Preprints of the 23rd IFAC World Congress
Busan, Republic of Korea, August 23-28, 2026

Smoothers for Lagrangian and Eulerian Grid-Based State Estimators ⋆

Jakub Matoušek * **Jindřich Duník** *

* *Dept. of Cyber., Univ. of West Bohemia in Pilsen Univerzitní 8, 306 14 Pilsen, Czech Republic Email: {matoujak, dunikj}@kky.zcu.cz*

Abstract: This paper addresses state-estimation problems in non-linear stochastic dynamical systems, with particular focus on the smoothing stage. The main contribution is the derivation of the Lagrangian grid-based smoother. In addition, the existing Eulerian smoothing formulations are collected and organized into a coherent framework that clarifies their relationships and computational structure. A unified overview of the analytical computational and memory complexities of both Eulerian and Lagrangian smoothers is also provided. The proposed Lagrangian smoother has been implemented in MATLAB®, and the code is publicly available *.

* https://github.com/pesslovany/Matlab-LagrangianPMF-simulated-smoothing

Keywords: Non-linear systems; State estimation; Grid based filters; Smoothing; Point-mass filter

1. INTRODUCTION

State estimation is essential in numerous areas of modern society, including navigation, timekeeping, speech and image processing, fault detection, optimal control, and tracking. Consequently, it has been a subject of considerable research interest for several decades.

This paper deals with Bayesian state estimation for discrete-time stochastic state-space models. The focus lies on numerical solutions of the associated *functional recursive relations* (FRRs) that yield the probability-density functions (PDFs) conditioned on the measurements. Numerical methods are indispensable for highly non-linear or non-Gaussian models, in which the FRRs lack closed-form solutions and local filters—such as the extended or unscented Kalman filter—fail to achieve acceptable performance (Matoušek et al., 2025).

Two principal numerical strategies for solving the FRRs are reported in the literature: approaches based on *stochastic* and on *deterministic* integration rules. The former relies on Monte Carlo (MC) integration and leads to the family of particle estimators (PEs) (Doucet et al., 2001). The latter employs deterministic quadrature rules, for example the midpoint rule, and results in the grid-based (point-mass) estimators (GbEs) (Šimandl et al., 2006a). Standard GbEs follow an Eulerian philosophy, i.e., the grid is independent of the model dynamics, and exhibit computational complexity $\mathcal{O}(N^2)$ with respect to the number of grid points N. For invertible dynamics, *Lagrangian* GbEs can be adopted (LGbEs); here the grid is generated according to the dynamics, and the complexity reduces to $\mathcal{O}(N \log_2 N)$ (Matoušek et al., 2025; Matoušek et al., 2025; Matoušek et al., 2023a), the same as the PEs.

It has been shown that GbEs offer increased resilience to initialisation errors and measurement outliers (Ånonsen and Hagen, 2010; Ma et al., 2023; Anonsen and Hallingstad, 2006; Orguner et al., 2010). An additional advantage is their deterministic character: identical data produce identical estimates across repeated runs and computing platforms, this is beneficial for safety critical applications. Recently, tensor-decomposition algorithms are being explored as a means to mitigate the curse of dimensionality in GbEs (Gehlen et al., 2024; Matoušek et al., 2024); such techniques are applicable precisely because of the structured nature of the underlying grids, and cannot be applied to PEs.

The Lagrangian grid-based filter (LGbFs) were derived and tested for both continuous and discrete time dynamics models in (Matoušek et al., 2023b,a). A lower computational complexity w.r.t discrete LGbFs was achieved using a spectral differentiation for continuous-time solution in (Matoušek et al., 2025). A tutorial like introduction to LGbFs can be found in (Matoušek et al., 2025). The LGbF gives qualitatively the same results as standard grid-based filter (GbF), the only difference is computational complexity and storage complexity. Same for the smoothing, therefore the focus of this paper is laid on the complexities only.

To the authors' knowledge, an up-to-date overview of smoothing techniques for GbEs is lacking, and the smoothing step has not yet been derived for the newly proposed efficient Lagrangian GbEs (LGbEs).

Accordingly, this paper pursues three objectives, namely to:

(i) Provide an up-to-date overview of smoothing for Eulerian GbEs (EGbEs);
(ii) Derive the smoothing for Lagrangian GbEs;
(iii) Release MATLAB® implementation of proposed smoother, enabling other researchers to experiment with it readily.

2. MODEL AND BAYESIAN ESTIMATION

The following discrete-time state-space model of a non-linear stochastic dynamic system with additive noises

⋆ This work was supported by the Czech Science Foundation (GACR) under grant GA 25-16919J.

©2026 the authors. Accepted by IFAC for publication under a Creative Commons Licence CC-BY-NC-ND

$$\mathbf{x}_{k+1} = \mathbf{f}_k(\mathbf{x}_k) + \mathbf{w}_k, \quad (1)$$
$$\mathbf{z}_k = \mathbf{h}_k(\mathbf{x}_k) + \mathbf{v}_k, \quad (2)$$

is considered, where the vectors $\mathbf{x}_k \in \mathbb{R}^{n_x}$, and $\mathbf{z}_k \in \mathbb{R}^{n_z}$ represent the *unknown* state of the model and measurement at time instant k, respectively. The state dynamics $\mathbf{f}_k : \mathbb{R}^{n_x} \to \mathbb{R}^{n_x}$, and measurement function $\mathbf{h}_k : \mathbb{R}^{n_x} \to \mathbb{R}^{n_z}$ are supposed to be *known*. Particular realizations of the state and measurement noises $\mathbf{w}_k$ and $\mathbf{v}_k$ are *unknown*, but their PDFs, i.e., the state noise PDF $p_{\mathbf{w}_k}(\mathbf{w}_k)$ and the measurement noise PDF $p_{\mathbf{v}_k}(\mathbf{v}_k)$, are supposed to be *known*, the initial state PDF $p_{\mathbf{x}_0}(\mathbf{x}_0)$ is also *known*. The noises and initial state random variable (RV) are independent.

