University of Tennessee, Knoxville

TRACE: Tennessee Research and Creative Exchange

Doctoral Dissertations

Graduate School

8-2016

Advanced sequential Monte Carlo methods and their applications to sparse sensor network for detection and estimation

Kai Kang
University of Tennessee, Knoxville, kkang2@vols.utk.edu

Follow this and additional works at: https://trace.tennessee.edu/utk_graddiss

 Part of the Applied Statistics Commons, Probability Commons, and the Statistical Theory Commons

Recommended Citation

Kang, Kai, "Advanced sequential Monte Carlo methods and their applications to sparse sensor network for detection and estimation. " PhD diss., University of Tennessee, 2016.
https://trace.tennessee.edu/utk_graddiss/3933

This Dissertation is brought to you for free and open access by the Graduate School at TRACE: Tennessee Research and Creative Exchange. It has been accepted for inclusion in Doctoral Dissertations by an authorized administrator of TRACE: Tennessee Research and Creative Exchange. For more information, please contact trace@utk.edu.

To the Graduate Council:

I am submitting herewith a dissertation written by Kai Kang entitled "Advanced sequential Monte Carlo methods and their applications to sparse sensor network for detection and estimation." I have examined the final electronic copy of this dissertation for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Doctor of Philosophy, with a major in Mathematics.

Vasileios Maroulas, Major Professor

We have read this dissertation and recommend its acceptance:

Michael Berry, Kody Law, Russell Zaretzki

Accepted for the Council:

Carolyn R. Hodges

Vice Provost and Dean of the Graduate School

(Original signatures are on file with official student records.)

Advanced sequential Monte Carlo methods and their applications to sparse sensor network for detection and estimation

A Dissertation Presented for the

Doctor of Philosophy

Degree

The University of Tennessee, Knoxville

Kai Kang

August 2016

© by Kai Kang, 2016
All Rights Reserved.

To my beloved parents, Kang, Jian and Qin, Ping.

Acknowledgements

I would like to express my deep appreciation for my advisor Professor Vasileios Maroulas. You have been a tremendous mentor for me. I would like to thank you for encouraging my research and for guiding me to grow as a research scientist. Your advice on both my research as well as on my career have been invaluable. I would also like to thank Professor Michael Berry, Professor Kody Law, Professor Russell Zaretzki for serving as my committee members. I also want to thank you for letting my defense be an enjoyable moment, and for your brilliant comments and suggestions. Thanks to you. Furthermore, Professor Maroulas's financial support from Air Force Office of Scientific Research and Simons Foundation is gratefully acknowledged.

A special thanks to my family. Words can not express how grateful I am to my mother and my father for all of the sacrifices that you've made on my behalf. Your prayer for me was what sustained me thus far. Moreover, I would like to thank my colleagues and friends Andrew Marchese, Xiaoyang Pan, Liguo Wang, Fei Xing, Ernest Jum, Eddie Tu and Joshua Mike. Thanks to you for the fruitful discussions on so many interesting problems which have always been joyful.

Abstract

The general state space models present a flexible framework for modeling dynamic systems and therefore have vast applications in many disciplines such as engineering, economics, biology, etc. However, optimal estimation problems of non-linear non-Gaussian state space models are analytically intractable in general. Sequential Monte Carlo (SMC) methods become a very popular class of simulation-based methods for the solution of optimal estimation problems. The advantages of SMC methods in comparison with classical filtering methods such as Kalman Filter and Extended Kalman Filter are that they are able to handle non-linear non-Gaussian scenarios without relying on any local linearization techniques. In this thesis, we present an advanced SMC method and the study of its asymptotic behavior. We apply the proposed SMC method in a target tracking problem using different observation models. Specifically, a distributed SMC algorithm is developed for a wireless sensor network (WSN) that incorporates with an informative-sensor detection technique. The novel SMC algorithm is designed to surmount the degeneracy problem by employing a multilevel Markov chain Monte Carlo (MCMC) procedure constructed by engaging drift homotopy and likelihood bridging techniques. The observations are gathered only from the informative sensors, which are sensing useful observations of the nearby moving targets. The detection of those informative sensors, which are typically a small portion of the WSN, is taking place by using a sparsity-aware matrix decomposition technique. Simulation results showcase that our algorithm outperforms

current popular tracking algorithms such as bootstrap filter and auxiliary particle filter in many scenarios.

Table of Contents

1 Introduction **1**

2 Review of existing methods **7**

2.1 Problem formulation 7

2.2 Sparsity methods 10

2.3 Monte Carlo Filtering methods 17

2.3.1 Hidden Markov Model 17

2.3.2 Monte Carlo and Importance Sampling Methods 22

2.3.3 Sequential Monte Carlo Methods 25

2.3.4 Importance Density 28

3 A novel methodology **31**

3.1 Introduction 31

3.2 Detecting dynamic objects 32

3.2.1 Analysis of Data Covariance 32

3.2.2 Sparse Data Association 34

3.3 An Advanced Sequential Monte Carlo method 37

3.3.1 Drift homotopy likelihood bridging particle filter 37

3.3.2 The MCMC sampler: Generalized hybrid Monte Carlo 41

3.4 A theoretical treatment 46

3.4.1 Drift Homotopy and Likelihood Bridging Particle Filtering . . 47

3.4.2 Almost Sure Convergence of the DHLB particle filter 48

4 Numerical applications **63**

4.1 Double well potential dynamics 63

4.1.1 Double well potential with rugged energy landscape 63

4.2 Multi-target tracking problem 69

4.2.1 Multi-target tracking in a wireless sensor network without fusion center 69

4.3 A learning parameter in DHLB-PF 76

4.3.1 Smooth double well potential dynamics 77

4.3.2 Case 1: Linear Gaussian model 80

4.3.3 Case 2: Nonlinear non-Gaussian model 81

Bibliography **86**

Vita **99**

Chapter 1

Introduction

The deployment of sensor networks allows the collection and distributed processing of information in challenging environments whose structure is unknown and is dynamically varying with time. In such environments, the network itself, as well as humans, are prone to spatiotemporally unpredictable threats that may be generated due to malicious attacks, functional failures and even human errors. Thus, effective and rapid detection and tracking of such threats are essential.

However, multi-object identification and trajectory estimation require first a robust association of sensors information with targets across space and time. Targets present in the sensing field affect only a small portion of the deployed wireless sensor networks (WSNs). Thus, given the limited resources, it is pertinent to identify the sensors that acquire information-bearing observations about the targets and use only those which provide such information. Many existing tracking techniques require all sensors to be active Ahmed et al. (2010); Hlinka et al. (2013); Olfati-Saber (2005); Ozdemir et al. (2009); Zhu et al. (2009) or at least multiple local fusion centers are required for data processing Coates (2004) which may be resource-consuming given the locality of the objects affecting only a small number of sensors.

To this end, a distributed algorithmic framework is developed here that associates targets with the sensors which acquire informative measurements about these targets,

and subsequently performs tracking using only these informative sensors. Specifically, sensors which are positioned close to the same target acquire data measurements that tend to be correlated. Such correlations induce a sparse structure in the sensor data covariance matrix. Sparsity is an attribute found in many natural and man-made signals, and it has been exploited in a wide range of applications including sparse regression, sub-Nyquist sampling and statistical inference, e.g., see Candès et al. (2006); Tibshirani (1996a).

To facilitate association of sensor measurements with targets a pertinent framework is derived to analyze the sensor data covariance into sparse factors whose support will indicate subsets of sensors sensing the same target. Different from Guan et al. (2012); Hoyer (2004a); Lee and Seung (2001); Lin (2007a); Ulfarsson and Solo (2008); Zou et al. (2006), the matrix factorization scheme developed here does not impose structural requirements to the unknown factors such as orthogonality and/or positivity of the factor entries. Covariance sparse factorization was also proposed in Schizas (2013) to identify sensing units acquiring information about static sources in a monitored field. However, the work in Schizas (2013) is dealing with stationary settings where the sources of information present in the field are immobile, while the assumed linearity in the data models is very restrictive.

Once the information of pertinent sensors has been acquired based on the support of the sparse factors which are obtained via sparsity-aware data covariance matrix decomposition, the next step of our framework is to estimate the objects' trajectories. This task involves the reconstruction of some unknown state quantities of the system of interest from noisy observations. We call this type of problems the filtering problem. In most cases, a mathematical model (typically presented in the form of stochastic or deterministic partial differential equations) that describes the dynamics of system and the model of observation are available. Therefore, the prior knowledge about the system and likelihood function provided through an observation model allow us to formulate the problem into Bayesian framework, and all the statistical inference on the unknown state can be obtained based on the posterior distribution

in the light of Bayesian theory. In many scenarios, observations arrive sequentially in time (as in the WSN environment the active sensors keep sensing their nearby field and obtain new measurements) and we need to perform the statistical inference on-line, hence it motivates the development of mathematical methods that can update the posterior distribution sequentially in time as the latest observation becomes available. A recursive filtering scheme provides the mechanism that processes the observations sequentially using only most current measurement. There is no need to store all the available data in the sensors and reprocess them every time in the computation when new observation arrives. This computational simplicity is another motivation for sequential (recursive) methods. Such a task of estimating the state of a system that varies in time based on the observations is common in many disciplines and applications other than target tracking, including finance, economics, pollution monitoring, communications, terrain referenced navigation, audio engineering, see the partial list, Kang and Maroulas (2013); Kang et al. (2014); Liu (2008); Mahler and Maroulas (2013); Maroulas and Nebenfuhr (2015); Maroulas et al. (2015); Hamilton and Susmel (1994); Law et al. (2015); Rabiner and Juang (1993); Jelinek (1997); Hamilton (1989); Hamilton and Raj (2013); Kim et al. (1999, 1998); Bunke and Caelli (2001); Koski (2001).

Statistically, tracking of objects boils down to estimate the posterior conditional distribution, $p(x^i_{0:k}|y^j_{1:k})$, for each target i with state history $x^i_{0:k} = (x^i_0, \ldots, x^i_k)$, and associated observational history $y^j_{1:k} = (y^j_1, \ldots, y^j_k)$. The objects' state vector is related to pertinent dynamics expressed by a stochastic differential equation (SDE). The associated data history $y^j_{1:k}$ is given by the observation process which provides the likelihood.

There are many proposed methods that solve different types of filtering problems. In the case where the system and observation models are linear and noise is distributed according to a Gaussian distribution, an analytic solution of the posterior distribution is tractable and optimal using the Kalman filter Kalman (1960); Welch and Bishop (2006) which provides analytic formulation for updating the mean and covariance

recursively. If the system model is partially observed, finite state-space Markov model, the Hidden Markov Model filter is possible to provide an analytic solution Cappé et al. (2005); Smith et al. (2013). However, there are only a few cases where the analytic solutions are tractable. The real system is oftentimes much more complicated and typically contains non-Gaussian noise and nonlinear dynamics in which cases it is almost impossible to compute the analytic solution of the posterior distribution. Many approximation schemes are therefore developed over the last few decades. For example, the extended Kalman filter Bar-Shalom et al. (2004); Ristic et al. (2004) uses Taylor series expansion to linearize the nonlinear functions and approximates the posterior by a Gaussian density, therefore performance of EKF will be highly degraded in the cases that exist severe nonlinearity and non-Gaussianity. Since in most practical scenarios, the posterior distribution is much more complex than a Gaussian distribution, the Gaussian sum filter Sorenson and Alspach (1971) attempts to approximate the posterior distribution by a Gaussian mixture model (GM) which is a weighted sum of Gaussian densities. Ideally, the approximation can be made as accurate as required by choosing appropriate weights and components; however, it is not a trivial task to operate it on-line and in some cases the components of GM can grow exponentially Ristic et al. (2004). Another class of nonlinear filters employs statistical linearization techniques (different from Taylor expansion) to approximate the posterior distribution by a Gaussian distribution. For example, the unscented Kalman filter Julier et al. (1995); Wan and Van Der Merwe (2000) uses unscented transformation and choose a set of samples to determine the mean and covariance of the posterior. These filters play a crucial role in the target tracking scenario where they carry out the state estimation of moving targets recursively in time. Relying on these filtering methods, there are a plethora of strategies which address the multi-target tracking problem, for example see the partial list of Baum and Hanebeck (2013, 2012); Kang et al. (2014); Maroulas et al. (2015); Liu and Chen (1998); Mahler (2007); Maroulas and Stinis (2012); Vo et al. (2005); Mahler and Maroulas (2013); Maroulas and Nebenfuhr (2015) and references therein. The aforementioned linear/non-linear

filters are to a certain extent very successful; nevertheless, they are also very limited due to the restrictive assumptions (such as linearity and Gaussian posterior) and can present poor performances in large amount of real applications where substantial nonlinearity, non-Gaussianity and high-dimensionality exhibited in the systems of interest.

Sequential Monte Carlo (SMC) methods, or equivalently particle filters Cappé et al. (2005); Smith et al. (2013), are a set of simulation-based methods that present very attractive advances in computing the posterior distribution when the system is highly non-linear and posterior is severely non-Gaussian. The essential idea of particle filter is to use a set of weighted particles (samples) to form an empirical distribution that estimates the posterior distribution and the accuracy of the approximation only depends on the number of samples Crisan and Doucet (2002). Therefore, it is a powerful alternative in the case of substantial nonlinearity and non-Gaussianity.

Although it has been a popular approach, it may degenerate if several time steps and multiple targets are involved, or it may require an extremely large number of particles. The reason is that typically only a few samples have dominant weights and the rest of weights is close to zero. It has actually been shown in Snyder et al. (2008) that only one sample carries the approximation and has a nonzero weight for high dimensional problems. Indeed, the problem of degeneracy carries over in the multi-target tracking problem and when it comes to tracking in a WSN then few samples need to be considered due to stringent energy capabilities of sensors. To this end, the tracking process here is carried out via, what we call, a drift homotopy likelihood bridging particle filter algorithm (DHLB-PF). We engage the idea of appending an extra Markov chain Monte Carlo (MCMC) step Berzuini and Gilks (2001); Gilks and Berzuini (2001); Weare (2009) after the resampling step which aims to move the particles to statistically significant regions. At the modified resampling step, particles are resampled based not only on the current observation (as in the standard resampling technique) but also according to the previous one. The idea is that highly weighted "children" particles have been

produced by highly weighted "parent" particles. On the other hand, at the same time, the nature of the posterior distribution needs to be preserved. Consequently, at the appended MCMC step, the drift homotopy likelihood bridging (DHLB) technique considers a two-fold simultaneous approach. On one hand, a sequence of SDEs with modified drifts that interpolate between the original and modified driftsis considered. These SDEs correspond to modified dynamics of the objects in the WSN, thus a sequence of appropriate transition densities, $f_\ell(x_k|x_{k-1}),\ \ell = 0, \ldots, L$, is taken into account. On the other hand, at the same time, we consider a sequence of likelihood densities, $g^m(y_k|y_{k-1}),\ 0 \leq m \leq 1$. The interpolation of dynamics and likelihood bridgingengage multiple levels, $\ell,\ m$ for which at $\ell, m = 0$ the modified dynamics and the uniform likelihoodare, in effect, as opposed at $\ell = L, m = 1$ where the original transition density (corresponding to the original SDE which represents the dynamics of objects) and likelihood density are. All levels from 0 to $L-1$ are auxiliary and aim to facilitate the MCMC sampling that may start with a very poor initial condition. Specifically, using an appropriate MCMC scheme, one samples particles at level ℓ with an initial condition the particles at level $\ell - 1$. The DHLBtechnique allows us to gradually morph a particle with a low weight to a particle with a significant weight while at the same time respecting the nature of the posterior distribution.

The thesis is organized as follows. A review of existing methods is given in chapter 2 discussing popular sensor scheduling schemes and sequential Monte Caro filters. In chapter 3, we present a novel distributed tracking algorithm comprising a informative sensor detection method and a new particle filter, also a almost surely convergence of the particle filter is shown. The last chapter exhibits extensive numerical experiments showing convincing performance of the DHLB-PF algorithm.

Chapter 2

Review of existing methods

2.1 Problem formulation

Consider a wireless sensor network (WSN) with a total number of p sensors. Each sensor is able to communicate with its neighboring sensors which are within its communication range. The immediate neighborhood for sensor ρ, $\rho = 1, \cdots, p$, is denoted by $\mathcal{A}_\rho$, while the entire WSN is modeled as an undirected graph whose inter-sensor links are symmetric. The connectivity information of the WSN is summarized by a $p \times p$ adjacency matrix A whose (ρ, ρ')th entry will be 1 if sensors ρ and ρ' are connected and zero otherwise. Sensors monitor a field on which an unknown and time-varying number of multiple moving targets is present. The targets on the field are sensed via measurements, denoted by $y_\rho(k)$, acquired at sensor ρ at time instant k. For example, as shown in Fig.2.1, each sensor can commute with its nearby neighbors (connected by red lines) and objects s_1, s_2 in the field are sensed by only a small portion of the sensor network in the colored regions. Moreover, we consider that only one moving object, say the jth target, is close to sensor ρ whereas the rest are sufficiently far thus their contribution in $y_\rho(k)$ is very small. This can be realized when targets are sufficiently separated in space such that no more than one target is positioned inside the sensing region of a sensor. As the sensing range (and sensing

region) of the sensors is reduced, objects may be located closer while ensuring the previous assumption of one dominant target.

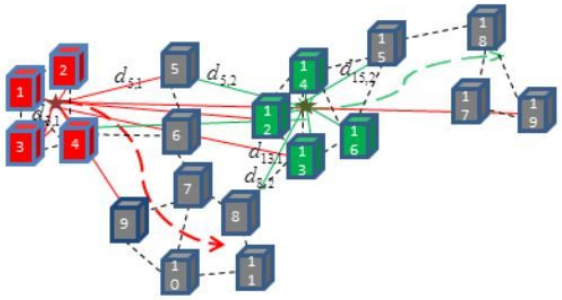

Figure 2.1: A sensor network observing a field. Figure courtesy from Ren et al. (2015).

Let's assume at time instant k there are M_k objects in the sensing field of the WSN. The observation model for the local measurements acquired at an arbitrary sensor $\rho, \rho = 1, \cdots, p$ is given as

$$y_\rho(k) = \sum_{m=1}^{M_k} b_{\rho m}(k) s_m(k) + w_\rho(k), \tag{2.1}$$

where $s_m(k)$ denotes the intensity of a signal emitted from target m, and $b_{\rho m}(k) = d_{\rho m}^{-2}(k)$ quantifies the signal attenuation due to wireless transmission, where $d_{\rho m}(k)$ is the Euclidean distance between a sensor ρ and a target m at time instant k. The driving noise, $w_\rho(k)$, denotes zero-mean white sensing noise with variance equal to σ_w^2. After fusing all sensor measurements into a vector, eq. (2.1) can be summarized in a regression form

$$\mathbf{y}_k = \mathcal{B}_k \mathbf{s}_k + \mathbf{w}_k, \tag{2.2}$$

where $\mathbf{s}_k = [s_1(k), \cdots, s_{M_k}(k)]^T$ contains the intensities of signals emitted by all targets at time instant k. The noise $\mathbf{w}_k$ has covariance $\sigma_w^2 I$. The vector $\mathbf{y}_k =$

$[y_1(k), \cdots, y_p(k)]^T$ holds all the observations of the p sensors, and $\mathcal{B}_k = [b_1, \cdots, b_{M_k}]$ is a $p \times M_k$ matrix that corresponds to the unknown regression vectors. Let $b_m = [b_{1m}, \cdots, b_{pm}]^T$ correspond to the mth column of $\mathcal{B}_k$, which contains the attenuation a target signal is experiencing to reach all p sensors in the network. The entries in b_m with large amplitudes (i.e. close distance between a sensor and an object) indicate the informative sensors, while the entries with small amplitudes imply otherwise. Since targets in practice affectonly a small number of sensors, namely the ones close to the vicinity of a target, the regression vector b_m is expected to be sparse, i.e., only a few of the entries are of relatively large amplitudes and the rest of the entries are close to zero. Based on the measurement model of eq.(2.2), and given that the entries of $\mathbf{s}_k$ are uncorrelated, the data covariance can be obtained by the following formula

$$\Sigma_{y,k} = \mathcal{B}_k D_s {\mathcal{B}_k}^T + \sigma_w^2 I_p = \bar{H}_k {\bar{H}_k}^T + \sigma_w^2 I_p, \tag{2.3}$$

where D_s is the diagonal covariance matrix of the source vector s_k, namely, $D_s = diag(\sigma_1^2, \cdots, \sigma_{M_k}^2)$, and $\bar{H}_k = \mathcal{B}_k D_s^{1/2}$. Let us define

$$M_{y,k} = \Sigma_{y,k} - \sigma_w^2 I_p = \bar{H}_k {\bar{H}_k}^T. \tag{2.4}$$

Notice that $\mathcal{B}_k$ is a sparse matrix and D_s is diagonal matrix, thus $\bar{H}_k = [\bar{h}_{1,k}, \cdots, \bar{h}_{M_k,k}]$ is also a sparse matrix that has the same sparsity structure with $\mathcal{B}_k$. Hence, the large amplitude entries of $\bar{H}_t$ also indicate the informative sensors that are close to the moving targets at time instant k. Therefore, according to eq. (2.4), if the matrix $M_{y,k}$ can be decomposed as the product of two sparse matrices, then it is possible to identify the set of informative sensors, as well as associate targets with sensor observations which plays a crucial role in the likelihood function related to trajectory estimation of the objects in the WSN. Once the informative sensors have been recovered, the second phase proceeds with tracking the objects. What follows next is a survey of pertinent matrix decomposition methods involving certain constrains, such as sparsity, as well

as outline of sequential Monte Carlo techniques which will be employed in the tracking stage.

2.2 Sparsity methods

The first yet crucial component of the distributed tracking algorithm is to identify the sensors that acquire informative measurements about sources of interest. There are a number of existing methods that address the related problem of sensor selection in WSNs. For example, the algorithm proposed in Gupta et al. (2006) optimized the sensor scheduling problem while tracking the moving targets at each time instant. It chooses the sensor randomly according to some probability distribution which is chosen so as to minimize the expected steady-state error covariance. The scheme was developed based on Kalman filter and Riccati recursion, therefore it is limited to linear Gaussian dynamics and Gaussian posterior distribution. It is not straightforward to generalize such scheme to a more general case, for example, when it has no restrictions on the distributional properties of the system noises. Since communication power consumption accounts for 70% of the total power in a sensor network, some sensor selection techniques are developed based on power constraints. For example, the work by Thatte and Mitra (2008) presented a method for sensor selection which focuses on minimizing the power used for communication within SN rather than considering resources used for sensing and data processing. The sensor network considered in Thatte and Mitra (2008) consists of a fusion center with different types of topologies, such as star topology, branch topology and linear topology, see Fig.2.2.

