[300, 600]$  is considered for  $N_p$ ). Note that the SDBSA is outperformed by the DBSA in terms of  $RMSE_N$ ; for instance,  $RMSE_N(\text{SDBSA})$  is about 1.31 times larger than  $RMSE_N(\text{DBSA})$  for  $N_p = 500$ . However, this result is achieved at the price of a significantly higher complexity; in fact,  $\text{CTB}(\text{SDBSA})$  is approximately equal to  $2 \cdot 10^{-3} \cdot \text{CTB}(\text{DBSA})$ .

## VI. Conclusions

In this manuscript, factor graph methods have been exploited to develop new smoothing algorithms based on the interconnection of two Bayesian filters in the forward pass and of two backward information filters in the backward pass. This has allowed us to develop a new approximate method for Bayesian smoothing, called *double Bayesian smoothing*. Four double Bayesian smoothers have been derived for the class of conditionally linear Gaussian systems and have been compared, in terms of both accuracy and execution time, with other smoothing algorithms for two specific dynamic models. Our simulation results lead to the conclusion that the devised algorithms can achieve a better complexity-accuracy tradeoff and a better tracking capability than other smoothing techniques recently appeared in the literature.

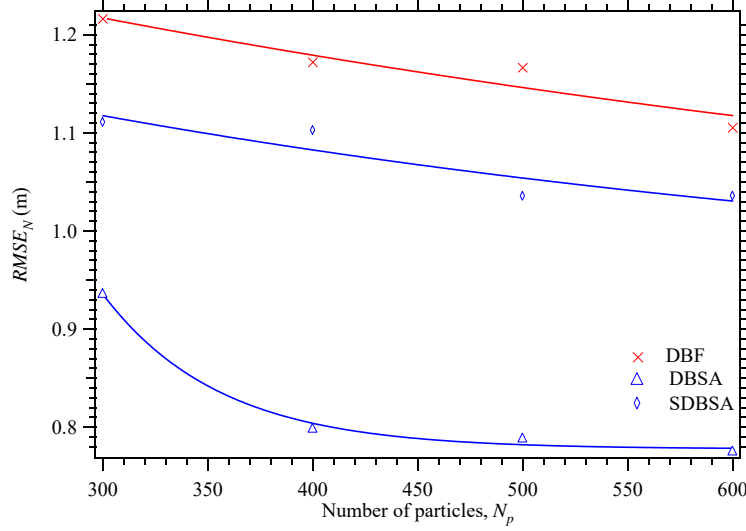


Figure 8: RMSE performance versus  $N_p$  for the nonlinear component ( $RMSE_N$ ) of the state of SSM #2; two smoothing algorithms (the DBSA and the SDBSA) and one filtering technique (DBF) are considered.

## Acknowledgment

We would like to thank the anonymous Reviewers for their constructive comments, that really helped us to improve the quality of this manuscript.

## Appendix A

In this Appendix, the derivation of the expressions of various messages evaluated in each of the three phases the DBSA consists of is sketched.

**Phase I** - Formulas (39) and (40), referring to the message  $\tilde{m}_1(\mathbf{x}_k)$  (38), can be easily computed by applying eqs. (IV.6)-(IV.8) of ref. [21, Table 4, p.1304] in their backward form (with  $A \rightarrow \mathbf{I}_D$ ,  $X \rightarrow \mathbf{F}_k \mathbf{x}_k$ ,  $Z \rightarrow \mathbf{x}_{k+1}$  and  $Y \rightarrow \mathbf{u}_k + \mathbf{w}_k$ ) and, then, eqs. (III.5)-(III.6) of [21, Table 3, p.1304] (with  $A \rightarrow \mathbf{F}_k$ ,  $X \rightarrow \mathbf{x}_k$  and  $Y \rightarrow \mathbf{F}_k \mathbf{x}_k$ ).

**Phase II** -Step 1) The message  $m_2^{(n)}(\mathbf{x}_k)$  (42) results from merging, in the  $\text{BIF}_2 \rightarrow \text{BIF}_1$  block, the statistical information about the *nonlinear* state component conveyed by the message  $m_1^{(n-1)}(\mathbf{x}_k^{(N)})$  (41) (and, consequently, by its  $N_p$  components  $\{m_{1,j}^{(n-1)}(\mathbf{x}_k^{(N)}) = W_{1,k,j}^{(n-1)} \delta(\mathbf{x}_k^{(N)} - \mathbf{x}_{k,j}^{(N)})\}$ ) with those provided by the pseudo-measurement  $\mathbf{z}_k^{(L)}$  (21) about the *linear* state component. The method employed for processing this pseudo-measurement is the same as that developed for MPF and can be summarised as follows (additional mathematical details can be found in [23, Sec. IV, p. 1527]):

1) The particles  $\mathbf{x}_{k,j}^{(N)}$  and  $\mathbf{x}_{\text{be},k+1}^{(N)}$ , conveyed by the messages  $m_1^{(n-1)}(\mathbf{x}_k^{(N)})$  (41) and  $\tilde{m}_{\text{be}}(\mathbf{x}_{k+1}^{(N)})$  (37), respectively, are employed to compute the  $j$ -th realization  $\mathbf{z}_{k,j}^{(L)}$  (49) of  $\mathbf{z}_k^{(L)}$  for  $j = 1, 2, \dots, N_p$ .