2.1 Bayesian Estimation

The main goal of the state estimation in the Bayesian framework is to find the PDF of the state $\mathbf{x}_k$ conditioned on all measurements $\mathbf{z}^l = [\mathbf{z}_0, \mathbf{z}_1, \ldots, \mathbf{z}_l]$ up to the time instant l, i.e., the conditional PDF $p(\mathbf{x}_k|\mathbf{z}^l), \forall k$, is sought. If $k = l$, then the estimation task is *filtering*, if $k > l$, then it is *prediction*, and if $k < l$, the task is *smoothing*.

2.2 Bayesian Recursive Relations for Prediction and Filtering

The general solution to the filtering and predictive state estimation is given by the Bayesian recursive relations (BRRs) for the filtering and predictive conditional PDFs computation (Anderson and Moore, 1979)

$$p(\mathbf{x}_k|\mathbf{z}^k) = \frac{p(\mathbf{x}_k|\mathbf{z}^{k-1})p(\mathbf{z}_k|\mathbf{x}_k)}{p(\mathbf{z}_k|\mathbf{z}^{k-1})}, \quad (3)$$
$$p(\mathbf{x}_{k+1}|\mathbf{z}^k) = \int p(\mathbf{x}_{k+1}|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{z}^k)d\mathbf{x}_k, \quad (4)$$

where $p(\mathbf{x}_{k+1}|\mathbf{z}^k)$ is the one-step predictive PDF computed by the Chapman-Kolmogorov equation (CKE (4) and $p(\mathbf{x}_k|\mathbf{z}^k)$ is the filtering PDF computed by the Bayes' rule (3). The PDFs $p(\mathbf{x}_{k+1}|\mathbf{x}_k) = p_{\mathbf{w}_k}(\mathbf{x}_{k+1} - \mathbf{f}_k(\mathbf{x}_k))$ and $p(\mathbf{z}_k|\mathbf{x}_k) = p_{\mathbf{v}_k}(\mathbf{z}_k - \mathbf{h}_k(\mathbf{x}_k))$ are the state transition PDF obtained from (1) and the measurement PDF obtained from (2), respectively. The PDF $p(\mathbf{z}_k|\mathbf{z}^{k-1}) = \int p(\mathbf{x}_k|\mathbf{z}^{k-1})p(\mathbf{z}_k|\mathbf{x}_k)\, d\mathbf{x}_k$ is the one-step predictive PDF of the measurement. The recursion (3), (4) starts from $p(\mathbf{x}_0|\mathbf{z}^{-1}) = p(\mathbf{x}_0)$.

2.3 Recursive Relations for Smoothing

Smoothing is a state estimation task of finding more accurate estimates than filtering ones by going through the past filtering estimates and correcting them based on future measurements knowledge. Now, two main approaches to smoothing exist. The first is the standard forward-backward smoothing (Särkkä, 2013), and the second one is the two-filter smoother (Briers et al., 2010).

From the perspective of computational and storage complexity, which are the main topics of this paper, both approaches are equivalent and can solve all three sub-groups of smoothing that are fixed-interval, fixed-lag, and fixed-point smoothing (Anderson and Moore, 1979). Therefore, only the forward-backward smoother, probably the more common approach, is considered further.

The one-step recursive relation for the smoothing PDF $p(\mathbf{x}_l|\mathbf{z}^{k_f})$, where k_f is the step of last obtained measurement and $l = 0, 1, ..., k_f - 1$, is (Särkkä and Solin, 2019; Královec and Šimandl, 2005)

$$\overbrace{p(\mathbf{x}_l|\mathbf{z}^k)}^{\text{smoothing}} = \overbrace{p(\mathbf{x}_l|\mathbf{z}^l)}^{\text{filtering}} \int \frac{\overbrace{p(\mathbf{x}_{l+1}|\mathbf{z}^k)}^{\text{previous smoothing}}}{\underbrace{p(\mathbf{x}_{l+1}|\mathbf{z}^l)}_{\text{predictive}}} \overbrace{p(\mathbf{x}_{l+1}|\mathbf{x}_l)}^{\text{transition}} d\mathbf{x}_{l+1}, \quad (5)$$

where $l < k$. In (5), the PDFs are denoted by their names for easier understanding of the equation and the overall concept. Calculation of the smoothed estimate can be thus seen as a two pass algorithm. Firstly, the filter is run (measurement and time updates performed) forward in time, after that the smoother goes backward in time correcting the previously estimated values. The BRR (3), (4) and the relation (5) form FRRs.