Another interesting sensor activation control algorithm was proposed by Krishnamurthy et al. (2008), where the decision is made by taking into account an activation control utility function. This cost function, based on which both modes of activation and sleeping are evaluated, takes into consideration of the measurement contribution as well as the power consumption of each sensor. It employs the standard game theoretic framework to define the utility function and controls the sensor activation

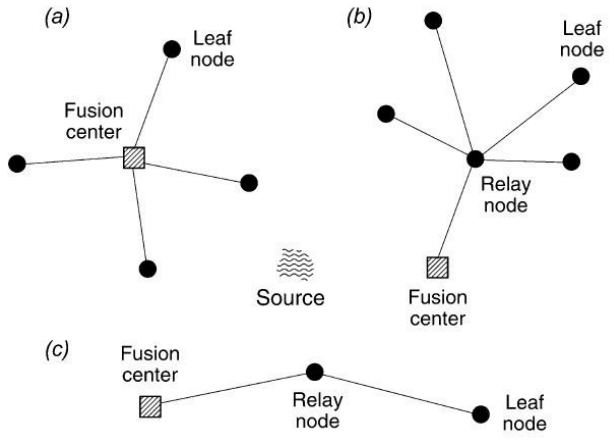

Figure 2.2: The sensor network topology: (a) star, (b) branch, (c) linear topology. Figure Courtesy of Thatte and Mitra (2008).

by considering the correlated equilibria. Convex optimization technique has also been employed for sensor selection problem. As proposed by Joshi and Boyd (2009), the selection of sensors is based on the maximum-likelihood estimate of state which is represented in terms of sensor measurements. The quality of estimation can be written as the volume of the $\eta-$confidence ellipsoid, then the subset of sensors is chosen to minimize the log volume of the resulting confidence ellipsoid. In the work of Fuemmeler and Veeravalli (2010), sleeping strategies for sensors are designed based on the assumption that the sensing ranges of the sensors completely cover the region of interest with no overlap. Each sensor can be in one of the two states: awake or asleep. Further, object sensing can be operated only in the awake state. The dynamics of moving object to be tracked is described by a first-order discrete Markov chain whose state space consists of $n + 1$ state, where n denotes the number of sensors and the $(n + 1)$th state means the object is out of the sensing range. The design of sleeping policies is developed based on the tracking and predicting locations of objects.

Under a relatively simple assumption which is that the distribution of the measurement vector is assumed Gaussian under two hypothesis (event occurring, event not occurring), the work by Bajović et al. (2011) proposed a method of selecting sensor subset by considering the Kullback-Leibler(KL) distance of the probability distributions of the measurement vectors, however a fusion center is also assumed in their framework.

Sensors that are located close to a target acquire data measurements that tend to be correlated, and it turns out that the covariance matrix of the sensor measurements can be analyzed into sparse factors. The problem of determining the measurement-informative sensors will boil down to the task of decomposing the data covariance matrix into sparse factors and locate their support entries. Related decomposition techniques have been studied in the literature.

In fact, the nonnegative matrix and tensor factorization have been a great interest in matrix factorization since it provides hidden components with physical or physiological meaning and interpretations. Nonnegative matrix factorization (NMF) and its extension to 3D nonnegative tensor factorization (NTF) aim to reconstruct latent nonnegative common structures from typically redundant raw data. These techniques have vast applications in data analysis such as pattern recognition, segmentation, clustering, dimensionality reduction, text mining, signal and image processing and gene separation Lee and Seung (1999); Berry et al. (2007); Sajda et al. (2004); Cichocki et al. (2006); Brunet et al. (2004).

The following nonnegative constrained linear form has been used as the underlying model in NMF

$$Y = WH + N, \tag{2.5}$$

where $Y = (y_{ik}) \in \mathbb{R}^{I\times K}$ is a matrix of observations, $W = (w_{ij}) \in \mathbb{R}_+^{I\times J}$ is an unknown nonnegative basis matrix, $H = (h_{jk}) \in \mathbb{R}_+^{J\times K}$ is matrix of unknown hidden nonnegative components, and $N = (n_{ik}) \in \mathbb{R}^{I\times K}$ denotes a matrix of noise or errors, where the notation $\mathbb{R}_+$ denotes the nonnegative subspace. The low rank J is assumed

to be known, or it can be easily estimated via singular value decomposition (SVD). The decomposition can be achieved by alternating minimization of a suitable cost function, or optionally a set of cost functions, subject to nonnegativity constraints. The cost function can be typically determined by a prior knowledge on statistical distribution of noise. Also, other natural constraints such as sparsity, smoothness are often assumed in the decomposition process.

For normally distributed noisy perturbations, the cost function is usually the regularized squared Euclidean distance given in the following form,

$$D_F(Y||WH) = \frac{1}{2}\|Y - WH\|_F^2 + \alpha_W J_W(W) + \alpha_H J_H(H), w_{ij} \geq 0, h_{jk} \geq 0, \forall i, j, k \tag{2.6}$$

where $\|\cdot\|_F$ denotes the Frobenius norm, α_W and α_H are nonnegative regularization parameters, and the terms $J_W(W)$, and $J_H(H)$ are used to impose certain characteristics.

For non-Gaussian distributed noise we can use the $\alpha-$ or $\beta-$divergence Cichocki et al. (2006). For instance, the $\alpha-$divergence can be expressed as

$$D_W^{\alpha}(Y||WH) = \frac{1}{\alpha(\alpha-1)} \sum_{ik} \left(y_{ik}^{\alpha} (WH)_{ik}^{1-\alpha} - \alpha y_{ik} + (\alpha - 1)(WH)_{ik} \right), \alpha \neq 0 \tag{2.7}$$

where $(WH)_{ik}$ denotes the ikth entry of WH.

The following methods are typically used for performing the alternating minimization of a given cost function (such as eq.(2.6) and eq.(2.7)) , multiplicative update rules, projected gradient, fixed point alternating least square(ALS), and quasi-Newton.

The multiplicative update rules are the most commonly used algorithms for optimizing the cost function Lee and Seung (1999); Berry et al. (2007). For instance, by applying the standard gradient descent technique to the cost function and selecting suitable learning rates we obtain the following algorithm. The entry $w_{i,j}$ is updated recursively by $w_{ij} \frac{[(YH^T)_{ij} - \alpha_W \Phi_W(w_{ij})]_+}{(WHH^T)_{ij} + \epsilon}$ and h_{jk} is updated by

$w_{jk}\frac{[(W^TY)_{jk}-\alpha_H\Phi_H(h_{jk})]_+}{(W^TWH)_{jk}+\epsilon}$, where the operator $[\cdot]_+$ is defined as $[x]_+ = max\{\epsilon, x\}$ for small ϵ, and $\Phi_W(w_{ij}) = \partial J_W(W)/\partial w_{ij}$, $\Phi_H(h_{jk}) = \partial J_H(H)/\partial h_{jk}$. Similarly, we can derive an alternative NMF multiplicative rule for normalized weights by applying gradient descent technique to the $\alpha-$divergence, then the entry w_{ij} is updated by $w_{ij}\left(\frac{\sum_{j=1}^{K}(y_{ik}/(WH)_{ik})^\alpha h_{jk}}{\sum_{k=1}^{K}h_{jk}}\right)^{1/\alpha}$ and h_{jk} is updated by $\left(\frac{\sum_{i=1}^{I}w_{ij}(y_{ik}/(WH)_{ik})^\alpha}{\sum_{k=1}^{K}w_{ij}}\right)^{1/\alpha}$, Multiplicative algorithms are relatively simple and typically parameter free, but with relatively slow convergence speed Cichocki et al. (2007a). A second method for solving the problem is the projected gradient method Lin (2007b). According to this method, the non-negative basis matrix W can be recurrently updated by $W = [W - \eta_W P_W]_+$. Similarly, the updated matrix H is given by $H = [H - \eta_H P_H]_+$, where P_W and P_H are descent directions for W and H. There are several rules for choosing the adaptive learning rates η_W and η_H, see Cichocki et al. (2007b) for details. A third very powerful technique is the fixed point ALS algorithm. Fixed point alternating least squares (FP-ALS) algorithms proposed by Cichocki et al. (2007a) do not directly employ the gradient descent technique but attempt to establish iterative algorithms based on the Karush-Kuhn-Tucker (KKT) conditions. For instance, one can compute the gradients of the cost function and set it equal to zero and we can have the FP-ALS algorithm which updates W by $[(YH^T - \alpha_W\Phi_W(W))(HH^T)^{-1}]_+$ and updates H by $[(W^TW)^{-1}(W^TY - \alpha_H\Phi_H(H))]_+$, where $\Phi_W(W) = \partial J_W(W)/\partial W, \Phi_H(H) = \partial J_H(H)/\partial H$. Instead of minimizing the global cost function we can minimize the set of local cost functions defined as

$$D_F^j(Y^j||w_jH_j) = \frac{1}{2}\|Y^j - w_jh_j\|_F^2 + \alpha_W^j J_W(w_j) + \alpha_H^j J_H(h_j), j = 1\cdots, J, \tag{2.8}$$

where

$$Y^j = Y - \sum_{r\neq j} w_r\bar{h}_r = Y - WH + w_j\bar{h}_j, \tag{2.9}$$

$w_j \in \mathbb{R}_+^{I\times 1}$ are columns of W, $\bar{h}_j \in \mathbb{R}_+^{1\times K}$ are rows of H. Form stationary point and under sparsity constraint, i.e., $J_W(w_j) = \|w_j\|_1$ and $J_H(\bar{h}_j) = \|\bar{h}_j\|_1$, we can obtain a new set of sequential learning rules, the so-called hierarchical ALS (HALS) algorithm Cichocki et al. (2007a) that updates w_j by $\frac{1}{\bar{h}_j\bar{h}_j^T}[Y^j\bar{h}_j^T - \alpha_H^j]_+$ and updates $\bar{h}_j$ by $\frac{1}{w_j^T w_j}[w_j^T Y^j - \alpha_H^j]_+, j = 1, 2, \cdots, J$. Furthermore, the quasi-Newton (QN) method Zdunek and Cichocki (2007); Cichocki et al. (2007a) can be used if H is estimated with solving a highly over-determined system of linear equations :$H^T W^T = Y^T$, i.e., for $K \gg J$. In order to improve the performance of the NMF algorithms and to reduce the risk of getting stuck in local minima of a cost function subject to nonconvex alternating minimization rule, a hierarchical multi-stage procedure combined with multistart initialization has been developed and details of the algorithms can be found in Cichocki et al. (2006, 2007b,a).

A very popular and interesting decomposition algorithm with sparseness constraints was presented by Hoyer (2004b). The algorithm was an extension on the non-negative matrix factorization (NMF) technique. It introduced a measure of sparsity and devised a projected gradient descent algorithm for NMF with sparsity constraints. To impose sparsity constraints on only one matrix W or H, this algorithm uses a multiplicative update rule for updating the counter matrix which suffers from slow convergence. Pascual-Montano et al. (2006) suggested that non-smooth NMF (nsNMF), which was also developed based on multiplicative update rules, outperformed previous sparse NMF variants on their numerical experiments. Pauca et al. (2006) proposed a constrained NMF (CNMF) formulation,

$$\min_{W,H} \|Y - WH\|_F^2 + \alpha\|W\|_F^2 + \beta\|H\|_F^2, s.t. W, H \geq 0 \tag{2.10}$$

where α and β are regularization parameterss. A sparse algorithm using the following least squares,

$$\min_{H} \|Y - WH\|_F^2 + \beta\|H\|_F^2, \tag{2.11}$$

has been used in Pauca et al. (2004); Gao and Church (2005). This algorithm forces negative values in H to be zero in the computation, in which case, no theory guarantees the convergence of the algorithm. However, it is suggested in the literature that L_1-norm based formulations would be more appropriate than L_2-norm based formulations so as to control sparsity Tibshirani (1996b). Kim and Park (2007) proposed sparse NMFs based on alternating non-negative-constrained least squares. They introduced two formulations and the corresponding algorithms for sparse NMFs, i.e. the so-called SNMF/R formulation for sparse H as following

$$\min_{W,H} \frac{1}{2}\{\|Y - WH\|_F^2 + \eta\|W\|_F^2 + \beta \sum_{j=1}^{n} \|H(:,j)\|_1^2\}, s.t. W, H \geq 0, \tag{2.12}$$

where $H(:,j)$ is the jth column vector of H, $\eta > 0$ is a parameter to suppress $\|W\|_F^2$, and $\beta > 0$ is a regularization parameter to balance the trade-off between the accuracy of the approximation and the sparseness of H. The SNMF/L formulation for sparse W is given as follows

$$\min_{W,H} \frac{1}{2}\{\|Y - WH\|_F^2 + \eta\|H\|_F^2 + \alpha \sum_{j=1}^{n} \|W(:,j)\|_1^2\}, s.t. W, H \geq 0, \tag{2.13}$$

where $W(i,:)$ is the ith row vector of W, $\eta > 0$ is a parameter to suppress $\|H\|_F^2$, and $\alpha > 0$ is a regularization parameter to balance the trade-off between the accuracy of the approximation and the sparseness of W. Their formulations imposed the sparsity on a factor of NMF utilize L_1-norm minimization and the corresponding algorithms are based on alternating non-negativity constrained least squares (ANLS) which iterates the following until convergence,

$$\min_{H} \|(W, \sqrt{\beta} e_{1\times k})^T H - (Y, 0_{1\times n})\|_F^2, s.t. H \geq 0 \tag{2.14}$$

where $e_{1\times k} \in \mathbb{R}^{1\times k}$ is a row vector with all components equal to one and $0_{1\times n} \in \mathbb{R}^{1\times n}$ is a zero vector, the motation (A, a) denotes appending the matrix A a vector a and

$$\min_{W} \|(H^T, \sqrt{\eta} I_k)^T W^T - (Y^T, 0_{k\times m})\|_F^2, s.t. W \geq 0 \tag{2.15}$$

where I_k is an $k \times k$ identity matrix and $0_{k\times m}$ is a zero matrix of size $k \times m$. Eq.(2.14) minimizes L_1-norm of the columns of H which imposes sparsity on H. For SNMF/L the algorithm runs in the same fashion.

Also, PCA techniques are also involved in developing the sparse matrix decomposition methods, for example, in d'Aspremont et al. (2007), they proposed a direct approach for sparse principle components analysis by directly incorporating a sparsity criterion in the PCA formulation, then forming a convex relaxation of the problem. The work in Zhou et al. (2012) proposed a fast NMF using low-rank approximation technique to alleviate the problem of slow convergence for NMF algorithms. Since the major bottleneck of the slow convergence is caused by the matrix multiplications with large matrices. Zhou et al. (2012) suggested to replace the large matrices by much smaller ones, and therefore achieved a much lower time complexity and space complexity.

We provided a brief summary on the popular matrix decomposition methods which based on the NMF techniques and also can be used in the case where further constraints, such as sparsity, are required, and next we move to the sequential Monte Carlo filters.

2.3 Monte Carlo Filtering methods

2.3.1 Hidden Markov Model

The available dynamics system model and the observation model in the filtering problems typically form a Hidden Markov Model (abbreviated HMM), or equivalently state-space model. The hidden Markov model is loosely speaking a Markov chain

observed through noisy perturbations. The dynamical model compromises a Markov chain, which we denote by $\{x_k\}_{k\geq 0}$, where $k \in \mathbb{N}$ denotes time instant. We assume this Markov chain take values in some general state space. Nevertheless, the Markov chain is hidden which means the stochastic process $\{x_k\}_{k\geq 0}$ is not observable. Observer has access to another stochastic process $\{y_k\}_{k\geq 0}$ which is linked to the Markov chain such that x_k, for any k, governs the distribution of the corresponding y_k. All statistical inference on the hidden Markov chain $\{x_k\}_{k\geq 0}$ has to be made via $\{y_k\}_{k\geq 0}$ only, as $\{x_k\}_{k\geq 0}$ is not observed.

We first briefly revisit the definitions regarding the Markov chain on a general state space. Assume that two real valued stochastic processes $\{x_k\}_{k\geq 0}$ and $\{y_k\}_{k\geq 0}$ defined on a probability space denoted by $(\Omega, \mathcal{F}, P)$, where Ω is the state space, $\mathcal{F}$ is the $\sigma-$algebra and P is the probability measure, and $x_k \in \mathbb{R}^{n_x}$, and let $(\mathbb{R}^{n_x}, \mathcal{B}(\mathbb{R}^{n_x}))$ be the measurable space, where $\mathcal{B}(\mathbb{R}^{n_x})$ denotes the Borel $\sigma-$algebra over $\mathbb{R}^{n_x}$. We then provide the definition of a transition kernel and a Markov chain in the following

Definition 2.1. *The function $K(\cdot,\cdot) : (\mathbb{R}^{n_x}, \mathcal{B}(\mathbb{R}^{n_x})) \to [0,1]$ is called a Markov kernel or a transition kernel if*

1. *for each $x \in \mathbb{R}^{n_x}$ the mapping $A \in \mathcal{B}(\mathbb{R}^{n_x}) \to K(x, A)$ is a probability measure on $(\mathbb{R}^{n_x}, \mathcal{B}(\mathbb{R}^{n_x}))$,*

2. *for each $A \in \mathcal{B}(\mathbb{R}^{n_x})$ the mapping $x \in \mathbb{R}^{n_x} \to K(x, A)$ is a $\mathcal{B}(\mathbb{R}^{n_x})-$measurable function.*

Definition 2.2. *A sequence of random variables $\{x_k\}_{k\in\mathbb{N}}$ on a probability space $(\Omega, \mathcal{F}, P)$ mapping into $(\mathbb{R}^n, \mathcal{B}(\mathbb{R}^n))$ is called a Markov chain with transition kernel K if for all $n \in \mathbb{N}$ and $A \in \mathcal{B}(\mathbb{R}^n)$ one has*

$$P(x_{k+1} \in A | x_1, \cdots, x_k) = P(x_{k+1} \in A | x_k) = K(x_k, A)$$

almost surely. The distribution

$$\pi_0(A) = P(x_1 \in A), \forall A \in \mathcal{B}(\mathbb{R}^n)$$

is called the initial distribution.

Definition 2.3. *The Markov chain is said to be time-homogeneous if the transition kernel is independent of time instant k.*

To this end, a formal definition of a hidden Markov model can be summarized as follows

Definition 2.4. *A hidden Markov model is a bivariate discrete time process $\{x_k, y_k\}_{k\geq 0}$, where $\{x_k\}_{k\geq 0}$ is a Markov chain and, conditional on $\{x_k\}$, $\{y_k\}_{k\geq 0}$ is a sequence of independent random variables such that the conditional distribution of y_k only depends on x_k.*

A graphical illustration of dependencies is given in Fig.2.3. The distribution of y_k conditionally on the past observations $y_0, \cdots, y_{k-1}$ and the past values of the state $x_0, \cdots, x_k$, is determined by x_k only. We should point out that such graphical illustration is only used to provide an intuitive perspective rather than mathematically rigorousness. Also, the Markov chain we consider is time-homogeneous and the conditional distribution of y_k given x_k does not depend on k either.

More specifically, let us consider the following generic nonlinear dynamical system of interest described in state-space form

- System model

$$x_k = a(x_{k-1}, u_{k-1}) \tag{2.16}$$

- Observation model

$$y_k = b(x_k, v_k) \tag{2.17}$$

Equations (2.16) and (2.17) mean the hidden state x_k and the observations y_k are assumed to be generated by nonlinear functions $a(\cdot)$ and $b(\cdot)$. The state and

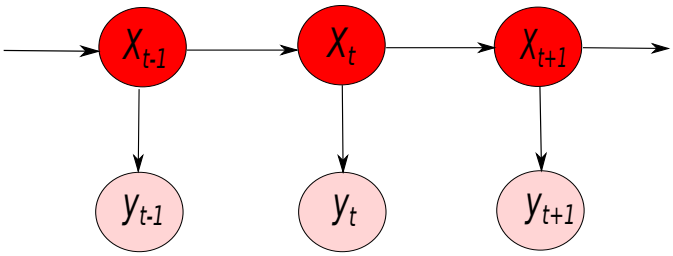

Figure 2.3: Graphical model demonstrating the dependencies (depicted by arrow lines) of hidden Markov model between hidden states x_t and the observation y_t.

observation noise u_k and v_k are assumed to be independent. We make the assumption that the stochastic processes $\{x_k\}$ and $\{y_k\}$ generated by (2.16) and (2.17) form a hidden Markov model. We further assume that the initial state x_0 is distributed according to a known distribution $\pi_0(x_0)$.