2) The pseudo-measurement  $\mathbf{z}_{k,j}^{(L)}$  (49) is exploited to generate the (particle-dependent) pdf

$$f_j^{(n)}(\mathbf{x}_k^{(L)}) = \mathcal{N}(\mathbf{x}_k^{(L)}; \tilde{\eta}_{k,j}, \tilde{\mathbf{C}}_{k,j}), \quad (125)$$

that conveys pseudo-measurement information about  $\mathbf{x}_k^{(L)}$  for any  $j$ ; the covariance matrix  $\tilde{\mathbf{C}}_{k,j}$  and the mean vector  $\tilde{\eta}_{k,j}$  of this message are computed on the basis of the precision matrix  $\tilde{\mathbf{W}}_{k,j}$  (47) and the transformed mean vector  $\tilde{\mathbf{w}}_{k,j}$  (48), respectively.

3) The messages  $\{m_{1,j}^{(n-1)}(\mathbf{x}_k^{(N)})\}$  are merged with the pdfs  $\{f_j^{(n)}(\mathbf{x}_k^{(L)})\}$  to generate the message  $m_2^{(n)}(\mathbf{x}_k)$  (42). The approach we adopt to achieve this result is based on the fact that the message  $m_1^{(n-1)}(\mathbf{x}_k^{(N)})$  and

the pdf  $f_j^{(n)}(\mathbf{x}_k^{(L)})$  refer to the same particle (i.e., to the  $j$ -th particle  $\mathbf{x}_{k,j}^{(N)}$ , but provide *complementary* information (since they refer to the two different components of the overall state  $\mathbf{x}_k$ ). This allows us to condense the statistical information conveyed by the sets  $\{m_{1,j}^{(n-1)}(\mathbf{x}_k^{(N)})\}$  and  $\{f_j^{(n)}(\mathbf{x}_k^{(L)})\}$  in the joint pdf

$$f^{(n)}(\mathbf{x}_k^{(L)}, \mathbf{x}_k^{(N)}) \triangleq w_p \sum_{j=1}^{N_p} m_{1,j}^{(n-1)}(\mathbf{x}_k^{(N)}) f_j^{(n)}(\mathbf{x}_k^{(L)}). \quad (126)$$

referring to the whole state  $\mathbf{x}_k$ . Then, the message  $m_2^{(n)}(\mathbf{x}_k)$  (42) is computed by projecting the pdf  $f^{(k)}(\mathbf{x}_k^{(L)}, \mathbf{x}_k^{(N)})$  (126) onto a single Gaussian pdf having the same *mean* and *covariance*.

Steps 2 and 3) The expression (52) of  $\tilde{m}_3^{(n)}(\mathbf{x}_k)$  represents a straightforward application of formula no. 2 of ref. [23, App. A, TABLE I] (with  $\mathbf{W}_1 \rightarrow \mathbf{W}_{1,k}$ ,  $\mathbf{W}_2 \rightarrow \mathbf{W}_{2,k}$ ,  $\mathbf{w}_1 \rightarrow \mathbf{w}_{1,k}$  and  $\mathbf{w}_2 \rightarrow \mathbf{w}_{2,k}$ ). The same considerations apply to the derivation of the expression (58) of  $m_4^{(n)}(\mathbf{x}_k)$ .

Step 4) The expression (63) of the weight  $w_{3,k,j}^{(n)}$  is derived as follows. First, we substitute the expression (19) of  $f(\mathbf{x}_{k+1}^{(N)} | \mathbf{x}_k^{(N)}, \mathbf{x}_k^{(L)})$ , and the expressions of the messages  $\tilde{m}_{be}(\mathbf{x}_{k+1}^{(N)})$  (37) and  $m_1^{(n)}(\mathbf{x}_k^{(L)})$  (61) in the *right-hand side* (RHS) of Eq. (62). Then, the resulting integral is solved by applying formula no. 1 of [23, App. A, TABLE II] in the integration with respect to  $\mathbf{x}_k^{(L)}$  and the sifting property of the Dirac delta function in the integration with respect to  $\mathbf{x}_{k+1}^{(N)}$ .

