Wiley
International Journal of Distributed Sensor Networks
Volume 2026, Article ID 6527524, 19 pages
https://doi.org/10.1155/dsn/6527524

Research Article

Speeding Up Sequential Markov Chain Monte Carlo Methods in the Context of Large Volumes of Data From Distributed Sensor Networks

Allan De Freitas,[1] **François Septier**,[2] **and Lyudmila Mihaylova**[3]

[1]*Department of Electrical, Electronic & Computer Engineering, University of Pretoria, Pretoria, South Africa*
[2]*Université Bretagne Sud, UMR CNRS 6205, LMBA, Vannes, France*
[3]*School of Electrical and Electronic Engineering, The University of Sheffield, Sheffield, UK*

Correspondence should be addressed to Allan De Freitas; allan.defreitas@up.ac.za

Received 3 December 2024; Revised 9 October 2025; Accepted 8 December 2025

Academic Editor: Marco Scarpa

Advances in digital sensors, digital data storage, and communications have resulted in systems being capable of accumulating large collections of data. In light of dealing with the challenges that large volumes of data present, this work proposes solutions to inference and filtering problems within the Bayesian framework. Two novel sequential Markov chain Monte Carlo (SMCMC) frameworks are proposed for nonlinear and non-Gaussian state space models, able to deal with large volumes of data (or observations). These are SMCMC frameworks relying on two key ideas: (1) a divide-and-conquer type approach computing local filtering distributions, each using a subset of the data, and (2) subsampling the large data and utilizing a smaller subset for filtering and inference. Simulation results highlight the large computational savings that can reach 90% by the proposed algorithms when compared with a state-of-the-art SMCMC approach.

Keywords: adaptive subsampling; big data; distributed sensor network; parallel processing; sequential Markov chain Monte Carlo

1. Introduction

Advances in technology have led to an explosion of large volumes of data in many scientific and engineering domains [1–3]. In general, increases in data stream complexity result in increased computational complexity and algorithmic instability [4]. Bayesian methods have been a popular choice for data processing in many applications with low volumes of data for several decades. They have several advantages over competing approaches which include the ability to incorporate prior knowledge, a principled approach to quantifying uncertainty, model flexibility, and reduction in overfitting [5]. Even so, there has been a slow adoption of Bayesian methods in the era of large volumes of data due to a general lack of scalable solutions [5]. However, recent developments in the efficiency of Bayesian methods for various processing tasks have demonstrated that viable Bayesian solutions exist, such as the development of scalable Gaussian processes for regression [6].

In this paper, the processing task of interest is the estimation of the current signal state using only observations available up to the present time from a sequence of large data volumes. The need for signal estimation occurs in a wide variety of fields, for example, object tracking in sensor networks [7], econometrics [8], and biomedical image processing [9]. In a Bayesian framework, this involves the sequential inference of the filtering distribution associated with a state space model. The solution is referred to as the Kalman filter [10] when the state space model is linear and Gaussian. However, there is typically no analytically tractable solution when the state space model is nonlinear and/or non-Gaussian. Several algorithms which achieve sequential inference in such systems through approximations have been proposed. One such class of techniques is referred to as

sequential Monte Carlo (SMC) methods [11], or particle filters (PFs), which involve a weighted discrete approximation of the filtering distribution and utilize importance sampling. PFs have been successfully applied to many areas.

1.1. Related Works. In recent years, there has been an increased focus in the literature to develop novel theories and methods to address large-scale problems. In the context of sequential Bayesian inference, large-scale problems can be described by a high-dimensional state space and a large number of observations. In high-dimensional problems, the PF is prone to weight degeneracy [12, 13]. In [14], a Markov chain Monte Carlo (MCMC) kernel was applied after the resampling stage of the PF to reduce degeneracy. Several approaches have focused on exploiting local spatial dependencies in high-dimensional state space models by dividing the state space into a collection of lower dimensional subspaces within the context of particle filtering. These include the block PF [15, 16], the space–time PF [17], nested SMC [18], and divide-and-conquer SMC [19]. A related but promising alternative to the PF framework is the sequential MCMC (SMCMC) method [20–23], which has been successfully applied in several challenging areas [24, 25]. In contrast to importance sampling, used within a PF framework, the SMCMC method utilizes MCMC sampling, which does not rely on a weighting mechanism. Both importance sampling and MCMC sampling are susceptible to the curse of dimensionality. However, the latter has been shown to empirically perform better in high-dimensional systems [26], and the authors in [27] established conditions under which SMCMC can outperform standard SMC in terms of asymptotic variance of the corresponding Monte Carlo (MC) estimators through statistical analysis. In addition, the authors in [28] discuss different MCMC kernels for high-dimensional SMCMC. This work was extended in [29] where a composite Metropolis–Hastings (MH) kernel using invertible particle flow within the SMCMC framework to improve the acceptance rate for high-dimensional problems is presented.

The focus of this paper is not on addressing the curse of dimensionality, but rather on the challenges introduced by a large number of observations. In this case, sampling methods can lead to computational complexity which is problematic in time-sensitive filtering applications. Research on computationally efficient implementations of SMC methods has focused on making the structure of the PF parallel [30], particularly the resampling step [31], which can then be used in distributed processing applications [32]. However, this typically includes approximations to achieve a solution and still requires processing of all the data which can be particularly problematic when the computational complexity of evaluating a single observation is large.

Within the MCMC framework, there have been many different approaches proposed for dealing with large datasets for *static* systems. Two fundamentally different strategies exist: the development of dataset compression techniques applied prior to Bayesian computation with classical approaches and the development of scalable methods for Bayesian computation without dataset compression. A recent example of the former is weighted subsets of data points, referred to as coresets, which can be used for posterior inference [33]. However, there are many limitations and open challenges related to coreset construction [34], and it is common practice to apply redundant information removal with classical signal preprocessing techniques in time-sensitive filtering applications. The focus of the paper is thus on the latter strategy of scalable Bayesian computation without dataset compression.

The proposed methods can be categorized as either parallel or iterative strategies. In terms of parallel strategies, approaches can be further categorized as one-shot or multishot learning. In the former case, only one round of communication is required. Given the focus on time-sensitive filtering applications, only one-shot learning approaches are further explored (refer to [34] for a review covering multishot learning approaches). Techniques based on single-shot learning, also referred to as divide and conquer,[1] focus on subdividing the measurements and running separate MCMC samplers in parallel on each subdivided set of measurements. The samples from the separate MCMC samplers, referred to as local samples, are then combined to obtain samples from the complete posterior distribution, referred to as global samples. The divide-and-conquer techniques differ in how the local samples are combined to obtain the global samples. In [37], global samples are obtained as a weighted average of the local samples. This approach is only theoretically valid under a Gaussian assumption. In [38], the local posterior from the separate MCMC samplers is approximated as a Gaussian or with a Gaussian kernel density estimation. Global samples can then be obtained through the product of the local densities. This work was further extended for time series analysis in [39]. This idea is also further developed in [40] by representing the discrete kernel density estimation as a continuous Weierstrass transform. In [41], the combination is based on the geometric median of the local posteriors which are approximated with Weiszfeld's algorithm by embedding the local posteriors in a reproducing kernel Hilbert space. Divide-and-conquer techniques typically struggle in applications where the local posteriors substantially differ, and if they do not satisfy Gaussian assumptions. In [42, 43], a divide-and-conquer strategy is proposed which attempts to overcome the challenge of different local posteriors and relaxes the Gaussian assumption to a more general assumption of a posterior distribution from the exponential family. The approach is based on the expectation propagation (EP) algorithm. In this approach, the separate MCMC samplers exchange sufficient statistics, resulting in each individual MCMC sampler converging to the global posterior. Recently, more complex models have been proposed to address specific issues such as mode collapse, model mismatch, and underrepresented tails when combining results from different local posteriors. Notable examples include [44], which presents a parallel active inference method incorporating Gaussian process surrogate modeling and active learning; [35], which introduces a method based on applying affine transformations to the local posterior samples; and [36], which uses diffusion generative modeling to fit density approximations to the local posteriors.

Iterative strategies rely on computing an approximation based on a subset or minibatch of the entire dataset. Methods which adopt this approach can be categorized as exact or inexact, depending on whether the target distribution remains the true posterior distribution. Exact methods typically impose strict constraints on the posterior distribution or have limited computational speedups due to various reasons. They can be further decomposed into approaches which rely on an unbiased estimation of the likelihood using unbiased estimates of the log likelihood [45, 48] and approaches based on a factorized version of the MH acceptance probability. Recent examples of the latter include approaches that focus on delayed acceptance schemes [46, 47] and methods which assume a strictly positive lower bound on the log likelihood that accumulates tractably when summed over many observations [48–50], additionally utilizing control variates [51, 52]. Inexact methods approximate the MH acceptance ratio to within an error tolerance, implicitly trading off exactness for increased computational gain. Inexact approaches with random sampling of the minibatches include approaches based on confidence intervals [53–55] and an approach based on the noise-tolerant Barker acceptance test [56]. An inexact approach with sampling informed by measured summary statistics and a fixed size of observations in a minibatch is proposed in [57]. SMC samplers [58], population MCMC approaches [59], adaptive importance sampling [60, 61], and MCMC with tensor factorization [62] are other approaches that have been proven efficient.

1.2. Contributions. The key contributions of this paper consist of the development and comparison of two different frameworks for efficient Bayesian filtering with large volumes of data: (i) A divide-and-conquer SMCMC framework is proposed for dynamic systems based on the processing of batches of data in parallel. This is demonstrated with an implementation via the EP algorithm and gives a computationally efficient parallelized solution. (ii) A subsampling SMCMC framework is proposed for dynamic systems. This is demonstrated with an implementation based on adaptive subsampling. Initial ideas for the implemented approach and results were introduced in [63], while in this paper, the framework is generalized, and the likelihood functions are derived for applications with measurement origin uncertainty, that is, by solving the data association problem. (iii) Extensive evaluation and validation of the developed approaches is performed and compared with the state-of-the-art SMCMC approach over two testing examples, including tracking in a complex wireless sensor network (WSN) with clutter.

This paper presents unpublished methods and results that are part of the first author's dissertation [64]. The structure of the frameworks proposed in this paper is presented in Figure 1. While the generic SMCMC framework utilizes all available data (as shown in Figure 1a), the two proposed SMCMC frameworks employ a fraction of the data. The subsampling SMCMC framework (as shown in Figure 1b) is demonstrated with an approach which performs uniform subsampling by a pseudolikelihood term derived in this paper. The proposed divide-and-conquer SMCMC framework (as shown in Figure 1c) has a different mechanism to select the data by computing local filtering distributions.

The rest of the paper is organized as follows. Section 2 gives the problem formulation. Section 3 presents the divide-and-conquer SMCMC framework, and Section 4 presents the subsampling SMCMC framework. Section 5 yields the performance validation results. Conclusions are summarized in Section 6. The expressions of the likelihood calculation needed for Section 5 are given in the Appendix.