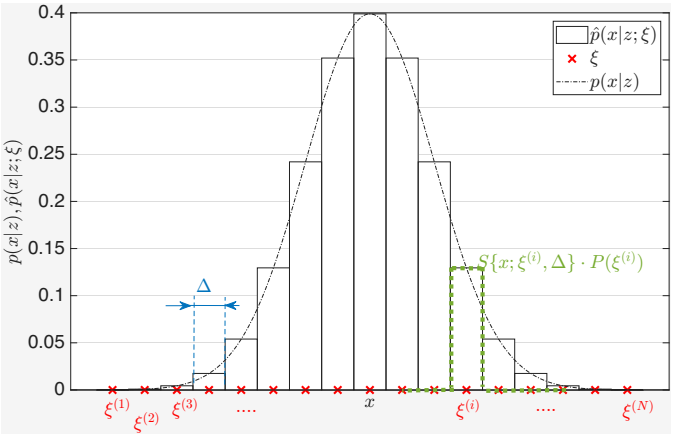


Fig. 1. Points-mass density

3. GRID BASED FILTERS AND PREDICTORS

This section briefly presents the Eulerian and Lagrangian GbFs.

3.1 Point-Mass Density

The GbFs are based on a piecewise constant approximation of the sought PDFs, as can be seen in Figure 1. Mathematically the conditional PMD can be described as

$$p(\mathbf{x}_k|\mathbf{z}^l; \Xi_k) \triangleq \sum_{i=1}^{N} P_{k|l}(\xi_k^{(i)}) S(\mathbf{x}_k; \boldsymbol{\xi}_k^{(i)}), \quad (6)$$

where $\Xi_k = \{\boldsymbol{\xi}_k^{(i)}\}_{i=1}^N, \boldsymbol{\xi}_k^{(i)} \in \mathbb{R}^{n_x}$ is the grid, $N = \prod_{i=1}^{n_x} N_i$ is the number of grid points per dimension which is usually driven be the requested maximal computational complexity, $P_{k|l}(\xi_k^{(i)}) \propto p(\boldsymbol{\xi}_k^{(i)}|\mathbf{z}^l)$ is a normalised value of the PDF [1] $p(\mathbf{x}_k|\mathbf{z}^l)$ defining the height of the PMD at the neighbourhood of grid point $\boldsymbol{\xi}_k^{(i)}$, further called as weight, and $S(\mathbf{x}_k; \boldsymbol{\xi}_k^{(i)})$ is an indicator function for the non-overlapping neighbourhood of the said point with a volume $\delta_k^{(i)}$. For computationally efficient implementation, considered also in this paper, it is beneficial to assume the equidistantly spaced grid with $\delta_k^{(i)} = \delta_k$.

The standard Euler GbFs are implemented using column vectors of weights (vectorized representation), further denoted with double column as $P_{k|l}^{(:)} = [P_{k|l}^{(1)}, ..., P_{k|l}^{(N)}]^T$.

[1] Whenever it does not affect the readability the weights notation is simplified to $P_{k|l}^{(i)} = P_{k|l}(\xi_k^{(i)})$.

However, the efficient Lagrangian-based implementation exploits the underlying physical nature of the state. Therefore, the weights are stored as a tensor $P_{k|l} \in \mathbb{R}^{N_1 \times N_2 \times \ldots \times N_{n_x}}$ and their element indexed as $P_{k|l}^{(i_1,\ldots,i_{n_x})}$.

3.2 Measurement Update

The measurement update is the same for the standard Eulerian GbFs and LGbFs. The PMD weights are updated based on the Bayes' rule (3)

$$P_{k|k}(\boldsymbol{\xi}_k^{(i)}) \propto \underbrace{p(\mathbf{z}_k|\mathbf{x}_k = \boldsymbol{\xi}_k^{(i)})}_{L^{(i)}} P_{k|k-1}(\boldsymbol{\xi}_k^{(i)}). \tag{7}$$

This operation has a computational complexity, w.r.t. number of points, $\mathcal{O}(N)$, and can be implemented by simple point-wise product of likelihood weights $L^{(i)}$ with prior weights $P_{k+1|k}$ as

$$P_{k|k} = L \odot P_{k|k-1}, \tag{8}$$

where $\odot$ denotes the point-wise Hadamard product.

Having numerical solution for the Bayes' rule (3), which is the same for all GbFs, the CKE (4) can be numerically solved using the Eulerian and Lagrangian integration perspective [2]. Both solutions follows.

3.3 Eulerian Time Update

The Eulerian update can be seen as brute force numerical solution to the CKE (4) with the PMD weight evolution described as (Šimandl et al., 2006b)

$$P_{k+1|k}(\boldsymbol{\xi}_{k+1}^{(j)}) = \sum_{i=1}^{N} \underbrace{p(\boldsymbol{\xi}_{k+1}^{(j)}|\mathbf{x}_k = \boldsymbol{\xi}_k^{(i)})}_{\mathbf{T}_k^{(j,i)}} P_{k|k}(\boldsymbol{\xi}_k^{(i)})\delta_k, \forall j, \tag{9}$$

where $\boldsymbol{\xi}_{k+1}^{(j)}$ is the grid points of a newly designed grid $\boldsymbol{\Xi}_{k+1}$ and $\mathbf{T}_k^{(j,i)}$ is the (j,i)-th element of the transition probability matrix (TPM) $\mathbf{T}_k$. The position and size of the grid $\boldsymbol{\Xi}_{k+1}$ can be freely picked by the GbF designer, but it should well cover the part of the state-space where the predictive PMD is expected. The weight time-update can be implemented as a matrix-vector product

$$P_{k+1|k}^{(:)} = \mathbf{T}_k \, P_{k|k}^{(:)}. \tag{10}$$

However, the computational complexity of either (9) and its equivalent (10) is $\mathcal{O}(N^2)$, and its reduction has been the main motivation for the Lagrangian formulation employment (Matoušek et al., 2025).