The joint probability density of states and observations denoted by $\pi_{0:T,0:T}(x_{0:T}, y_{0:T})$, is given by

$$\pi_{0:T,0:T}(x_{0:T}, y_{0:T}) = \pi_0(x_0)g(y_0|x_0)\prod_{k=1}^{T} f(x_k|x_{k-1})g(y_k|x_k), \tag{2.18}$$

where f is the transition density and g is the likelihood. Therefore, a sample from (2.18) can be simulated according to Alg. 1

Algorithm 1 Generating N i.i.d Random Samples from a hidden Markov Model

1: **Initialization at time** $k = 0$, $\{x_0^i\}_{i=1,\cdots,N} \sim \pi_0(x_0)$, $y_0^i \sim g(y_0|x_0^i), i = 1, \cdots, N$
2: **for** $k = 1 : T$ **do**
3: Sample $x_k^i \sim f(x_k|x_{k-1}^i), i = 1, \cdots, N$
4: Sample $y_k^i \sim g(y_k^i|x_{k-1}^i), i = 1, \cdots, N$
5: **end for**
6: $(x_0^i, \cdots, x_T^i, y_0^i, \cdots, y_T^i), i = 1, \cdots, N$ are N i.i.d samples from $\pi_{0:T,0:T}(x_{0:T}, y_{0:T})$

The statistical inference for the general nonlinear dynamic system involves computing the posterior distribution of a collection of state variables $x_{i:j} = \{x_i, \cdots, x_j\}$ conditioned on a batch of observations, $y_{0:k} = \{y_0, \cdots, y_k\}$, denoted by $\pi_{i:j|0:k}(x_{i:j}|y_{0:k})$. One may be interested in predicting the state x_k based on the past observations, i.e. $j > k$, or in refining the estimation x_k given past, current, or future observations ($j < k$). These are respectively the prediction and smoothing problems. For $i = j = k$, it is the filtering problem. A closed form of the posterior distribution can be computed in only in a very specific cases, for example, the linear Gaussian model and the discrete hidden Markov model. In the vast majority of cases, nonlinearity or non-Gaussianity preclude an analytic solution, e.g., see Ristic et al. (2004); Cappé et al. (2005). For nonlinear dynamic systems, There are also many filtering methods that can be used to make statistical inference. The extended Kalman filter (EKF) and its variants are designed to solve nonlinear filtering problem based on linearization of the state and observation models. However, the EKF is known to perform poorly Ristic et al. (2004) if the system exhibits substantial nonlinearity and if the state and the observation noise are significantly non-Gaussian. Another alternative is the Gaussian sum filter which approximates the posterior distribution by a Gaussian mixture model Alspach and Sorenson (1972); Kulhavỳ (1990). More recently, some grid-based algorithms have been developed that use a set of deterministic points to represent the posterior distribution accurately. For example, the unscented Kalman filter (UKF) which is based on the sigma points Ristic et al. (2004). Whereas these techniques have been applied successfully in certain cases Valverde and Terzija (2011); Chatzi and Smyth (2009); He et al. (2011); Horwood and Poore (2011), they are valid only if the posterior distribution can be closely approximated by a Gaussian distribution. Towards that end, SMC methods have been proposed to bypass the stringent assumptions on the dynamics and posterior distribution.

The Monte Carlo methods for filtering problem have been existing for several decades and the pioneering works can be tracked back to Handschin and Mayne

(1969); Handschin (1970). The bootstrap filter Gordon et al. (1993) was the first practically successful sequential Monte Carlo method for nonlinear filtering and many more similar independent works followed afterwards Kitagawa (1996); Del Moral (1996); Liu and Chen (1998). A key advantage of SMC methods is its great generality which allows statistical inference of the full posterior distribution in general state-space model which can be non-linear and non-Gaussian. The accuracy of the estimation depends only the number of samples. What follows next we discuss the associated sequential Monte Carlo methods starting with the standard Monte Carlo technique.

2.3.2 Monte Carlo and Importance Sampling Methods

In the context of filtering problem, we are interest in computing the posterior distribution at some time instant k, i.e., $\pi_{0:k|0:k}(x_{0:k}|y_{0:k})$. In the light of Monte Carlo methods, we assume that one can generate N identical independently distributed (i.i.d) samples, called particles herein, from the posterior distribution $\pi_{0:k|0:k}(x_{0:k}|y_{0:k})$, i.e. $\{x^i_{0:k}\}_{i=0,\cdots,N}$, then an empirical distribution that approximates the posterior distribution is represented by

$$\pi^N_{0:k|0:k}(dx_{0:k}|y_{0:k}) = \frac{1}{N}\sum_{i=1}^{N}\delta_{x^i_{0:k}}(dx_{0:k}) \tag{2.19}$$

where $\delta_{x^i_{0:k}}(dx_{0:k})$ denotes the delta-Dirac mass located at sample point $x^i_{0:k}$. Then the expectation of $f(x)$ with respect to the posterior distribution, i.e. $\mathbb{E}(f(x))$, is estimated using the empirical measure $\frac{1}{N}\sum_{i=1}^{N} f(x^i_{0:k})$. By Owen (2013), we have the following strong law of large number (SLLN).

Theorem 2.5. *If the variance of $f(x_{0:k})$ is finite, i.e. $\sigma_f^2 < +\infty$, then*

$$\mathbb{E}_N(f(x)) \xrightarrow{a.s.} \mathbb{E}(f(x)), N \to +\infty. \tag{2.20}$$

where $\xrightarrow{a.s.}$ *denotes almost surely convergence.*

A central limit theory also holds, e.g. see Chung (2001), which is summarized below,

Theorem 2.6. *The central limit theory holds if samples are i.i.d and the variance of* $f(x_{0:k})$ *is finite,*

$$\sqrt{N}\left(\mathbb{E}_N(f(x)) - \mathbb{E}(f(x))\right) \xrightarrow{d} \mathcal{N}(0, \sigma_f^2) \tag{2.21}$$

where $\xrightarrow{d}$ *denotes convergence in distribution.*

In the case where we cannot draw i.i.d samples from posterior distribution, we resort an alternative method called importance sampling.

Oftentimes, it is a rather formidable task to draw i.i.d samples from the posterior distribution $\pi_{0:k|0:k}(x_{0:k}|y_{0:k})$. An alternative solution is the importance sampling method, which introduces a proposal distribution, called importance sampling distribution, that differs the posterior distribution but has a similar shape. Let denote the importance sampling distribution by $\tilde{\pi}_{0:k|0:k}(x_{0:k}|y_{0:k})$ and assume the support of $\pi_{0:k|0:k}(x_{0:k}|y_{0:k})$ is a subset of the support of $\tilde{\pi}_{0:k|0:k}(x_{0:k}|y_{0:k})$. Then we have that

$$\mathbb{E}(f(x_{0:k})) = \frac{\int f(x_{0:k}) w(x_{0:k}) \tilde{\pi}_{0:k|0:k}(x_{0:k}|y_{0:k}) dx_{0:k}}{\int w(x_{0:k}) \tilde{\pi}_{0:k|0:k}(x_{0:k}|y_{0:k}) dx_{0:k}}, \tag{2.22}$$

where $w(x_{0:k})$ is called the importance weight which is given by

$$w(x_{0:k}) = \frac{\pi_{0:k|0:k}(x_{0:k}|y_{0:k})}{\tilde{\pi}_{0:k|0:k}(x_{0:k}|y_{0:k})}. \tag{2.23}$$

Eq.(2.22) suggests if we manage to generate N i.i.d samples from $\tilde{\pi}_{0:k|0:k}(x_{0:k}|y_{0:k})$, i.e. $\{x_{0:k}^i, i = 1, \cdots, N\}$, then the expectation (2.22) is approximated by

$$\mathbb{E}_N(f(x_{0:k})) = \frac{\frac{1}{N}\sum_{i=1}^{N} f(x_{0:k}^i) w(x_{0:k}^i)}{\frac{1}{N}\sum_{j=1}^{N} w(x_{0:k}^i)} = \sum_{i=1}^{N} f(x_{0:k}^i) \tilde{w}(x_{0:k}^i), \tag{2.24}$$

where $\tilde{w}(x^i_{0:k})$ is called the normalized importance weights given by

$$\tilde{w}(x^i_{0:k}) = \frac{w(x^i_{0:k})}{\sum_{j=1}^{N} w(x^j_{0:k})}. \tag{2.25}$$

The importance sampling also follows a SLLN and CLT, e.g. see Owen (2013); Cappé et al. (2005), given below,

Theorem 2.7. *Suppose $\tilde{\pi}_{0:k|0:k}(x_{0:k}|y_{0:k})$ is a probability density function with $\tilde{\pi}_{0:k|0:k}(x_{0:k}|y_{0:k}) > 0$ whenever $\pi_{0:k|0:k}(x_{0:k}|y_{0:k}) > 0$. Let $\{x^i_{0:k}, i = 1, \cdots, N\}$ be i.i.d samples follows $\tilde{\pi}_{0:k|0:k}(x_{0:k}|y_{0:k})$, and $\mathbb{E}_N(f(x_{0:k}))$ the self-normalized importance sampling estimate given by eq.(2.24), then*

$$\mathbb{E}_N(f(x_{0:k})) \xrightarrow{a.s} \mathbb{E}(f(x_{0:k})), N \to \infty.$$

where $\xrightarrow{a.s}$ denotes almost surely convergence.

Theorem 2.8. *Let f be a measurable function such that $\mathbb{E}(|f|) < \infty$. Assume that $\{x^i_{0:k}, i = 1 \cdots, N\}$ are i.i.d samples follows $\tilde{\pi}_{0:k|0:k}(x_{0:k}|y_{0:k})$ and f satisfies*

$$\mathbb{E}_{\tilde{\pi}_{0:k|0:k}}(1 + f^2)(\frac{\pi_{0:k|0:k}}{\tilde{\pi}_{0:k|0:k}})^2 < \infty,$$

then

$$\sqrt{N}(\mathbb{E}_N(f(x_{0:k})) - \mathbb{E}(f(x_{0:k})) \xrightarrow{d} N(0, \sigma^2_{\tilde{\pi}}(f)),$$

where $\sigma_{\tilde{\pi}(0:k|0:k)}(f) = \mathbb{E}_{\tilde{\pi}(0:k|0:k)}(f - \mathbb{E}(f))^2(\frac{\pi_{0:k|0:k}}{\tilde{\pi}_{0:k|0:k}})^2$.

The importance sampling procedure essentially provide a empirical measure of the posterior distribution by

$$\pi^N_{0:k|0:k}(x_{0:k}|y_{0:k}) = \sum_{i=1}^{N} \tilde{w}(x^i_{0:k}) \delta_{x^i_{0:k}}(dx_{0:k}). \tag{2.26}$$

The importance sampling is a very useful tool when simple Monte Carlo is hard to operate due to complexity of posterior distribution. However, to generalize the basic idea of importance sampling to a sequential setting, we need to design the importance sampling distribution in a nontrivial way. This is because it is crucial to design an efficient algorithm that is able to update the posterior distribution using only the most recent observation instead of reprocessing all the old observations. We will discuss the strategy for designing the sequential importance sampling scheme in the following section.

2.3.3 Sequential Monte Carlo Methods

First, the importance distribution, $\tilde{\pi}_{0:k|0:k}(x_{0:k}|y_{0:k})$, is written as a product according to the following

$$\tilde{\pi}_{0:k|0:k}(x_{0:k}|y_{0:k}) = \tilde{\pi}_{0:k|0:k}(x_{0:k}|y_{0:k})\tilde{\pi}_{k|k}(x_k|x_{k-1}, y_k). \tag{2.27}$$

The first factor in the decomposition of eq.(2.27) implies that we keep the old path and the second signifies that we predict the path to current time instant. The unnormalized importance weight w_k at time instant k can be computed from the weight at previous time instant $k-1$ as follows

$$w_k^i = \frac{\pi(x_{0:k}|y_{0:k})}{\tilde{\pi}(x_{0:k}|y_{0:k})} \propto w_{k-1}^i \frac{f(x_k^i|x_{k-1}^i)g(y_k|x_k^i)}{\tilde{\pi}_k(x_k^i|x_{k-1}^i, y_k)} \tag{2.28}$$

This decomposition indicates that the importance weights can be updated sequentially without having to reconsider the past observations. The basic sequential importance sampling method is summarized in Algorithm 2.

SIS can only output accurate estimation for short time periods since the importance weights will become highly degenerate after a few time steps meaning the probability mass concentrate on only a small portion of the particles and most of the particle contribute nothing significant to the empirical distribution. The solution

Algorithm 2 Sequential Importance Sampling (SIS)

Initialization at time $k = 0$, **generate** N **i.i.d samples** $\{x_0^i\}_{i=1,\cdots,N} \sim \pi_0(x_0), i = 1, \cdots, N$
for $k = 1 : T$ **do**
  for $i = 1, \cdots, N$ **do**
    Sample $x_k^i \sim \tilde{\pi}(x_k|x_{k-1}^i), i = 1, \cdots, N$
    Compute importance weight: $\tilde{w}_k^i = \tilde{w}_{k-1}^i \frac{g(y_k|x_k^i)f(x_k^i|x_{k-1}^i)}{\tilde{\pi}(x_k|x_{k-1}^i, y_k)}, i = 1, \cdots, N$
    Normalized importance weight: $w_i^i = \frac{\tilde{w}_k^i}{\sum_{j=1}^{N} \tilde{w}_k^j}, i = 1, \cdots, N.$
  end for
end for

proposed by Gordon et al. (1993) to reduce the degeneracy of the importance weights is based on the concept of resampling. The idea consists in resampling in the current particle cloud using the normalized importance weights as probabilities of selection. Therefore, the particles with small importance weights are discarded, whereas those with large importance weights are generated more copies of themselves proportional to their weights. After importance sampling, all importance weights are set to be $\frac{1}{N}$. The generic particle filter is given in Algorithm 3.

The bootstrap filter proposed by Gordon et al. (1993) uses the state transition density as the importance sampling distribution. Therefore, the importance weight becomes

$$w_k^i \propto w_{k-1}^i g(y_k|x_k^i). \tag{2.29}$$

The resampling is performed at every time instant so that the importance weight at previous time instant is always $1/N$ and can be ignored. In bootstrap filter the importance weight only depends on the likelihood of the most current observation. Also, the transition density is used as importance distribution.

Recall that the particle filter schemes generate a particle could associated each particle with proper weights. The weighted particles provide a weighted empirical distribution on the path state (or state space for filtering distribution) given in eq.(2.26). The auxiliary particle filter (APF) proposed by Pitt and Shephard (1999)

Algorithm 3 Generic Particle Filter

Initialization at time $k = 0$**, generate** N **i.i.d samples** $\{x_0^i\}_{i=1,\cdots,N} \sim \pi_0(x_0), i = 1, \cdots, N$
for $k = 1 : T$ **do**
  for $i = 1, \cdots, N$ **do**
    Sample $x_k^i \sim \tilde{\pi}(x_k|x_{k-1}^i), i = 1, \cdots, N$
    Compute importance weight: $\tilde{w}_k^i = \tilde{w}_{k-1}^i \frac{g(y_k|x_k^i) f(x_k^i|x_{k-1}^i)}{\tilde{\pi}(x_k|x_{k-1}^i, y_k)}, i = 1, \cdots, N$
  end for
  Normalized importance weight: $w_i^i = \frac{\tilde{w}_k^i}{\sum_{j=1}^N \tilde{w}_k^j}, i = 1, \cdots, N.$
  if Resampling **then**
    Select N particle indices $j_i \in \{1, \cdots, N\}$ according to the importance weights $\{w_k^j, j = 1, \cdots, N\}$.
    Set $x_{k-1}^i = \tilde{x}_{k-1}^{j_i}$ and $w_{k-1}^i = 1/N, i = 1, \cdots, N$
  else
    set $x_{k-1}^i = \tilde{x}_{k-1}^i, i = 1, \cdots, N.$
  end if
end for

uses a different resampling scheme which attempts to favor particles that are more likely to survive at the next time step. We briefly describe the formulation of APF in the following.

Let us assume the empirical posterior distribution at a given time instant $k-1$, i.e. $\pi_{0:k-1|0:k-1}^N$ is available. Then the posterior distribution at time instant k can be approximated by

$$\pi_{0:k|0:k}^N(dx_{0:k}|y_{0:k}) \approx \frac{1}{C} \sum_{i=1}^N w_{k-1}^i \delta_{x_{0:k-1}^i}(dx_{0:k-1}) g(y_k|x_k) f(x_k|x_{k-1}^i) dx_k, \tag{2.30}$$

where the normalizing constant is $C = \sum_{i=1}^N w_{k-1}^i \int f(x_k|x_{k-1}^i) g(y_k|x_k) dx_k$. The proposal for the new path state $x_{0:k}^i$ is generated by

$$\begin{aligned} \tilde{\pi}_{0:k|0:k}(dx_{0:k}|y_{0:k}) &= \tilde{\pi}_{0:k-1|0:k}(dx_{0:k-1}|y_{0:k}) \tilde{\pi}_k(dx_k|x_{k-1}, y_k) \\ &= \left(\sum_{i=1}^N \nu_{k-1}^i \delta_{x_{0:k}^i}(dx_{0:k-1}) \right) \cdot \tilde{\pi}_k(dx_k|x_{k-1}^i, y_k) \end{aligned} \tag{2.31}$$

where $\nu_{k-1}^i > 0$ is the weight of $x_{0:k-1}^i$ and $\sum_{i=1}^{N} \nu_{k-1}^i = 1$. The factorization compromises two components, a marginal proposal $\tilde{\pi}_{0:k-1|0:k}(\cdot)$ for the past path state $x_{0:k-1}$ and a conditional proposal $\tilde{\pi}_k(\cdot)$ for the current state x_k. The key feature of this proposal is that the first component depends explicitly on the observations up to current time instant k. It can be seen as a discrete distribution centered upon the old particles $\{x_{0:k-1}^i\}$ with the corresponding probability mass designed to be $\{\nu_{k-1}^i\}$. The rationale is to propose the past paths $x_{0:k-1}$ from their marginal conditional distribution $\pi_{0:k-1|0:k}$, i.e.

$$\begin{aligned} & \pi_{0:k-1|0:k}(dx_{0:k-1}|y_{0:k}) \\ \propto & \int \pi_{0:k-1|0:k-1}(dx_{0:k-1}|y_{0:k-1}) f(x_k|x_{k-1}) g(y_k|x_k) dx_k \\ = & \sum_{i=1}^{N} w_{k-1}^i \delta_{x_{0:k-1}^i}(dx_{0:k-1}) \int f(x_k|x_{k-1}^i) g(y_k|x_k) dx_k. \end{aligned} \tag{2.32}$$

The integral can be estimated by any means available. Using the proposal mechanism in eq.(2.31) we can define a generalized importance weight as follows

$$\tilde{w}_k^i = \frac{w_{k-1}^i}{\nu_{k-1}^i} \cdot \frac{g(y_k|x_k) f(x_k^i|x_{k-1}^i)}{q_k(x_k^i|x_{k-1}^i, y_k)}. \tag{2.33}$$

A summary of the APF algorithm is given in Algorithm 4.

2.3.4 Importance Density

The choice of importance density is crucial for the design of efficient particle filter algorithm. When the SMC method is interpreted as a Monte Carlo sampling scheme, the optimal choice of the importance density is the posterior distribution of interest, i.e. $\pi_{0:k|0:k}(x_{0:k}|y_{0:k})$. However, it is known that generic particle filter (Algorithm.3) suffers from degeneracy which means the samples are of negligible weights after a few steps of iterations, and it is an inevitable issue since the variance of the importance weights increase over time Doucet et al. (2000). A brute force solution is to increase

Algorithm 4 Auxiliary Particle Filter

Initialization at time $k = 0$**, generate** N **i.i.d samples** $\{x_0^i\}_{i=1,\cdots,N} \sim q_0(x_0|y_0), \tilde{w} = \frac{g(y_0|\tilde{x}_0^i)\pi_0(\tilde{x}_0^i)}{q_0(\tilde{x}_0^i|y_0)}, i = 1, \cdots, N.$
for $k = 1 : T$ **do**
    Select N particles indices $j_i \in \{1, \cdots, N\}$ according to weights $\{\nu_{k-1}^i\}_{i=1}^N$.
    for $i = 1, \cdots, N$ **do**
        set $x_{k-1}^i = \tilde{x}_{k-1}^{j_i}$
        set $u_{k-1}^i = \frac{w_{k-1}^{j_i}}{\nu_{k-1}^{j_i}}$
    end for
    for $i = 1 \cdots, N$ **do**
        Sample $\tilde{x}_k^i = \tilde{\pi}_k(\tilde{x}_k^i|x_{k-1}^i, y_k)$
        Compute weight by $\tilde{w}_k^i = u_{k-1}^i \frac{g(y_k|\tilde{x}_k^i)f(\tilde{x}_k^i|x_{k-1}^i)}{\tilde{\pi}_k(x_k^i|x_{k-1}^i, y_k)}$
    end for
    Normalization $w_k^i = \frac{\tilde{w}_k^i}{\sum_{j=1}^N \tilde{w}_k^j}, i = 1, \cdots, N.$
end for

the sample size, however, it is very impractical and inefficient. In order to alleviate the issue of degeneracy, one can choose a good importance density such that the variance of the importance weights is small. Generally, one can use effective sample size as a measure of the degeneracy. The effective sample size is defined as follows

$$\text{ESS} = \frac{(\sum_{i=1}^N w_k^i)^2}{\sum_{i=1}^N (w_k^i)^2} \tag{2.34}$$

where $w_k^i = \frac{\pi_{k|0:k}(x_k^i|y_{0:k})}{\tilde{\pi}_k(x_k^i|x_{k-1}^i, y_k)}$. Ideally, the optimal importance density can be found by minimizing the $var(w_k^i)$ conditional upon the simulated trajectory $x_{0:k-1}^i$ and the observations $y_{0:k}$. The following proposition by Doucet et al. (2000) provides such importance density.

Proposition 2.9. *The following importance density minimizes the variance of the importance weight conditional upon* $x^i_{0:k-1}$ *and* $y_{0:k}$*,*

$$\tilde{\pi}_k(x_{0:k}|x^i_{0:k-1}, y_{0:k}) = \pi_k(x_k|x^i_{k-1}, y_k). \tag{2.35}$$

Then we have

$$\begin{aligned} w^i_k &\propto w^i_{k-1} \frac{g(y_k|x^i_k) f(x^i_k|x^i_{k-1})}{\tilde{\pi}_k(x^i_{0:k}|x^i_{0:k-1}, y_{0:k})} \\ &= w^i_{k-1} \int g(y_k|x_k) f(x_k|x^i_{k-1}) dx_k \end{aligned} \tag{2.36}$$

The optimal choice of importance density (2.36) will reduce the $var(w^i_k)$ to zero since any generated sample x^i_k will have the same weight by (2.36). However, generating samples from (2.35) and evaluating weights for such samples by (2.36) are not straightforward tasks. A more convenient option is to use the importance density based on the dynamics model as in Algorithm.3. However, there are cases when the likelihood function related to observation model eq.(2.17) belongs to the exponential family of distributions, i.e. $p(y_k|x_k) \propto exp\{y_k g(x_k) - b(x_k)\}$ where $g(\cdot)$ is a known function and b is the function given in eq.(2.17). In this case as shown in Lemma 1 Evangelou and Maroulas (2015), the optimal density is skewed and a skewed normal proposal density has been suggested.