2. Problem Formulation

Consider $x_k \in \mathbb{R}^{n_x}$, the latent state vector of interest at time t_k with $k = 1, \cdots, T \in \mathbb{N}$. The evolution of the latent state vector is given by a Markov process. Thus, the joint distribution of the latent state vector is

$$p(x_{1:T}) = \mu(x_0) \prod_{k=1}^{T} f_k(x_k|x_{k-1}), \tag{1}$$

where $\mu(x_0)$ is an initial probability density function (pdf) and $f_k(x_k|x_{k-1})$ is referred to as the state transition pdf. The data received up till time t_k is represented by $z_{1:k} = \{z_1, \cdots, z_k\}$. The data received at each time t_k are represented by a set $z_k = \{z_k^1, \cdots, z_k^{M_k}\}$, where M_k is the total amount of data and $z_k^i \in \mathbb{R}^{n_z}$. Each z_k^i is a partial and noisy function of x_k. The data is considered conditionally independent, resulting in

$$p(z_{1:T}|x_{1:T}) = \prod_{k=1}^{T} g_k(z_k|x_k), \tag{2}$$

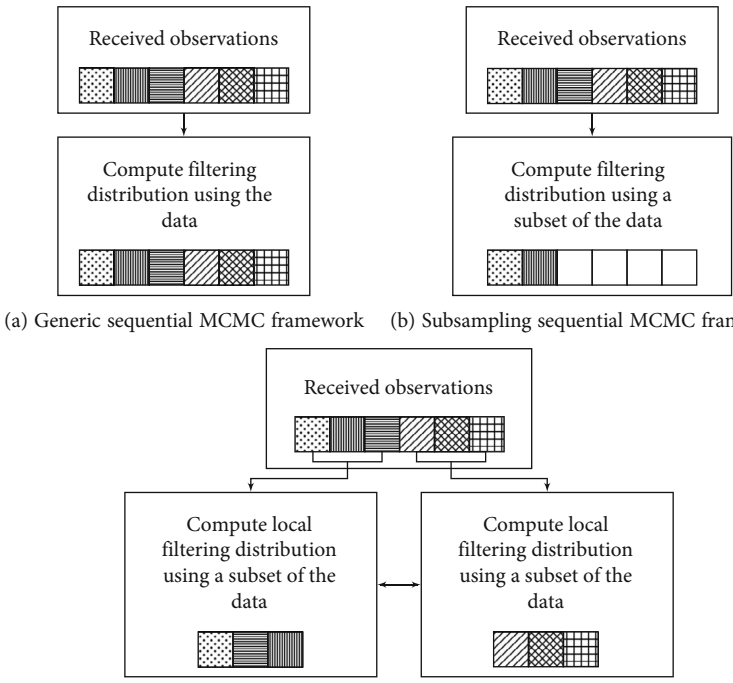
where $g_k(z_k|x_k)$ is referred to as the likelihood pdf. The inference problem is described as the estimation of the latent state vector given the data. In a Bayesian framework, this is encapsulated by the posterior distribution:

$$p(x_{1:k}|z_{1:k}) \propto g_k(z_k|x_k) f_k(x_k|x_{k-1}) p(x_{1:k-1}|z_{1:k-1}). \tag{3}$$

Further, in this paper, the focus is on applications where online estimates of the latent state vector are required. This translates into the problem of sequentially determining a marginal of the posterior distribution referred to as the filtering distribution, $p(x_k|z_{1:k})$. The filtering distribution infers the latent state vector at the current time step for the data observed thus far:

$$p(x_k|z_{1:k}) \propto g_k(z_k|x_k) \int f_k(x_k|x_{k-1}) p(x_{k-1}|z_{1:k-1}) dx_{k-1}. \tag{4}$$

The filtering distribution represents the online updated uncertainty about the latent state vector after assimilating the most recent data. From this distribution, it is possible to extract point estimates and any other posterior expectations needed for decision-making.



(a) Generic sequential MCMC framework (b) Subsampling sequential MCMC framework

(c) Divide-and-conquer sequential MCMC framework with two computing nodes

Figure 1: How data are processed in the proposed SMCMC frameworks compared with the generic SMCMC framework at each discrete time step.

2.1. SMCMC. An analytical solution to Equation (4) is typically intractable when the state space model is characterized by nonlinearities and/or non-Gaussian noise. To overcome this issue, sampling-based approaches approximate the filtering distribution by a discrete set of samples. In the case of SMCMC, sampling from the filtering distribution directly is more computationally expensive than sampling from the posterior distribution [28]. To obtain samples from the posterior distribution, SMCMC methods replace the posterior distribution at the previous time t_{k-1} by an empirical approximation based on previous iterations of the algorithm in the previous recursion. Therefore, the posterior distribution is given by

$$\tilde{p}(x_{1:k}|z_{1:k}) \propto g_k(z_k|x_k) f_k(x_k|x_{k-1}) \hat{p}(x_{1:k-1}|z_{1:k-1}), \quad (5)$$

where

$$\hat{p}(x_{1:k-1}|z_{1:k-1}) \approx \frac{1}{N} \sum_{j=N_b+1}^{N_b+N} \delta_{X^j_{k-1,1:k-1}}(dx_{1:k-1}), \quad (6)$$

where $\{X^j_{k-1,1:k-1}\}_{j=N_b+1}^{N_b+N}$ denotes the N post–burn-in states of a Markov chain with kernel $\mathcal{K}_{k-1}$ and invariant distribution, $p(x_{1:k-1}|z_{1:k-1})$ (each sample lies in the path space, $\mathbb{R}^{(k-1)n_x}$), N_b is the number of discarded burn-in samples, j is the sample index, and δ_x is the Dirac measure. Finally, an approximation for the filtering distribution is represented by the last N samples at time t_k:

$$\tilde{p}(x_k|z_{1:k}) \approx \frac{1}{N} \sum_{j=N_b+1}^{N_b+N} \delta_{X^j_{k,k}}(dx_k). \quad (7)$$

This procedure is summarized by Algorithm 1.

It is important to note that the method as described is forward-only since the focus is on determining the filtering distribution. Fixed-lag smoothing could be layered on top by augmenting the kernel if desired. Also, because the latent process is Markovian, the filtering distribution at time t_k depends only on the most recent state. As a result, only the samples from the previous step, $\{X^j_{k-1,k-1}\}_{j=N_b+1}^{N_b+N}$, need to be stored, rather than complete path histories.

3. Divide-and-Conquer SMCMC

A natural mechanism to tackle the challenge of large amounts of data in an MCMC framework is to divide the data into batches for processing and then to combine the results [37–39, 43]. This approach is reliant on the availability of a processing platform which supports parallel processing of batches of data. The optimal strategy for separating data into subsets will also be dependent on the processing platform's processing capabilities. In the case of a sequential data stream, the separation of data into subsets, parallel processing, and combination of the results must be completed within each time step. This emphasizes the additional requirement for an efficient communication link between the computing nodes within the processing platform and to favor approaches which transmit minimal information

1: **if** $k = 1$ **then**
2: **for** $j = 1,...,N + N_b$ **do**
3: Sample $X_{1,1}^{j} \sim \mathscr{K}_1(X_{1,1}^{j-1}, \cdot)$ with $\mathscr{K}_1$ an MCMC kernel of invariant distribution $p(x_1|z_1) \propto \mu(x_0)g_1(z_1|x_1)$
4: **end for**
5: **else if** $k \geq 2$ **then**
6: Sample $X_{k,1:k}^{j} \sim \mathscr{K}_n(X_{k,1:k}^{j-1}, \cdot)$ with $\mathscr{K}_k$ an MCMC kernel of invariant distribution Equation (5)
7: **end if**
8: Approximation of the smoothing distribution with the following empirical measure,

$$p(x_{1:k}|z_{1:k}) \approx \frac{1}{N} \sum_{j=N_b+1}^{N_b+N} \delta_{X_{k,1:k}^{j}}(dx_{1:k})$$

9: Approximation of the filtering distribution according to Equation (7)

Algorithm 1:Generic sequential Markov chain Monte Carlo [28].

between computing nodes. How accurately the derived global filtering distribution approximates the true global filtering distribution is dependent on the local filtering distributions and the mechanism used to combine them. The divide-and-conquer SMCMC framework is summarized in Algorithm 2. The overhead associated with processing the data into batches and distributing the batches to computing nodes is considered outside the scope of this paper. It is worth noting that in certain applications, this processing overhead is naturally nonexistent. For example, in the case of a distributed sensor network, subsets of measurements occur naturally as each sensor in the network collects data.

3.1. Expectation Propagation Sequential Markov Chain Monte Carlo (EP-SMCMC). To illustrate the divide-and-conquer SMCMC framework, consider the EP-SMCMC approach. In this case, utilization of concepts from EP [43], which is a variational message passing scheme [65], is used for the combination of the results from the computing nodes. This approach is particularly well-suited for SMCMC with large volumes of data since the intercomputing node overhead is restricted to a small number of fixed parameters, and the EP framework allows for the incorporation of inference from all the other computing nodes as a prior in the inference step for any given computing node.

Consider the case where the set of measurements, z_k, can be divided into D disjoint subsets, $z_{k,1}, \cdots, z_{k,D}$ of size $M_{\mathrm{k,1}}$, $\cdots, M_{\mathrm{k,D}}$ (each $z_{\mathrm{k,d}}$ contains $M_{\mathrm{k,d}}$ vectors in $\mathbb{R}^{n_z}$), with the assumption that the subsets are conditionally independent, such that $z_k = \bigcup_{d=1}^{D} z_{k,d}$ and $z_{k,i} \bigcap z_{k,j} = \varnothing : i \neq j$. The posterior distribution in Equation (5) is further factored:

$$\tilde{p}(x_{1:k}|z_{1:k}) \propto f_k(x_k|x_{k-1})\widehat{p}(x_{1:k-1}|z_{1:k-1}) \prod_{d=1}^{D} g_{k,d}(z_{k,d}|x_k). \tag{8}$$

The D subsets of measurements are processed in parallel on D computing nodes. Sampling from the global filtering distribution is achieved by first approximating the likelihood of the $D-1$ sets of measurements from the other computing nodes with a distribution from the exponential family:

$$\pi(x_k|\eta) = h(x_k)g(\eta) \exp\left\{\eta^T u(x_k)\right\}, \tag{9}$$

where $u(x_k)$ are fixed sufficient statistics of the chosen family, $h(x_k)$ is a base measure, and $g(\eta)$ is the normalizer. The vector η collects the natural parameters (NPs) for that family which uniquely specifies the density $\pi(\bullet|\eta)$. In the proposed approach, the NPs, which are independent of the number of measurements in the subset, are cheaply distributed between computing nodes. Thus, the local posterior distribution for an individual computing node is given by

$$\tilde{p}_d(x_{1:k}|z_{1:k}) \propto g_{k,d}(z_{k,d}|x_k)f_k(x_k|x_{k-1})\widehat{p}(x_{1:k-1}|z_{1:k-1}) \prod_{i \neq d} \pi(x_k|\eta_i). \tag{10}$$

Each local posterior distribution, Equation (10), is thus an approximation of the global posterior distribution in Equation (8) in which each true likelihood $g_{k,i}(\bullet|x_k)$ is replaced by its exponential family proxy $\pi(\bullet|\eta_i)$.

The algorithm proceeds iteratively, beginning with the application of an appropriate MCMC kernel with invariant distribution Equation (10) on each computing node. The NPs of each computing node, η_d, are then determined. This is achieved by firstly considering the local filtering distribution:

$$\tilde{p}_d(x_k|z_{1:k}) \propto g_{k,d}(z_{k,d}|x_k)\widehat{p}(x_k|z_{1:k-1}) \prod_{i \neq d} \pi(x_k|\eta_i). \tag{11}$$

A discrete approximation for the local filtering distribution can be cheaply obtained from the MCMC samples drawn from the local posterior distribution as in Equation (7). Further, by replacing the likelihood expression with the approximate likelihood term, we obtain

$$p_d(x_k|z_{1:k}) \propto \pi(x_k|\eta_d)\widehat{p}(x_k|z_{1:k-1}) \prod_{i \neq d} \pi(x_k|\eta_i). \tag{12}$$

The idea is to select the NPs, η_d, in a way that results in the minimization of $\mathrm{KL}(\tilde{p}_d(x_k|z_{1:k})\|p_d(x_k|z_{1:k}))$, where $\mathrm{KL}(\cdot)$ refers to the Kullback–Leibler divergence. It has been shown [66] that the minimization occurs when

1: **for** k = 1,...,T **do**
2: [IF REQUIRED] Separate data into batches and distribute to computing nodes
3: **for** d = 1,...,D **do** {Computing node index (complete in parallel)}
4: Generate discrete approximation of the filtering distribution local to each computing node according to Algorithm 1.
5: Extract compressed representative information of the local filtering distribution approximation.
6: Share the information with the other computing nodes, i.e. $D \setminus d$ computing nodes.
7: **end for**
8: Combine the information from all the computing nodes to obtain an approximate global filtering distribution.
9: **end for**

Algorithm 2: Divide-and-conquer sequential Markov chain Monte Carlo.

$$\mathbb{E}_{\tilde{p}_d(\mathrm{x}_k|\mathrm{z}_{1:k})}[u(x_k)] = \mathbb{E}_{p_d(\mathrm{x}_k|\mathrm{z}_{1:k})}[u(x_k)], \tag{3.6}$$

where $\mathbb{E}[\cdot]$ represents the mathematical expectation, which corresponds to matching the expected sufficient statistics. Approximating the discrete distributions with the same exponential family as the likelihood term approximation, that is, $\pi(x_k|\eta_{f,d}) \approx p_d(x_k|z_{1:k})$ and $\pi(x_k|\eta_{p,d}) \approx p(x_k|z_{1:k-1})$, results in the NPs being determined by

$$\eta_d = \eta_{f,d} - \left(\eta_{p,d} + \sum_{i \neq d} \eta_i\right), \tag{13}$$

where $\eta_{p,d}$ and $\eta_{f,d}$ are the NPs of the approximated predictive and filtering distribution, respectively. Finally, the NPs are distributed to all $D \setminus d$ computing nodes, followed by the next iteration.