To sum up, in each GbF recursion step there are two grids: $\boldsymbol{\Xi}_k$, which carries the prior $P_{k|k-1}$ and posterior $P_{k|k}$, and $\boldsymbol{\Xi}_{k+1}$, which carries the next prior $P_{k+1|k}$ and posterior $P_{k+1|k+1}$.

3.4 Lagrangian Time Update

The LGbFs can be used under the assumption that the model dynamics in (1) are invertible. Here for simplicity we will present the algorithms for linear dynamics i.e., further denoted as $\mathbf{f}_k(\mathbf{x}_k) = \mathbf{F}_k\mathbf{x}_k$ and $\mathbf{F}_k^{-1}$ exists. An extension of the LGbF to non-linear invertible dynamics can be seen in (Dunik et al., 2026), and the proposed smoothing approach can be readily applied in that setting as well. The main underlying idea is to split the prediction solution into two sub-solutions. Thus, the sub-solution to the *advection* given by the $\mathbf{x}_{k+1}^{\text{adv}} = \mathbf{F}_k\mathbf{x}_k$, and sub-solution to the *diffusion* driven by the noise term $\mathbf{w}_k$ are solved separately.

The *advection* solution is given by the grid movement. If the new grid is designed such that

$$\boldsymbol{\xi}_{k+1}^{(i)} = \mathbf{F}_k\boldsymbol{\xi}_k^{(i)}, \forall i, \tag{11}$$

and the weights are kept constant, i.e., $P_{k+1|k}^{\text{adv}} = P_{k|k}$. The resulting PMD $p^{\text{adv}}(\mathbf{x}_{k+1}|\mathbf{z}^k; \boldsymbol{\Xi}_{k+1})$ is the advection solution. Because the dynamics is linear, the grid keeps its regular structure.

What is left to be solved is the *diffusion* part corresponding to model

$$\mathbf{x}_{k+1} = \mathbf{x}_{k+1}^{\text{adv}} + \mathbf{w}_k. \tag{12}$$

This model describes a sum of two random variables (RVs), and it is well known that the PDF of a sum of two RVs is equal to the convolution of their PDFs (resp. PMDs). By convolution theorem, the convolution can be efficiently solved by discrete Fourier transform (DFT), and thus the diffusion solution is given as a PMD weight update

$$P_{k+1|k} = \mathcal{F}^{-1}\left(\mathcal{F}(W_k) \odot \mathcal{F}(P_{k+1|k}^{\text{adv}})\right), \tag{13}$$

where $\mathcal{F}(\cdot)$ denotes the DFT, $\mathcal{F}^{-1}(\cdot)$ inverse DFT (IDFT), and tensor $W_k = p(\boldsymbol{\xi}_{k+1}^{\text{mean}}|\boldsymbol{\Xi}_k) \in \mathbb{R}^{N_1 \times N_2 \times \ldots \times N_{n_x}}$ is the transition probability evaluated at $\boldsymbol{\xi}_{k+1}^{\text{mean}}$ being the mean over all points in $\boldsymbol{\Xi}_{k+1}$. If N_i is set odd $\forall i$, then $\boldsymbol{\xi}_{k+1}^{\text{mean}}$ is one of the grid points in $\boldsymbol{\Xi}_{k+1}$, which simplifies the calculations. Compared to the Eulerian formulation, the movement of the grid (11) and the weight update (13) are synchronized and form the Lagrangian grid-based prediction.

Note that the Lagrangian flow (11) of the grid would, of course, inherently lead to a degradation of the grid (infinite growth or shrinking, based on stability of the model), and also to ignorance of incoming measurement information. That is why after the measurement update (and before the time-update), the grid is redesigned and the weights interpolated. Let the estimate of first two filtering PDF moments be $\hat{\mathbf{x}}_{k|k}$, and $\mathbf{C}_{k|k}$, then (due to linearity of the dynamics) it is known, that the predictive first two moments are $\hat{\mathbf{x}}_{k+1|k} = \mathbf{F}_k\hat{\mathbf{x}}_{k|k}$, $\mathbf{C}_{k+1|k} = \mathbf{F}_k\mathbf{C}_{k|k}\mathbf{F}_k^T + \mathbf{Q}_k$. Therefore, taken into account the flow of the grid (11), the filtering PMD is interpolated on a grid $\boldsymbol{\Xi}_k^{\text{interp}}$ forming weights $P_{k|k}^{\text{interp}} = P_{k|k}^{\text{adv}}$. This grid covers an area with center $\hat{\mathbf{x}}_{k|k}$ and size given by σ-ellipse belonging to covariance $\mathbf{C}_{k|k} + \mathbf{F}_k^{-1}\mathbf{Q}_k\mathbf{F}_k^{-T}$. This ensures that after the grid is flown it will correctly cover the predictive PMD [3].

To sum it up, for each LGbF recursion step there are three grids:

- Filtering - $\boldsymbol{\Xi}_k$ - which carries the prior $P_{k|k-1}$ and posterior $P_{k|k}$,
- Filtering interpolated - $\boldsymbol{\Xi}_k^{\text{interp}}$ - which carries the interpolated posterior $P_{k|k}^{\text{interp}}$,

[2] The name is derived from the Lagrangian description of flow in fluid dynamics, where the observer moves with the flow. This contrasts with the Eulerian approach, in which the observer remains fixed in space and observes the flow passing by (Maljaars, 2019).