Step 5) - The derivation of the expression (68) of the weight  $w_{2,k,j}^{(n)}$  is similar to that illustrated for the particle weights originating from the pseudo-measurements in dual MPF and can be summarised as follows (additional mathematical details can be found in ref. [23, Sec. V, pp. 1528-1529]). Two different Gaussian densities are derived for the random vector  $\mathbf{z}_k^{(N)}$  (24), *conditioned on*  $\mathbf{x}_k^{(N)}$ . The expression of the first density originates from the definition (24) and from the knowledge of the *joint* pdf of  $\mathbf{x}_k^{(L)}$  and  $\mathbf{x}_{k+1}^{(L)}$ ; this joint density is obtained from: a) the statistical information provided by the message  $m_1^{(n)}(\mathbf{x}_k^{(L)})$  (61) and the pdf  $\mathcal{N}(\mathbf{x}_{k+1}^{(L)}, \tilde{\eta}_{be,k+1} \tilde{\mathbf{C}}_{be,k+1})$  (resulting from integrating out the dependence of  $\tilde{m}_{be}(\mathbf{x}_{k+1})$  (36) on  $\mathbf{x}_k^{(N)}$ ); b) the Markov model  $f(\mathbf{x}_{k+1}^{(L)} | \mathbf{x}_k^{(N)}, \mathbf{x}_k^{(L)})$  (26). This leads to the pdf

$$f_1^{(n)}(\mathbf{z}_k^{(N)} | \mathbf{x}_k^{(N)}) = \mathcal{N}(\mathbf{z}_k^{(N)}; \tilde{\eta}_{z,k}^{(n)}(\mathbf{x}_k^{(N)}), \tilde{\mathbf{C}}_{z,k}^{(n)}(\mathbf{x}_k^{(N)})), \quad (127)$$

where

$$\tilde{\eta}_{z,k}^{(n)}(\mathbf{x}_k^{(N)}) = \tilde{\eta}_{be,k+1} - \mathbf{A}_k^{(L)}(\mathbf{x}_k^{(N)}) \tilde{\eta}_{1,k}^{(n)} \quad (128)$$

and

$$\tilde{\mathbf{C}}_{z,k}^{(n)}(\mathbf{x}_k^{(N)}) = \tilde{\mathbf{C}}_{be,k+1} - \mathbf{A}_k^{(L)}(\mathbf{x}_k^{(N)}) \tilde{\mathbf{C}}_{1,k}^{(n)}(\mathbf{A}_k^{(L)}(\mathbf{x}_k^{(N)}))^T. \quad (129)$$

The second pdf of  $\mathbf{z}_k^{(N)}$ , instead, results from the fact that this vector  $\mathbf{z}_k^{(N)}$  must equal the sum (25); consequently, it is given by

$$f_2(\mathbf{z}_k^{(N)} | \mathbf{x}_k^{(N)}) = \mathcal{N}(\mathbf{z}_k^{(N)}; \mathbf{f}_k^{(L)}(\mathbf{x}_k^{(N)}), \mathbf{C}_w^{(N)}). \quad (130)$$

Given the pdfs (127) and (130), the message  $\tilde{m}_3^{(n)}(\mathbf{x}_k^{(N)})$  is expressed by their *correlation*, i.e. it is computed as

$$\tilde{m}_3^{(n)}(\mathbf{x}_k^{(N)}) = \int f_1^{(n)}(\mathbf{z}_k^{(N)} | \mathbf{x}_k^{(N)}) \cdot f_2(\mathbf{z}_k^{(N)} | \mathbf{x}_k^{(N)}) d\mathbf{z}_k^{(N)}. \quad (131)$$

Substituting Eqs. (127) and (130) in the RHS of the last expression, setting  $\mathbf{x}_k^{(N)} = \mathbf{x}_{k,j}^{(N)}$  and applying formula no. 4 of ref. [23, Table II] to the evaluation of the resulting integral yields Eq. (68); note that  $\tilde{\eta}_{z,k,j}^{(n)}$  (70) and  $\tilde{\mathbf{C}}_{z,k,j}^{(n)}$  (71) represent the values taken on by  $\tilde{\eta}_{z,k}^{(n)}(\mathbf{x}_k^{(N)})$  (128) and  $\tilde{\mathbf{C}}_{z,k}^{(n)}(\mathbf{x}_k^{(N)})$  (129), respectively, for  $\mathbf{x}_k^{(N)} = \mathbf{x}_{k,j}^{(N)}$ .

Step 7) The expression (82) of the weight  $w_{5,k,j}^{(n)}$  is derived as follows. First, we substitute the expressions (61) and (20) of  $m_1^{(n)}(\mathbf{x}_k^{(L)})$  and  $f(\mathbf{y}_k|\mathbf{x}_k^{(N)}, \mathbf{x}_k^{(L)})$ , respectively, in the RHS of Eq. (77). Then, solving the resulting integral (see formula no. 1 of ref. [23, App. A, TABLE II]) produces Eq. (78). Finally, setting  $\mathbf{x}_k^{(N)} = \mathbf{x}_{k,j}^{(N)}$  in Eq. (78) yields Eq. (82).

**Phase III** - The expression (100) of the message  $\tilde{m}_{\text{be}}(\mathbf{x}_k)$  results from the application of formula no. 2 of ref. [23, App. A, TABLE I] to Eq. (99).