The number of iterations is dependent on the rate of convergence of η_d and is treated as a fixed parameter. The EP-SMCMC method is summarized as Algorithm 3.

4. Subsampling SMCMC

An alternative to divide-and-conquer SMCMC is to only evaluate a subset of the data. In this approach, the requirements on the processing platform, including the ability to support dedicated parallel processing of batches of data and efficient communication between computing nodes, are no longer essential. However, it is assumed that the processor has access to the entire dataset. In a sequential setting, the challenge is in selecting a subsample of the data at each time step which, when processed by an MCMC kernel, is able to generate samples which accurately approximate the filtering distribution, Equation (7). This is a twofold problem since the subsampling mechanism is required to determine both which data to process, as well as how much of the data to process. In the case of a sequential data stream, the computational complexity required to generate the subsample of data to be evaluated is required to be minimal to maximize the time available for inference. The subsampling SMCMC framework is represented algorithmically by Algorithm 1, with the exception that the MCMC kernel only processes a subset of the data. It is important to note that the choice of MCMC kernel will impact how accurately the filtering distribution approximates the true filtering distribution.

4.1. Adaptive Subsampling Sequential Markov Chain Monte Carlo (AS-SMCMC). To demonstrate the subsampling SMCMC framework, consider the AS-SMCMC approach. In this case, the confidence MH kernel proposed in [55] is considered. The confidence sampler leverages confidence bounds based on concentration inequalities to automatically select the subsample.

Unlike in the generic SMCMC and EP-SMCMC approaches, AS-SMCMC is restricted to applications with MH-based MCMC kernels, although this is not true of the general subsampling SMCMC framework. This limitation is based on the inherent link between the adaptive subsampling mechanism and the MH kernel. However, MH-based kernels are a popular choice in SMCMC applications [22, 23, 67], making it a viable option in many scenarios. In general, the classic MH kernel operates by generating samples from the posterior distribution by generating samples from a known proposal distribution $X^*_{k,1:k} \sim q(x_{1:k}|X^{j-1}_{n,1:k})$. The proposed sample is accepted as the current state of the chain, $X^j_{k,1:k}$, with probability

$$u < \frac{p\left(X^*_{k,1:k}|z_{1:k}\right) q\left(X^{j-1}_{k,1:k}\middle|X^*_{k,1:k}\right)}{p\left(X^{j-1}_{k,1:k}|z_{1:k}\right) q\left(X^*_{k,1:k}\middle|X^{j-1}_{k,1:k}\right)}, \tag{14}$$

where u represents a sample from a uniform distribution $u \sim \mathrm{U}_{[0,1]}$. This expression can be further developed by expanding with Bayes' rule:

$$u < \frac{f_k\left(X^*_{k,k}\middle|X^*_{k,k-1}\right) q\left(X^{j-1}_{k,1:k}\middle|X^*_{k,1:k}\right)}{f_k\left(X^{j-1}_{k,k}\middle|X^{j-1}_{k,k-1}\right) q\left(X^*_{k,1:k}\middle|X^{j-1}_{k,1:k}\right)} \prod_{i=1}^{M_k} \frac{g_k\left(z^i_k\middle|X^*_{k,k}\right)}{g_k\left(z^i_k\middle|X^{j-1}_{k,k}\right)}. \tag{15}$$

The previous state of the chain is stored as the current state, $X^j_{k,1:k} = X^{j-1}_{k,1:k}$, when the proposed sample does not meet this criterion. Further manipulation of this expression leads to a form with the likelihoods isolated:

$$\psi\left(X^{j-1}_{k,1:k}, X^*_{k,1:k}\right) < \Lambda^{M_k}\left(X^{j-1}_{k,1:k}, X^*_{k,1:k}\right), \tag{16}$$

where the term
$\psi(X^{j-1}_{k,1:k}, X^*_{k,1:k}) = (1/M_k) \log [u(f_k(X^{j-1}_{k,k}|X^{j-1}_{k,k-1}) q(X^*_{k,1:k}|X^{j-1}_{k,1:k}$

1: **for** $k = 1,...,T$ **do**
2: **for** $d = 1,...,D$ **do** {Computing node index (completed in parallel)}
3: **for** $L = 1,...,L$ **do** {EP iteration index}
4: Generate discrete approximation of the local posterior distribution using the same procedure as Algorithm 1 where Equation (10) is the invariant target distribution.
5: Determine the NPs of the approximated likelihood term, η_d, according to Equation (13).
6: Distribute the NPs of the approximated likelihood term to the set $D \setminus d$ computing nodes.
7: **end for**
8: **end for**
9: Combine the information from all the computing nodes to obtain an approximate global filtering distribution.
10: **end for**

Algorithm 3: Expectation propagation sequential Markov chain Monte Carlo.

$)/f_k(X^*_{k,k}|X^*_{k,k-1})q(X^{j-1}_{k,1:k}|X^*_{k,1:k}))]$ and the term $\Lambda^{M_k}(X^{j-1}_{k,1:k}, X^*_{k,1:k}) = (1/M_k)\sum_{i=1}^{M_k} \log [g_k(z^i_k|X^*_{k,k})/g_k(z^i_k|X^{j-1}_{k,k})]$. When the number of measurements is very large, the log likelihood ratio becomes the most computationally expensive part of the generic SMCMC algorithm. To reduce the computational complexity, an MC approximation for the log likelihood ratio has been proposed [55]:

$$\Lambda^{S_j}\left(X^{j-1}_{k,1:k}, X^*_{k,1:k}\right) = \frac{1}{S_j}\sum_{i=1}^{S_j} \log\left[\frac{g_k\left(z^{i,*}_k|X^*_{k,k}\right)}{g_k\left(z^{i,*}_k\middle|X^{j-1}_{k,k}\right)}\right], \quad (17)$$

where the set $z^*_k = \{z^{1,*}_k, \cdots, z^{S_j,*}_k\}$ is drawn uniformly without replacement from the original set of M_k measurements.

The difficulty which arises is in selecting a minimum value for S_j that results in a set of subsampled measurements that contain enough information to make the correct decision in the MH step. To overcome this difficulty in standard MCMC for static inference, the authors in [54] proposed to use concentration inequalities which provide a probabilistic bound on how functions of independent random variables deviate from their expectation. In this case, the independent random variables are the log likelihood ratio terms. Thus, it is possible to obtain a bound on the deviation of the MC approximation in Equation (17) from the complete log likelihood ratio:

$$\left(\left|\Lambda^{S_j}\left(X^{j-1}_{k,1:k}, X^*_{k,1:k}\right) - \Lambda^{M_k}\left(X^{j-1}_{k,1:k}, X^*_{k,1:k}\right)\right| \le c_{S_j}\right) \ge 1 - \delta_{S_j}, \quad (18)$$

where $\delta_{S_j} > 0$ and c_{S_j} is dependent on which inequality is used. There are several inequalities which could be used; in this paper, the empirical Bernstein inequality [55] is used, which results in

$$c_{S_j} = \sqrt{\frac{2V_{S_j}\log\left(3/\delta_{S_j}\right)}{S_j}} + \frac{3R\log\left(3/\delta_{S_j}\right)}{S_j}, \quad (19)$$

where V_{S_j} represents the sample variance of the log likelihood ratio and R is the range given by

$$R = \max_{1\le i\le M_k}\left\{\log\left[\frac{g_k\left(\mathbf{z}^i_k|X^*_{k,k}\right)}{g_k\left(\mathbf{z}^i_k\middle|X^{j-1}_{k,k}\right)}\right]\right\} - \min_{1\le i\le M_k}\left\{\log\left[\frac{g_k\left(\mathbf{z}^i_k|X^*_{k,k}\right)}{g_k\left(\mathbf{z}^i_k\middle|X^{j-1}_{k,k}\right)}\right]\right\}. \quad (20)$$

It is required to relate the expression in Equation (16) in terms of the MC approximation of Equation (17). Since the MC approximation is bounded, it can be stated that it is not possible to make a decision when the value of $\psi(X^{j-1}_{k,1:k}, X^*_{k,1:k})$ falls within the region specified by the bound. Thus, it is required that $|\Lambda^{S_j}(X^{j-1}_{k,1:k}, X^*_{k,1:k}) - \psi(X^{j-1}_{k,1:k}, X^*_{k,1:k})| > c_{S_j}$, where $|\cdot|$ represents the absolute value, in order to be able to make a decision, with probability at least $1 - \delta_{S_j}$.

This forms the underlying principle for the creation of a stopping rule [54, 68]. Let $\delta_s \in (0, 1)$ be a user-specified input parameter. The idea is to sequentially increase the size of S_j while at the same time checking if the stopping criterion, $|\Lambda^{S_j}(X^{j-1}_{k,1:k}, X^*_{k,1:k}) - \psi(X^{j-1}_{k,1:k}, X^*_{k,1:k})| > c_{S_j}$, is met. If the stopping criterion is never met, then this will result in $S_j = M_k$, that is, requiring the evaluation of all the measurements at the time step t_k. Selecting $\delta_{S_j} = (p_s - 1/p_s S_j^{p_s})\delta_s$ results in $\sum_{S_j\ge 1}\delta_{S_j} \le \delta_s$. The event

$$\mathscr{E} = \bigcap_{S_j\ge 1}\left\{\left|\Lambda^{S_j}\left(X^{j-1}_{k,1:k}, X^*_{k,1:k}\right) - \Lambda^{M_k}\left(X^{j-1}_{k,1:k}, X^*_{k,1:k}\right)\right| \le c_{S_j}\right\} \quad (21)$$

thus holds with probability at least $1 - \delta_s$ by a union-bound argument.

This iterative procedure allows for an adaptive size of the number of measurements required to be evaluated. However, there is cause for concern with the definition of the stopping rule. That is the fact that the range, R, used in the calculation of Equation (19), is dependent on the log likelihood for all M_k measurements. Calculating this range would thus inherently require at least the same number of calculations as the generic SMCMC approach in Algorithm 1. In certain applications, it may be possible to obtain an expression for the range which is independent of the measurements; however, this is not the case in general. To avoid

1: Determine initial proxy parameters.
2: **for** $k = 1,...,T$ **do**
3: **for** $j = 1,...,N + N_b$ **do**
4: **if** $j = 1 \vee N_b$ **then**
5: Update proxy parameters.
6: **end if**
7: **if** $k = 1$ **then**
8: **for** $j = 1,...,N + N_b$ **do**
9: Sample $X_{1,1}^{j} \sim \mathscr{K}_1(X_{1,1}^{j-1}, \cdot)$ with $\mathscr{K}_1$ a confidence MH based kernel (for example Algorithm 5) with invariant distribution $p(x_1|z_1) \propto \mu(x_0) g_k(z_1|x_1)$
10: **end for**
11: **else if** $k \geq 2$ **then**
12: Sample $X_{k,1:k}^{j} \sim \mathscr{K}_n(X_{k,1:k}^{j-1}, \cdot)$ with $\mathscr{K}_k$ a confidence MH based kernel (for example Algorithm 5) with invariant distribution Equation (5)
13: **end if**
14: **end for**
15: Approximation of the smoothing distribution with the following empirical measure,

$$p(x_{1:k}|z_{1:k}) \approx \frac{1}{N} \sum_{j=N_b+1}^{N_b+N} \delta_{X_{k,1:k}^{j}}(dx_{1:k})$$

16:Approximation of the filtering distribution with Equation (7)
17: **end for**

Algorithm 4: Adaptive subsampling sequential Markov chain Monte Carlo.

computing the range, it has been proposed to substitute a computationally cheap upper bound of the range [55]. A control variate, referred to as a proxy, was introduced to enable the computation of an upper bound of the range:

$$\wp_i\left(X_{k,k}^{j-1}, X_{k,k}^{*}\right) \approx \log\left[\frac{g_k\left(z_k^i \middle| X_{k,k}^{*}\right)}{g_k\left(z_k^i \middle| X_{k,k}^{j-1}\right)}\right]. \tag{22}$$

This results in the MC approximation of Equation (17) being augmented into

$$\Lambda^{S_j}\left(X_{k,1:k}^{j-1}, X_{k,1:k}^{*}\right) = \frac{1}{S_j} \sum_{i=1}^{S_j} \log\left[\frac{g_k\left(z_k^{i,*} \middle| X_{k,k}^{*}\right)}{g_k\left(z_k^{i,*} \middle| X_{k,k}^{j-1}\right)}\right] - \wp_i\left(X_{k,k}^{j-1}, X_{k,k}^{*}\right). \tag{23}$$