[3] The interpolated or predictive grid (depending on the design choice) forms a parallelogram; see Matoušek and Duník (2025) for details. Nevertheless, the grid remains structured and does not introduce any issues for the following steps of the algorithm.

- Predictive - $\boldsymbol{\Xi}_{k+1}$ ($\boldsymbol{\xi}_{k+1}^{(:)} = \mathbf{F}_k \boldsymbol{\xi}_k^{(:),\text{interp}}, \forall i$) - which carries the advected $P_{k|k}^{\text{adv}}$ and next prior $P_{k+1|k}$.

The grid evolutions for EGbF and LGbF are illustrated in Figure 2.

4. BACKWARD-FORWARD SMOOTHING UPDATES

In this section, two approaches to grid-based smoothing are presented: one reducing computational complexity at the cost of increased memory by storing quantities during the forward pass, and the other reducing memory requirements at the expense of additional computations. Both approaches are formulated for Eulerian and Lagrangian settings. The Eulerian case follows existing formulations and is included for completeness. In contrast, smoothing for the Lagrangian grid-based filter (LGbF) has not been previously derived, and the Eulerian smoothing formulation cannot be directly applied. Therefore, a novel derivation of the LGbF smoothing step is provided.

4.1 Minimized Computational Complexity

Algorithms in this subsection minimize the computational complexity at the cost of large memory requirements.

Eulerian Smoothing Using the point-mass approximation to (5), the smoothing PMD weights can be computed as (Šimandl et al., 2006a)

$$P_{l|k}(\boldsymbol{\xi}_l^{(j)}) = P_{l|l}(\boldsymbol{\xi}_l^{(j)}) \sum_{i=1}^{N} \frac{P_{l+1|k}(\boldsymbol{\xi}_{l+1}^{(i)})}{P_{l+1|l}(\boldsymbol{\xi}_{l+1}^{(i)})} p(\boldsymbol{\xi}_{l+1}^{(i)} | \mathbf{x}_l = \boldsymbol{\xi}_l^{(j)}), \tag{14}$$

where the smoothing PMD has the same grid support as the filtering PDF, i.e., $\boldsymbol{\Xi}_l$. Similarly to the prediction step, (14) can be implemented using element-wise vector and matrix products as

$$\overbrace{P_{l|k}^{(:)}}^{\text{on } \boldsymbol{\Xi}_l} = \overbrace{P_{l|l}^{(:)}}^{\text{on } \boldsymbol{\Xi}_l} \odot \overbrace{\left(\underbrace{\left(P_{l+1|k}^{(:)} \oslash P_{l+1|l}^{(:)} \right)^T \mathbf{T}_l}_{\text{on } \boldsymbol{\Xi}_{l+1}} \right)^T}^{\text{on } \boldsymbol{\Xi}_l}, \tag{15}$$

where $\oslash$ is a Hadamard division. Note that smoothed $P_{l|k}^{(:)}$ and filtering $P_{l|l}^{(:)}$ weights are on $\boldsymbol{\Xi}_l$, predictive $P_{l+1|l}^{(:)}$, and last smoothed $P_{l+1|k}^{(:)}$ weights are on $\boldsymbol{\Xi}_{l+1}$, and the TPM $\mathbf{T}_l$ (10) transforms the results of Hadamard division from time $l+1$ and grid $\boldsymbol{\Xi}_{l+1}$ to time l, and grid $\boldsymbol{\Xi}_l$, so that the rest of the calculation can be performed. All of the mentioned quantities, except smoothing weights, are known and saved from the forward filter run.

The computational complexity of this algorithm is driven by the product of transition probability matrix $\mathbf{T}_l$ with PMD weights and it is $\mathcal{O}(N^2)$. The storage complexity is maximal when the smoothing is initialized and it is comprising of: all filtering weights - $k_f N$, all prediction weights - $k_f N$, all transition probabilities - $k_f N^2$, last smoothing weights - N, plus some other variables, such as the grid that are rather negligible. The grid can be fully reconstructed just from its center and rotation matrix, i.e., it is not necessary to store all the grid points. An efficient implementation can be found in the GitHub code.

Lagrangian Smoothing The Lagrangian smoother derivation is based on (15) with the difference that the TPM is evaluated in the interpolated grid $\mathbf{T}_l^{(j,i)} = p_{\mathbf{w}_k}(\boldsymbol{\xi}_{l+1}^{(j)} - \mathbf{F}_l \boldsymbol{\xi}_l^{(i),\text{interp}})$, because the LGbF was used in the forward run. This leads to an issue, the $\left(P_{l+1|k}^{(:)} \oslash P_{l+1|l}^{(:)} \right)^T$ in (15) is not a valid operation anymore as in the Lagrangian implementation the predictive $P_{l+1|l}^{(:)}$ weights and the smoothed weights $P_{l+1|k}^{(:)}$ from previous iteration are on different grids. Namely, the predictive weights are on $\boldsymbol{\Xi}_{l+1}$ and last smoothed weights are on $\boldsymbol{\Xi}_{l+1}^{\text{interp}}$. Therefore, last smoothed weights $P_{l+1|k}^{(:)}$ has to be interpolated on $\boldsymbol{\Xi}_{l+1}$ to form $P_{l+1|k}^{\text{fix}}$. To the authors' knowledge, there is no workaround for this issue; however, the interpolation step has negligible computational complexity compared to the time update itself.