## Appendix B Computational complexity of the devised double Bayesian smoothers

In this appendix, the computational complexity of the tasks accomplished *in a single recursion* of backward filtering and smoothing of the DBSA is assessed in terms of flops. Moreover, we comment on how the illustrated results can be also exploited to assess the computational complexity of a single recursion of the DDBSA. In the following,  $\mathcal{C}_{\mathbf{H}}, \mathcal{C}_{\mathbf{B}}, \mathcal{C}_{\mathbf{F}}, \mathcal{C}_{\mathbf{A}^{(L)}} \text{ and } \mathcal{C}_{\mathbf{A}^{(N)}}$ , and  $\mathcal{C}_{\mathbf{g}}, \mathcal{C}_{\mathbf{f}^{(L)}}, \mathcal{C}_{\mathbf{f}^{(N)}}$  and  $\mathcal{C}_{\mathbf{f}_k}$  denote the cost due to the evaluation of the matrices  $\mathbf{H}_k, \mathbf{B}_k, \mathbf{F}_k, \mathbf{A}_k^{(L)}(\mathbf{x}_k^{(N)})$  and  $\mathbf{A}_k^{(N)}(\mathbf{x}_k^{(N)})$ , and of the functions  $\mathbf{g}_k(\mathbf{x}_k^{(N)}), \mathbf{f}_k^{(L)}(\mathbf{x}_k^{(N)}), \mathbf{f}_k^{(N)}(\mathbf{x}_k^{(N)})$  and  $\mathbf{f}_k(\mathbf{x}_k)$ , respectively. Moreover, similarly as [26], it is assumed that the computation of the inverse of any covariance matrix involves a Cholesky decomposition of the matrix itself and the inversion of a lower or upper triangular matrix. Finally, it is assumed that the computation of the determinant of any matrix involves a Cholesky decomposition of the matrix itself and the product of the diagonal entries of a triangular matrix.

**Phase I** - The overall computational cost of this task is evaluated as (see Eqs. (39)-(40) and (47)-(49))

$$\begin{aligned} \mathcal{C}_1 = & \mathcal{C}_{\mathbf{w}_{1,k}} + \mathcal{C}_{\mathbf{w}_{1,k}} + N_p \left( \mathcal{C}_{\mathbf{z}_{k,j}^{(L)}} + \mathcal{C}_{\tilde{\mathbf{w}}_{k,j}} \right. \\ & \left. + \mathcal{C}_{\tilde{\mathbf{c}}_{k,j}} + \mathcal{C}_{\tilde{\eta}_{k,j}} \right) \triangleq \mathcal{C}_{\text{bp}}^{(1)}. \end{aligned} \quad (132)$$

Moreover, we have that: 1) the cost  $\mathcal{C}_{\mathbf{w}_{1,k}}$  is equal to  $\mathcal{C}_{\mathbf{F}} + 26D^3/3 - D^2/2 + 5D/6$  flops; 2) the cost  $\mathcal{C}_{\mathbf{w}_{1,k}}$  is equal to  $4D^3 + 4D^2 - 2D$  flops (the cost for computing  $\mathcal{C}_{\mathbf{F}}$  has been already accounted for at point 1)); 3) the cost  $\mathcal{C}_{\mathbf{z}_{k,j}^{(L)}}$  is equal to  $\mathcal{C}_{\mathbf{f}^{(N)}} + D_N$  flops; 4) the cost  $\mathcal{C}_{\tilde{\mathbf{w}}_{k,j}}$  is equal to  $\mathcal{C}_{\mathbf{A}^{(N)}} + 4D_N^3 - 2D_N^2$  flops; 5) the cost  $\mathcal{C}_{\tilde{\mathbf{w}}_{k,j}}$  is equal to  $2D_N^3 + D_N^2 - D_N$  flops (the cost for computing  $\mathcal{C}_{\mathbf{A}^{(N)}}$  has been already accounted for at point 4)); 6) the cost  $\mathcal{C}_{\tilde{\mathbf{c}}_{k,j}}$  is equal to  $2D_N^3/3 + 3D_N^2/2 + 5D_N/6$  flops; 7) the cost  $\mathcal{C}_{\tilde{\eta}_{k,j}}$  is equal to  $2D_N^2 - D_N$  flops. The expressions listed at points 1)-2) can be exploited for the DDBSA too; in the last case, however,  $D_N = 0$  and  $D = D_L$  must be assumed.

**Phase II** - The overall computational cost of this task is evaluated as

$$\begin{aligned} \mathcal{C}_2 = & n_i \left( \mathcal{C}_{\text{pm}^{(1)}} + \mathcal{C}_{\text{be1}^{(1)}} + \mathcal{C}_{\text{sm}^{(1)}} + \mathcal{C}_{\text{bp}^{(2)}} + \right. \\ & \left. \mathcal{C}_{\text{pm}^{(2)}} + \mathcal{C}_{\text{ms}^{(2)}} + \mathcal{C}_{\text{be2}^{(2)}} + \mathcal{C}_{\text{sm}^{(2)}} \right). \end{aligned} \quad (133)$$

The terms appearing in the RHS of the last equation can be computed as follows. First of all, we have that

$$\mathcal{C}_{\text{pm}^{(1)}} = \mathcal{C}_{\eta_{2,k}^{(n)}} + \mathcal{C}_{\mathbf{C}_{2,k}^{(n)}}, \quad (134)$$

where (see Eqs. (43)-(44)): 1) the cost  $\mathcal{C}_{\eta_{2,k}^{(n)}}$  is equal to  $2N_p D - D$  flops; 2) the cost  $\mathcal{C}_{\mathbf{C}_{2,k}^{(n)}}$  is equal to  $5N_p D_L^2 + 4N_p D_N^2 + 4N_p D_L D_N + D_L^2 + D_N^2 + D_L D_N$  flops. The expressions listed at points 1)-2) can be exploited for the DDBSA too; in the last case, however,  $D_N = 0$  and  $D = D_L$  must be assumed.