The proxy also has the additional advantage of reducing the sample variance V_{S_m}. With the inclusion of the proxy, the range in Equation (20) has the following form:[2]

$$R = \max_{1 \leq i \leq M_k} \left\{ \log\left[\frac{g_k\left(z^i \middle| X_{k,k}^{*}\right)}{g_k\left(z^i \middle| X_{k,k}^{j-1}\right)}\right] - \wp_i\left(X_{k,k}^{j-1}, X_{k,k}^{*}\right) \right\} - \min_{1 \leq i \leq M_k} \left\{ \log\left[\frac{g_k\left(z^i \middle| X_{k,k}^{*}\right)}{g_k\left(z^i \middle| X_{k,k}^{j-1}\right)}\right] - \wp_i\left(X_{k,k}^{j-1}, X_{k,k}^{*}\right) \right\}. \tag{24}$$

In [55], it was proposed to utilize a Taylor series as an approximation for the log likelihood, $\ell_i(x_k) = \log\ g_k(z_k^i|x_k)$, since the Taylor series is cheap to compute and Taylor's theorem can be used to obtain an upper bound on the range. The focus of this paper is on the reduction of computational complexity, and thus, a first-order Taylor series is considered. However, the result can be trivially expanded for higher order Taylor series. In this case, the Taylor series approximation of the log likelihood is given by

$$\widehat{\ell}_i(x_k) = \ell_i(x_k^+) + (\nabla \ell_i)_{x_k^+}^{\top} \cdot (x_k - x_k^+), \tag{25}$$

where $(\nabla \ell_i)_{x_k^+}$ represents the gradient of $\ell_i(x_k)$ evaluated at x_k^+. This results in the following form of the proxy

$$\wp_i\left(X_{k,k}^{j-1}, X_{k,k}^{*}\right) = \widehat{\ell}_i\left(X_{k,k}^{*}\right) - \widehat{\ell}_i\left(X_{k,k}^{j-1}\right), \tag{26}$$

$$= (\nabla \ell_i)_{x_k^+}^{\top} \cdot \left(X_{k,k}^{*} - X_{k,k}^{j-1}\right). \tag{27}$$

An upper bound for the range can be derived:

$$R \leq 2 \max_{1 \leq i \leq M_k} \left\{ \left| \log\left[\frac{g_k\left(z_k^i \middle| X_{k,k}^{*}\right)}{g_k\left(z_k^i \middle| X_{k,k}^{j-1}\right)}\right] - \wp_i\left(X_{k,k}^{*}, X_{k,k}^{j-1}\right) \right| \right\}, \tag{28}$$

$$\leq 2 \max_{1 \leq i \leq M_k} \left\{ \left| \ell_i\left(X_{k,k}^{*}\right) - \ell_i\left(X_{k,k}^{j-1}\right) - \widehat{\ell}_i\left(X_{k,k}^{*}\right) + \widehat{\ell}_i X_{k,k}^{j-1} \right| \right\}, \tag{29}$$

$$\leq 2 \max_{1 \leq i \leq M_k} \left\{ \left| B\left(X_{k,k}^{j-1}\right) - B\left(X_{k,k}^{*}\right) \right| \right\}, \tag{30}$$

1: Given: $X_{k,1:k}^{j-1}$, $z_k = \{z_k^1, \cdots, z_k^{M_k}\}$, and δ_s.
2: Initialize: number of sub-sampled measurements, $S_j = 0$, Approximate log likelihood ratio subtracted by proxy, $\Lambda = 0$, set of sub-sampled measurements, $\mathbf{z}_k^* = \varnothing$, initial batchsize, $b = 1$, while loop counter, $w = 0$.
3: Sample $X_{k,1:k}^* \sim q(x_{1:k}|X_{k,1:k}^{j-1})$ and $u \sim \mathrm{U}_{[0,1]}$.
4: Compute $\psi(X_{k,1:k}^{j-1}, X_{k,1:k}^*)$ as defined in Equation (17).
5: Compute an upper bound for the range, R, according to Equation (29).
6: Compute the proxy, $\{\wp_i(X_{k,k}^*, X_{k,k}^{j-1})\}_{i=1}^{M_k}$, according to Equation (27).
7: DONE = FALSE
8: **while** DONE == FALSE **do**
9: $w = w + 1$
10: $\{z_k^{S_j+1,*}, \cdots, z_k^{b,*}\} \sim_{w/repl.} z_k \setminus z_k^*$
11: $z_k^* = z_k^* \cup \{z_k^{S_j+1,*}, \cdots, z_k^{b,*}\}$
12: $\Lambda = (1/b)(S_j\Lambda + \sum_{i=S_j+1}^{b}[\log(g_k(z_k^{i,*}|X_{k,k}^*)/g_k(z_k^{i,*}|X_{k,k}^{j-1})) - \wp_i(X_{k,k}^*, X_{k,k}^{j-1})])$
13: $S_j = b$
14: $\delta_w = (p_s - 1/p_s w^{p_s})\delta_s$
15: Compute c according to Equation (20) utilizing δ_w.
16: $b = \gamma_s S_j \wedge M_k$
17: **if** $|\Lambda + 1/M_k \sum_{i=1}^{M_k} \wp_i(X_{k,k}^*, X_{k,k}^{j-1}) - \psi(X_{k,1:k}^*, X_{k,1:k}^{j-1})| \geq c$ or $S_j == M_k$ **then**
18: DONE = TRUE
19: **end if**
20: **end while**
21: **if** $\Lambda > \psi(X_{k,1:k}^*, X_{k,1:k}^{j-1}) - 1/M_k \sum_{i=1}^{M_k} \wp_i(X_{k,k}^*, X_{k,k}^{j-1})$ **then**
22: $X_{k,1:k}^j = X_{k,1:k}^*$
23: **else**
24: $X_{k,1:k}^j = X_{k,1:k}^{j-1}$
25: **end if**

Algorithm 5: Confidence Metropolis–Hasting algorithm as $\mathscr{K}_k$ (j-th iteration).

where $B(x_k) = \ell_i(x_k) - \widehat{\ell}_i(x_k)$ is the remainder. The Taylor–Lagrange inequality states that if $|\nabla^2(\ell_i(x_k))| \leq Y$, where $\nabla^2(\ell_i(x_k))$ represents the Hessian of the log likelihood, on some interval $I = [a, b]$, then the remainder term, $B(x_k)$, can be upper bounded according to $|B(x_k)| \leq Y|x_k - x_k^+|^2/2$ on the same interval I. Finally, based on the triangle inequality, an upper bound on the range term is given by

$$R \leq 2 \max_{1 \leq i \leq M_k} \left| \left| B\left(X_{k,k}^*\right) \right| + \left| B\left(X_{k,k}^{j-1}\right) \right| \right|, \tag{31}$$

$$\leq \left| Y \left(\left| X_{k,k}^* - x_k^+ \right|^2 + \left| X_{k,k}^{j-1} - x_k^+ \right|^2 \right) \right|. \tag{32}$$

The AS-SMCMC approach is illustrated by Algorithm 4 and the confidence MH sampler in Algorithm 5.

5. Performance Validation

In this section, a comparison of the generic SMCMC algorithm [28] with the proposed AS-SMCMC algorithm and EP-SMCMC algorithm referred to as SMCMC, AS-SMCMC, and EP-SMCMC, respectively, is presented. All the algorithms were implemented in the interpreted language MATLAB (source code is available at https://github.com/allandf/speeding-up-smcmc-distributed-sensor-networks). The parallel processing for the EP-SMCMC algorithm was achieved in MATLAB with the parfor command. All simulations were performed on a mobile computer with Intel Core i7-4702HQ CPU @ 2.20 GHz with 16GB of RAM. All results are averaged over 50 independent runs. The length of the burn-in period, N_b, was determined based on the analysis of trace plots. However, there are also a wealth of convergence diagnostics (e.g., the Gelman–Rubin diagnostic [69]) used in static MCMC which are directly applicable to SMCMC that can be used for this purpose.

5.1. Algorithm Considerations

5.1.1. Choice of MCMC Kernel. The selection of the MCMC kernel for sampling has an overall impact on the general performance of SMCMC. There are a variety of kernels from the MCMC literature which can be employed in SMCMC. The only limitation in the context of this paper is for the AS-SMCMC where the kernel is restricted to the MH kernel or a composite MH kernel. An in-depth review of MCMC kernels in the context of SMCMC is provided in [28]. The same MCMC kernel is used for each approach in this paper unless otherwise stated.[3]

5.1.2. EP-SMCMC. For the examples presented in this paper, the member of the exponential family selected to

approximate the likelihood terms is the multivariate Gaussian distribution. For this case, the NPs are given by

$$\eta = \left(\Sigma^{-1}\mu, \Sigma^{-1}\right)^{\top}, \tag{33}$$

where μ and Σ represent the mean and covariance of the multivariate Gaussian distribution. In this case, the NP update in Equation (13) simplifies to

$$\Sigma_d^{-1}\mu_d = \Sigma_{p,d}^{-1}\mu_{p,d} - \left(\Sigma_{f,d}^{-1}\mu_{f,d} + \sum_{i\neq d}\Sigma_i^{-1}\mu_i\right), \tag{34}$$

$$\Sigma_d^{-1} = \Sigma_{p,d}^{-1} - \left(\Sigma_{f,d}^{-1} + \sum_{i\neq d}\Sigma_i^{-1}\right), \tag{35}$$

where standard techniques are used to obtain unbiased mean and covariance estimates for the discrete distributions. It is important to note that the difference between two positive definite matrices is not necessarily itself positive definite. Techniques, such as *SoftAbs* [70], can be used to ensure that the result remains positive definite.

5.1.3. AS-SMCMC. The AS-SMCMC approach is dependent on the existence of an expression for the range of the log likelihood ratio, or an upper bound on the range of the log likelihood ratio, in order to define a bound on the deviation of the MC approximation from the complete log likelihood ratio using concentration inequalities. This implicitly requires the range of the log likelihood ratio to be finite. However, this constraint is satisfied by a variety of observation models, as illustrated in the following examples.

5.1.3.1. Example 1: Dynamic Gaussian Process With Gaussian Likelihood. The first example is based on a Gaussian state space model with corresponding transition density and likelihood pdfs:

$$f_k(x_k|x_{k-1}) = \mathcal{N}(x_k; Ax_{k-1}, Q), \tag{36}$$

$$g_k\left(z_k^i|x_k\right) = \mathcal{N}\left(z_k^i; Hx_k, R\right). \tag{37}$$

The measurements are assumed independent, hence resulting in the joint likelihood expression for all measurements:

$$p(z_k|x_k) = \prod_{i=1}^{M_k} g_k\left(z_k^i|x_k\right). \tag{38}$$

An advantage of the Gaussian model in Equations (36) and (37) is that the Kalman filter [10] can be used as a benchmark for performance. Unless otherwise specified, the following parameters were utilized for all experiments. The filter parameters include the number of particles, SMCMC and AS-SMCMC, $N_p = 4000$, and EP-SMCMC, $N_p = 500$, for each computing node (number of computing nodes, $D = 4$); the number of EP iterations, $L = 2$; and the subsampling parameters, $\gamma_s = 1.2$, $\delta_s = 0.1$, and $p_s = 2$. The simulation parameters include the number of measurements at each time step, $M = 500$; the total simulation time, $T_{\text{tot}} = 20$ time steps; the transition density parameters, $Q = 0.08$ and $A = 0.9$; the likelihood parameters, $H = 1$ and $R = 2$; and the state space dimension size, $N_d = 1$.

TABLE 1: Algorithm computation time per time step.