Now, in the forward run, the grid was flown according to (11), and it holds that $\boldsymbol{\xi}_{l+1}^{(i)} = \mathbf{F}_l \boldsymbol{\xi}_l^{(i),\text{interp}}, \forall i$, respectively $\boldsymbol{\xi}_l^{(i),\text{interp}} = \mathbf{F}_l^{-1} \boldsymbol{\xi}_{l+1}^{(i)}, \forall i$. This means that the TPM $\mathbf{T}_l$ is Toeplitz. Therefore, the operation $\left(P_{l+1|k}^{(:)} \oslash P_{l+1|l}^{(:)} \right)^T \mathbf{T}_l$ in (15) can be performed efficiently by DFT convolution of weights in tensor form with a kernel given by a middle row of $\mathbf{T}_l$ matrix in the tensor form denoted as W_l, as in the forward run.

Then, the final calculation of smoothing weights in Lagrangian form reads

$$P_{l|k} = P_{l|l} \odot \mathcal{F}^{-1} \left(\mathcal{F} \left(P_{l+1|k}^{\text{fix}} \oslash P_{l+1|l} \right) \odot \mathcal{F}(W_l) \right), \tag{16}$$

where the DFT of TPM $\mathcal{F}(W_l)$ is saved during forward run. Then, the computational complexity is driven by the one DFT of weights and one IDFT, that is $2\mathcal{O}(N \log_2 N)$. The storage complexity is again maximal when the smoothing is initialized and it comprises of these main components: filtering weights - $k_f N$, prediction weights - $k_f N$, transition probabilities - $k_f N$, smoothing weights - N.

4.2 Minimized Memory Requirements

If the aim is to minimize the memory requirements, the number of quantities saved during the forward run can be greatly reduced at the cost of their repeated calculation during the smoothing. Both approaches can be also combined to achieve the desired ratio between storage and computational complexity.

Eulerian Smoothing The TPMs and filtering PMDs are not saved as they can be calculated from the saved prediction PMDs. The computational complexity is now given by $2\mathcal{O}(N^2)$ (the TPMs have to be both calculated and used), however, the storage complexity is reduced to the following main components: prediction weights - $k_f N$, smoothing weights - N.

Lagrangian Smoothing Here the transition probabilities are calculated again, therefore the computational complexity is driven by 2 DFTs and one IDFT, i.e. $3\mathcal{O}(N \log_2 N)$. In the Lagrangian case, however, it would be too complex to reconstruct the filtering PMD from the predictive so it needs to be still saved. The storage complexity is thus comprising from: prediction weights - $k_f N$, filtering weights - $k_f N$, smoothing weights - N.

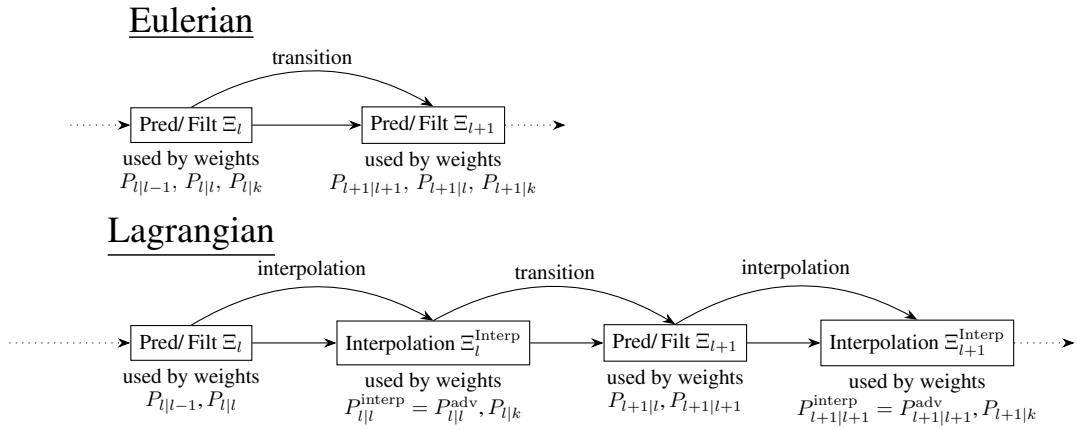


Fig. 2. Illustration of Eulerian and Lagrangian filters grid workflows.

5. SUMMARY AND UNIFICATION OF GRID-BASED ESTIMATORS

The Eulerian and Lagrangian-based filtering, prediction, and smoothing are summarized in a unified algorithmic form, closely reflecting the provided implementation. The standard *Eulerian GbE* with time update and smoothing complexity $\mathcal{O}(N^2)$, can be summarised as:

Algorithm 1: Eulerian grid-based filtering, prediction, and smoothing

(1) *Initialisation*: Set $k = 0$, construct the initial PMD with $\boldsymbol{\Xi}_{0|-1}$ and $P_{0|-1}$.
(2) *Meas. update:* Compute the filtering PMD weights $P^{(:)}_{k+1|k+1} = L^{(:)}_k \odot P^{(:)}_{k+1|k}$ (8).
(3) *Grid construction:* Construct the new grid $\boldsymbol{\Xi}_{k+1}$. Using local filter time update or transformed grid $\mathbf{f}_k(\boldsymbol{\xi}^{(i)}_k), \forall i$.
(4) *Time update:* Compute the predictive PMD weights $P^{(:)}_{k+1|k} = \mathbf{T}_k\, P^{(:)}_{k|k}$ (10).
(5) *Recursion:* If $k \leq k_f$ let $k = k + 1$ and go to step (2), else continue to the next step.
(6) *Smoothing initialisation:* Set $l = k_f$, and set the initial smoothed PMD as the last filtering PMD
(7) *Backward step:* If $l \geq 1$ set $l = l - 1$.
(8) *Smoothing update:* Calculate the smoothed weights as $P^{(:)}_{l|k} = P^{(:)}_{l|l} \odot \left(\left(P^{(:)}_{l+1|k} \oslash P^{(:)}_{l+1|l} \right)^T \mathbf{T}_l \right)$ (15),
(9) *Smoothing recursion:* Go to step (7).