Chapter 3

A novel methodology

3.1 Introduction

The combination of sensors selection techniques and filtering methods has been used in target tracking in a sensor network. For example, the classical extended Kalman filter (EKF) for tracking a single-target is developed which also incorporated with a probabilistic framework for sensor selection Lin et al. (2009). Further, a distributed multi-target tracking algorithm using EKF was developed in Ren and Schizas (2013). Based on the consensus-averaging techniques and the particle filtering framework, single-target tracking schemes have also been developed for SNs Dias and Bruno (2013); LIU et al. (2009). For multi-target tracking problems, there are proposed methods that incorporate data association with particle filtering where observations are provided from a single sensor Doucet et al. (2002); Gorji et al. (2009); Hue et al. (2002). The work by Ng et al. (2005) proposed method that employ probabilistic models on the number of targets and the target-measurement assignments for multi-target tracking. A centralized algorithm which relies on Markov chain Monte Carlo (MCMC) performs data association on the observations acquired at a single-sensor in polynomial time is considered in [38] and it's been extended to the case of sensor network in Oh (2012). Other centralized approaches that perform data association in

time utilize Monte Carlo filtering have also been developed for example see Doucet et al. (2002); Vermaak et al. (2005).

A distributed algorithm that incorporates joint probabilistic data association with Kalman filter has been proposed by Sandell and Olfati-Saber (2008). However, the linear Gaussian assumptions in the model are not suited for vast real-world tracking applications. A different approach is followed in Tharmarasa et al. (2011) where multiple fusion centers are allowed in the sensor network and evaluate the posterior Cramer-Rao lower bound that requires knowledge of the underlying data model. Then as long as the fusion centers know which targets they are watching, then they can select the sensors that give the smallest Cramer-Rao lower bound.

3.2 Detecting dynamic objects

3.2.1 Analysis of Data Covariance

The crucial task in identifying the sensors which provide informative data is the estimation of the data covariance matrix $\Sigma_{y,k}$ of eq.(2.3). Moreover, the data covariance matrix depends on time, and changes as the targets move and the correlation structure between the sensor data changes. The estimation process should consider more recent data while gradually "forgets" the old ones in the data history. Inspired by Ren et al. (2015), we consider a memory parameter $\gamma \in (0, 1)$, whose main goal is to amplify recent data and attenuate past measurements. In detail, the covariance matrix at time k is estimated by

$$\hat{\Sigma}_{y,k} = \sum_{\tau=0}^{k} \alpha_k(\tau)(y_\tau - \bar{y}_k)(y_\tau - \bar{y}_k)^T, \tag{3.1}$$

where $\bar{y}_k = \sum_{\tau=0}^{t} \alpha_k(\tau) y_\tau$ is the adaptive mean estimate and the weighting function is defined by $\alpha_t(\tau) := \frac{\gamma^{k-\tau}(1-\gamma)}{1-\gamma^{k+1}}$. Notice that the weighting function $\alpha_k(\tau)$ gives more weight to the more recent sensing data while gradually discards the previous data

as needed. Notice that as τ decreases the factor $\gamma^{k-\tau}$ also decreases and it becomes 1 while $\tau = t$ (for current observation). The scaling factor $\frac{1-\gamma}{1-\gamma^{k+1}}$ is introduced to ensures that the mean and covariance estimates are unbiased, i.e.,

$$\mathbb{E}(\frac{1-\gamma}{1-\gamma^{k+1}}\hat{\Sigma}_{y,k}) = \Sigma_{y,k},\ \mathbb{E}(\bar{y}_k) = \mathbb{E}(y_k).$$

Given the distributed nature of the sensor network, where no central processing center is available, the decomposition of the covariance matrix $M_{y,k}$ of eq.(2.4) has to take into account the decentralized topology of the network. To this end, the following pertinent formulation is considered to factorize the covariance estimate $\hat{M}_{y,k}$

$$\hat{H}_k = \arg\min_{H_k} \left\| A \odot (\hat{M}_{y,k} - H_k H_k^T) \right\|_F^2 + \sum_{m=1}^{r} \lambda_{m,k} \left\| h_{m,k} \right\|_1 , \tag{3.2}$$

where $\|\cdot\|_F$ denotes the Frobenius norm and $\|\cdot\|_1$ refers to norm one while $\lambda_{m,k}$ are nonnegative scalars for tuning the sparsity in column $h_{m,k}$, A is the adjacency matrix of the network communication graph indicating which sensors can communicate with other sensors. Notice that $\odot$ denotes entry-wise multiplication and also encapsulates the immediate neighboring communication constraints in the WSN. The minimization of only the first term in eq.(3.2) does not necessarily give a sparse matrix since $H_k H_k^T$ is invariant of unitary rotation, i.e. $H_k H_k^T = H_k U (H_k U)^T$ where U is a unitary matrix. Therefore, we employ ℓ_1-regularization mechanisms (second part of the cost function) to impose sparsity in H_k, see e.g., Tibshirani (1996a); Zou (2006), and determine therefore information-bearing sensors. Further, the parameters $\lambda_{m,k}$ control the number of zeros (sparsity level) inside the factors $h_{m,k}$. The minimization problem in (3.2) can be tackled by employing a coordinate descent approach where the cost is minimized with respect to an entry of H_k while keeping the rest of the entries fixed to their latest update. Thus, the entries of $\hat{H}_k$ can be obtained applying a cyclic minimization procedure until the update do not change significantly over consecutive

cycles, see Ren et al. (2015) for more details. A detailed description of the algorithm along with the actual problem of tracking is described in Sec. 3.2.2.

3.2.2 Sparse Data Association

Starting from the basic regularized matrix decomposition cost function in eq. (3.2), the following formulation is obtained to decompose the sensor data covariance matrix

$$\begin{aligned} & \left\| A \odot (\hat{M}_{y,k} - H_k H_k^T) \right\|_F^2 + \sum_{\rho=1}^{M_k} \lambda_\rho \left\| h_\rho \right\|_1 \\ = & \sum_{j=1}^{p} \sum_{j' \in \mathcal{A}_j \cup \{j\}} \left(\hat{M}_{y,k}(j,j') - \sum_{\ell=1}^{M_k} H_k(j,\ell) H_k(j',\ell) \right)^2 + \sum_{\rho=1}^{M_k} \lambda_\rho \| h_\rho \|_1, \end{aligned} \tag{3.3}$$

where $\mathcal{A}_j$ denotes the neighboring sensors of sensor j, A is the adjacency matrix of the WSN and $\odot$ indicates entry-wise multiplication. As explained in Section 3.2.1 the support (nonzero entries) of the factors h_ρ will point to the sensors acquiring observations that contain information about target ρ. An iterative algorithm is employed relying on coordinate descent. At each coordinate descent cycle k, we update all the entries in the matrix H_k, then move forward to the next cycle $m+1$. Specifically, during the updating process at step m, the entries in H_k are updated one by one along the columns. We denote the updated matrix after the $(m-1)$th cycle by $\hat{H}_k^{m-1}$. Then, at the beginning of the mth cycle, we set $\hat{H}_k^m = \hat{H}_k^{m-1}$ and update the entry value $\hat{H}_k^m(j,\rho)$ by minimizing (3.3) with respect to $H_k(j,\rho)$. After applying first-order optimality conditions it turns out that we can get the update values according to the following proposition.

Proposition 3.1. *The update $\hat{H}_k^m(j,\rho)$ value can be obtained as the value that minimizes eq. (3.3) among the following three candidates:*

1. *Zero*

2. the real positive roots of

$$4h^3 + 4\left(\sum_{\mu\in\mathcal{A}_j}\left[\hat{H}_k^m(\mu,\rho)\right]^2 - \delta_\mu^m(j,j,\rho)\right)h + \lambda_\rho \tag{3.4}$$

$$- \left(4\sum_{\mu\in N_j}\delta_\mu^m(j,\mu,\rho)\hat{H}_k^m(\mu,\rho)\right) = 0 \tag{3.5}$$

3. the real negative roots of

$$4h^3 + 4\left(\sum_{\mu\in N_j}\left[\hat{H}_k^m(\mu,\rho)\right]^2 - \delta_\mu^m(j,j,\rho)\right)h - \lambda_\rho \tag{3.6}$$

$$- \left(4\sum_{\mu\in N_j}\delta_\mu^m(j,\mu,\rho)\hat{H}_k^m(\mu,\rho)\right) = 0 \tag{3.7}$$

where $\delta_\mu^m(a,b,c) = M_{y,k}(a,b) - \sum_{\ell=1,\ell\neq c}^{r}\hat{H}_k^m(a,\ell)\hat{H}_k^m(\ell,c)$*, and sensor* s_j *is responsible for updating the* j*th row of matrix* $\hat{H}_k$*.*

Proof. Since at cycle m, we need to optimize the cost entry by entry column-wise, and suppose we want to update the value of entry $\hat{H}_k^m(j,\rho)$. At this point, all the entries before $\hat{H}_k^m(j,\rho)$ are updated while the rest are not. We treat all the values except for $\hat{H}_k^m(j,\rho)$ as constants. After some algebra the cost function of (3.3) is reduced to a cost function with respect to $\hat{H}_k^m(j,\rho)$ only, which is given as follow

$$\begin{aligned} C^m(j,\rho) &= (\hat{H}_k^m(j,\rho))^4 + \lambda_\rho|\hat{H}_k^m(j,\rho)| \\ &+ (\hat{H}_k^m(j,\rho))^2\left(2\sum_{\mu\in\mathcal{A}_j}(\hat{H}_k^m(\mu,\rho))^2 - 2\delta_M^m(j,j,\rho)\right) \\ &- \hat{H}_k^m(j,\rho)\left(4\sum_{\mu\in\mathcal{A}_j}\delta_M^m(j,\mu,\rho)\hat{H}_k^m(\mu,\rho)\right) \end{aligned} \tag{3.8}$$

Therefore, we need to solve the following optimization problem

$$\begin{aligned}\hat{H}_k^m(j,\rho) &= \arg\min_{H_k^m(j,\rho)} (H_k^m(j,\rho))^4 + \lambda_\rho |H_k^m(j,\rho)| \\ &+ (H_k^m(j,\rho))^2 \left(2 \sum_{\mu \in \mathcal{A}_j} (\hat{H}_k^m(\mu,\rho))^2 - 2\delta_M^m(j,j,\rho) \right) \\ &- H_k^m(j,\rho) \left(4 \sum_{\mu \in \mathcal{A}_j} \delta_M^m(j,\mu,\rho) \hat{H}_k^m(\mu,\rho) \right) \end{aligned} \tag{3.9}$$

Set $H_k^m(j,\rho) = h,\ |H_k^m(j,\rho)| = a, c_1 = \left(2 \sum_{\mu \in \mathcal{A}_j} (\hat{H}_k^m(\mu,\rho))^2 - 2\delta_M^m(j,j,\rho) \right), c_2 = \left(4 \sum_{\mu \in \mathcal{A}_j} \delta_M^m(j,\mu,\rho) \hat{H}_k^m(\mu,\rho) \right)$, the problem in (3.9) is summarized as

$$\begin{aligned}\hat{H}_k^m(j,\rho) &= \arg\min_h h^4 + \lambda_\rho a + c_1 h^2 - c_2 h \\ \text{subject to } h - a &\leq 0 \\ -h - a &\leq 0 \end{aligned} \tag{3.10}$$

Let define $f(h,a) = h^4 + \lambda_\rho a + c_1 h^2 - c_2 h, g_1(h,a) = h - a, g_2(h,a) = -h - a$. The Karush-Kuhn-Tucker necessary condition implies the following. If h^* and a^* are local optimum, then there exist constants ξ_1, ξ_2 such that

$$-\nabla f(h^*, a^*) = \xi_1 \nabla g_1(h^*, a^*) + \xi_2 \nabla g_2(h^*, a^*) \tag{3.11}$$

$$\xi_i g_i(h^*, a^*) = 0, i = 1, 2 \tag{3.12}$$

while $\xi_1, \xi_2 \geq 0$. If $h^* > 0$, then (3.11) and (3.12) imply that $\xi_2 = 0, \xi_1 = \lambda_\rho$. Substituting the values for ξ_1, ξ_2 in (3.11) proves the condition of positive candidate. The negative candidate in the proposition can be shown in the same fashion. □

We utilize a 'deflation' technique to obtain a distributed scheme that is more computationally efficient. Specifically, instead of applying the aforementioned

coordinate descent scheme to find jointly all factors in the matrix $H_k = [h_1, \cdots, h_{M_k}]$, we employ the method to find a single factor $h_\rho, \rho = 1, \cdots, M_k$, at each time. For example, initially, the first estimated column factor $\hat{h}_1$ can be obtained by setting $M_{y,k} = 1$ and using $\hat{M}^0_{y,k} = A \odot \hat{M}_{y,k}$ (namely, we employ the minimization scheme on a single column factor), then we use the estimated column factor $\hat{h}_\rho$ to calculate the deflated matrix $\hat{M}^\rho_{y,k} = A \odot \hat{M}^{\rho-1}_{y,k} - A \odot (\hat{h}_\rho \hat{h}^T_\rho)$, then we can obtain the next estimated column factor $\hat{h}_{\rho+1}$ by running the same method. The deflation based decomposition algorithm is given in Algorithm 5.

Algorithm 5 Deflation based distributed decomposition algorithm (DSMD)

1: Each sensor S_j initialize the jth row of the covariance matrix $\hat{M}^0_{y,k}(j,:) = A(j,:) \odot \hat{M}_y(j,:)$
2: **for** $\rho = 1, \cdots, r_d$ **do**
3: Each sensor initializes $\hat{H}^0(j,\rho) = \hat{H}_{1s}(j,\rho)$ where $\hat{H}_{1s}(j,\rho)$ is obtained via DSMD by setting $\lambda = 0$ and $\hat{M}^{\rho-1}_{x,d}$.
4: **for** $k = 1, 2, \cdots$ **do**
5: Each sensor S_j for $j = 1, \cdots, p$
6: Transmits $\hat{H}^{k-1}(j,\rho)$ to its neighbors in $\mathcal{A}_j$, and receives $\hat{H}^{k-1}(j,\rho)$ from $\mu \in \mathcal{A}_j$
7: Evaluates $\delta^k_M(j,\mu,\rho)$ for $\mu \in \mathcal{A}_j \cup \{j\}$ using $\{\hat{M}^{\rho-1}_{y,k}(j,j')\}_{j' \in \mathcal{A}_j \cup \{j\}}$.
8: Determine the updates $\{\hat{H}^k(j,\rho)\}^r_{\rho=1}$ by minimizing (**??**).
9: If $|C^k(j,\rho) - C^{k-1}(j,\rho)| \leq \epsilon$, the stop.
10: **end for**
11: Each sensor S_j updates $\hat{M}^\rho_{y,k}(j,j') = \hat{M}^{\rho-1}_{y,k}(j,j') - \hat{H}^k(j,\rho)\hat{H}^k(j',\rho)$ for $j' \in \mathcal{A}_j \cup \{j\}$.
12: **end for**

3.3 An Advanced Sequential Monte Carlo method

3.3.1 Drift homotopy likelihood bridging particle filter

Once the informative sensors have been recovered, the second phase proceeds with estimating the trajectories of the objects. The tracking procedure is based on a sequential Monte Carlo method. Consider that the dynamics of the objects in the

sensing field are described by the following stochastic differential equation (SDE)

$$dx_k = b(x_k)dt + c(x_k)dB_k, \tag{3.13}$$

where $x_k \in \mathbb{R}^n$ is the state of the system at time instant k, and B_k is a Brownian motion. The drift $b(\cdot)$ and the diffusion coefficient $c(\cdot)$ denote nonlinear maps from $\mathbb{R}^n$ to $\mathbb{R}^n$ which satisfy suitable regularity properties. However, x_k is not fully-observed but noisy data (expressed in eq. (2.2)) from the informative sensors in the WSN are collected.

The goal is to estimate the posterior distribution $p(x_{0:k}|y_{1:k})$ (or just $p(x_k|y_{1:k})$), where $x_{0:k} = \{x_0, \cdots, x_k\}$ is the discrete version of the states of the latent Markov process generated from (3.13) in $[0, k]$ by assimilating the observational history $y_{1:k} = \{y_1, \cdots, y_k\}$ relying on eq.(2.2). In the following description, we will describe the estimation of the posterior distribution, i.e. $p(x_{0:T}|y_{1:T})$, where T denotes the final time.

In the sequential setting of the hidden Markov model, the posterior distribution is obtained by using the following two-stage recursion

$$\textit{Prediction:} \quad \pi_{0:k|0:k-1}(x_{0:k}|y_{1:k-1}) = \pi_{0:k-1|0:k-1}(x_{0:k-1}|y_{1:k-1})f(x_k|x_{k-1}), \tag{3.14}$$

$$\textit{Update:} \quad \pi_{0:k|0:k}(x_{0:k}|y_{1:k}) = \frac{g(y_k|x_k)\pi_{0:k|0:k-1}(x_{0:k}|y_{1:k-1})}{\pi(y_k|y_{1:k-1})}, \tag{3.15}$$

where $p(x_{0:k-1}|y_{1:k-1})$ is the posterior distribution at time instant $k-1$, $f(x_k|x_{k-1})$ is the transition density associated with eq.(3.13), $g(y_k|x_k)$ is the likelihood related to eq.(2.2), and $p(y_k|y_{1:k-1})$ is the normalizing constant that is independent of the state. The posterior distribution, $p(x_{0:T}|y_{1:T})$, does not have an explicit form under general conditions, thus it needs to be approximated. The recursive formula given in eqs. (3.14) & (3.15) provides the basis of particle filtering which approximates the posterior distribution by a set of weighted samples, i.e. $\{w_k^i, x_k^i\}_{i=1}^N$.

Particle filter even with the resampling step may need a large number of samples to approximate the posterior distribution. Moreover, as time evolves, particle filter starts to degenerate. We mitigate this degeneracy problem of low weighted particles by appending a Markov chain Monte Carlo (MCMC) step after the particles resampling. The appended MCMC step though should respect the nature of the posterior filtering distribution. To this end, we propose the drift homotopy likelihood bridging particle filter (DHLB-PF).

The algorithmic framework of DHLB-PF performs as the generic particle filter up to the resampling step as shown in Algorithm 6 from line 1 to line 11. Although the resampling step aims to create copies of samples with high-valued weights, oftentimes, only one sample dominates (i.e. $w_k^i = 1$ for some fixed i) and all the rest are zero Snyder et al. (2008). Precisely, in order to move the particles to a statistically significant region at some time instant t, an MCMC step is appended after resampling. To this end, we consider first a modification on the resampling step in the generic particle filter. This modification consists of resampling the sample pair (x_{k-1}^i, x_k^i) instead of x_k^i according to weights w_k^i, $i = 1, \cdots, N$. Using Bayesian considerations one can show that the filtering distribution $p(x_k|y_{0:k})$ is preserved if one samples from $f(x_k|x_{k-1})g(y_k|x_k)$ where x_{k-1} is given by the modified resampling step Weare (2009). The important step here is to sustain the features of the filtering distribution. We proceed by designing multiple levels of intermediate stationary densities given by

$$\pi_{\ell,m}(x_k|x_{k-1}, y_k) \propto f_\ell(x_k|x_{k-1})g^m(y_k|x_k), \tag{3.16}$$

where $0 \leq \ell \leq L$ and $0 \leq m \leq 1$. When $(\ell, m) = (L, 1)$, the sample cloud is distributed according to the filtering distribution $p(x_k|y_{1:t})$ at time instant t. Eq.(3.16) suggests two ways of addressing the problem. First, the intermediate levels ℓ of the transition density, $f_\ell(x_k|x_{k-1})$, aids to move the samples from a low energy region to a high energy region (similar to simulated annealing). Simultaneously, the likelihood bridging expressed by the exponent m of the likelihood helps to introduce the

likelihood in a smooth way. Several times, particle clouds are far from the likelihood of the data and thus the update step of particle filtering fails because samples cannot move to statistically significant regions. The likelihood bridging allows the particles to explore the state space (e.g. at $m = 0$, g is the uniform distribution) with a higher degree of freedom. We explain below how we proceed with the sampling of eq.(3.16) keeping in mind that this is at the appended MCMC step in the particle filter algorithm.

We first start by constructing a sequence of SDEs with modified drifts and engage them in a sequential way with the original dynamics being the last one in the sequence. To this end, we define

$$dx_k = (1 - \epsilon_\ell)a(x_k)dt + \epsilon_\ell b(x_k)dt + c(x_k)dB_k, \tag{3.17}$$

where $\epsilon_\ell = \frac{\ell}{L}, \ell = 0, 1, \cdots, L$, and $a(\cdot)$ is a suitably modified drift term that is different from the original drift $b(\cdot)$. At the ℓth level, the transition density $f_\ell(x_k|x_{k-1})$ in eq.(3.16) is determined by the modified dynamics given in (3.17). The original dynamics given in eq. (3.13) are taken into account at terminal level $\ell = L$ and the modified one alone is considered when $\ell = 0$. The choice of $a(\cdot)$ is problem specific and its goal is to facilitate the sampling process. Simultaneously, as the level ℓ changes, the exponent of the likelihood density, m, changes from 0 to 1. This allows to gradually shrink the width of likelihood density from the uniform to the original.

After the resampling step at time t, there is an available sample cloud $x_k^i, i = 1, \cdots, N$. Consequently we generate a new set of samples by evolving $x_k^i, i = 1, \cdots, N$ according to a Markov chain transition kernel $K_{\ell,m}(dy|x)$ that leaves $p_{\ell,m}(x_k)$ of eq.(3.16) invariant. The samples from the previous level are used as the initial condition at current level. At each level, the acceptance rate is given by

$$\alpha_{\ell,m,k} = \min\{1, \frac{J_\ell(x_k|x_k')f_\ell(x_k'|x_{k-1})g^m(y_k|x_k')}{J_\ell(x_k'|x_k)f_\ell(x_k|x_{k-1})g^m(y_k|x_k)}\}, \tag{3.18}$$

where $f_\ell(x_k|x_{k-1})g^m(y_k|x_k)$ is the stationary distribution, x_k is the current sample state and x'_k is the proposed sample state generated through a proposal distribution $J_\ell(\cdot)$. Notice that, instead of using a uniform or Gaussian proposal, the design of the stationary distribution provides an intuitive proposal density, i.e. $f_\ell(x'_k|x_k)$. The appended multilevel MCMC step improves the quality of the sample cloud by moving the samples into statistically significant regions and therefore obtain a better empirical representation of the filtering distribution. The operation is illustrated in the following Fig.3.1 which shows that at time instant t, one starts with the particle x_k^{*i} and go through the MCMC procedure to obtain a better sample $x_k^{L,i}$. The novel particle filter algorithm is tabulated in Algorithm 6.