Algorithm	Time (s)	Computational gain (%)
	$M = 500$	
SMCMC	114.75	0
AS-SMCMC	69.54	39.4
EP-SMCMC	9.89	91.38
	$M = 5000$	
SMCMC	1087.93	0
AS-SMCMC	274.60	74.76
EP-SMCMC	96.40	91.14

For this example, the MCMC kernel used is a composite MH kernel [28] consisting of two consecutive draws from MH kernels. The proposal distribution used for the first MH kernel is

$$q_1\left(x_{k-1}\middle|X_{k,k-1:k}^j\right) = p(x_{k-1}|x_k, z_{1:k}), \tag{39}$$

$$= \sum_{m=N_b+1}^{N_b+N} \frac{f_k\left(x_k = X_{k,k}^j\middle|X_{k,k-1}^m\right)}{\sum_i^N f_k\left(x_k = X_{k,k}^j\middle|X_{k,k-1}^i\right)}\delta_{X_{k,1:k}^m}(dx_{1:k}). \tag{40}$$

The invariant distribution for the second MH kernel is given by the following conditional posterior:

$$p\left(x_k\middle|X_{k,k-1}^j, z_{1:k}\right) = g_k(z_k|x_k)f_k\left(x_k\middle|X_{k,k-1}^j\right). \tag{41}$$

The proposal distribution for the second MH kernel in the case of SMCMC and AS-SMCMC is

$$q_2\left(x_k\middle|X_{k,k-1:k}^j\right) = f_k\left(x_k\middle|X_{k,k-1}^j\right). \tag{42}$$

In the case of EP-SMCMC, the conditional posterior for local computing node d is given by

$$p_d\left(x_k\middle|X_{k,k-1}^j, z_{1:k}\right) = g_{k,d}(z_{k,d}|x_k)f_k\left(x_k\middle|X_{k,k-1}^j\right)\prod_{i\neq d}\pi(x_k|\eta_i). \tag{43}$$

The proposal distribution for the second MH kernel in this case is selected as

$$q_2\left(x_k\middle|X_{k,k-1:k}^j\right) \propto f_k\left(x_k\middle|X_{k,k-1}^j\right)\prod_{i\neq d}\pi(x_k|\eta_i), = \mathcal{N}\left(x_k; \mu_q, \Sigma_q\right), \tag{44}$$

where μ_q and Σ_q are derived from the NPs $\eta_q = \eta_{g,d} + \sum_{i\neq d}\eta_i$ and $\eta_{g,d}$ represents the NPs of the transition density, $f_k(x_k|X_{k,k-1}^j)$. Table 1 illustrates the computational complexity of

TABLE 2: Acceptance rates for the first refinement step.

Algorithm	Acceptance rate (min, median, mean, max)
SMCMC	(30.93, 94.35, 89.72, 96.57)
AS-SMCMC	(30.90, 94.43, 89.70, 96.54)
EP-SMCMC ($L = 1$)	(34.15, 93.99, 89.47, 94.86)
EP-SMCMC ($L = 2$)	(30.86, 94.54, 89.77, 96.59)

TABLE 3: Acceptance rates for the second refinement step.

Algorithm	Acceptance rate (min, median, mean, max)
SMCMC	(8.82, 23.44, 21.26, 25.78)
AS-SMCMC	(9.04, 24.24, 21.95, 26.75)
EP-SMCMC ($L = 1$)	(19.76, 42.07, 38.46, 45.01)
EP-SMCMC ($L = 2$)	(72.11, 76.24, 75.86, 77.69)

the algorithms for 500 and 5000 measurements. It is interesting to note that an increase in measurements leads to an increase in computational savings in AS-SMCMC. It is important to note that in reality, the computational cost of EP-SMCMC may increase due to processing the measurements into batches and communicating the batches to computing nodes.

Although the focus of this paper is not related to the optimal selection of MCMC kernels, it is interesting to note the difference in acceptance rates illustrated in Tables 2 and 3. In Table 2, the acceptance probabilities for the different algorithms do not differ significantly. This is expected since all three algorithms utilize the same proposal distribution and acceptance ratio for the first refinement step, and additionally, this refinement step is not dependent on the data.

Table 3 highlights the improvement in the acceptance ratio for the EP-SMCMC in this scenario. The increase during the first EP iteration is due to the relative decrease in the number of measurements processed by each computing node. The large increase during the second EP iteration is due to a smarter proposal distribution which incorporates the information about the measurements from the other computing nodes.

The Kolmogorov–Smirnov (KS) statistic is used to gauge the relative accuracy of correctly approximating the filtering distribution of interest empirically by the algorithms. The KS statistic is given by

$$\mathrm{KS} = \max_{x} \left(\widehat{F}(x) - G(x) \right), \tag{45}$$

where $\widehat{F}(x)$ is an empirical cumulative distribution function (cdf) and $G(x)$ is a continuous cdf. In this setting, $\widehat{F}(x)$ is the empirical cdf of the discrete filtering distribution estimated by the SMCMC algorithms, and $G(x)$ is the cdf of a Gaussian distribution with parameters updated by a Kalman filter.

For EP-SMCMC, the samples from all D computing nodes at the final EP iteration are considered. It is worthwhile mentioning that the transmission of the samples from the D computing nodes to a single computing node was utilized in this experiment but is not necessary when only estimates are required to be extracted. For example, since the samples in SMCMC are unweighted, the global mean can be established through the averaging of the individual local means. The KS statistic for several different filter configurations is illustrated in Figure 2 for both the cases of 500 and 5000 measurements. It is first noted that the SMCMC and AS-SMCMC share almost identical performance. This was expected as the goal of AS-SMCMC is to make the same accept or reject decision in the embedded MCMC algorithms as in SMCMC, only while evaluating less measurements. From Figure 2a, it can be seen that the performance of the EP-SMCMC varies depending on the configuration. Doubling the number of computing nodes, while halving the number of samples, conserves the total number of samples while further increasing the computational efficiency at the cost of an increase in error. While in the other extreme case, increasing the number of samples while keeping the number of computing nodes fixed significantly increases the accuracy while decreasing the computational gain. The case of N_p equal to 1000 results in the same number of samples for all three algorithms. It is clear that even in this scenario, there is an increase in performance, which can be attributed to the increased acceptance rate which results in a more diverse empirical cdf. The EP-SMCMC algorithm is also well-suited in this specific example due to the Gaussian nature of the model and the utilization of the Gaussian density for the approximate likelihood terms.

5.1.3.2. Example 2: WSN Tracking in Clutter. In this example, the application of target tracking with clutter in a complex WSN [71, 72] is considered. The WSN consists of 400 sensors randomly distributed in a region with an area of $1000 \times 1000\,\mathrm{m}^2$. Each sensor has a sensing range, r_s. The state vector consists of the position and velocity of a target in a two-dimensional space, $x_k = [x_k, \dot{x}_k, y_k, \dot{y}_k]^\top$. The motion that the target adheres to is assumed to be the nearly constant velocity model with corresponding state transition density described by

$$f_k(x_k | x_{k-1}) = \mathcal{N}(x_k ; Ax_{k-1}, \Gamma v_{k-1}), \tag{46}$$

where $A = \mathrm{diag}\,(A_1, A_1)$, $A_1 = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix}$, $\Gamma = \begin{bmatrix} T/2 & 1 & 0 & 0 \\ 0 & 0 & T/2 & 1 \end{bmatrix}$, v_{k-1} is the system dynamic noise, and T is the sampling interval. To capture a broad spectrum of motion patterns, the trajectory is assumed to evolve with linear velocity segments, while sharp changes occur primarily at turning points (most notably in the velocity direction). As such, the process noise in the system dynamics is modeled as a mixture of two Gaussian distributions:

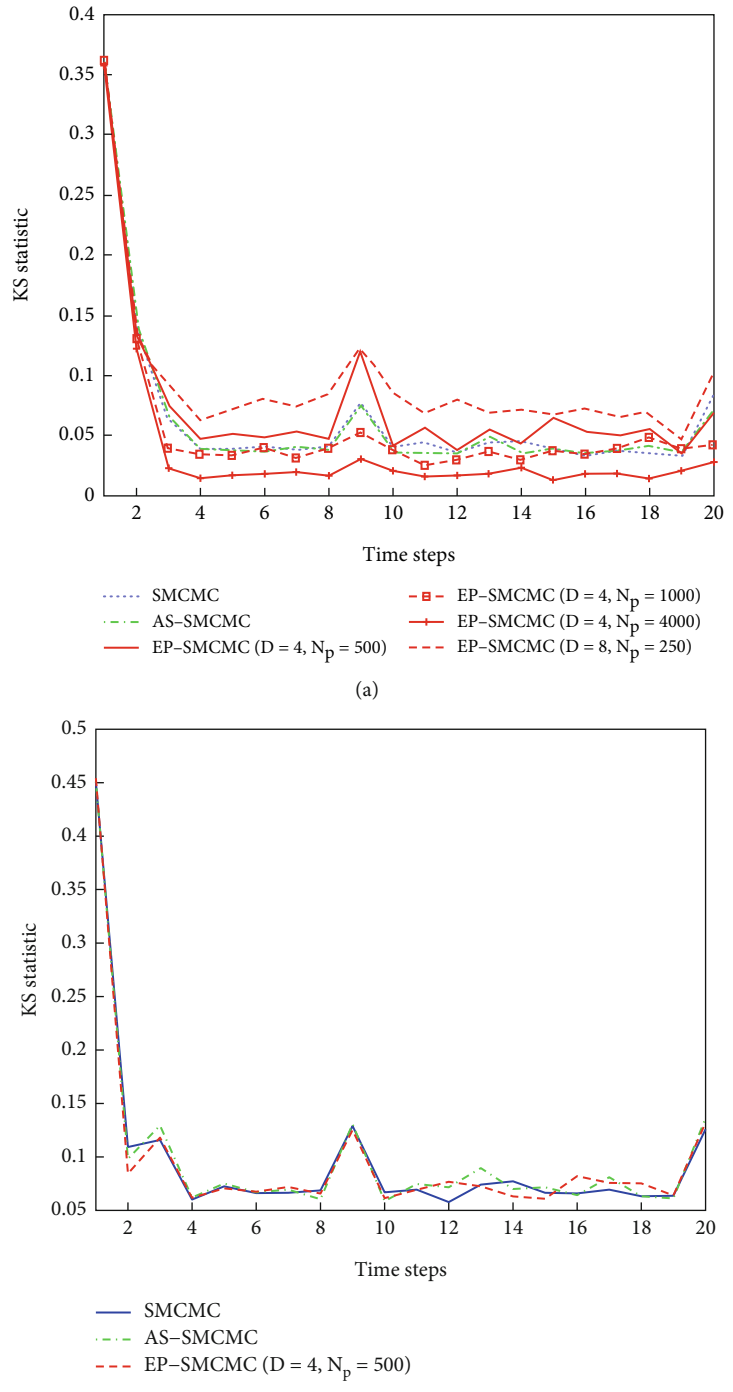


FIGURE 2: The KS statistic for several configurations of the SMCMC-based algorithms relative to the Kalman filter. (a) Comparison of the KS statistic for the case of 500 measurements. SMCMC, AS-SMCMC, and EP-SMCMC ($D = 4, N_p = 500$) and EP-SMCMC ($D = 8, N_p = 250$) correspond to an equivalent total of 4000 particles. The other results indicate improvements due to an increase in particles. (b) Comparison of the KS statistic for the case of 5000 measurements.

$$p(v_{k-1}) = \alpha\mathcal{N}(0, Q_1) + (1-\alpha)\mathcal{N}(0, Q_2), \tag{47}$$

where $Q_1 = \text{diag}\,(\sigma^2, \sigma_1^2)$, $Q_2 = \text{diag}\,(\sigma^2, \sigma_2^2)$, and σ denote a common and fixed standard deviation for both the x and y components; $\sigma_1 \ll \sigma_2$ are standard deviations allowing the model to account for gradual velocity variations and sudden directional shifts, respectively. The coefficient $\alpha \in [0\ 1]$ specifies the relative contribution of the two Gaussian components.

A single run of the scenario of interest is illustrated in Figure 3.

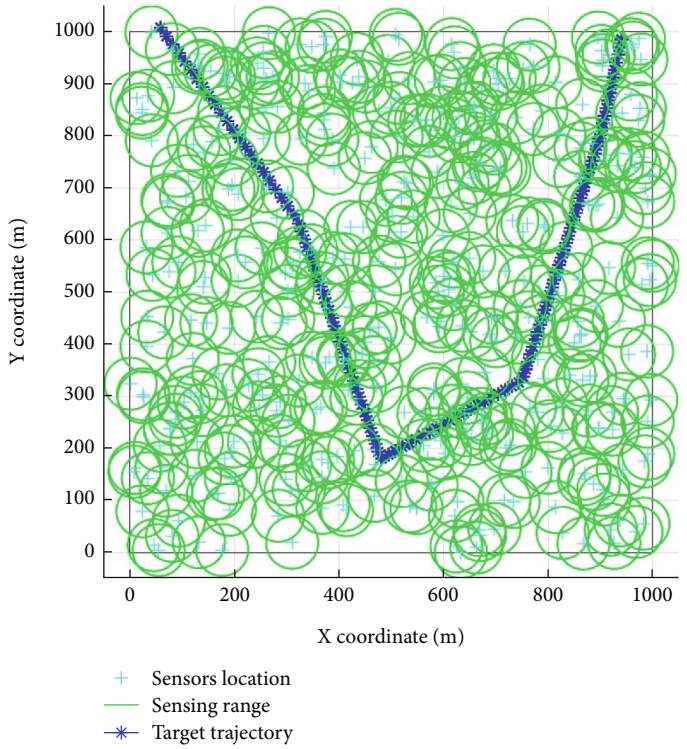


Figure 3: True target trajectory in the xy plane and sensor locations with respective sensing ranges.