The efficient *Lagrangian GbE* with time update and smoothing complexity $2\mathcal{O}(N \log_2 N)$ can be summarised as:

Algorithm 2: Lagrangian grid-based filtering, prediction, and smoothing

(1) *Initialisation:* Set $k = 0$, construct the initial PMD $\boldsymbol{\Xi}_{0|-1}$, $P_{0|-1}$.
(2) *Meas. update:* : Compute the filtering PMD weights $P_{k|k} = L_k \odot P_{k|k-1}$ (8).
(3) *Interpolated grid design:* Calculate the filtering estimate $\hat{\mathbf{x}}_k$, $\hat{\mathbf{C}}_{k|k}$. Design interpolation grid $\boldsymbol{\Xi}^{\text{interp}}_k$ with mean $\hat{\mathbf{x}}_k$ and size corresponding to $\hat{\mathbf{C}}_{k|k} + \mathbf{F}_k^{-1}\mathbf{Q}_k\mathbf{F}_k^{-T}$.
(4) *Interpolation:* Interpolate the PMD filtering weights $P_{k|k}$ from $\boldsymbol{\Xi}_k$ to $\boldsymbol{\Xi}^{\text{interp}}_k$ forming interpolated weights $P^{\text{interp}}_{k|k}$.
(5) *Time update - advection*: Form the advection solution by flowing the grid $\boldsymbol{\xi}_{k+1} = \mathbf{F}_k\boldsymbol{\xi}^{\text{interp}}_k$ and setting weights $P^{\text{adv}}_{k+1|k} = P^{\text{interp}}_{k|k}$.
(6) *Time update - diffusion:* Compute the predictive PMD weights $P_{k+1|k} = \mathcal{F}^{-1}\left(\mathcal{F}(W_k) \odot \mathcal{F}(P^{\text{adv}}_{k+1|k})\right)$ (13).
(7) *Recursion:* If $k \leq k_f$ let $k = k + 1$ and go to step (2), else continue to the next step.
(8) *Smoothing initialisation:* Set $l = k_f$, and set the initial smoothed PMD as the filtering PMD
(9) *Backward step:* If $l \geq 1$ set $l = l - 1$
(10) *Interpolation:* Interpolate the smoothed weights $P_{l+1|k}$ on $\boldsymbol{\Xi}^{\text{interp}}_{l+1}$ from previous step to $\boldsymbol{\Xi}_{l+1}$ forming weights $P^{\text{fix}}_{l+1|k}$.
(11) *Smoothing update:* Calculate the smoothed weights as $P_{l|k} = P_{l|l} \odot \mathcal{F}^{-1}\left(\mathcal{F}\left(P^{\text{fix}}_{l+1|k} \oslash P_{l+1|l}\right) \odot \mathcal{F}(W_l)\right)$ (16),
(12) *Smoothing recursion:* Go to step (9).

6. IMPLEMENTATION AND SUMMARY

The accuracy and consistency of LGbE and EGbE are qualitatively the same. Numerical comparisons of (Lagrangian) grid-based filters with other estimators have been presented in several earlier works (Matoušek et al., 2025; Matoušek and Duník, 2025; Anonsen and Hallingstad, 2006; Orguner et al., 2010; Matoušek et al., 2026) and are therefore omitted here due to space limitations. Instead, we provide the full implementation of the proposed estimators together with their analytical computational and memory complexities. We also include an illustrative example of empirical computational-complexity results to facilitate interpretation of the analytical expressions.

The simulation code is available at GitHub repository. The main experiment can be executed through *main.m*, where a flag enables or disables smoothing. The system dynamics and measurement equations are defined in *initModel.m*, which supports arbitrary state and measurement dimensions. Users can freely modify the model to suit their needs.

Analytical computational and storage complexities for discussed approaches are summarised in Table 1. An illustrative

Table 1. Summary of practical results and analytical complexities.

Practical computational time [s]

Method	Min. Cmpxl.	Min. Storage
Eulerian Smoother	0.0021	0.62
Lagrangian Smoother	0.0014	0.0018
Eulerian Filter	4.67	
Lagrangian Filter	0.003	

Analytical computational complexity

Method	Min. Cmpxl.	Min. Storage
Eulerian Smoother	$2\mathcal{O}(N^2)$	$\mathcal{O}(N^2)$
Lagrangian Smoother	$2\mathcal{O}(N \log N)$	$3\mathcal{O}(N \log N)$
Eulerian Filter	$2\mathcal{O}(N^2)$	
Lagrangian Filter	$3\mathcal{O}(N \log N)$	

Analytical storage complexity [doubles]

Method	Min. Cmpxl.	Min. Storage
Eulerian Smoother	$2k_f N + k_f N^2 + N$	$k_f N + N$
Lagrangian Smoother	$3k_f N + N$	$2k_f N + N$

RMSE

Method	First dimension	Second dimension
Lagrangian Filter	1.2651	1.0794
Lagrangian Smoother	1.1813	1.0215

example of empirical computational complexity for a two-dimensional model with 201 points per dimension, obtained on a MacBook Air M1, is provided ibidem. For completeness, reports the root-mean-square errors of the Lagrangian filter and its corresponding smoother for the same model and 100 Monte-Carlo simulations are also provided. As expected, the smoother attains lower estimation errors, supporting the correctness of both the derivation and its implementation. Since the smoothing procedures employed here correspond to the standard backward-pass formulations applied to the underlying grid-based filters, their accuracy characteristics (Root mean square error (RMSE), average standard deviation (ASTD), averaged normalised estimation error squared (ANEES)) do not exhibit any qualitative behaviour beyond what is already well established in the literature. Consequently, the specific choice of model is not critical, and additional numerical examples would be redundant.