The second term appearing in the RHS of Eq. (133) is evaluated as

$$\mathcal{C}_{\text{be1}^{(1)}} = \mathcal{C}_{\mathbf{C}_{3,k}^{(n)}} + \mathcal{C}_{\eta_{3,k}^{(n)}} + \mathcal{C}_{\mathbf{w}_{3,k}^{(n)}} + \mathcal{C}_{\mathbf{w}_{3,k}^{(n)}}, \quad (135)$$

where (see Eqs. (55)-(56)): 1) the cost  $\mathcal{C}_{\mathbf{C}_{3,k}^{(n)}}$  is equal to  $14D^3/3 + D^2/2 + 5D/6$  flops; 2) the cost  $\mathcal{C}_{\eta_{3,k}^{(n)}}$  is equal to  $4D^2 - D$  flops (the cost for computing  $\mathcal{C}_{\mathbf{w}_k^{(n)}}$  has been already accounted for at point 1)); 3) the cost

$\mathcal{C}_{\mathbf{W}_{3,k}^{(n)}}$  is equal to  $2D^3/3 + 3D^2/2 + 5D/6$  flops; 4) the cost  $\mathcal{C}_{\mathbf{W}_{3,k}^{(n)}}$  is equal to  $2D^2 - D$  flops. The expressions listed at points 1)-4) can be exploited for the DDBSA too; in the last case, however,  $D_N = 0$  and  $D = D_L$  must be assumed.

The third term appearing in the RHS of Eq. (133) is computed as

$$\mathcal{C}_{\text{sm}^{(1)}} = \mathcal{C}_{\mathbf{W}_{4,k}^{(n)}} + \mathcal{C}_{\mathbf{W}_{4,k}^{(n)}} + \mathcal{C}_{\mathbf{C}_{4,k}^{(n)}} + \mathcal{C}_{\eta_{4,k}^{(n)}}, \quad (136)$$

where (see Eqs. (59)-(60)): 1) the cost  $\mathcal{C}_{\mathbf{W}_{4,k}^{(n)}}$  is equal to  $D^2$  flops; 2) the cost  $\mathcal{C}_{\mathbf{W}_{4,k}^{(n)}}$  is equal to  $D$  flops; 3) the cost  $\mathcal{C}_{\mathbf{C}_{4,k}^{(n)}}$  is equal to  $2D^3/3 + 3D^2/2 + 5D/6$  flops; 4) the cost  $\mathcal{C}_{\eta_{4,k}^{(n)}}$  is equal to  $2D^2 - D$  flops. The expressions listed at points 1)-4) can be exploited for the DDBSA too; in the last case, however,  $D_N = 0$  and  $D = D_L$  must be assumed.

The fourth term appearing in the RHS of Eq. (133) is given by

$$\mathcal{C}_{\text{bp}^{(2)}} = N_p \left( \mathcal{C}_{\eta_{3,k,j}^{(N)}} + \mathcal{C}_{\mathbf{C}_{3,k,j}^{(N)}} + \mathcal{C}_{D_{3,k,j}^{(n)}} + \mathcal{C}_{Z_{3,k,j}^{(n)}} \right), \quad (137)$$

where (see Eqs. (66)-(67) and (64)-(65)): 1) the cost  $\mathcal{C}_{\eta_{3,k,j}^{(N)}}$  is equal to  $\mathcal{C}_{\mathbf{A}^{(N)}} + \mathcal{C}_{\mathbf{F}^{(N)}} + 2D_L D_N$  flops; 2) the cost  $\mathcal{C}_{\mathbf{C}_{3,k,j}^{(N)}}$  is equal to  $2D_L^2 D_N + 2D_L D_N^2 - D_L D_N$  flops (the cost for computing  $\mathcal{C}_{\mathbf{A}^{(N)}}$  and  $\mathcal{C}_{\mathbf{F}^{(N)}}$  has been already accounted for at point 1)); 3) the cost  $\mathcal{C}_{D_{3,k,j}^{(n)}}$  is equal to  $D_N^3/3 + D_N^2 + 5D_N/3 + 2$  flops; 4) the cost  $\mathcal{C}_{Z_{3,k,j}^{(n)}}$  is equal to  $2D_N^2 + 2D_N - 1$  flops.