Each sensor in the network collects measurements while the target is within the sensing range. Thus, the set of active sensors collecting measurements is a time-dependent set, represented by $\mathscr{L}_k$. The total number of measurements received at the l-th sensor is given by $M_{k,l} = M^x_{k,l} + M^c_{k,l}$, where $M^x_{k,l}$ is the number of target-originated measurements and $M^c_{k,l}$ is the number of clutter measurements. This setting is representative of a high-sampling frequency (higher than the sampling frequency of the estimation process) and/or high-resolution sensing modality(ies). The number of target and clutter measurements is Poisson distributed with mean λ_X and λ_C, respectively. The likelihood density for the l-th sensor's measurements therefore takes the following form [73]:

$$p(z_{k,l}|x_k) = \frac{e^{-\mu_k}}{M_{k,l}!} \prod_{i=1}^{M_{k,l}} \left(\lambda_C p_C\left(z^i_{k,l}\right) + \lambda_X p_X\left(z^i_{k,l}|x_k\right)\right), \quad (48)$$

where $\mu_k = \lambda_C + \lambda_T$, $p_X(\cdot)$, and $p_C(\cdot)$ denote the single-measurement likelihoods for target and clutter measurements, respectively. Each individual measurement is a point in the two-dimensional observation space, $z^i_{k,l} = [z^i_{x,k}, z^i_{y,k}]^\top$. In the case of a measurement from a target, the likelihood is modeled as $p_X(z^i_{k,l}|x_k) = \mathcal{N}(z^i_{k,l}; x_k, \Sigma)$. Clutter measurements are independent of the target state and are uniformly distributed over the visible region of the sensor. Hence, the clutter likelihood takes the form of $p_C(z^i_{k,l}) = U_{R_x}(z^i_{x,k}) U_{R_y}(z^i_{y,k}) = 1/A_c$, where $A_c = \pi r_s^2$ represents the clutter area. The sensors are assumed conditionally independent, leading to

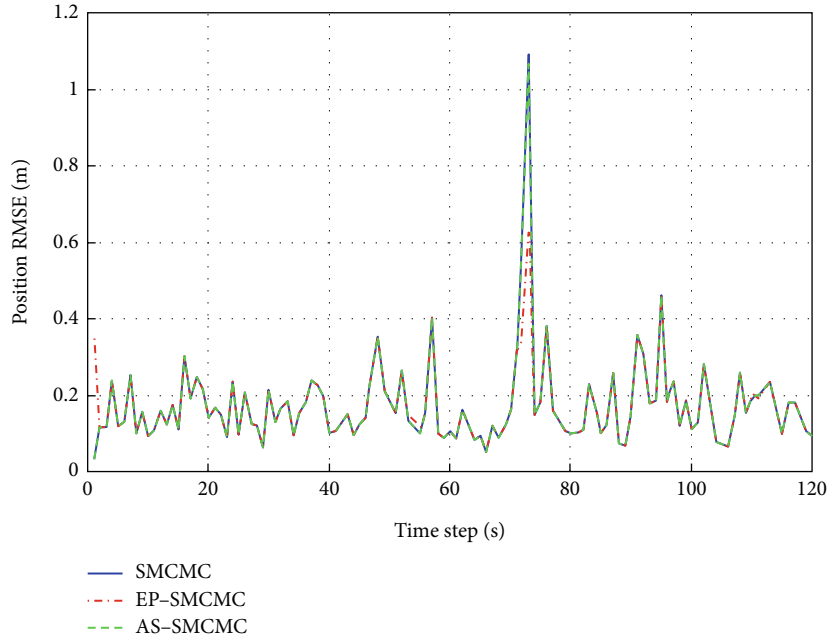
$$p(z_k|x_k) = \prod_{l \in \mathscr{L}_k} p(z_{k,l}|x_k). \quad (49)$$

The following parameters, unless otherwise specified, were used for all experiments. The filter parameters include the number of particles, for SMCMC and AS-SMCMC, $N_p = 4000$, and EP-SMCMC, $N_p = 500$, for each computing node (number of computing nodes, $D = 4$); the covariance associated with the proposal for the refinement step, $\Sigma_r = 0.01I$; and the subsampling parameters, $\gamma_s = 2$, $\delta_s = 0.1$, and $p_s = 1.1$. The simulation parameters include a total running time, $T = 120$ s, with sampling time, $T = 1$ s; the parameters associated with the motion model, $\alpha = 0.5$, $\sigma = 1$, $\sigma_1 = 0$, and $\sigma_2 = 10$; the target observation model parameters, $\lambda_X = 1000$ and $\Sigma = 16I$; the clutter parameters, $\lambda_C = 1000$; and sensor range, $r_s = 50$ m.

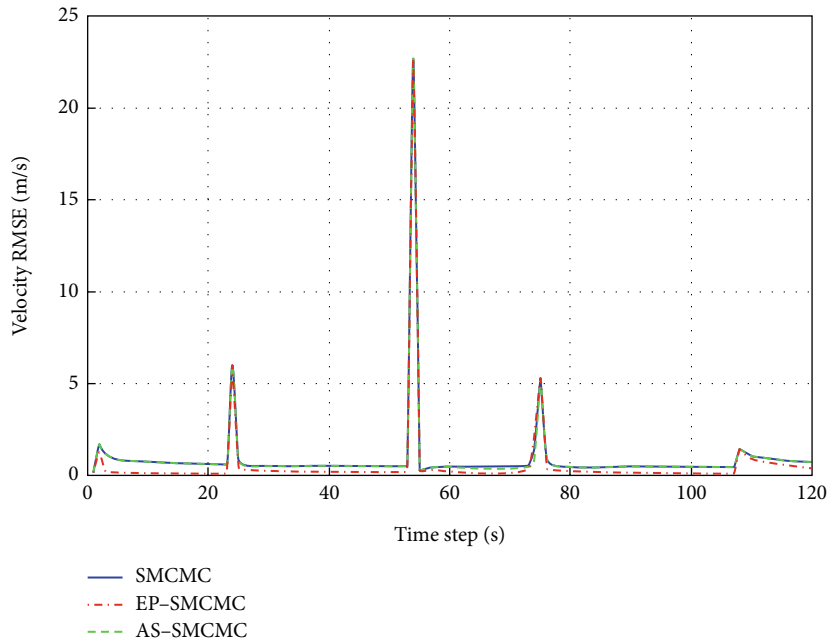
For this example, a composite MH kernel [28] consisting of a joint draw and a local refinement step on the current state only, for all three algorithms, is utilized. The proposal distribution for the joint draw in the SMCMC and AS-SMCMC is given by

$$q_1\left(x_k, x_{k-1} \middle| X^j_{k,k-1:k}\right) = f_k(x_k|x_{k-1}) \times \frac{1}{N} \sum_{j=N_b+1}^{N+N_b} \delta_{X^j_{k,k-1}}(dx_{k-1}). \quad (50)$$

The local refinement step consists of two parts. Firstly, a symmetric random walk with covariance Σ_r is applied to the position components with the following proposal distribution:

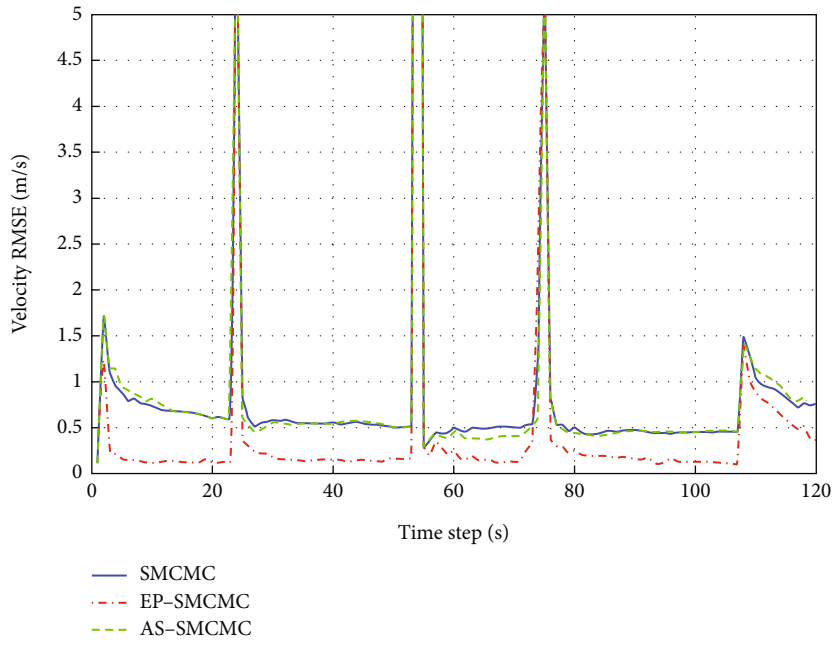


(a) RMSE averaged over the position dimensions



(b) RMSE averaged over the velocity dimensions

Figure 4: Continued.



(c) Zoomed in view of the RMSE averaged over the velocity dimensions

FIGURE 4: A comparison of the position and velocity RMSEs for the WSN tracking example.

TABLE 4: Average algorithm computation time per time step.

Algorithm	Time (s)	Computational gain (%)
SMCMC	5.13	0
AS-SMCMC	4.86	5.32
EP-SMCMC	0.92	82.12

$$q_{2,p}\left(x_k(\Omega)\middle|X^j_{k,k-1:k}\right) = \mathcal{N}\left(X^j_{k,k}(\Omega), \Sigma_r\right), \tag{51}$$

where $x_k(\Omega) = [x_k, y_k]^T$. Secondly, a Gibbs-based draw of the velocity components by the mixture conditional implied by Equation (46) is applied.

In the case of EP-SMCMC, the proposal distribution for the joint draw is given by

$$q_1\left(x_k, x_{k-1}\middle|X^j_{k,k-1:k}\right) \propto f_k\left(x_k\middle|X^j_{k,k-1}\right)\prod_{i\neq d}\pi(x_k|\eta_i) \times \frac{1}{N}\sum_{j=N_b+1}^{N+N_b}\delta_{X^j_{k,k-1}}(dx_{k-1}) = \mathcal{N}\left(x_k; \mu_q, \Sigma_q\right) \times \frac{1}{N}\sum_{j=N_b+1}^{N+N_b}\delta_{X^j_{k,k-1}}(dx_{k-1}),$$

where μ_q and Σ_q are derived from the NPs $\eta_q = \eta_{g,d} + \sum_{i\neq d}\eta_i$ and $\eta_{g,d}$ represents the NPs of the transition density, $f_k(x_k|X^j_{k,k-1})$. The same local refinement step is used as previously described for SMCMC and AS-SMCMC in EP-SMCMC.

It is interesting to note that in this example, the likelihood expression, given in Equation (47), is independent of the target's velocities. Therefore, when determining the NPs of the approximate likelihood terms using Equation (35), the subtraction of the precision terms between the posterior and predictive posterior distributions was forced to zero for all the dimensions related to target velocity. This eliminates potential numerical problems that could arise in the empirical estimation of the NPs from a finite number of samples.

Figure 4 reports the root mean square error (RMSE) for position and velocity. In general, when many measurements are available, all algorithms produce accurate estimates. Notably, the position RMSE in Figure 4a is fairly spiky. This reflects large fluctuations in the number of available measurements from one time step to the next as the target leaves and/or enters multiple sensors' fields of view. More sensors observing the target provide more information and thus lower RMSE.

The most prominent spike in position RMSE occurs around Time Step 70, caused by two consecutive time steps in which the target passes through a region with no sensor coverage, leading to filter coasting. The resulting increase in RMSE is nearly identical for SMCMC and AS-SMCMC, but smaller for EP-SMCMC. This difference is linked to EP-SMCMC's more accurate velocity estimation, as shown in Figure 4b,c.

In Figure 4b, the effect of the target's sharp turns, visible in the target trajectory in Figure 3, is large spikes in the velocity RMSE. In Figure 4c, EP-SMCMC yields the lowest velocity RMSE. This is attributed to the second EP iteration, where the smarter proposal distribution incorporates the information about the measurements from the other computing nodes, as observed in Example 1.

Computation time varies at each time step due to fluctuations in the number of measurements arising from changes

in sensor coverage. Total computation time was recorded in MATLAB and divided by the number of time steps; the resulting average time per time step for each algorithm is reported in Table 4. In this scenario, the computational gain of AS-SMCMC is marginal, whereas EP-SMCMC achieves a substantial speedup. The difference stems from their underlying mechanisms. For both methods, the dominant cost lies in processing measurements. EP-SMCMC reduces this cost by partitioning the dataset; with a proportional increase in computing nodes, finer partitioning yields a more predictable reduction in runtime. By contrast, the speedup from AS-SMCMC is less predictable because it depends on the characteristics of the filtering distribution and several algorithmic settings.