3.3.2 The MCMC sampler: Generalized hybrid Monte Carlo

We employ a particularly effective Markov chain Monte Carlo method , the so called generalized Hybrid Monte Carlo (GHMC), in Step 15 of Alg.6 for the solution of multi-targt tracking in a WSN. In this section, we briefly describe the GHMC sampler in the general context of sampling from some target distribution $f_\ell(x_k|x_{k-1})g^m(y_k|x_k)$.

Consider a system whose state is determined by $\mathcal{T}$ real-valued state variables, i.e. $\{x_k^i\}_{i=1}^{\mathcal{T}}$. and associate them with a set of momentum variables $\{p_k^i\}_{i=1}^{\mathcal{T}}$, where x_k^i, p_k^i are N dimensional real valued vectors. In the context of Monte Carlo simulation, the momenta serve the purpose of constructing the associated Hamiltonian system. The auxiliary kinetic energy is defined as $\mathcal{K}(p_k) = \sum_{i=1}^{\mathcal{T}} \frac{|(p_k^i)^2|}{2}$. The total Hamiltonian of the system is given by $\mathcal{H}(x_k, p_k) = \mathcal{U}(x_k) + \mathcal{K}(p_k)$, where $\mathcal{U}(x_k)$ is the potential energy defined by $\mathcal{U}(x_k) = -log(f_\ell(x_k|x_{k-1})g^m(y_k|x_k))$. A generalized hybrid Monte Carlo (GHMC) Toral and Ferreira (1994) suggested that we consider the following

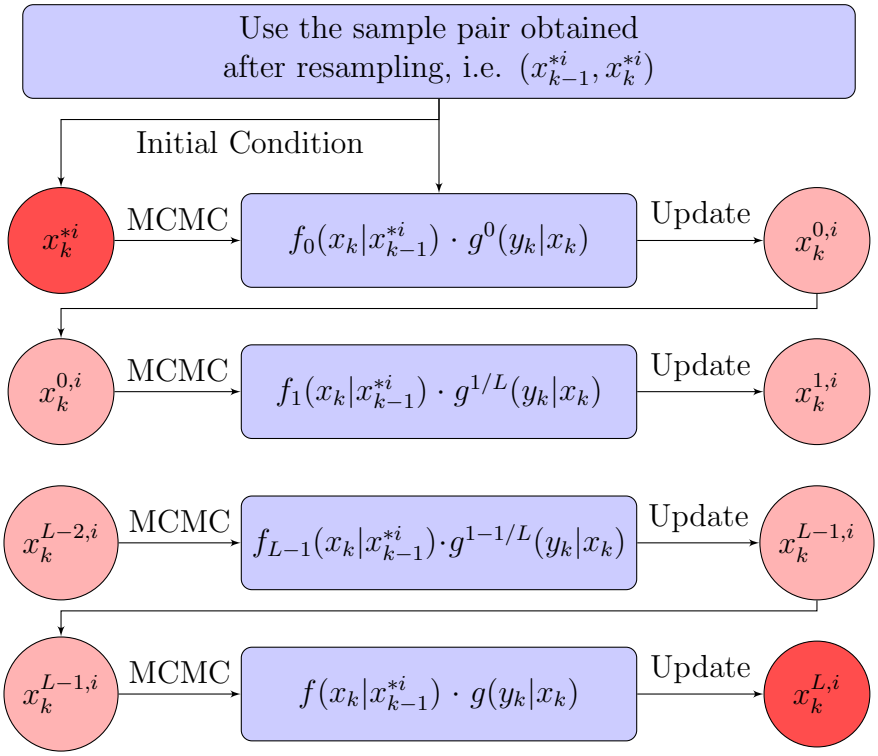

Figure 3.1: This diagram exhibits the procedure of the appended MCMC sampling. One starts with the sample pair (x_{k-1}^{*i}, x_k^{*i}) after resampling where x_{k-1}^{*i} is fixed during the samping and x_k^{*i} is used as the initial condition. After the first level, one obtains $x_k^{0,i}$, then use it as the initial condition for the MCMC sampling at the next level. Finally, the original sample x_k^{*i} is improved by the sample $x_k^{L,i}$ after the final level after MCMC sampling.

Algorithm 6 Drift homotopy likelihood bridging particle filter (DHLB-PF)

for $i = 1, \cdots, N$ **do**
  Draw i.i.d samples $x_0^i \sim p_0(x)$, and $w_0^i = \frac{1}{N}$, where $p_0(x)$ is the known initial distribution.
end for
for $t = 1, \cdots, T$ **do**
  for $i = 1, \cdots, N$ **do**
    Draw sample $x_{k-1}^i \sim p(x_{k-1}|y_{1:t-1})$
    Propagation: $x_k^i \sim p(x_k|x_{k-1}^i)$
    Particle weight: $\hat{w}_k^i = p(y_k|x_k^i)$
  end for
  Weights Normalization: $w_k^i = \hat{w}_k^i / \sum_{j=1}^{N} \hat{w}_k^j, i = 1, \cdots, N.$
  Resampling: Generate N independent uniform random variables $\{\theta^i\}_{i=1}^N$ from $[0, 1]$. For $i, j = 1, \ldots, N$, let $\{x_{k-1}^{*i}, x_k^{*i}\} = \{x_{k-1}^j, x_k^j\}$ where

$$\sum_{l=1}^{j-1} w_k^l \leq \theta^j < \sum_{l=1}^{j} w_k^l$$

  MCMC step:
  for $\ell = 0, \cdots, L$ **do**
    for $m = 0, \cdots, 1$ **do**
      for $i = 1, \cdots, N$ **do**
        With initial value x_k^{*i}, sample through MCMC the stationary distribution

$$\pi_\ell(x_k|x_{k-1}, y_k) \propto g^m(y_k|x_k) f_\ell(x_k|x_{k-1}^{*i})$$

      end for
    end for
  end for
end for

generalized Hamiltonian system:

$$\begin{array}{rcl} \frac{dx_k}{dt} & = & \sum_{s=1}^{N} \mathcal{A}^s P_k^s, \\ \frac{dP_k^s}{dt} & = & -(\mathcal{A}^s)^{\mathcal{T}} \frac{d\mathcal{U}}{dx_k}, \end{array} \tag{3.19}$$

where $s = 1, \cdots, N, x_k = (x_k^1, x_k^2, \cdots, x_K^{\mathcal{T}})^{\mathcal{T}}$ and $x_k^i = ((x_k^i)^1, \cdots, (x_k^i)^N)^{\mathcal{T}}, i = 1, 2, \cdots, \mathcal{T}$. Similarly, $p_k = (p_k^1, p_k^2, \cdots, p_k^{\mathcal{T}})^{\mathcal{T}}$ and $p_k^i = ((p_k^i)^1, \cdots, (p_k^i)^N)^{\mathcal{T}}, i = 1, 2, \cdots, \mathcal{T}$. Especially, $p_k^s = ((p_k^1)^s, (p_k^2)^s, \cdots, (p_k^{\mathcal{T}})^s)^T$, for $s = 1, \cdots, N$. Notice

that in our simulation in Section 4.2.1, the $\mathcal{T}$ state variables correspond to the states on the path between two consecutive observations, i.e. $\mathcal{T} = 1/\Delta t$.

Suppose the matrix $\mathcal{A} = \{\mathcal{A}_{ij}\}, 1 \leq i, j \leq \mathcal{T}$, then for each state variable X_i and the associated momentum P_i, the generalized Hamiltonian system is

$$\begin{array}{rcl} \frac{dx_k^i}{dt} & = & \sum_{j=1}^{\mathcal{T}} \mathcal{A}_{ij} p_k^j, \\ \frac{dp_k^i}{dt} & = & -(\mathcal{A})^{\mathcal{T}} \frac{d\mathcal{U}}{dx_k} = \sum_{j=1}^{T} \mathcal{A}_{ji} \frac{d\mathcal{U}}{dx_k^j}. \end{array} \tag{3.20}$$

$\mathcal{A}$ is an arbitrary $\mathcal{T} \times \mathcal{T}$ matrix which should be chosen to enhance the effectiveness of proposal (i.e. reduce the correlation between two consecutive proposals) and it was suggested Alexander et al. (2005a) that $\mathcal{A} = circ(1, exp(-\alpha), exp(-2\alpha), \cdots, exp(-(\mathcal{T}-1)\alpha))$, explicitly

$$\mathcal{A} = \begin{pmatrix} 1 & exp(-(\mathcal{T}-1)\alpha) & \cdots & exp(-\alpha) \\ exp(-\alpha) & 1 & \cdots & exp(-2\alpha)) \\ \vdots & \vdots & \ddots & \vdots \\ exp(-(\mathcal{T}-1)\alpha) & exp(-(\mathcal{T}-2)\alpha) & \cdots & 1 \end{pmatrix}.$$

We evolve the system of (3.20) by performing some numerical scheme and then propose a new state $((x_k^i)', (p_k^i)')$. A commonly used numerical scheme for solving Hamiltonian system is the leapfrog discretization Frenkel and Smit (2001). The Metropolis step afterwards will accept the new state $((x_k^i)', (p_k^i)')$ with probability

$$\begin{array}{rcl} \alpha_i & = & \min\{1, \frac{\bar{\mathcal{F}}((x_k^i)', (p_k^i)')}{\bar{\mathcal{F}}(x_k^i, p_k^i)}\} \\ & = & \min\{1, \exp(-\mathcal{H}((x_k^i)', (p_k^i)') + \mathcal{H}((x_k^i)', (p_k^i)')\} \\ & = & \min\{1, \exp(\mathcal{U}(x_k^i) - \mathcal{U}((x_k^i)') + \mathcal{K}(p_k^i) - \mathcal{K}((p_k^i)')\}, \end{array} \tag{3.21}$$

where we define the joint distribution of state x_k and auxiliary momentum p_k as $\bar{\mathcal{F}}(x_k, p_k) \propto exp(-\mathcal{H}(x_k, p_k))$.

In GHMC, the proposal comes from the molecular dynamics simulation. As we mentioned above, the leapfrog scheme is employed for evolving the dynamics in (3.20).

The leapfrog scheme first updates the momentum at a half time interval, then evolves the position over a full time interval. At the end, the momentum is updated over the other half time interval. Notice that the time index here is different from the time index we use for the state of system, in GHMC the time index is auxiliary and only used for proposing a new state. Suppose Δt is the discretized time interval, the leapfrog algorithm operates as follows

1. The momentum variable is updated first

$$p_k^i(t + \Delta t/2) = p_k^i(t) - \frac{\Delta t}{2}\frac{\partial \mathcal{U}}{\partial x_k^i(t)},$$

2. Then state over a full time interval Δt is updated

$$x_k^i(t + \Delta t) = x_k^i(t) + \Delta t \frac{\partial \mathcal{K}}{\partial p_k^i(t + \Delta t)},$$

3. Update the momentum over the other half of the time interval

$$p_k^i(t + \Delta t) = p_k^i(t + \Delta t/2) - \frac{\Delta t}{2}\frac{\partial \mathcal{U}}{\partial x_k^i(t + \Delta t)}.$$

Algorithm 7 Generalized hybrid Monte Carlo algorithm (GHMC)

1: Generate initial position state x_0 from some known distribution.
2: **for** $t = 1, \cdots, M$ **do**
3: Generate a momentum variable $p_t \sim p(p) \propto \exp(\mathcal{K}(p))$, which is a standard Gaussian distribution $\mathcal{N}(0, 1)$.
4: Run L steps leapfrog algorithm with initial state (x_t, p_t) and step size Δt to obtain a new state (x_t^*, p_t^*).
5: Set $x_{t+1} = x_t^*$ with probability

$$\alpha_k = \min(1, \exp(\mathcal{U}(x_t) - \mathcal{U}(x_t^*) + \mathcal{K}(p_t) - \mathcal{K}(p_t^*))).$$

and $x_{t+1} = x_t$ with probability $1 - \alpha_t$.
6: **end for**

3.4 A theoretical treatment

We assume the signal process $\{x_k\}_{k\geq 0}$ is a Markov process with known initial distribution, i.e. $x_0 \sim \pi_0(x_0)$. Then by Definition 2.2 we have that

$$P(x_k \in A | x_{k-1} = x_{k-1}^*) = \int_A K(x_{k-1}^*, dx_k), A \in \mathcal{B}(\mathbb{R}^{n_x}). \tag{3.22}$$

The observations are conditionally independent of X and

$$P(y_k \in B | x_k = x_k^*) = \int_B g(dy_k | x_k^*), B \in \mathcal{B}(\mathbb{R}^{n_y}). \tag{3.23}$$

where x_k denotes the random variable and x_k^* denotes a realization of x_k.

Based on the Markovian assumption and Bayes' theorem, the joint posterior distribution can be updated through the following recursion given by eq.(3.14) and eq.(3.15)

In many cases, we are interested in the filtering distribution and the recursion formula writes

$$\text{Prediction: } \pi_{k|k-1}(dx_k) = \int_{\mathbb{R}^{n_x}} \pi_{k-1|k-1}(dx_{k-1}) K(x_{k-1}, dx_k) \tag{3.24}$$

$$\text{Update: } \pi_{k|k}(dx_k) = \frac{g(y_k|x_k)\pi_{k|k-1}(dx_k)}{\int_{\mathbb{R}^{n_x}} g(y_k|x_k)\pi_{k|k-1}(dx_k)} \tag{3.25}$$

Based on the definition, the Markov transition kernel defines an operator on measurable functions and measures as follows

$$\begin{aligned} \nu K(A) &= \int \nu(dx) K(x, A), \\ K\phi(x) &= \int K(x, dy)\phi(y), \end{aligned}$$

where ν is a measure, and ϕ is a measurable function.

3.4.1 Drift Homotopy and Likelihood Bridging Particle Filtering

The particle filter is a simulation-based method that estimates the statistical inference based on the posterior distribution at time instant k. Essentially, it provides a cloud of N suitably weighted particles, i.e $\{x_k^i, w_k^i, i = 1, \cdots, N\}$, whose empirical distribution is used as an approximation of the posterior distribution. We denote the empirical filtering distribution by

$$\pi_{k|0:k}^N(dx_k|y_{0:k}) = \frac{1}{N}\sum_{i=1}^{N}\delta_{x_k^i}(dx_k), \tag{3.26}$$

where $\delta_{x_k^i}(dx_k)$ denotes the delta-Dirac mass located at point $x_k^i, i = 1, \cdots, N$. Any statistical inference will be carried out using the empirical measure defined by (3.26). Another importance feature of the method is that particle filter is a recursive algorithm in the sense that the empirical measure at next time instant $k+1, i.e.\ \pi_{k+1|0:k+1}^N(dx_{k+1}|y_{0:k})$ is obtained by evolving the current N particles through some transition density and reassign the importance weights appropriately. Our proposed novel particle filter inherits the same essential idea and focus on resolving the degeneracy problem of the framework by appending a well designed multilevel MCMC procedure, the goal of which is to redeploy the particle cloud so that a better (in the sense that more particles will locate in statistically significant region) empirical posterior distribution can be obtained. Even though the particle filter framework is versatile for non-linear and non-Gaussian case, it might be very inefficient in the case where a complicated posterior distributed presents and an efficient transition density is hard to find. In such cases, most particles will be wander around the places with low probability mass and result in a poor representation of posterior distribution. The multilevel MCMC procedure aims to mitigate the problem. To this end, the proposed DHLB-PF algorithm operates in the following order and later in order to analyze the DHLB-PF algorithm, we will treat each of these steps as a function

- Initialization: $x_0^i \sim \pi_0(x_0), i = 1, \cdots, N$, where π_0 is known.
- Prediction: $x_k^i \sim \pi_{k-1|k-1}^N K(x_{k-1}^i, dx_k), i = 1, \cdots, N$.
- Resampling: $x_k^i \sim \tilde{\pi}_{k|k}^N(dx_k)$, where $\tilde{\pi}_{k|k}^N = \sum_{i=1}^{N} w_i^k \delta_{x_{k|k}^i}(dx_k)$
- Redeployment: $x_k^i \xrightarrow{DHLB} x_k^{*i}, i = 1, \cdots, N$.

The prerequisite of the algorithm is that we are able to generate i.i.d samples from the known initial distribution of state random variable $\pi_0(x_0)$, or we can generate i.i.d samples from an instrumental distribution and compute the corresponding importance weights. The algorithm takes three extra steps to evolve the weighted particle cloud (equally weighted if samples are i.i.d). The output of the particle filter is particle cloud which provides an empirical measure that approximate the posterior distribution at final time instant. In the following section, we discuss in what sense the empirical measure provided by our DHLB-PF algorithm is close to the posterior distribution and under minimal condition it converges to the posterior distribution asymptoticly.

3.4.2 Almost Sure Convergence of the DHLB particle filter

Let $\mathcal{M} = \mathcal{P}(\mathbb{R}^{n_x})$ be the set of probability measures on the n_x-dimensional Euclidean space $\mathbb{R}^{n_x}$ endowed with the topology of weak convergence. We say that a sequence of probability measures $\{\pi_N\}_{N\geq 1}$, where $\pi_N \in \mathcal{P}(\mathbb{R}^{n_x}), \forall N$, converges weakly to another probability measure $\pi \in \mathcal{P}(\mathbb{R}^{n_x})$ if for any $h \in C_b(\mathbb{R}^{n_x})$ such that

$$\lim_{N\to\infty} (\pi_N, h) = (\pi, h),$$

where $(\cdot, \cdot)$ denotes the inner product, i.e. $(\pi, h) = \int h d\pi$, $C_b(\mathbb{R}^{n_x})$ denotes the set of bounded continues functions on $\mathbb{R}^{n_x}$. We simply write the weak convergence as $\lim_{N\to\infty} \pi_N = \pi$. It turns out that the weak convergence can be determined by a countable set of continuous bounded functions, that is, $\exists \mathcal{H} = \{h_i\}_{i>0}$, and $h_i \in$

$C_b(\mathbb{R}^{n_x}), \forall i$, such that,

$$\lim_{N\to\infty} \pi_N = \pi \ \textit{iff} \ \lim_{N\to\infty} (\pi_N, h_i) = (\pi, h_i), \forall h_i \in \mathcal{H}.$$

We define a distance on the space $\mathcal{P}(\mathbb{R}^{n_x})$ by using the functions in $\mathcal{H}$ as follows,

$$d(\pi, \nu) = \sum_{i=1}^{N} \frac{|(\pi, h_i) - (\nu, h_i)|}{2^i \|h_i\|}, \tag{3.27}$$

where $\|h_i\| = sup_{x\in\mathbb{R}^{n_x}} |h_i(x)|$ is the supremum norm on $C_b(\mathbb{R}^{n_x})$. It is easy to see that weak convergence by definition indicates that the distance between probability measures and the limit probability measure goes to zero and vice versa, i.e.

$$\lim_{N\to\infty} d(\pi_N, \pi) = 0 \ \textit{iff} \ \lim_{N\to\infty} \pi_N = \pi. \tag{3.28}$$

Now we have stated the convergence property we want to study. In the context of filtering, π_N will be the empirical measure given by the algorithm, where N typically corresponds to the sample size of the particle cloud and π will be the posterior distribution at some fixed time instant. At any time instant, we want the empirical distribution π_N to get closer to π as we increase the sample size N. In the following, we will discuss how we can achieve this convergence and what conditions we need to impose.

We define the prediction step of the particle filter algorithm at time instant k as a mapping, $q_k : \mathcal{P}(\mathbb{R}^{n_x})$ to $\mathcal{P}(\mathbb{R}^{n_x})$, namely

$$q_k(\pi)(dx_k) = \pi K(dx_k) = \int_{\mathbb{R}^{n_x}} K(x_{k-1}, dx_k)\pi(dx_{k-1}), \tag{3.29}$$

for all $\pi \in \mathcal{P}(\mathbb{R}^{n_x})$. Therefore, we have that $(q_k(\pi), h) = (\pi, Kh), \forall h \in C_b(\mathbb{R}^{n_x})$, and $\pi_{k|0:k-1} = q_k(\pi_{k-1|0:k-1})$. It turns out that in order to guarantee the weak convergence, the function $q_k(\cdot)$ has to be continuous. We require the transition kernel K, which defines the mapping $q_k(\cdot)$, to be Feller, that is

Definition 3.2. *A transition kernel is Feller if the following holds*

$$Kh \in C_b(\mathbb{R}^{n_x}), \forall h \in C_b(\mathbb{R}^{n_x}). \tag{3.30}$$

Hence, if $\lim_{N\to\infty} \pi_N = \pi$, then by definition 3.4.2 we have

$$\lim_{N\to\infty} (q_k(\pi_N), h) = \lim_{N\to\infty} (\pi_N, Kh) = (\pi, Kh) = (q_k(\pi), h), \forall h \in C_b(\mathbb{R}^{n_x}). \tag{3.31}$$

Therefore, $q_k(\cdot)$ is continuous.

We now define the update step of the algorithm to be another mapping from $\mathcal{P}(\mathbb{R}^{n_x})$ to $\mathcal{P}(\mathbb{R}^{n_x})$. Namely, we define $u_k : \mathcal{P}(\mathbb{R}^{n_x}) \to \mathcal{P}(\mathbb{R}^{n_x})$ as

$$(u_k(\pi), h) = \frac{(\pi, hg)}{(\pi, g)}, \forall h \in C_b(\mathbb{R}^{n_x}). \tag{3.32}$$

Therefore, $\pi_{k|k} = u_k(\pi_{k|k-1}) = u_k \circ q_k(\pi_{k-1|k-1})$. Again, the continuity of function $u_k(\cdot)$ is required to guarantee a convergent algorithm. To ensure the continuity of the function $u_k(\cdot)$, we need that $g(y_k|\cdot)$ is a continuous bounded and strictly positive function, i.e.

$$g(y_k|\cdot) \in C_b(\mathbb{R}^{n_x}), g(y_k|x_k) > 0, \forall x_k \in \mathbb{R}^{n_x}. \tag{3.33}$$

Then based on the definition, we can verify that if $\lim_{N\to\infty} \pi_N = \pi$, then

$$\lim_{N\to\infty} (u_k(\pi_N), h) = \frac{\lim_{N\to\infty} (\pi_N, hg)}{\lim_{N\to\infty} (\pi_N, g)} = \frac{(\pi, hg)}{\pi, g} = (u_k(\pi), h), \forall h \in C_b(\mathbb{R}^{n_x}) \tag{3.34}$$

Therefore, $u_k(\cdot)$ is continuous.