The fifth term appearing in the RHS of Eq. (133) is evaluated as

$$\mathcal{C}_{\text{pm}^{(2)}} = N_p \left( \mathcal{C}_{\tilde{\eta}_{z,k,j}^{(n)}} + \mathcal{C}_{\tilde{\mathbf{C}}_{z,k,j}^{(n)}} + \mathcal{C}_{\tilde{\mathbf{W}}_{2,k,j}^{(n)}} + \mathcal{C}_{\tilde{\mathbf{w}}_{2,k,j}^{(n)}} + \mathcal{C}_{D_{2,k,j}^{(n)}} + \mathcal{C}_{Z_{2,k,j}^{(n)}} \right), \quad (138)$$

where (see Eqs. (69)-(74)): 1) the cost  $\mathcal{C}_{\tilde{\eta}_{z,k,j}^{(n)}}$  is equal to  $\mathcal{C}_{\mathbf{A}^{(L)}} + 2D_L^2$  flops; 2) the cost  $\mathcal{C}_{\tilde{\mathbf{C}}_{z,k,j}^{(n)}}$  is equal to  $4D_L^3 - D_L^2$  flops (the cost for computing  $\mathcal{C}_{\mathbf{A}^{(L)}}$  has been already accounted for at point 1)); 3) the cost  $\mathcal{C}_{\tilde{\mathbf{W}}_{2,k,j}^{(n)}}$  is equal to  $2D_L^3/3 + 5D_L^2/2 + 5D_L/6$  flops; 4) the cost  $\mathcal{C}_{\tilde{\mathbf{w}}_{2,k,j}^{(n)}}$  is equal to  $\mathcal{C}_{\mathbf{F}^{(L)}} + 4D_L^2 - D_L$  flops; 5) the cost  $\mathcal{C}_{D_{2,k,j}^{(n)}}$  is equal to  $D_L^3/3 + 2D_L^2 + 5D_L/3 + 2$  flops; 6) the cost  $\mathcal{C}_{Z_{2,k,j}^{(n)}}$  is equal to  $6D_L^2 + 3D_L - 1$  flops.

The sixth term appearing in the RHS of Eq. (133) is computed as

$$\mathcal{C}_{\text{ms}^{(2)}} = N_p \left( \mathcal{C}_{\tilde{\eta}_{5,k,j}^{(n)}} + \mathcal{C}_{\tilde{\mathbf{C}}_{5,k,j}^{(n)}} + \mathcal{C}_{D_{5,k,j}^{(n)}} + \mathcal{C}_{Z_{5,k,j}^{(n)}} \right), \quad (139)$$

where (see Eqs. (83)-(86)): 1) the cost  $\mathcal{C}_{\tilde{\eta}_{5,k,j}^{(n)}}$  is equal to  $\mathcal{C}_{\mathbf{B}} + \mathcal{C}_{\mathbf{g}} + 2PD_L$  flops; 2) the cost  $\mathcal{C}_{\tilde{\mathbf{C}}_{5,k,j}^{(n)}}$  is equal to  $2PD_L^2 + 2P^2 D_L - PD_L$  flops (the cost for computing  $\mathcal{C}_{\mathbf{B}}$  has been already accounted for at point 1)); 3) the cost  $\mathcal{C}_{D_{5,k,j}^{(n)}}$  is equal to  $D_L^3/3 + D_L^2 + 5D_L/3 + 2$  flops; 4) the cost  $\mathcal{C}_{Z_{5,k,j}^{(n)}}$  is equal to  $2P^3/3 + 7P^2/2 + 17P/6 - 1$  flops. It is important to note that, if the forward weights  $\{w_{\text{fe},k,j}\}$  are reused, the cost  $\mathcal{C}_{\text{ms}^{(2)}}$  appearing in Eq. (139) is equal to zero.

The seventh term appearing in the RHS of Eq. (133) is given by

$$\mathcal{C}_{\text{be}2^{(2)}} = N_p \left( \mathcal{C}_{D_{6,k,j}^{(n)}} + \mathcal{C}_{Z_{6,k,j}^{(n)}} + \mathcal{C}_{w_{6,k,j}^{(n)}} \right), \quad (140)$$

where the costs  $\mathcal{C}_{D_{6,k,j}^{(n)}}$  and  $\mathcal{C}_{Z_{6,k,j}^{(n)}}$  are equal to 2 flops, and the cost  $\mathcal{C}_{w_{6,k,j}^{(n)}}$  is equal to 3 flops (see Eqs. (89)-(91)). If the forward weights  $\{w_{\text{fe},k,j}\}$  are reused, the costs  $\mathcal{C}_{D_{6,k,j}^{(n)}}$  and  $\mathcal{C}_{Z_{6,k,j}^{(n)}}$  are equal to 1 flops, whereas the cost  $\mathcal{C}_{w_{6,k,j}^{(n)}}$  remains unchanged.

The last term appearing in the RHS of Eq. (133) is evaluated as

$$\mathcal{C}_{\text{sm}^{(2)}} = \mathcal{C}_{w_{1,k,j}^{(n)}} + \mathcal{C}_{W_{1,k,j}^{(n)}}, \quad (141)$$

where the costs  $\mathcal{C}_{w_{1,k,j}^{(n)}}$  and  $\mathcal{C}_{W_{1,k,j}^{(n)}}$  are equal to  $N_p$  and  $2N_p - 1$  flops, respectively (see Eqs. (95)-(96)).