6. Conclusions

This paper presents two novel frameworks for SMCMC for dealing with large volumes of data in dynamic systems. The power of the frameworks is demonstrated through two examples, with comparisons to a state-of-the-art SMCMC algorithm. Implementations of both frameworks showed that, under certain conditions, Bayesian filtering complexity can be reduced by up to 90% with negligible loss in estimation accuracy.

The first framework is divide-and-conquer SMCMC which is illustrated with an implementation based on an EP algorithm. The computational gain from divide-and-conquer SMCMC is achieved through efficient parallel processing of data partitions. In both examples considered, computational benefits scaled with the number of data partitions when matched to available compute nodes, making it the most promising approach for raw speedups. However, performance depends critically on how local filtering distributions are recombined. Large discrepancies among the local filtering distributions can yield poor global approximations and degrade estimation accuracy. This limitation was not fully explored in this paper and should be examined in future work. Additionally, another avenue of interest for future work is the implementation of more efficient MCMC methods, robust to heterogeneous local posteriors (such as [35]), within the divide-and-conquer SMCMC framework.

The second framework is subsampling SMCMC which is illustrated with an implementation based on adaptive subsampling. In this case, the computational gain is based on the processing of an adaptive subsample of the data. The attainable speedup is problem and design-dependent, and careful construction of the confidence MH kernel is needed to maximize potential computational gain. The greatest gains can be expected when the filtering distribution is relatively concentrated. In the worst case, it reverts to using all measurements, sacrificing speedup but preserving estimation accuracy and thus offering estimation accuracy robustness. Future work will focus on the implementation of inexact iterative strategies which maximize performance under real-time computational budgets, such as [57], within the subsampling SMCMC framework.

Both frameworks have flexible structures; however, in sensor network applications, the divide-and-conquer approach has the added advantage of potentially avoiding intersensor data transmission, reducing communication overhead. The implementations presented in this paper rely on the conditional independence of measurements. If this was violated, it would result in the introduction of a bias, as in [15]. Developing methods to negate this bias is the next big challenge for big data algorithms in both static and dynamic scenarios.

Appendix A

The AS-SMCMC algorithm requires an upper bound on the range of the log likelihood ratio, as described in Equation (32). The upper bound is dependent on the Hessian of the log likelihood. In the examples exhibited in Section 5, the upper bound is independent of the data and is computed offline prior to tracking. In Example 1, the likelihood for the i-th measurement is given by

$$p\left(z_k^i|x_k\right) = \mathcal{N}\left(z_k^i; Hx_k, R\right), \tag{A.1}$$

with corresponding log likelihood,

$$\begin{aligned}\ell_i(x_k) = &-\frac{1}{2}\log\left(|R|\right) - \frac{N_d}{2}\log\left(2\pi\right)\\ &-\frac{1}{2}\left(z_k^i - Hx_k\right)^\top R^{-1}\left(z_k^i - Hx_k\right).\end{aligned} \tag{A.2}$$

The gradient of the log likelihood is

$$\nabla\ell_i(x_k) = H^\top R^{-1}\left(z_k^i - Hx_k\right). \tag{A.3}$$

Finally, the Hessian of the log likelihood is given by

$$\nabla^2\ell_i(x_k) = H^\top R^{-1}H. \tag{A.4}$$

In Example 2, the likelihood for the i-th measurement is given by

$$p(z_{i,k}|x_k) = \lambda_C p_C\left(z_k^i\right) + \lambda_X p_X\left(z_k^i|x_k\right), \tag{A.5}$$

with corresponding log likelihood,

$$\ell_i(x_k) = \log\left(\lambda_C p_C\left(z_k^i\right) + \lambda_X p_X\left(z_k^i|x_k\right)\right). \tag{A.6}$$

The gradient of the log likelihood is

$$\nabla\ell_i(x_k) = \frac{\lambda_X \nabla p_X\left(z_k^i|x_k\right)}{\left(\lambda_C p_C\left(z_k^i\right) + \lambda_X p_X\left(z_k^i|x_k\right)\right)}. \tag{A.7}$$

Finally, the Hessian of the log likelihood is given by

$$\nabla^2 \ell_i(x_k) = \frac{\left(\lambda_C p_C\left(z_k^i\right) + \lambda_X p_X\left(z_k^i|x_k\right)\right)\lambda_X \nabla p_X\left(z_k^i|x_k\right)}{\left(\lambda_C p_C\left(z_k^i\right) + \lambda_X p_X\left(z_k^i|x_k\right)\right)^2} + \frac{\lambda_X^2 \nabla p_X\left(z_k^i|x_k\right)\left(\nabla p_X\left(z_k^i|x_k\right)\right)^\top}{\left(\lambda_C p_C\left(z_k^i\right) + \lambda_X p_X\left(z_k^i|x_k\right)\right)^2}. \tag{A.8}$$

Data Availability Statement

The data that support the findings of this study are available from the corresponding author upon reasonable request.

Disclosure

Conflicts of Interest

The authors declare no conflicts of interest.

Funding

This work was supported by the UK Engineering and Physical Sciences Research Council (EPSRC) under the Bayesian Tracking and Reasoning over Time (BTaRoT) grant EP/K021516/1 and by the European Commission's Seventh Framework Programme (FP7, 2013-
2017) through the TRAcking in compleX sensor systems (TRAX) project, Grant Agreement No. 607400.

Acknowledgments

We acknowledge the support from the UK Engineering and Physical Sciences Research Council (EPSRC) via the Bayesian Tracking and Reasoning over Time (BTaRoT) (Grant EP/K021516/1) and the [FP7 2013-2017] TRAcking in compleX sensor systems (TRAX) (Grant Agreement No. 607400). For the purpose of open access, the authors have applied a Creative Commons Attribution (CC BY) license to any Author Accepted Manuscript version arising from this submission.

Endnotes

[1]Note that in the MCMC literature, "divide and conquer" typically refers to partition-sample-recombine strategies [35, 36], which differs from the classical algorithmic usage of the term where recursive problem decomposition is essential.

[2]It is also required to amend the MH acceptance accordingly to take the inclusion of the proxy into account.

[3]In the case of an MH kernel, the classic MH kernel is used for the standard SMCMC [28] and confidence MH kernel, as defined in Algorithm 5, for AS-SMCMC.

References

[1] O. J. Reichman, M. B. Jones, and M. P. Schildhauer, "Challenges and Opportunities of Open Data in Ecology," *Science* 331, no. 6018 (2011): 703–705, https://doi.org/10.1126/science.1197962.

[2] B. Ahlgren, M. Hidell, and E. C.-H. Ngai, "Internet of Things for Smart Cities: Interoperability and Open Data," *IEEE Internet Computing* 20, no. 6 (2016): 52–56, https://doi.org/10.1109/MIC.2016.124.

[3] K. P. Seng, L. M. Ang, and E. Ngharamike, "Artificial Intelligence Internet of Things: A New Paradigm of Distributed Sensor Networks," *International Journal of Distributed Sensor Networks* 18, no. 3 (2022): 155014772110628, https://doi.org/10.1177/15501477211062835.

[4] J. Fan, F. Han, and H. Liu, "Challenges of Big Data Analysis," *National Science Review* 1, no. 2 (2014): 293–314, https://doi.org/10.1093/nsr/nwt032.

[5] J. Zhu, J. Chen, W. Hu, and B. Zhang, "Big Learning With Bayesian Methods," *National Science Review* 4, no. 4 (2017): 627–651, https://doi.org/10.1093/nsr/nwx044.

[6] H. Liu, Y.-S. Ong, X. Shen, and J. Cai, "When Gaussian Process Meets Big Data: A Review of Scalable GPs," *IEEE Transactions on Neural Networks and Learning Systems* 31, no. 11 (2020): 4405–4423, https://doi.org/10.1109/TNNLS.2019.2957109.

[7] T. Shao, "Distributed Filtering in Sensor Networks Based on Linear Minimum Mean Square Error Criterion With Limited Sensing Range," *International Journal of Distributed Sensor Networks* 18, no. 7 (2022): 15501329221110810, https://doi.org/10.1177/15501329221110810.

[8] A. C. Harvey, "Applications of the Kalman Filter in Econometrics," in *Advances in Econometrics: Fifth World Congress*, ed. T. F. Bewley (Cambridge University Press, 1987), 285–312, https://doi.org/10.1017/CCOL0521344301.008.

[9] T. M. Deserno, *Biomedical Image Processing* (Springer-Verlag, 2011).

[10] R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems," *Journal of Basic Engineering* 82, no. 1 (1960): 35–45, https://doi.org/10.1115/1.3662552.

[11] O. Cappe, S. J. Godsill, and E. Moulines, "An Overview of Existing Methods and Recent Advances in Sequential Monte Carlo," *Proceedings of the IEEE* 95, no. 5 (2007): 899–924, https://doi.org/10.1109/JPROC.2007.893250.

[12] T. Bengtsson, P. Bickel, and B. Li, "Curse-of-Dimensionality Revisited: Collapse of the Particle Filter in Very Large Scale Systems," *Probability and Statistics: Essays in Honor of David A. Freedman* 2 (2008): 316–334, https://doi.org/10.1214/193940307000000518.

[13] P. J. van Leeuwen, H. R. Künsch, L. Nerger, R. Potthast, and S. Reich, "Particle Filters for High-Dimensional Geoscience Applications: A Review," *Quarterly Journal of the Royal Meteorological Society* 145, no. 723 (2019): 2335–2365, https://doi.org/10.1002/qj.3551.

[14] W. R. Gilks and C. Berzuini, "Following a Moving Target-Monte Carlo Inference for Dynamic Bayesian Models," *Journal of the Royal Statistical Society: Series B* 63, no. 1 (2001): 127–146, https://doi.org/10.1111/1467-9868.00280.

[15] P. Rebeschini and R. van Handel, "Can Local Particle Filters Beat the Curse of Dimensionality?," *Annals of Applied Probability* 25, no. 5 (2015): 2809–2866, https://doi.org/10.1214/14-AAP1061.

[16] T. Li, C. Sbarufatti, and F. Cadini, "Multiple Local Particle Filter for High-Dimensional System Identification," *Mechanical Systems and Signal Processing* 209 (2024): 111060, https://doi.org/10.1016/j.ymssp.2023.111060.

[17] A. Beskos, D. Crisan, A. Jasra, K. Kamatani, and Y. Zhou, "A Stable Particle Filter for a Class of High-Dimensional State-Space Models," *Advances in Applied Probability* 49, no. 1 (2017): 24–48, https://doi.org/10.1017/apr.2016.77.

[18] C. A. Naesseth, F. Lindsten, and T. B. Schön, "High-Dimensional Filtering Using Nested Sequential Monte Carlo," *IEEE Transactions on Signal Processing* 67, no. 16 (2019): 4177–4188, https://doi.org/10.1109/TSP.2019.2926035.

[19] F. R. Crucinio and A. M. Johansen, "A Divide-and-Conquer Sequential Monte Carlo Approach to High Dimensional Filtering," *Statistica Sinica* 34 (2024): 1093–1113.

[20] A. Doucet, S. Godsill, and C. Andrieu, "On Sequential Monte Carlo Sampling Methods for Bayesian Filtering," *Statistics and Computing* 10, no. 3 (2000): 197–208, https://doi.org/10.1023/A:1008935410038.

[21] S. Centanni and M. Minozzo, "A Monte Carlo Approach to Filtering for a Class of Marked Doubly Stochastic Poisson Processes," *Journal of the American Statistical Association* 101, no. 476 (2006): 1582–1597, https://doi.org/10.1198/016214506000000276.

[22] Z. Khan, T. Balch, and F. Dellaert, "MCMC-Based Particle Filtering for Tracking a Variable Number of Interacting Targets," *IEEE Transactions on Pattern Analysis and Machine Intelligence* 27, no. 11 (2005): 1805–1819, https://doi.org/10.1109/TPAMI.2005.223.

[23] F. Septier, S. K. Pang, A. Carmi, and S. Godsill, "On MCMC-Based Particle Methods for Bayesian Filtering: Application to Multitarget Tracking," in *Proceedings of the 2009 3rd IEEE International Workshop on Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP)* (IEEE, 2009), 360–363, https://doi.org/10.1109/CAMSAP.2009.5413256.