We have now defined the two recursive steps of the particle filter as continuous functions on the space of probability measures under suitable conditions on the transition kernel and likelihood function. Let us now consider at any time instant k, the algorithm can be seen as a composition of the two recursive steps and therefore define another function that maps the initial probability distribution to the posterior

distribution at time instant k, i.e., we define

$$s_{1:k} = s_k \circ s_{k-1} \circ \cdots \circ s_1, \text{where } s_k = u_k \circ q_k, \tag{3.35}$$

where $\circ$ denotes the composition of functions. It is clear that s_k and $s_{1:k}$ are continuous since the u_k and q_k are continuous functions. Now we have defined the recursion formula of particle filter as a continuous function, i.e.,

$$\pi_{k|k} = s_k(\pi_{k-1|k-1}) = s_{1:k}(\pi_0). \tag{3.36}$$

The particle filter is a simulation-based realization of (3.36). The idea is to use random samples to estimate the posterior probability measure. Therefore, a random perturbation occurs at every time instant k. We define this random perturbation by

$$m_N(\pi)(w) = \frac{1}{N}\sum_{i=1}^{N} \delta_{x_i(w)} \tag{3.37}$$

where $w \in \Omega$ and $x_i : \Omega \to \mathbb{R}^{n_x}$ are i.i.d random variables following the probability distribution π. For a realization of the i.i.d random variables, eq.(3.37) provide an empirical distribution.

The last step of the proposed algorithm uses an MCMC sampler to redeploy the particles. This procedure evolves the particles through some Markov transition kernel for certain steps and uses the end points of the Markov chain as the new particle cloud, hence it can be defined as

$$m_N(\pi)\bar{K}^M(w) = \frac{1}{N}\sum_{i=1}^{N} \bar{K}^M \delta_{x_i(w)} \tag{3.38}$$

where $\bar{K}^M$ denotes the operation of evolving particles through the MCMC transition kernel $\bar{K}$ M steps, which turns out to another transition kernel as discussed below.

Notice that the transition kernel $\bar{K}$ of a Markov chain describes the probability of moving the state one step forward, i.e. for all $m \in \mathbb{N}$ we have

$$\bar{K}(x, A) = P(X_{m+1} \in A | X_m = x). \tag{3.39}$$

The M step transition kernel can be obtained inductively by

$$\bar{K}^M(x, A) = \int_{\mathbb{R}^n} \bar{K}^{M-1}(y, A)\bar{K}(x, dy) = \int_{\mathbb{R}^n} \bar{K}(y, A)\bar{K}^{M-1}(x, dy). \tag{3.40}$$

Now consider integrating over the conditional distribution of the previous step, we have the following

$$\begin{aligned}
& \mathbb{P}(X_{m+1} \in A | X_m = x) = \bar{K}(x, A), \\
& \mathbb{P}(X_{m+2} \in A | X_m = x) \\
= & \int_{\mathbb{R}^n} \mathbb{P}(X_{m+2} \in A | X_{m+1} = y, X_m = x)\mathbb{P}(X_{m+1} \in dy | X_m = x) \\
= & \int_{\mathbb{R}^n} \mathbb{P}(X_{m+2} \in A | X_{m+1} = y)\bar{K}(x, dy) = \bar{K}^2(x, A), \\
\cdots & \\
& \mathbb{P}(X_{m+M} \in A | X_m = x) \\
= & \int_{\mathbb{R}^n} \mathbb{P}(X_{m+M} \in A | X_{m+M-1}, X_m = x)\mathbb{P}(X_{m+M-1} \in dy | X_m = x) \\
= & \int_{\mathbb{R}^n} \mathbb{P}(X_{m+M} \in A | X_{m+M-1} = y)K^{M-1}(x, dy) \\
= & \bar{K}^M(x, A).
\end{aligned}$$

The M step transition probability from state x to A is $\mathbb{P}(X_{m+M} \in A | X_m = x) = \bar{K}^M(x, A)$.

To ensure the weak convergence of the algorithm, we need to ensure that the accumulated errors of the random perturbation will not explode, and for any fixed time instant, the empirical measure should converge to the posterior distribution

asymptotically. The key to guarantee this feature is the following condition,

$$\lim_{N\to\infty} \pi_N = \pi \Longrightarrow \lim_{N\to\infty} m^N \bar{K}^M(\pi_N) = \pi, \forall \pi_N, \pi \in \mathcal{P}(\mathbb{R}^{n_x}). \tag{3.41}$$

Now, let's consider a simple case of our algorithm where the redeployment step uses only one level, that is, $L = 0$. Namely, one directly sample from the posterior distribution without intermediate levels. We show that the perturbation steps (resampling and redeployment) satisfy the following lemma.

Lemma 3.3. *The random perturbation*

$$m_N(\pi)\bar{K}^M(w) = \frac{1}{N}\sum_{i=1}^{N} \bar{K}^M \delta_{x_i(w)}$$

satisfy the following condition for almost all $w \in \Omega$

$$\lim_{N\to\infty} \pi_N = \pi \Longrightarrow \lim_{N\to\infty} m^N \bar{K}^M(\pi_N) = \pi, \forall \pi_N, \pi \in \mathcal{P}(\mathbb{R}^{n_x})$$

Proof. Let $\pi_N, \pi \in \mathcal{P}(\mathbb{R}^{n_x})$ be a sequence of probability measures and its mean, that is, $\lim_{N\to\infty} \pi_N = \pi$ weakly. We know that the random perturbation $m_N(\pi_N)$ generates N i.i.d samples from π, i.e.$\{x_i, i = 1, \cdots, N\}$. We assume the transition kernel $\bar{K}$ is reversible and has stationary distribution π. Let $\{x_i^1, i = 1, \cdots, N\}$ be the set of random variables evolved one step through the transition kernel, then it is clear that $P(x_i^1 \in A) = \pi\bar{K}(A) = \pi(A)$ which indicates that $\{x_i^1, i = 1, \cdots, N\}$ still follows the distribution π. Also, we know for each transition kernel $K(\cdot, \cdot)$ there exists a random mapping representation, that is a measurable function $\Phi : \mathbb{R}^{n_x} \times [0, 1] \to \mathbb{R}^{n_x}$ which satisfies $P(\Phi(x, Z) \in A) = K(x, A), x \in \mathbb{R}^{n_x}, A \in \mathcal{B}(\mathbb{R}^{n_x})$, where the random variable Z is uniformly distributed. Then, the set of random variables $\{x_i^1, i = 1, \cdots, N\}$ can be defined as

$$x_i^1 = \Phi(x_{i-1}, Z_i), i \geq 2 \tag{3.42}$$

where $\{Z_i, i = 1, \cdots, N\}$ are a set of i.i.d random variables with uniform distribution. Since $\Phi()$ is measurable function, $\{x_i^1, i = 1, \cdots, N\}$ is a set of independent random variables. By induction, we have that $\{x_i^M, i = 1, \cdots, N\}$ are independent $\forall M < \infty$, where M denotes the steps of Markov chain.

Now let us consider the following

$$\begin{aligned}
& \mathbb{E}\left(((m_N K_M(\pi_N), h) - (\pi_N, h))^4\right) \\
= \; & \mathbb{E}\left((\frac{1}{N}\sum_{j=1}^{N} h(x_j^M) - (\pi_N, h))^4\right) \\
= \; & \mathbb{E}\left(\frac{1}{N^4}(\sum_{j=1}^{N}(h(x_j^M) - (\pi_N, h)))^4\right) \\
= \; & \frac{1}{N^4}\mathbb{E}\left(\sum_{j=1}^{N}(h(x_j^M) - (\pi_N, h))^4\right) \\
= \; & \frac{1}{N^4}\mathbb{E}\left(\sum_{1\le i,j,k,\ell\le N}\prod_{\tau=i,j,k,\ell}(h(x_\tau^M) - (\pi_N, h))\right) \qquad (3.43)
\end{aligned}$$

Notice that $\mathbb{E}\left(h(x_i^M) - (\pi_N, h)\right) = \mathbb{E}(h(x_i^M)) - (\pi_N, h) = 0$. Since $\{x_i^M, i = 1, \cdots, M\}$ are independent, then we have

$$\mathbb{E}\left(h(x_i^M) - (\pi_N, h)^3(h(x_j^M) - (\pi_N, h))\right) = 0,$$

$$\mathbb{E}\left(h(x_i^M) - (\pi_N, h)^2(h(x_j^M) - (\pi_N, h))(h(x_k^M) - (\pi_N, h))\right) = 0,$$

$$\mathbb{E}\left(h(x_i^M) - (\pi_N, h)(h(x_j^M) - (\pi_N, h))(h(x_k^M) - (\pi_N, h))(h(x_\ell^M) - (\pi_N, h))\right) = 0$$

, if i, j, k, ℓ are distinct. Therefore, eq.(3.43) reduces to the following

$$\begin{aligned}
& \frac{1}{N^4}\mathbb{E}\left(\sum_{1\le i,j,k,\ell\le N}\prod_{\tau=i,j,k,\ell}(h(x_\tau^M)-(\pi_N,h))\right)\\
= & \frac{1}{N^4}\sum_{j=1}^{N}\mathbb{E}(h(x_j^M)-(\pi_N,h))^4\\
+ & \frac{6}{N^4}\sum_{i,j=1,i\neq j}^{N}\mathbb{E}(h(x_i^M)-(\pi_N,h))^2(h(x_j^M)-(\pi_N,h))^2\\
\le & \frac{1}{N^4}\left(N\cdot 2^4\|h\|^4+6\cdot\frac{N(N-1)}{2}2^4\|h\|^4\right)\\
= & \frac{1}{N^4}\left(16N\|h\|^4+48N(N-1)\|h\|^4\right)\\
= & \frac{1}{N^4}\left(16N\|h\|^4+48N^2\|h\|^4-48N\|h\|^4\right)\le\frac{48\|h\|^4}{N^2}.
\end{aligned}$$

Then

$$\mathbb{E}\left(\sum_{N=1}^{\infty}\left((m_N\bar{K}_M(\pi_N),h)-(\pi_N,h)\right)^4\right)\le 48\|h\|^4\sum_{N=1}^{\infty}\frac{1}{N^2}<\infty \tag{3.44}$$

therefore,

$$\sum_{N=1}^{\infty}\left((m_N\bar{K}_M(\pi_N),h)-(\pi_N,h)\right)^4<\infty,\text{for most all }\omega\in\Omega. \tag{3.45}$$

hence,

$$\lim_{N\to\infty}|(m_N\bar{K}_M(\pi_N),h)-(\pi_N,h)|=0,\text{for almost all }\omega\in\Omega. \tag{3.46}$$

by definition, for almost all $\omega\in\Omega$, the following holds

$$\lim_{N\to\infty}m^N\bar{K}_M(\pi_N)=\pi,\forall\pi_N,\pi\in\mathcal{P}(\mathbb{R}^{n_x}). \tag{3.47}$$

By the same method, we also have that

$$\lim_{N\to\infty}m^N(\pi_N)=\pi,\forall\pi_N,\pi\in\mathcal{P}(\mathbb{R}^{n_x}). \tag{3.48}$$

□

Theorem 3.4. *If the transition kernel K is Feller and the likelihood function g is bounded, continuous and strictly positive, then for the proposed algorithm that uses one level MCMC, we have $\lim_{N\to\infty} \pi^N_{k|0:k} = \pi_{k|0:k}$ almost surely.*

Proof. Notice that the algorithm can be represented as following,

$$\begin{aligned}\pi^N_{k|0:k} &= m^N \bar{K}_M \circ u_k \circ m^N \circ q_k(\pi^N_{k-1|0:k-1}) = s^N_k(\pi^N_{k-1|0:k-1}) \\ \pi^N_{k|0:k} &= s^N_{1:k} \circ m^N(\pi_0) = s^N_{1:k}(\pi^N_0)\end{aligned}$$

where $\pi^N_0 = m^N(\pi_0)$. By Lemma 3.4.2, we have $\lim_{N\to\infty} \pi^N_0 = \pi_0$, then by the continuity of q_k, we have

$$\lim_{N\to\infty} \pi^N_0 = \pi_0 \Longrightarrow \lim_{N\to\infty} q_k(\pi^N_0) = q_k(\pi_0). \tag{3.49}$$

And by the property of m_N in Lemma 3.4.2, we have

$$\lim_{N\to\infty} q_k(\pi^N_0) = q_k(\pi_0) \Longrightarrow \lim_{N\to\infty} m^N(q_k(\pi^N_0)) = q_k(\pi_0). \tag{3.50}$$

Since u_k is also continuous as we showed before, we have

$$\lim_{N\to\infty} m^N(q_k(\pi^N_0)) = q_k(\pi_0) \Longrightarrow \lim_{N\to\infty} u_k(m^N(q_k(\pi^N_0))) = u_k(q_k(\pi_0)), \tag{3.51}$$

then, employ the property of $m^N \bar{K}_N$,

$$\lim_{N\to\infty} u_k(m^N(q_k(\pi^N_0))) = u_k(q_k(\pi_0)) \Longrightarrow \lim_{N\to\infty} m^N \bar{K}_N(u_k(m^N(q_k(\pi^N_0)))) = u_k(q_k(\pi_0)). \tag{3.52}$$

Therefore,

$$\lim_{N\to\infty} s^N_k(\pi^N_0) = s_k(\pi_0), \tag{3.53}$$

and by induction on time, it is clear that

$$\lim_{N\to\infty} s^N_{1:k}(\pi^N_0) = s_{1:k}(\pi_0). \tag{3.54}$$

□

Now let us consider the case where we use multiple levels of MCMC. In such case, the particles will evolve through different transition kernels designed with different stationary distributions. We define the following perturbation

$$m_N(\pi)\bar{K}^M_L(w) = \frac{1}{N}\sum_{i=1}^{N} \bar{K}^M_L \delta_{x_i(w)}. \tag{3.55}$$

where M denotes the length of Markov chain and L denotes the number of levels.

The key condition we need to check is the following

$$\lim_{N\to\infty} \pi_N = \pi \Longrightarrow \lim_{N\to\infty} m^N \bar{K}^M_L(\pi_N) = \pi, \forall \pi_N, \pi \in \mathcal{P}(\mathbb{R}^{n_x}). \tag{3.56}$$

To show the perturbation satisfies the required condition, the multilevel MCMC process considers all the levels of MCMC sampling before the final level as auxiliary levels. The effect of these auxiliary levels is that at the final level the initial distribution for the particles has been changed, however we will show that the required condition can still be fulfilled in this case. The following definition and proposition is needed for the proof.

Definition 3.5. *The total variation distance between two probability measures* $\nu_1(\cdot)$ *and* $\nu_2(\cdot)$ *is*

$$\|\nu_1(\cdot) - \nu_2(\cdot)\|_{tv} = \sup_A |\nu_1(A) - \nu_2(A)|.$$

Proposition 3.6. $\|\nu_1(\cdot) - \nu_2(\cdot)\|_{tv} = \frac{1}{b-a} \sup_{f:\mathcal{X}\to[a,b]} |\int f d\nu_1 - \int f d\nu_2|$ *for any* $a < b$.

Proof. See Roberts et al. (2004). □

Definition 3.7. *A Markov chain* $\{x_m\}_{m\geq 1} \in \mathbb{R}^n$ *is* $\Phi-$*irreducible if there exists a non-zero* $\sigma-$*finite measure* Φ *on* $\mathbb{R}^n$ *such that for all* $A \subset \mathbb{R}^n$ *with* $\Phi(A) > 0$*, and for all* $x \in \mathbb{R}^n$*, there exists a positive integer* $M = M(x, A)$ *such that* $K^M(x, A) > 0$*.*

Definition 3.8. *A Markov chain with stationary distribution* $\pi(\cdot)$ *is aperiodic if there do not exist* $d \geq 2$ *and disjoint subsets* $\mathcal{X}_1, \cdots, \mathcal{X}_d \subset \mathbb{R}^n$ *with* $K(x, \mathcal{X}_{i+1}) = 1$ *for all* $x \in \mathcal{X}_i$ *with* $1 \leq i \leq d-1$ *and* $K(x, \mathcal{X}_1) = 1$ *for all* $x \in \mathcal{X}_d$*, such that* $\pi(\mathcal{X}_i) > 0$ *for all* i*.*

Theorem 3.9. *If a Markov chain on a state space with countably generated* $\sigma-$*algebra is* $\Phi-$*irreducible and aperiodic, and has a stationary distribution* $\pi(\cdot)$*, the for* $\pi-$*a.e.,*

$$\lim_{M\to\infty} \|K^M(x, \cdot) - \pi(\cdot)\|_{tv} = 0.$$

In particular, $\lim_{M\to\infty} K^M(x, A) = \pi(A)$ *for all measurable* A*.*

Proof. See Roberts et al. (2004). □

Lemma 3.10. *The random perturbation*

$$m_N(\pi)\bar{K}_L^M(w) = \frac{1}{N}\sum_{i=1}^{N} \bar{K}_L^M \delta_{x_i(w)}$$

satisfies the following condition for almost all $w \in \Omega$*,*

$$\lim_{N\to\infty} \pi_N = \pi \Longrightarrow \lim_{N\to\infty} m^N \bar{K}_L^M(\pi_N) = \pi, \forall \pi_N, \pi \in \mathcal{P}(\mathbb{R}^{n_x}).$$

Proof. Let us consider the following expectation, where ν denotes the initial condition and $\bar{K}_L^M$ corresponds to the transition kernel at Mth step of the final level, is taken with respect to the joint distribution of $\{x_i, i = 1 \cdots, N\}$, say $W_{\nu,\bar{K}_L^M}$, by

the independence of the random variables, Proposition 3.6 and Theorem 3.9

$$\begin{aligned}
& \mathbb{E}_{\nu,\bar{K}_L^M}\left(((m_N\bar{K}_L^M(\pi_N),h)-(\pi_N,h))^4\right)\\
= \;& \mathbb{E}_{\nu,\bar{K}_L^M}\left((\frac{1}{N}\sum_{j=1}^{N}h(x_j^M)-(\pi_N,h))^4\right)\\
= \;& \mathbb{E}_{\nu,\bar{K}_L^M}\left(\frac{1}{N^4}(\sum_{j=1}^{N}(h(x_j^M)-(\pi_N,h)))^4\right)\\
= \;& \frac{1}{N^4}\mathbb{E}_{\nu,\bar{K}_L^M}\left(\sum_{j=1}^{N}(h(x_j^M)-(\pi_N,h))^4\right)\\
= \;& \frac{1}{N^4}\mathbb{E}_{\nu,\bar{K}_L^M}\left(\sum_{1\le i,j,k,\ell\le N}\prod_{\tau=i,j,k,\ell}(h(x_\tau^M)-(\pi_N,h))\right) \qquad (3.57)
\end{aligned}$$

Notice that $\forall\epsilon,\exists\mathcal{M},\forall M>\mathcal{M}$, such that the following inequality holds

$$\begin{aligned}
& \mathbb{E}_{\nu,\bar{K}_L^M}\left(h(x_i^M)-(\pi_N,h)\right)\\
= \;& \mathbb{E}_{\nu,\bar{K}_L^M}(h(x_i^M))-(\pi_N,h)\\
= \;& (\nu\bar{K}_L^M,h)-(\pi_N,h)\le\sup_{\|h\|\le B}|\int h\mathrm{d}(\nu\bar{K}_L^M)-\int h\mathrm{d}\pi_N|\\
= \;& 2B\|\nu\bar{K}_L^M-\pi_N\|_{tv}\le\epsilon
\end{aligned}$$

Since $\{x_i^M, i=1,\cdots,M\}$ are independent, then we have

$$\begin{aligned}
&\mathbb{E}_{\nu,\bar{K}_L^M}\left(h(x_i^M)-(\pi_N,h)^3(h(x_j^M)-(\pi_N,h))\right)=0,\\
&\mathbb{E}_{\nu,\bar{K}_L^M}\left(h(x_i^M)-(\pi_N,h)^2(h(x_j^M)-(\pi_N,h))(h(x_k^M)-(\pi_N,h))\right)=0,\\
&\mathbb{E}_{\nu,\bar{K}_L^M}\left(h(x_i^M)-(\pi_N,h)(h(x_j^M)-(\pi_N,h))(h(x_k^M)-(\pi_N,h))(h(x_\ell^M)-(\pi_N,h))\right)=0
\end{aligned}$$

if i,j,k,ℓ are distinct. Therefore, eq.(3.57) reduces to the following

$$
\begin{aligned}
&\frac{1}{N^4}\mathbb{E}_{\nu,\bar{K}_L^M}\left(\sum_{1\le i,j,k,\ell\le N}\prod_{\tau=i,j,k,\ell}(h(x_\tau^M)-(\pi_N,h))\right)\\
=\;&\frac{1}{N^4}\sum_{j=1}^{N}\mathbb{E}_{\nu,\bar{K}_L^M}(h(x_j^M)-(\pi_N,h))^4\\
+\;&\frac{6}{N^4}\sum_{i,j=1,i\neq j}^{N}\mathbb{E}_{\nu,\bar{K}_L^M}(h(x_i^M)-(\pi_N,h))^2(h(x_j^M)-(\pi_N,h))^2\\
+\;&\frac{4}{N^4}\sum_{i,j=1,i\neq j}^{N}\mathbb{E}_{\nu,\bar{K}_L^M}\left(|h(x_i^M)-(\pi_N,h)|^3|h(x_j^M)-(\pi_N,h)|\right)\\
+\;&\frac{12}{N^4}\sum_{\substack{i,j,k=1,\\ i\neq j,i\neq k,j\neq k}}^{N}\mathbb{E}_{\nu,\bar{K}_L^M}\left(|h(x_i^M)-(\pi_N,h)|^2|h(x_j^M)-(\pi_N,h)||h(x_k^M)-(\pi_N,h)|\right)\\
+\;&\frac{24}{N^4}\sum_{\substack{i,j,k,\ell=1\\ i\neq j\neq k\neq \ell}}^{N}\mathbb{E}_{\nu,\bar{K}_L^M}\Bigg(|h(x_i^M)-(\pi_N,h)||(h(x_j^M)-(\pi_N,h)||(h(x_k^M)-(\pi_N,h)|\\
&|(h(x_\ell^M)-(\pi_N,h)|\Bigg)
\end{aligned}
\tag{3.58}
$$

by taking $\epsilon=\frac{1}{N}$ the eq.(3.58) is bounded by the following

$$
\begin{aligned}
&\frac{1}{N^4}\mathbb{E}_{\nu,\bar{K}_L^M}\left(\sum_{1\le i,j,k,\ell\le N}\prod_{\tau=i,j,k,\ell}(h(x_\tau^M)-(\pi_N,h))\right)\\
\le\;&\frac{1}{N^4}\Bigg(N\cdot 2^4\|h\|^4+6\cdot\frac{N(N-1)}{2}2^4\|h\|^4+4\epsilon\cdot\frac{N(N-1)}{2}\|h\|^3\\
+\;&12\epsilon^2\cdot\frac{N(N-1)(N-2)}{6}\|h\|^2+24\epsilon^4\cdot\frac{N(N-1)(N-2)(N-3)}{24}\Bigg)\\
=\;&\frac{1}{N^4}\Bigg(16N\|h\|^4+48N(N-1)\|h\|^4+2(N-1)\|h\|^3\\
+\;&\frac{2}{N}(N-1)(N-2)\|h\|^2+\frac{(N-1)(N-2)(N-3)}{N^3}\Bigg)\\
\le\;&\frac{1}{N^4}\left(16N\|h\|^4+48N^2\|h\|^4-48N\|h\|^4+2\|h\|^3+2N\|h\|^2+1\right)\le\frac{C}{N^2}
\end{aligned}
$$

where $C < \infty$ is an appropriate constant. Then

$$\mathbb{E}_{\nu,\bar{K}_L^M}\left(\sum_{N=1}^{\infty}\left((m_N\bar{K}_L^M(\pi_N),h)-(\pi_N,h)\right)^4\right) \leq C\sum_{N=1}^{\infty}\frac{1}{N^2} < \infty \tag{3.59}$$

therefore,

$$\sum_{N=1}^{\infty}\left((m_N\bar{K}_L^M(\pi_N),h)-(\pi_N,h)\right)^4 < \infty, \text{for most all } \omega \in \Omega. \tag{3.60}$$

hence,

$$\lim_{N\to\infty}|(m_N\bar{K}_L^M(\pi_N),h)-(\pi_N,h)| = 0, \text{for almost all } \omega \in \Omega. \tag{3.61}$$

by definition, for almost all $\omega \in \Omega$, the following holds

$$\lim_{N\to\infty} m^N\bar{K}_L^M(\pi_N) = \pi, \forall\pi_N, \pi \in \mathcal{P}(\mathbb{R}^{n_x}). \tag{3.62}$$