**Phase III** - The overall computational cost of this task is evaluated as

$$\mathcal{C}_3 = \mathcal{C}_{\text{be}(2)} + \mathcal{C}_{\text{pm}(1)} + \mathcal{C}_{\text{be1}(1)} + \mathcal{C}_{\text{be}(1)}. \quad (142)$$

Here, the cost  $\mathcal{C}_{\text{be}(2)}$  is equal to  $\mathcal{C}_S(N_p)$ , that represents the total cost of a sampling step that involves a particle set of size  $N_p$ ; moreover, the costs  $\mathcal{C}_{\text{pm}(1)}$  and  $\mathcal{C}_{\text{be1}(1)}$  are the same as those appearing in the RHS of Eq. (133), and  $\mathcal{C}_{\text{be}(1)}$  is computed as (see Eqs. (102)-(105))

$$\begin{aligned} \mathcal{C}_{\text{be}(1)} = & \mathcal{C}_{\mathbf{W}_{\text{ms},k}} + \mathcal{C}_{\mathbf{w}_{\text{ms},k}} + \mathcal{C}_{\mathbf{W}_{\text{be2},k}} + \\ & \mathcal{C}_{\mathbf{w}_{\text{be2},k}} + \mathcal{C}_{\mathbf{C}_{\text{be}}} + \mathcal{C}_{\eta_{\text{be}}}. \end{aligned} \quad (143)$$

Moreover, we have that: 1) the cost  $\mathcal{C}_{\mathbf{W}_{\text{ms},k}}$  is equal to  $\mathcal{C}_{\mathbf{H}} + 2P^2D + 2PD^2 - D^2 - PD$  flops; 2) the cost  $\mathcal{C}_{\mathbf{w}_{\text{ms},k}}$  is equal to  $\mathcal{C}_{\mathbf{B}} + \mathcal{C}_{\mathbf{g}} + 2P^2D + 3PD + 2PD_L - P - D$  flops (the cost for computing  $\mathcal{C}_{\mathbf{H}}$  has been already accounted for at point 1)); 3) the cost  $\mathcal{C}_{\mathbf{W}_{\text{be2},k}}$  is equal to  $D^2$  flops; 4) the cost  $\mathcal{C}_{\mathbf{w}_{\text{be2},k}}$  is equal to  $D$  flops; 5) the cost  $\mathcal{C}_{\mathbf{C}_{\text{be}}}$  is equal to  $2D^3/3 + 3D^2/2 + 5D/6$  flops; 6) the cost  $\mathcal{C}_{\eta_{\text{be}}}$  is equal to  $2D^2 - D$  flops. The expressions listed at points 1)-6) can be exploited for the DDBSA too; in the last case, however,  $D_N = 0$  and  $D = D_L$  must be assumed. Note that the costs  $\mathcal{C}_{\mathbf{W}_{\text{ms},k}}$  and  $\mathcal{C}_{\mathbf{w}_{\text{ms},k}}$  (see points 1) and 2)) are ignored if the precision matrix  $\mathbf{W}_{\text{ms},k}$  and the transformed mean vector  $\mathbf{w}_{\text{ms},k}$  are stored in the forward pass (so that they do not need to be recomputed in the backward pass). Moreover, if the SDBSA or the SDDBSA is used, the cost  $\mathcal{C}_{\text{be}(2)}$  in the RHS of Eq. (142) becomes  $D_N(2N_p - 1)$  flops.

Finally, it is worth stressing that, if the DBSA or the DDBSA (the SDBSA or the SDDBSA) is employed, the overall computational complexity is obtained by multiplying the computational cost assessed for a single recursion by  $MT$  (by  $T$ ), where  $M$  and  $T$  denote the overall number of accomplished backward passes and the duration of the observation interval, respectively.

## References

- [1] B. Anderson and J. Moore, **Optimal Filtering**, Englewood Cliffs, NJ, Prentice-Hall, 1979.
- [2] S. Särkkä, **Bayesian Filtering and Smoothing**. Cambridge, U.K.: Cambridge Univ. Press, 2013.
- [3] A. Doucet, S. Godsill and C. Andrieu, “On Sequential Monte Carlo Sampling Methods for Bayesian Filtering”, *Statist. Comput.*, vol. 10, no. 3, pp. 197-208, 2000.
- [4] G. Kitagawa, “Non-Gaussian state-space modeling of nonstationary time series”, *Journal of the American Statistical Association*, vol. 82, pp. 1032-1063, 1987.
- [5] G. Kitagawa, “The two-filter formula for smoothing and an implementation of the Gaussian-sum smoother”, *Annals of the Institute of Statistical Mathematics*, vol. 46, pp. 605-623, 1994.
- [6] Y. Bresler, “Two-filter formula for discrete-time non-linear Bayesian smoothing”, *Int. Journal of Control*, vol. 43, no. 2, pp. 629-641, 1986.
- [7] B. N. Vo, B. T. Vo and R. P. S. Mahler, “Closed-Form Solutions to Forward-Backward Smoothing”, *IEEE Trans. Sig. Proc.*, vol. 60, no. 1, pp. 2-17, Jan. 2012.
- [8] S. Särkkä and J. Hartikainen, “On Gaussian optimal smoothing of non-linear state space models”, *IEEE Trans. Autom. Control*, vol. 55, no. 8, pp. 1938-1941, Aug. 2010.
- [9] J. Kokkala, A. Solin, and S. Särkkä, “Sigma-point filtering and smoothing-based parameter estimation in nonlinear dynamic systems”, *J. Adv. Inf. Fusion*, vol. 11, no. 1, pp. 15-30, 2016.
- [10] A. F. García-Fernández, L. Svensson and S. Särkkä, “Iterated posterior linearisation smoother”, *IEEE Trans. Autom. Control*, vol. 62, no. 4, pp. 2056-2063, Apr. 2017.
- [11] R. Doucet, A. Garivier, E. Moulines and J. Olsson, “Sequential Monte Carlo smoothing for general state space hidden Markov models”, *Ann. Appl. Probab.*, vol. 21, no. 6, pp. 2109-2145, 2011.