7. CONCLUSION

A smoothing method for the Lagrangian grid-based filter was proposed, together with a unified formulation of smoothing-algorithm designs for both Eulerian and Lagrangian grid-based estimators. Analytical storage and computational complexities were presented, and an illustrative example demonstrated how these complexities manifest in practice. Implementation of the proposed algorithm is publicly available.

REFERENCES

Anderson, B.D.O. and Moore, J.B. (1979). *Optimal Filtering*. Prentice Hall, New Jersey.

Anonsen, K. and Hallingstad, O. (2006). Terrain aided underwater navigation using point mass and particle filters. In *2006 IEEE/ION Position, Location, And Navigation Symposium*, 1027–1035.

Briers, M., Doucet, A., and Maskell, S. (2010). Smoothing algorithms for state–space models. *Annals of the Institute of Statistical Mathematics*, 62(1), 61–89.

Doucet, A., De Freitas, N., and Gordon, N. (2001). *Sequential Monte Carlo Methods in Practice*, chapter An Introduction to Sequential Monte Carlo Methods. Springer.

Dunik, J., Krejčí, J., Matoušek, J., Brandner, M., and Choe, Y. (2026). Lagrangian grid-based estimation of nonlinear systems with invertible dynamics. *in Proceedings of the 23rd IFAC World Congress (IFAC WC 2026)*. Under review.

Gehlen, J., Ulmke, M., Springer, J., Govaers, F., and Koch, W. (2024). Tensor decomposition based bearing-only target tracking - an analysis based on real data. In *2024 27th International Conference on Information Fusion (FUSION)*.

Královec, J. and Šimandl, M. (2005). Filtering, prediction and smoothing with point-mass approach. In *Automatic Control in Aerospace 2004*, 375–380. Elsevier, Oxford.

Ma, T., Ding, S., Li, Y., and Fan, J. (2023). A review of terrain aided navigation for underwater vehicles. *Ocean Engineering*, 281, 114779.

Maljaars, J. (2019). *When Euler meets Lagrange - Particle-Mesh Modeling of Advection Dominated Flows*. Ph.D. thesis, Technische Universiteit Delft, Nederland.

Matoušek, J., Duník, J., Govaers, F., and Gehlen, J. (2025). Diffusion in lagrangian grid-based predictors. In *2025 25th International Conference on Information Fusion (FUSION)*.

Matoušek, J., Krejčí, J., Dunik, J., and Zanetti, R. (2026). Tensor decompositions for online grid-based terrain-aided navigation. *2026 29th International Conference on Information Fusion (FUSION)*.

Matoušek, J., Duník, J., and Brandner, M. (2023a). Design of efficient point-mass filter with illustration in terrain aided navigation. In *26th International Conference on Information Fusion (FUSION)*. Charleston, USA, 2023.

Matoušek, J., Duník, J., Brandner, M., Park, C., and Choe, Y. (2023b). Efficient point mass predictor for continuous and discrete models with linear dynamics. In *22nd IFAC WC*. Yokohama, Japan, 2023.

Matoušek, J., Duník, J., and Straka, O. (2025). Lagrangian grid-based filters with application to terrain-aided navigation. *IEEE Signal Processing Magazine*.

Matoušek, J. and Duník, J. (2025). Efficient lagrangian grid based filter using spectral differentiation. *IEEE Transactions on Aerospace and Electronic Systems*, 1–12. doi:10.1109/TAES.2025.3636858.

Matoušek, J., Brandner, M., Duník, J., and Punčochář, I. (2024). Tensor train discrete grid-based filters: Breaking the curse of dimensionality. *IFAC-PapersOnLine*, 58(15), 19–24. 20th IFAC Symposium on System Identification SYSID 2024.

Orguner, U., Skoglar, P., Törnqvist, D., and Gustafsson, F. (2010). Combined point-mass and particle filter for target tracking. In *2010 IEEE Aerospace Conference*, 1–10.

Särkkä, S. (2013). *Bayesian Filtering and Smoothing*. Cambridge University Press.

Särkkä, S. and Solin, A. (2019). *Applied Stochastic Differential Equations*. Institute of Mathematical Statistics Textbooks. Cambridge University Press.

Šimandl, M., Královec, J., and Söderström, T. (2006a). Advanced point-mass method for nonlinear state estimation. *Automatica*, 42(7), 1133–1145. doi:https://doi.org/10.1016/j.automatica.2006.03.010.

Šimandl, M., Královec, J., and Söderström, T. (2006b). Advanced point-mass method for nonlinear state estimation. *Automatica*, 42(7), 1133–1145.

Ånonsen, K.B. and Hagen, O.K. (2010). An analysis of real-time terrain aided navigation results from a HUGIN AUV. In *2010 Marine Tech. Society and the Oceanic Engineering Society of the IEEE Conference*, 1–9. Seattle, WA, USA.