By the same method, we also have that

$$\lim_{N\to\infty} m^N(\pi_N) = \pi, \forall\pi_N, \pi \in \mathcal{P}(\mathbb{R}^{n_x}). \tag{3.63}$$

□

Theorem 3.11. *If the transition kernel K is Feller and the likelihood function g is bounded, continuous and strictly positive, then for the proposed algorithm uses multilevel MCMC, we have $\lim_{N\to\infty}\pi_{k|k}^N = \pi_{k|k}$ almost surely.*

Proof. Notice that the algorithm can be represented as following,

$$\begin{aligned}\pi_{k|0:k}^N &= m^N\bar{K}_L^M \circ u_k \circ m^N \circ q_k(\pi_{k-1|0:k-1}^N) = s_k^N(\pi_{k-1|0:k-1}^N)\\ \pi_{k|0:k}^N &= s_{1:k}^N \circ m^N(\pi_0) = s_{1:k}^N(\pi_0^N)\end{aligned}$$

where $\pi_0^N = m^N(\pi_0)$. By Lemma 3.4.2, we have $\lim_{N\to\infty} \pi_0^N = \pi_0$, then by the continuity of q_k, we have

$$\lim_{N\to\infty} \pi_0^N = \pi_0 \Longrightarrow \lim_{N\to\infty} q_k(\pi_0^N) = q_k(\pi_0). \tag{3.64}$$

And by the property of m_N in Lemma 3.4.2, we have

$$\lim_{N\to\infty} q_k(\pi_0^N) = q_k(\pi_0) \Longrightarrow \lim_{N\to\infty} m^N(q_k(\pi_0^N)) = q_k(\pi_0). \tag{3.65}$$

Since u_k is also continuous as we showed before, we have

$$\lim_{N\to\infty} m^N(q_k(\pi_0^N)) = q_k(\pi_0) \Longrightarrow \lim_{N\to\infty} u_k(m^N(q_k(\pi_0^N))) = u_k(q_k(\pi_0)), \tag{3.66}$$

then, employ the property of $m^N \bar{K}_L^M$,

$$\lim_{N\to\infty} u_k(m^N(q_k(\pi_0^N))) = u_k(q_k(\pi_0)) \Longrightarrow \lim_{N\to\infty} m^N \bar{K}_L^M(u_k(m^N(q_k(\pi_0^N)))) = u_k(q_k(\pi_0)). \tag{3.67}$$

Therefore,

$$\lim_{N\to\infty} s_k^N(\pi_0^N) = s_k(\pi_0), \tag{3.68}$$

and by induction on time, it is clear that

$$\lim_{N\to\infty} s_{1:k}^N(\pi_0^N) = s_{1:k}(\pi_0). \tag{3.69}$$

□

Chapter 4

Numerical applications

4.1 Double well potential dynamics

4.1.1 Double well potential with rugged energy landscape

Understanding the static and dynamical behaviors of complex physical systems is one of the most challenging problems of modern research in physics, chemistry and biology. The dynamics of many physical systems can be modeled as being driven by a double well potential. Such potential has vast applications in modeling dynamics in quantum mechanics, chemistry, and biology Janke (2007); Zwanzig (1988). For instance, ammonia molecule, which is a key ingredient of Ammonia Maser, has two equilibrium states. The nitrogen stays either on top or below of the hydrogens and the transition from one state to the other is blocked by a energy barrier. This phenomenon is well described by a double well potential. Also, in phase field models, in order to describe some given interfacial dynamics, one has to choose a well suited free energy function. The double well potential, as a popular choice of free energy function, is used to simulate the behavior of interface.

Furthermore, in many scientific disciplines such as physics and chemistry, a smooth double well potential may not be sufficient in describing the complicated characteristics of the objects. Therefore, a more sophisticated free energy model is

required. Examples such as spin glass, structural glass and protein folding are of great scientific interest. While the concepts in different research realms are in many respects quite diverse, certain common features coexist in all these systems. One of the most prominent joint key features of those fields is rugged free-energy landscape which generates multi-modality. For example, the study of Frauenfelder Zwanzig (1988) suggests that the potential surface of a protein might have a hierarchical structure with potential minima within potential minima, etc. The typical model of the rugged free-energy landscapes for the protein folding problem and spin glass problem is shown in Fig.4.1.

Let's consider a typical smooth double well potential function given by

$$U_\theta(x) = \frac{\theta}{2}(x^4 - 2x^2), \tag{4.1}$$

where $\theta \in \mathbb{R}$ is a parameter that controls the shape of the potential well and x denotes the state of dynamical system. A rugged energy landscape can be modeled by a potential function, defined in (4.1), superimposed by another oscillating function. The superimposed function describes many small potential barriers distributed in a random way. Potential model of this type has been used in Zwanzig (1988) to simulate the dynamical behavior of a protein. To this end, let us consider the following SDE

$$dx_k^\epsilon = -2\theta x_k^\epsilon({x_k^\epsilon}^2 - 1)dt - \frac{\epsilon}{\delta}(cos(\frac{x_k^\epsilon}{\delta}) - sin(\frac{x_k^\epsilon}{\delta}))dt + c(x_k)dB_k. \tag{4.2}$$

Notice that eq.(4.2) consists of two parts. The drift part, which describes the gradient flow of a rugged double well potential

$$U_\theta^\epsilon(x, x/\delta) = \frac{\theta}{2}(x^4 - 2x^2) + \epsilon(cos(\frac{x}{\delta}) + sin(\frac{x}{\delta})), \tag{4.3}$$

and the random perturbation part adjusted by a diffusion coefficient. A small noise diffusion coefficient, for example $c(x_k) = \frac{1}{4}$, weakens the effect of random noise and

yields rare transitions. Moreover, when $\epsilon = 0$, eq. (4.3) yields a smooth double well potential defined in eq. (4.1) which has two minima at ± 1, see Fig. 4.1. The state of the system described by eq. (4.2) wanders around one of the two equilibrium states, i.e. ± 1, depending on the initial condition. The frequency of transitions between the two states is determined by the stochastic perturbations and the effect of the oscillating term. For example, Fig. 4.2 illustrates the dynamics described by eq. (4.2) for 1,000 time steps. We observe that such transitions occur only once. If one increases the diffusion coefficient of eq. (4.2), then frequent jumps, which may not depict the reality, happen. One the other hand, such transitions and the evolution of the system is typically observed via data. Consider that the observations are additive Gaussian perturbations of the state process, i.e.

$$y_k = x_k + v_k, \tag{4.4}$$

where x_k generated by eq.(4.2), and the integrated noise v_k is distributed according to $\mathcal{N}(0, \sigma^2)$ and is independent of the noise in eq.(4.2).

Next, we describe the drift homotopy and likelihood bridging particle filter. Take into account the following $L+1$ levels of dynamics with the same diffusion coefficient,

$$\begin{aligned} dx^{\epsilon}_{k,\ell} &= -\beta(1-\epsilon_\ell)\left[2\theta x^{\epsilon}_{k,\ell}({x^{\epsilon}_{k,\ell}}^2 - 1) - \frac{\epsilon}{\delta}(cos(\frac{x^{\epsilon}_{k,\ell}}{\delta}) - sin(\frac{x^{\epsilon}_{k,\ell}}{\delta}))\right]dt \\ &- \epsilon_\ell\left[2\theta x^{\epsilon}_{k,\ell}({x^{\epsilon}_{k,\ell}}^2 - 1) - \frac{\epsilon}{\delta}(cos(\frac{x^{\epsilon}_{k,\ell}}{\delta}) - sin(\frac{x^{\epsilon}_{k,\ell}}{\delta}))\right]dt + \frac{1}{4}dB_k, \end{aligned} \tag{4.5}$$

where $\ell = 0, \cdots, L$ and $\beta \in [0, 1]$ is the coefficient controlling the steepness of the potential wells. Specifically, a smaller β corresponds to a double well potential with shallower wells, hence the samples will have a greater chance of moving between the two equilibria given the relatively small stochastic noise. The transition densities $f_\ell(x_k|x_{k-1})$, for $\ell = 0, \cdots, L$, are associated with the modified SDEs defined in eq.(4.5). The likelihood $g(y_k|x_k)$, which is defined by eq.(4.4), is introduced via the sequence of the bridging densities, i.e., $g^m(y_k|x_k)$ for $m = \epsilon_\ell$. Consider, now,

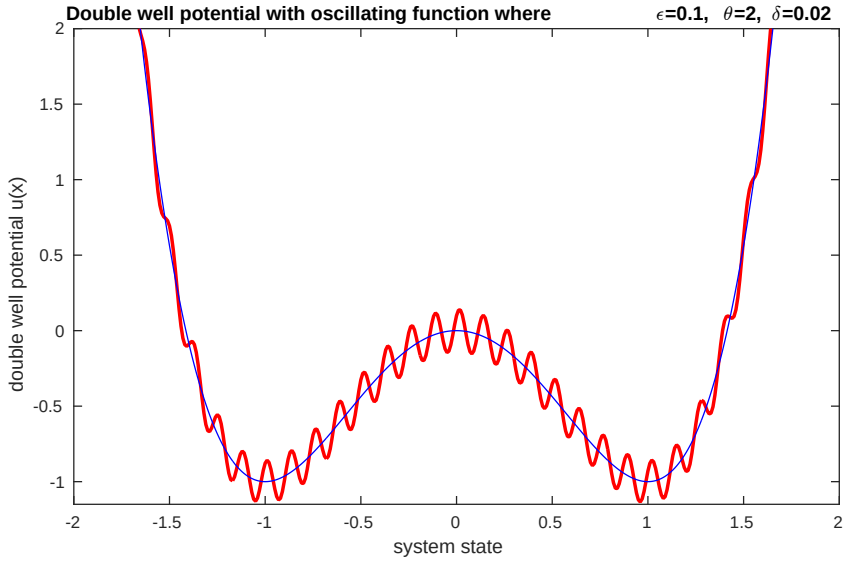

Figure 4.1: The blue line shows the smooth double well potential defined in eq.(4.1) and the red curve shows the oscillating double well potential (rugged free-energy landscape) defined in eq.(4.3).

a partition of the time interval from $k-1$ to k into subintervals, denoted by $x_{k-1} = x_{k-1}^0 \leq x_{k-1}^1 \leq \cdots \leq x_{k-1}^I = x_k$. The stationary distribution at level ℓ for the conditional path sampling is $g^m(y_k|x_k)\prod_{\lambda=0}^{I-1} f_\ell(x_{k-1}^{\lambda+1}|x_{k-1}^\lambda)$ which is proportional to

$$\exp\left[-\frac{m(y_k - x_t)^2}{2\sigma^2}\right]\prod_{\lambda=0}^{I-1}\exp\left[-\frac{(x_{k-1}^{\lambda+1} - \mathcal{K}_\ell(x_{k-1}^\lambda))^2}{2\Delta t}\right], \tag{4.6}$$

where $\ell = 0, \cdots, L$, $m = \frac{\ell}{L}$, and

$$\begin{aligned}\mathcal{K}_\ell(x_{k-1}^\lambda) &= -\beta(1-\epsilon_\ell)\left[2\theta x_{k-1}^\lambda({x_{k-1}^\lambda}^2 - 1) - \frac{\epsilon}{\delta}(cos(\frac{x_{k-1}^\lambda}{\delta}) - sin(\frac{x_{k-1}^\lambda}{\delta}))\right]\Delta t \\ &\quad -\epsilon_\ell\left[2\theta x_{k-1}^\lambda({x_{k-1}^\lambda}^2 - 1) - \frac{\epsilon}{\delta}(cos(\frac{x_{k-1}^\lambda}{\delta}) - sin(\frac{x_{k-1}^\lambda}{\delta}))\right]\Delta t.\end{aligned}$$

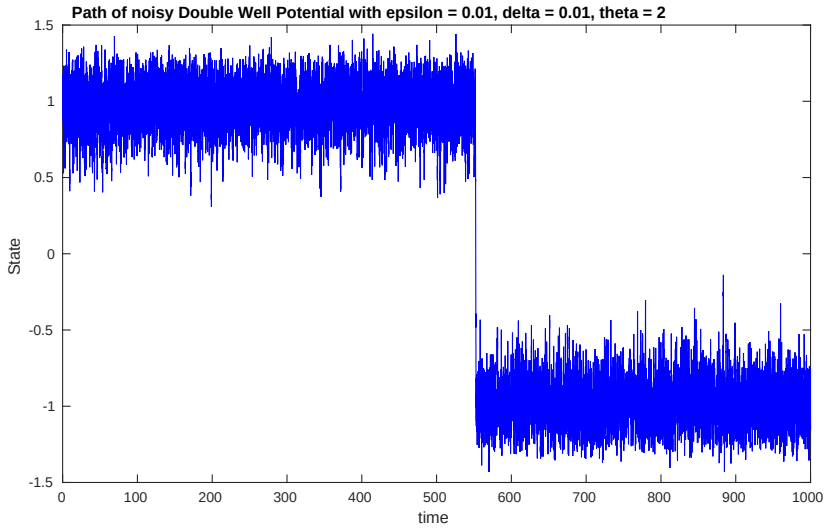

Figure 4.2: Trajectory described by eq. (4.2) with Gaussian noise of variance 0.25 and time step size 0.01. We perform 1000 time steps simulation and there is only one jump occurred.

In the numerical simulations, we define $\epsilon = 0.1, \theta = 2, \delta = 0.02$ and $\epsilon_\ell = \frac{\ell}{L},\ \ell = 0, \cdots, L$ for eq.(4.5). We track x_k over the time interval $[0, T]$, where $T = 10$ is the terminal time. Starting at $k = 0$, the observations arrive at every 0.1 time step and transition occurs at integer-valued time steps specifically $y_k = 1$ when $k = 2i + 1,\ i = 0, \cdots, 4$, and $y_k = -1$ when $k = 2i,\ i = 1, \cdots, 4$. The variance of the Gaussian noise, v_k, in the observation model is set to be $\sigma^2 = 0.05$. In addition, the Euler-Maruyama discretization scheme is employed to simulate the dynamics (4.2) where time increment is $\Delta t = 0.01$. For the appended MCMC step, we employ the Metropolis-Hastings scheme and set $L = 50$. For each stationary distribution in the sequence, we run 10 MCMC steps. The tracking results of 3 different particle filters are shown in Fig. 4.3. It indicates that the generic particle filter (also called sequential importance sampling resampling or abbreviated SISR) can fail to capture the rare transitions. APF is able to capture the jumps between two equilibrium states when enough observations become available. DHLB-PF can track the rare transitions with a lot less data. To show the algorithm is very effective, we plot the change of the

effective sample sizes (ESS) of the three particle filters as the level increases in Fig. 4.4 at the time when the first jump from 1 to -1 occurred. A detailed description is given later in Section 4.3. The ESS for all three particle filters behaves very similarly when jumps occur, therefore the corresponding figures are omitted. The ESS of DHLB-PF increases from 0 to around 800 out of 800 samples, whereas the ESS's of the other two algorithms are close to zero.

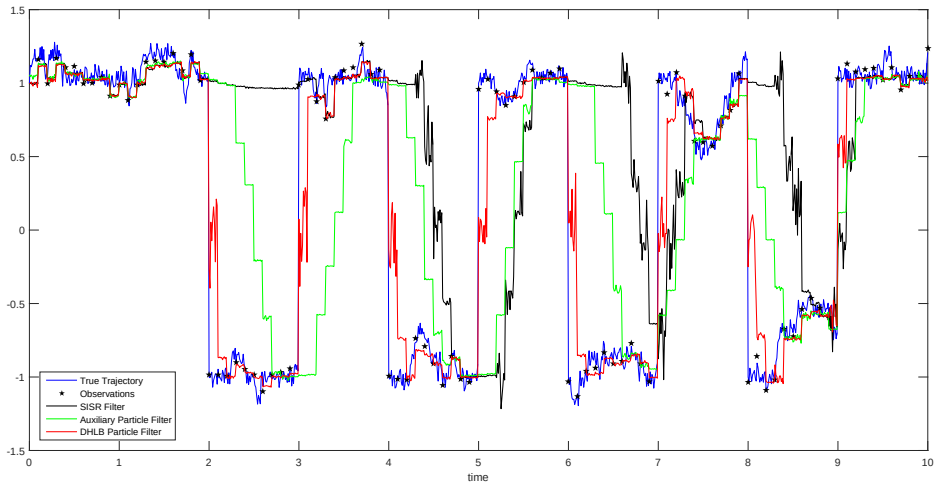

Figure 4.3: This picture shows the tracking results of three particle filters. We use 800 samples for each filter and 50 levels for the DHLB-PF algorithm.

Notice that our algorithm outperforms the other two popular filter schemes and behaves very stable as time evolves. The speed of relocation of the empirical filtering distribution to the true filtering distribution indicates the efficiency of the algorithm, especially in the case of rare transitions. However, as we can see in Fig. 4.3, the SISR filter and APF are not able to capture the jumps immediately. This phenomenon is compounded in the sequential setting since the errors are cumulated as time evolves, it coincides with the numerical results where the performances of the other filters are very unstable in tracking the rare transitions. With the given setting in the observation model, the true filtering distribution will move most of its probability mass around -1 after 2 or 3 observations. Fig. 4.5 shows that DHLB-PF needs about the same number of observations to relocate the empirical filtering distribution. SISR

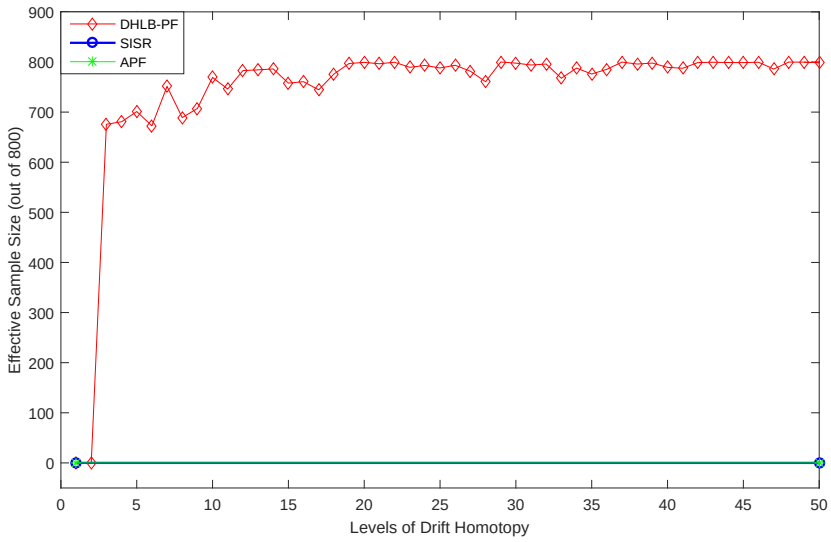

Figure 4.4: The comparison of the effective sample sizes at the time instant 2 when the first rare transition occurred. As shown in the figure, DHLB-PF is able to bring up the effective sample size as level increases, whereas the effective sample sizes of the other two methods, GPF and APF, are close to zero. (Notice that there is no levels for GPF and APF hence the ESS for these two filters does not change in the figure.)

fails to relocate the empirical filtering distribution at the first jump and it needs lots of observation data to move the empirical filtering distribution. APF performs better than SISR as it moves the empirical filtering distribution to the statistically significant region much more quickly than SISR, however still much slower than DHLB-PF.