- [12] G. Kitagawa, “Monte Carlo filter and smoother for non-Gaussian nonlinear state space models”, *J. Comput. Graph. Statist.*, vol. 5, no. 1, pp. 1–25, 1996.
- [13] S. J. Godsill, A. Doucet, and M. West, “Monte Carlo smoothing for nonlinear time series”, *J. Amer. Statist. Assoc.*, vol. 99, no. 465, pp. 156–168, Mar. 2004.
- [14] F. Lindsten and T. B. Schön, “Backward simulation methods for Monte Carlo statistical inference”, *Foundat. Trends Mach. Learn.*, vol. 6, no. 1, pp. 1–143, 2013.
- [15] R. Hostettler and S. Särkkä, “Rao–Blackwellized Gaussian Smoothing”, *IEEE Trans. Autom. Control*, vol. 64, no. 1, pp. 305–312, Jan. 2019.
- [16] M. Briers, A. Doucet and S. Maskell, “Smoothing algorithms for state-space models”, *Ann. Inst. Statist. Math.*, vol. 62, no. 1, pp. 61–89, Feb. 2010.
- [17] W. Fong, S. J. Godsill, A. Doucet and M. West, “Monte Carlo smoothing with application to audio signal enhancement”, *IEEE Trans. Signal Process.*, vol. 50, no. 2, pp. 438–449, Feb. 2002.
- [18] F. Lindsten, P. Bunch, S. Särkkä, T. B. Schön and S. J. Godsill, “Rao-Blackwellized Particle Smoothers for Conditionally Linear Gaussian Models”, *IEEE J. Sel. Topics in Sig. Proc.*, vol. 10, no. 2, pp. 353–365, March 2016.
- [19] G. M. Vitetta, E. Sirignano and F. Montorsi, “Particle Smoothing for Conditionally Linear Gaussian Models as Message Passing over Factor Graphs”, *IEEE Trans. Sig. Proc.*, vol. 66, no. 14, pp. 3633–3648, July 2018.
- [20] G. M. Vitetta, P. Di Viesti and E. Sirignano, “Multiple Bayesian Filtering as Message Passing”, submitted to the *IEEE Trans. Sig. Proc.*, February 2019 (available on arXiv at <https://arxiv.org/abs/1907.01358>)
- [21] H.-A. Loeliger, J. Dauwels, Junli Hu, S. Korl, Li Ping, F. R. Kschischang, “The Factor Graph Approach to Model-Based Signal Processing”, *IEEE Proc.*, vol. 95, no. 6, pp. 1295–1322, June 2007.
- [22] F. R. Kschischang, B. Frey, and H. Loeliger, “Factor Graphs and the Sum-Product Algorithm”, *IEEE Trans. Inf. Theory*, vol. 41, no. 2, pp. 498–519, Feb. 2001.
- [23] G. M. Vitetta, E. Sirignano, P. Di Viesti and F. Montorsi, “Marginalized Particle Filtering and Related Techniques as Message Passing”, *IEEE Trans. Sig. Proc.*, vol. 67, no. 6, pp. 1522–1535, Mar. 2019.
- [24] T. Schön, F. Gustafsson, P.-J. Nordlund, “Marginalized Particle Filters for Mixed Linear/Nonlinear State-Space Models”, *IEEE Trans. Sig. Proc.*, vol. 53, no. 7, pp. 2279–2289, July 2005.
- [25] M. S. Arulampalam, S. Maskell, N. Gordon and T. Clapp, “A Tutorial on Particle Filters for Online Nonlinear/Non-Gaussian Bayesian Tracking”, *IEEE Trans. Sig. Proc.*, vol. 50, no. 2, pp. 174–188, Feb. 2002.
- [26] B. Ait-El-Fquih and I. Hoteit, “A variational Bayesian multiple particle filtering scheme for large-dimensional systems”, *IEEE Trans. Sig. Proc.*, vol. 64, no. 20, pp. 5409–5422, Oct. 2016.