[24] R. Lamberti, F. Septier, N. Salman, and L. Mihaylova, "Sequential Markov Chain Monte Carlo for Multi-Target Tracking With Correlated RSS Measurements," in *Proceedings of the 2015 IEEE Tenth International Conference on Intelligent Sensors, Sensor Networks and Information Processing (ISSNIP)* (IEEE, 2015), 1–6, https://doi.org/10.1109/ISSNIP.2015.7106901.

[25] R. Lamberti, F. Septier, N. Salman, and L. Mihaylova, "Gradient-Based Sequential Markov Chain Monte Carlo for Multitarget Tracking With Correlated Measurements," *IEEE Transactions on Signal and Information Processing over Networks* 4, no. 3 (2018): 510–518, https://doi.org/10.1109/TSIPN.2017.2756563.

[26] F. Septier, J. Cornebise, S. Godsill, and Y. Delignon, "A Comparative Study of Monte-Carlo Methods for Multitarget Tracking," in *Proceedings of the 2011 IEEE Statistical Signal Processing Workshop (SSP)* (IEEE, 2011), 205–208, https://doi.org/10.1109/SSP.2011.5967660.

[27] A. Finke, A. Doucet, and A. M. Johansen, "Limit Theorems for Sequential MCMC Methods," *Advances in Applied Probability* 52, no. 2 (2020): 377–403, https://doi.org/10.1017/apr.2020.9.

[28] F. Septier and G. W. Peters, "Langevin and Hamiltonian Based Sequential MCMC for Efficient Bayesian Filtering in High-Dimensional Spaces," *IEEE Journal of Selected Topics in Signal Processing* 10, no. 2 (2016): 312–327, https://doi.org/10.1109/JSTSP.2015.2497211.

[29] Y. Li, S. Pal, and M. J. Coates, "Invertible Particle-Flow-Based Sequential MCMC With Extension to Gaussian Mixture Noise Models," *IEEE Transactions on Signal Processing* 67, no. 9 (2019): 2499–2512, https://doi.org/10.1109/TSP.2019.2905816.

[30] M. Bolic, P. M. Djuric, and S. Hong, "Resampling Algorithms and Architectures for Distributed Particle Filters," *IEEE Transactions on Signal Processing* 53, no. 7 (2005): 2442–2450, https://doi.org/10.1109/TSP.2005.849185.

[31] T. Li, M. Bolic, and P. Djuric, "Resampling Methods for Particle Filtering: Classification, Implementation, and Strategies," *IEEE Signal Processing Magazine* 32, no. 3 (2015): 70–86, https://doi.org/10.1109/MSP.2014.2330626.

[32] J. Read, K. Achutegui, and J. Míguez, "A Distributed Particle Filter for Nonlinear Tracking in Wireless Sensor Networks," *Signal Processing* 98 (2014): 121–134, https://doi.org/10.1016/j.sigpro.2013.11.020.

[33] J. Huggins, T. Campbell, and T. Broderick, "Coresets for Scalable Bayesian Logistic Regression," in *Proceedings of Advances in Neural Information Processing Systems (NIPS 2016)* (Curran Associates, Inc., 2016), https://doi.org/10.48550/arXiv.1605.06423.

[34] S. Winter, T. Campbell, L. Lin, S. Srivastava, and D. B. Dunson, "Emerging Directions in Bayesian Computation," *Statistical Science* 39, no. 1 (2024): 62–89, https://doi.org/10.1214/23-STS919.

[35] C. Vyner, C. Nemeth, and C. Sherlock, "SwISS: A Scalable Markov Chain Monte Carlo Divide-and-Conquer Strategy," *Stat* 12, no. 1 (2023): e523, https://doi.org/10.1002/sta4.523.

[36] C. Trojan, P. Fearnhead, and C. Nemeth, "Diffusion Generative Modelling for Divide-and-Conquer MCMC," 2024): https://doi.org/10.48550/arXiv.2406.11664.

[37] S. L. Scott, A. W. Blocker, F. V. Bonassi, H. A. Chipman, E. I. George, and R. E. McCulloch, "Bayes and Big Data: The Consensus Monte Carlo Algorithm," in *Big Data and Information Theory* (Routledge, 2022), 8–18.

[38] W. Neiswanger, C. Wang, and E. Xing, "Asymptotically Exact, Embarrassingly Parallel MCMC): https://doi.org/10.48550/arXiv.1311.4780.

[39] R. Casarin, R. V. Craiu, and F. Leisen, "Embarrassingly Parallel Sequential Markov-Chain Monte Carlo for Large Sets of Time Series," *Statistics and its Interface* 9, no. 4 (2016): 497–508, https://doi.org/10.4310/SII.2016.v9.n4.a9.

[40] X. Wang and D. B. Dunson, "Parallel MCMC via Weierstrass Sampler," 2014): https://doi.org/10.48550/arXiv.1312.4605.

[41] S. Minsker, S. Srivastava, L. Lin, and D. B. Dunson, "Robust and Scalable Bayes via a Median of Subset Posterior Measures," *Journal of Machine Learning Research* 18, no. 124 (2014): 1–40.

[42] M. Xu, Y. W. Teh, J. Zhu, and B. Zhang, "Distributed Context-Aware Bayesian Posterior Sampling via Expectation Propagation," *Advances in Neural Information Processing Systems* (2014.

[43] A. Vehtari, A. Gelman, T. Sivula, et al., "Expectation Propagation as a Way of Life: A Framework for Bayesian Inference on Partitioned Data," *Journal of Machine Learning Research* 21, no. 17 (2020): 1–53.

[44] D. A. De Souza, D. Mesquita, S. Kaski, and L. Acerbi, "Parallel MCMC Without Embarrassing Failures," in *Proceedings of the 25th International Conference on Artificial Intelligence and Statistics* (PMLR, 2022), 1786–1804.

[45] C. Andrieu and G. O. Roberts, "The Pseudo-Marginal Approach for Efficient Monte Carlo Computations," *Annals of Statistics* 37, no. 2 (2009): 697–725, https://doi.org/10.1214/07-AOS574.

[46] M. Banterle, C. Grazian, A. Lee, and C. P. Robert, "Accelerating Metropolis-Hastings Algorithms by Delayed Acceptance," *Foundations of Data Science* 1, no. 2 (2019): 103–128, https://doi.org/10.3934/fods.2019005.

[47] M. Quiroz, M.-N. Tran, M. Villani, and R. Kohn, "Speeding Up MCMC by Delayed Acceptance and Data Subsampling," *Journal of Computational and Graphical Statistics* 27, no. 1 (2018): 12–22, https://doi.org/10.1080/10618600.2017.1307117.

[48] D. Maclaurin and R. P. Adams, "Firefly Monte Carlo: Exact MCMC With Subsets of Data," 2015): https://doi.org/10.48550/arXiv.1403.5693.

[49] R. Zhang, A. F. Cooper, and C. M. De Sa, "Asymptotically Optimal Exact Minibatch Metropolis-Hastings," *Advances in Neural Information Processing Systems* 33 (2020): 19500–19510.

[50] W. Yuan and G. Wang, "Markov Chain Monte Carlo Without Evaluating the Target: An Auxiliary Variable Approach," 2024): https://doi.org/10.48550/arXiv.2406.05242.

[51] R. Cornish, P. Vanetti, A. Bouchard-Côté, G. Deligiannidis, and A. Doucet, "Scalable Metropolis-Hastings for Exact Bayesian Inference With Large Datasets," in *Proceedings of the 36th International Conference on Machine Learning* (PMLR, 2019), 1351–1360.

[52] E. Prado, C. Nemeth, and C. Sherlock, "Metropolis–Hastings With Scalable Subsampling," 2024): https://doi.org/10.48550/arXiv.2407.19602.

[53] A. Korattikara, Y. Chen, and M. Welling, "Austerity in MCMC Land: Cutting the Metropolis-Hastings Budget," *Proceedings of the 31st International Conference on Machine Learning* 32, no. 1 (2014): 181–189.

[54] R. Bardenet, A. Doucet, and C. Holmes, "Towards Scaling Up Markov Chain Monte Carlo: An Adaptive Subsampling Approach," in *Proceedings of the 31st International Conference on Machine Learning* (PMLR, 2014), 405–413.

[55] R. Bardenet, A. Doucet, and C. Holmes, "On Markov Chain Monte Carlo Methods for Tall Data," *Journal of Machine Learning Research* 18, no. 47 (2017): 1–43, https://doi.org/10.5555/3122009.3153003.

[56] D. Seita, X. Pan, H. Chen, and J. Canny, "An Efficient Minibatch Acceptance Test for Metropolis-Hastings): https://doi.org/10.48550/arXiv.1610.06848.

[57] F. Maire, N. Friel, and P. Alquier, "Informed Sub-Sampling MCMC: Approximate Bayesian Inference for Large Datasets," *Statistics and Computing* 29, no. 3 (2019): 449–482, https://doi.org/10.1007/s11222-018-9817-3.

[58] P. Del Moral, A. Doucet, and A. Jasra, "An Adaptive Sequential Monte Carlo Method for Approximate Bayesian Computation," *Statistics and Computing* 22, no. 5 (2012): 1009–1020, https://doi.org/10.1007/s11222-011-9271-y.

[59] O. Cappé, A. Guillin, J. M. Marin, and C. P. Robert, "Population Monte Carlo," *Journal of Computational and Graphical Statistics* 13, no. 4 (2004): 907–929, https://doi.org/10.1198/106186004X12803.

[60] L. Martino, V. Elvira, D. Luengo, and J. Corander, "Layered Adaptive Importance Sampling," *Statistics and Computing* 27, no. 3 (2017): 599–623, https://doi.org/10.1007/s11222-016-9642-5.

[61] V. Elvira, L. Martino, D. Luengo, and M. Bugallo, "Improving Population Monte Carlo: Alternative Weighting and Resampling Schemes," *Signal Processing* 131 (2017): 77–91, https://doi.org/10.1016/j.sigpro.2016.07.012.

[62] Y. Yang and D. Dunson, "Sequential Markov Chain Monte Carlo," 2013): https://doi.org/10.48550/arXiv.1308.3861.

[63] A. De Freitas, F. Septier, L. Mihaylova, and S. Godsill, "How Can Subsampling Reduce Complexity in Sequential MCMC Methods and Deal With Big Data in Target Tracking?," in *Proceedings of the 18th International Conference on Information Fusion (Fusion)* (IEEE, 2015), 134–141.

[64] A. De Freitas, *Sequential Monte Carlo Methods for Crowd and Extended Object Tracking and Dealing With Tall Data* (PhD diss., University of Sheffield, 2017).

[65] T. P. Minka, *A Family of Algorithms for Approximate Bayesian Inference* (PhD diss, Massachusetts Institute of Technology, 2001).

[66] C. M. Bishop, *Pattern Recognition and Machine Learning* (Springer-Verlag, 2006).

[67] L. Mihaylova, A. Y. Carmi, F. Septier, A. Gning, S. K. Pang, and S. Godsill, "Overview of Bayesian Sequential Monte Carlo Methods for Group and Extended Object Tracking," *Digital Signal Processing* 25, no. 1 (2014): 1–16, https://doi.org/10.1016/j.dsp.2013.11.006.

[68] V. Mnih, C. Szepesvári, and J.-Y. Audibert, "Empirical Bernstein Stopping," in *Proceedings of the 25th International Conference on Machine Learning* (Association for Computing Machinery, 2008), 672–679, https://doi.org/10.1145/1390156.1390241.

[69] A. Gelman and D. B. Rubin, "Inference From Iterative Simulation Using Multiple Sequences," *Statistical Science* 7, no. 4 (1992): 457–511, https://doi.org/10.1214/ss/1177011136.

[70] M. Betancourt, "A General Metric for Riemannian Manifold Hamiltonian Monte Carlo," in *International Conference on Geometric Science of Information* (Springer, 2013), 327–334.

[71] A. Gning and L. Mihaylova, "Dynamic Clustering and Belief Propagation for Distributed Inference in Random Sensor Networks With Deficient Links," in *Proceedings of the 12th International Conference on Information Fusion* (IEEE, 2009), 656–663.

[72] X. Liu, L. Mihaylova, J. George, and T. Pham, "Gaussian Process Upper Confidence Bounds in Distributed Point Target Tracking Over Wireless Sensor Networks," *IEEE Journal of Selected Topics in Signal Processing* 17, no. 1 (2023): 295–310, https://doi.org/10.1109/JSTSP.2022.3223521.

[73] K. Gilholm and D. Salmond, "Spatial Distribution Model for Tracking Extended Objects," *IEE Proceedings-Radar, Sonar and Navigation* 152, no. 5 (2005): 364–371, https://doi.org/10.1049/ip-rsn:20045114.