4.2 Multi-target tracking problem

4.2.1 Multi-target tracking in a wireless sensor network without fusion center

Adopting the DHLB-PF and the sparsity aware matrix decomposition frameworks, this section employs this novel framework in a multi-target tracking problem in a

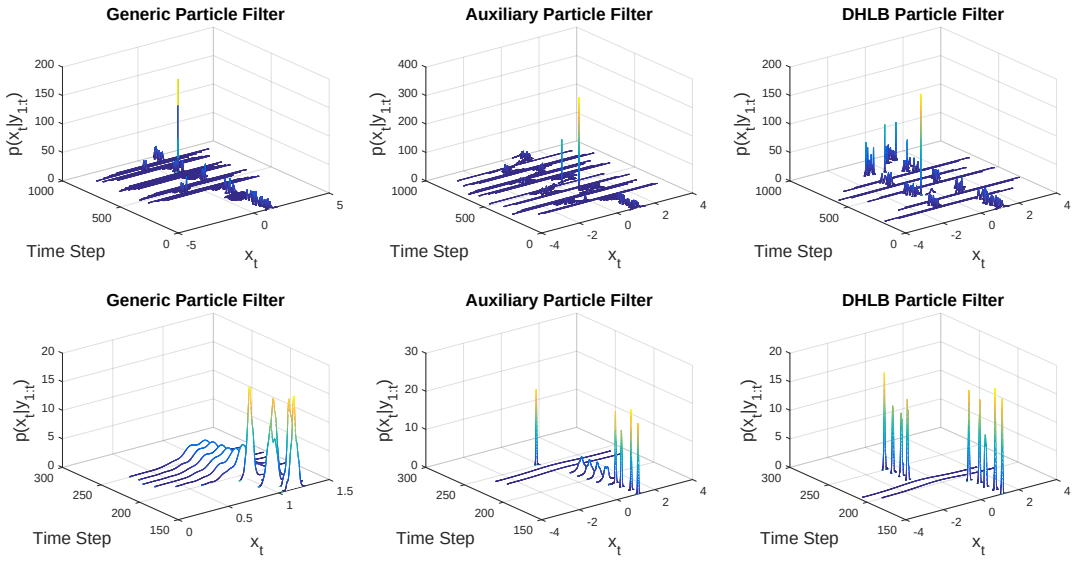

Figure 4.5: The first row in the figure shows the empirical filtering distribution $p(x_k|y_{1:k})$ at each time step when observation is available. The second row in the figure zooms out around the time steps when the first rare transition occurs. It is clear that GPF fails to relocate the empirical filtering distribution, and APF takes about 6 observation data to relocate the empirical filtering distribution, whereas DHLB-PF only needs two observational data to move the empirical filtering distribution.

wireless sensor network (WSN). We first discuss the problem and how we produce the synthetic data, then present numerical results.

Problem Formulation

We assume that the targets move in a 2 dimensional space. The state of the m^{th} target at time instant k is represented by a vector $\mathbf{x}_k^m = [x_k^m, \dot{x}_k^m, y_k^m, \dot{y}_k^m]$, where $(x_k^m, \dot{x}_k^m)$ and $(y_k^m, \dot{y}_k^m)$ are the position and velocity on the x and y axes respectively. At each time instant k, there are M_k moving targets, and the m^{th} target evolves according to the following dynamics

$$\mathbf{x}_k^m = \mathbf{B}\mathbf{x}_{k-1}^m + \mathbf{C}u_k^m, \tag{4.7}$$

where the matrices $\mathbf{B} = diag\{P, P\}$ and $\mathbf{B} = diag\{Q, Q\}$ are as follows $\mathbf{P} = \begin{pmatrix} 1 & \delta T \\ 0 & 1 \end{pmatrix}$, $\mathbf{Q} = (\delta T^2/2, \delta T)'$, where $(\cdot)'$ denotes the transpose of a vector, and δT denotes the time lag between observations which is set to be 1 in our numerical experiments. The model noise u_k^m is a Gaussian random vector that consists of two independent Gaussian random variables, i.e. $u_k^m = (u_{x,t}^m, u_{y,t}^m)$, with $\mu_u^m = 0$ and covariance $\Sigma_u^m = diag\{\sigma_x^2, \sigma_y^2\}$ where $\sigma_x^2 = 0.7, \sigma_y^2 = 0.7$ is used in the simulations.

The synthesized target tracks were created by evolving a number of targets according to (4.11) and recording the state of each target at each time step. The observations were obtained based on the model (2.2). The number of targets at each time instant is $M_k = 10$ for all k.

Drift homotopy and likelihood bridging

The modified dynamics of the m^{th} target based on the drift homotopy is given by

$$\mathbf{x}_k^{\ell,m} = \mathbf{B}\mathbf{x}_{k-1}^{\ell,m} + \mathbf{D}^{\ell,m} + \mathbf{C}u_k^m, \tag{4.8}$$

where $\mathbf{D}^{\ell,m} = (1-\epsilon_\ell)(\mu^m_{x,k-1}\frac{\delta T^2}{2}, \mu^m_{x,k-1}\delta T, \mu^m_{y,k-1}\frac{\delta T^2}{2}, \mu^m_{y,k-1}\delta T)'$, and $\epsilon_\ell = \ell/L$, and $\ell = 0, \cdots, L$. In the numerical experiments we set $L = 10$, namely, 10 levels of drift homotopy and likelihood bridging, and for the n^{th} sample at time instant $t-1$ we define

$$\begin{aligned}
\mu^m_{n,x,k-1} &= \frac{\bar{\mu}^m_x - x^m_{n,k-1}}{\delta T^2/2} - \frac{2\dot{x}^m_{n,k-1}}{\delta T}, \\
\mu^m_{n,y,k-1} &= \frac{\bar{\mu}^m_y - y^m_{n,k-1}}{\delta T^2/2} - \frac{2\dot{y}^m_{n,k-1}}{\delta T},
\end{aligned}$$

where $\bar{\mu}^m_x$ and $\bar{\mu}^m_y$ are mean drifts which offset the individual sample's properties such that

$$\begin{aligned}
\bar{\mu}^m_x &= \frac{1}{N}\sum_{n'=1}^{N}(x^m_{n',k-1} + \dot{x}^m_{n',k-1}\delta T), \\
\bar{\mu}^m_y &= \frac{1}{N}\sum_{n'=1}^{N}(y^m_{n',k-1} + \dot{y}^m_{n',k-1}\delta T).
\end{aligned}$$

In order to represent the path between two observations, we need to partition the time interval between time instants $k-1$ and k using $\Delta t = 0.1$. We consider a partition on the time interval from $k-1$ to k into I subintervals, denoted the states by $x_{k-1} = x^0_{k-1} \leq x^1_{k-1} \leq \cdots \leq x^I_{k-1} = x_k$. The stationary distribution at level ℓ for the nth sample using drift homotopy and likelihood bridging can be expressed as

$$\prod_{m=1}^{M_k}\left[g^{\epsilon_\ell}_m(y^m_{k,\hat{h}_{m,k}}|\mathbf{x}^m_{n,k})\prod_{\lambda=0}^{I-1}f_{\ell,m}(\mathbf{x}^{m,\lambda+1}_{n,k-1}|\mathbf{x}^{m,\lambda}_{n,k-1})\right],$$

where $y^m_{k,\hat{h}_{m,k}}$ are the observations of informative sensors instead of all sensor measurements. Specifically, the stationary distribution at level ℓ is proportional to

$$\begin{aligned}
&\prod_{m=1}^{M_k}\exp\Bigg(\frac{-\|y^m_{k,\hat{h}_{m,t}} - d_m(\mathbf{x}^m_{n,k})\|^2}{2\sigma_w^2} \\
&+ \sum_{\lambda=0}^{I-1}\left[\frac{(x^{m,\lambda+1}_{n,k-1} - \mathcal{S}_\ell(x^{m,\lambda}_{n,k-1}))^2}{(\Delta t)^4\sigma_x^2/2} + \frac{(y^{m,\lambda+1}_{n,k-1} - \mathcal{S}_\ell(y^{m,\lambda}_{n,k-1}))^2}{(\Delta t)^4\sigma_y^2/2}\right]\Bigg)
\end{aligned} \tag{4.9}$$

where $\mathcal{S}_\ell(\cdot)$ is defined as

$$\begin{aligned}\mathcal{S}_\ell(x^{m,\lambda}_{n,k-1}) &= x^{m,\lambda-1}_{n,k-1} + x^{m,\lambda-1}_{n,k-1}\Delta t + (1-\epsilon_\ell)\mu^m_{n,x,k-1}\frac{\Delta t^2}{2} + \frac{\Delta t^2}{2}u^m_{x,k}\\ \mathcal{S}_\ell(y^{m,\lambda}_{n,k-1}) &= y^{m,\lambda-1}_{n,k-1} + y^{m,\lambda-1}_{n,k-1}\Delta t + (1-\epsilon_\ell)\mu^m_{n,y,k-1}\frac{\Delta t^2}{2} + \frac{\Delta t^2}{2}u^m_{y,k}\end{aligned}$$

and

$$d_m(\mathbf{x}^m_{n,k}) = [d^{-2}_{\rho_1,m,n}, d^{-2}_{\rho_2,m,n}, \cdots, d^{-2}_{\rho_{|\hat{h}_{m,k}|},m,n}]^T,$$

where $|\hat{h}_{m,k}|$ denotes the number of informative sensors at time instant k around target m, namely it is the number of nonzero entries in the vector $\hat{h}_{m,k}$ which is obtained from the decomposition algorithm, $\rho_1, \cdots, \rho_{|\hat{h}_{m,k}|}$ are the indices for informative sensors nearby target m and $d_{\rho_i,m,n}, i = 1, \cdots, |\hat{h}_{m,k}|$ denotes the Euclidean distance between sensor ρ_i and target m, and n denotes the sample index in the particle cloud. We should emphasize that the implementation and choice of MCMC sampler in the appended MCMC step is crucial for the algorithm to work efficiently. A naive choice of an MCMC method will increase the computational costs. In this application of multi-target tracking in WSN, we use the generalized hybrid Monte Carlo sampler as described in Section 3.3.2.

Simulation results

Fig. 4.6 displays the tracking estimates based on three particle filters, i.e. the generic particle filter (GPF), the auxiliary particle filter (APF) and our approach the called drift homotopy likelihood bridging particle filter (DHLB-PF). We observe that the DHLB-PF successfully detects the tracks of the moving targets in the sensing field, whereas, the other two popular particle filters fail to follow the moving targets in the sensing field. The performances of three particle filters are measured by the root-mean-square error (RMSE) at time k which is defined with reference to the true target

tracks by the following formula,

$$\mathrm{RMSE}(k) = \sqrt{\frac{\sum_{m=1}^{M_k} \|\mathbf{x}_k^m - E(\mathbf{x}_k^m|y_1, \cdots, y_k)\|^2}{M_k}}, \tag{4.10}$$

where $E(\mathbf{x}_k^m|y_1, \cdots, y_k)$ is the conditional expectation estimate provided byDHLB-PF, and $\mathbf{x}_k^m$ denotes the true state of target m at time instant k. We only extract position information to calculate the RMSE at each time instant and the result is shown in Fig. 4.7. The number of active sensors during the tracking process is provided in Fig.

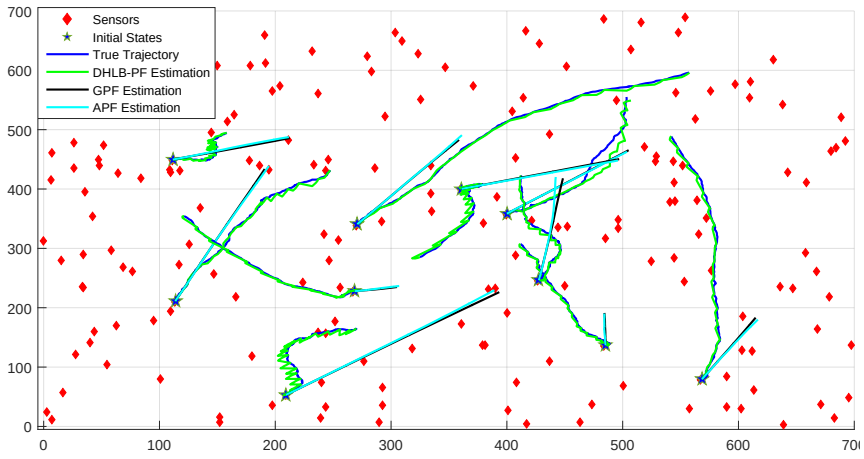

Figure 4.6: Tracking results of three particle filters in the wireless sensor network, including GPF, APF and DHLB-PF. The red diamonds denote the sensors in the field (200 sensor are deployed). The Blue stars are the initial positions of the 10 moving targets. The blue trajectories denote the true moving tracks of the targets and the green curves denote the tracking estimates obtained using DHLB-PF algorithm. Black track and cyan track represent the tracking results of GPF (10000 samples) and APF(1000 samples) respectively. We assume the targets are moving from time instant 0 to time instant 17. As we can see from the results, the DHLB-PF is competent for following the moving targets in the wireless sensor network and outperforms the other two particle filters.

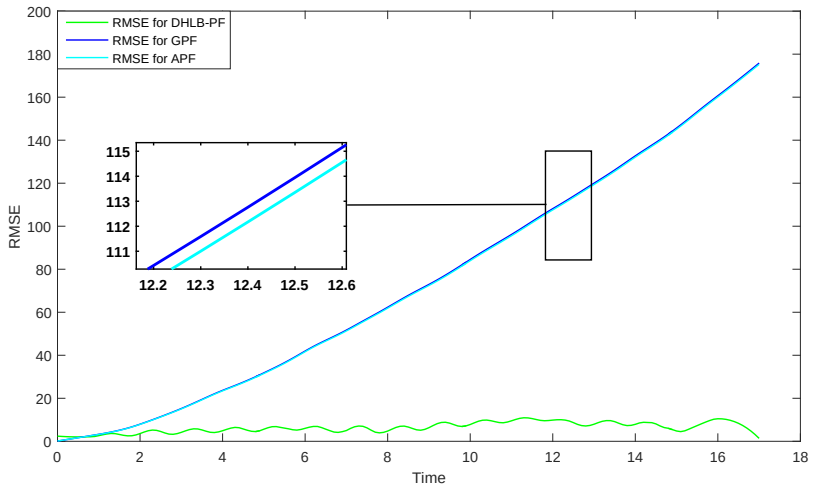

Figure 4.7: RMS errors comparison of GPF, APF and DHLB-PF algorithms. The RMS error is computed using eq. (4.10). The results show that the RMS error maintains at the very low level as the time evolves for DHLB-PF. However, the RMS errors for the other two particle filters blow up quickly. The small window shows magnified difference of RMS errors between GPF and APF in which we can see that APF performs slightly better than GPF.

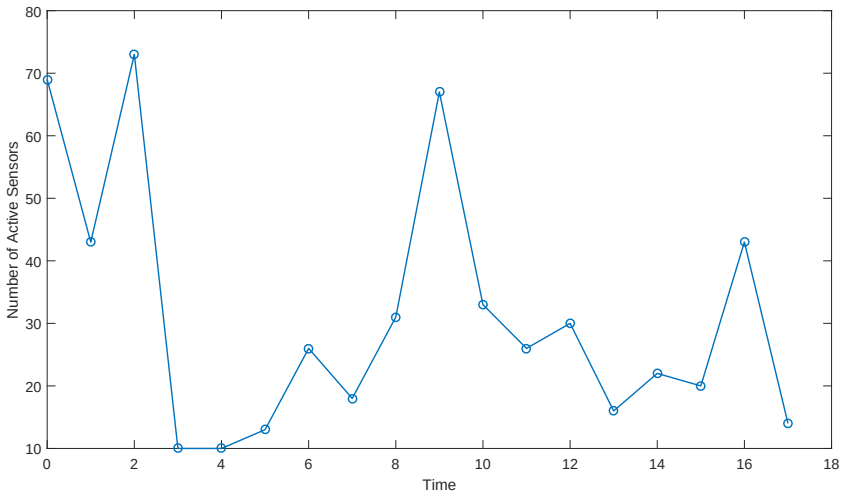

Figure 4.8: The number of active sensors during the tracking process. The total number of sensors deployed in the WSN is 200. As we can see, only a small portion of the WSN is effected during the tracking process.

4.3 A learning parameter in DHLB-PF

Section 4.2 as well as the studies in Maroulas and Stinis (2012); Kang and Maroulas (2013); Kang et al. (2014) consider a fixed number of L levels which were employed. However, in most cases, the MCMC may achieve a convergent result prior to going through all the auxiliary levels of drift homotopy. In other words, computational time is unnecessarily spent. However, if one considers a surveillance distributed sensor network Ren et al. (2015) which employs a particle filter method for monitoring threats, then it is of paramount importance to execute quickly and accurately the procedure since there exist stringent power constraints. In other words, a technique, which decreases the number of levels such that computational time is saved, is urgently needed. Therefore, we introduce a learning method within the MCMC sampler in the particle filter. The learning method automatically adjusts the number of levels, ℓ_k, at a given time k.

The learning criterion is the effective sample size (ESS) as defined in eq.(2.34). The ESS is a measure of how much the samples at any given time k contribute to the approximation of the filtering distribution.The novel learning drift homotopy particle filter calculates the ESS after each level of drift homotopy at each time step when observations are available. Suppose that one generates N i.i.d samples from the importance distribution $q(x)$, another equivalent ESS formula is defined by,

$$\mathrm{ESS}_\ell = \frac{N}{1+\mathrm{CV}^2_{N,\ell}},$$

where $\mathrm{CV}_{N,\ell}$ is the coefficient of variation of the normalized weights given by

$$\mathrm{CV}_{N,\ell} = \left(\frac{1}{N} \sum_{i=1}^{N} \left(\frac{N w_i^\ell}{\sum_{j=1}^{N} w_j^\ell} - 1 \right)^2 \right)^{1/2},$$

where w_i^ℓ and w_j^ℓ denote the importance weights for the i^{th} and j^{th} particles respectively after ℓ^{th} level of drift homotopy. Notice the weights w_i^ℓ and w_j^ℓ are still

calculated by substituting x_k^i by x_{k,ℓ_k}^i, where x_{k,ℓ_k}^i denotes the i^{th} sample at ℓ_k^{th} level at time step k. Based on the definition of ESS, its value is between 1 and N. If the particles with equal weights $\frac{1}{N}$ are considered, then the $\mathrm{CV}_{N,\ell}$ will be equal to zero. On the other hand, if all the normalized weights but one are null, then the $\mathrm{CV}_{N,\ell}^2$ will reach its maximum value $N-1$ and therefore the ESS will be just 1. Also, the ESS reveals that using N weighed samples generated from the importance density to approximate the filtering distribution is equivalent to using $\frac{N}{1+\mathrm{CV}_{N,\ell}^2}$ i.i.d samples drawn from the filtering distribution Cappé et al. (2005); Liu (2008).

We can employs the ESS at each time step when observations are available. Since ESS indicates the number of samples that essentially contributes to the estimation, if the ESS exceeds an appropriate threshold, it implies that the MCMC step converged to the filtering distribution, and therefore no more levels in the drift homotopy are needed.

In the following, we present two examples to illustrate the advantages of introducing the learning parameter. The first numerical experiment is performed for the double well potential dynamics and the second is a multi-target tracking problem with linear and non-linear observation models.

4.3.1 Smooth double well potential dynamics

For the double well potential dynamics, we employ the Metropolis-Hasting algorithm as the MCMC sampler at each level $\ell_k = 0, \cdots, L$ of each time step k. One may fix both the number of levels, and the steps of MCMC sampling at each time step. However, this leads to unnecessary consumption of the computational time which may be disastrous if one tracks threats with a distributed surveillance sensor network. Therefore, we use the ESS as the parameter that controls levels such that the number of levels, which are needed in order to reach a convergent result, decreases drastically. Moreover, a comparison of two ways of implementations is shown in Table 4.1 and Table 4.2. Table 4.1 displays the results of a DHLB-PF with fixed number of levels

Table 4.1: This table shows, for fixed 40 levels of drift homotopy and 1 level of likelihood bridging and 150 MCMC steps, the error at each time step which is simply the difference between true state and estimation. The number of particles is considered to be 10. The experiment considers the smooth double well potential energy.

Time steps	L	MCMC steps	Error
1	40	150	0.016962
2	40	150	0.058091
3	40	150	0.026314
4	40	150	0.046993
5	40	150	0.032146
6	40	150	0.012817
7	40	150	0.012912
8	40	150	0.034872
9	40	150	0.044780
10	40	150	0.026893
11	40	150	0.029236
12	40	150	0.015401
13	40	150	0.071464
14	40	150	0.036774
15	40	150	0.038505
16	40	150	0.039479
17	40	150	0.071885
18	40	150	0.064279
19	40	150	0.000960
20	40	150	0.002162

($L = 40$) and 150 MCMC steps. Table 4.2 uses the learning DHLB-PF where the ESS threshold is set to be 75%. One may observe that we have comparable errors however with significantly less levels of drift homotopy and MCMC steps. Also, the tracking error, which is defined as the distance between estimated state and the true state, is given in the tables for demonstration of the filtering estimation of the partially observed diffusion. In fact, there were a few instances ($t = 2, 6, 14$) where sampling directly from the modified dynamics ($\ell = 0$) was sufficient for the filter to reach a convergent result.

Table 4.2: This table shows the levels performed before the final level $L = 40$ and the MCMC steps, and the error at each time step which is simply the difference between true state and estimation. The sample size is set to be 10. The ESS threshold is set to be 75% of the sample size. The experiment considers the smooth double well potential energy.

Time steps	ℓ_k	MCMC steps	Error
1	22	10	0.049871
2	0	1	0.046551
3	1	29	0.048307
4	2	49	0.044896
5	7	83	0.049962
6	0	12	0.004473
7	1	37	0.049107
8	6	57	0.045660
9	24	5	0.048551
10	5	97	0.047620
11	15	15	0.043014
12	9	92	0.046975
13	9	58	0.048404
14	0	45	0.049052
15	3	29	0.047892
16	11	82	0.046791
17	5	70	0.046585
18	6	15	0.041894
19	10	35	0.049664
20	6	44	0.047678

We next study the performance for a multi-target tracking problem. We consider two observation models: a linear Gaussian model and a nonlinear non-Gaussian model.

4.3.2 Case 1: Linear Gaussian model

Suppose there are m targets and the state vector of the m^{th} target at time k is represented via $\mathbf{x}_k^m = [x_k^m, \dot{x}_k^m, y_k^m, \dot{y}_k^m]$, where $(x_k^m, \dot{x}_k^m)$ and $(y_k^m, \dot{y}_k^m)$ are the position and velocity on the x and y axes respectively. The dynamic of each target is given by

$$\mathbf{x}_k^m = \mathbf{A}_1 \mathbf{x}_{k-1}^m + \mathbf{A}_2 \mathbf{u}_k^m, \tag{4.11}$$

where the matrices $\mathbf{A}_1$ and $\mathbf{A}_2$ are as follows

$$\mathbf{A}_1 = \begin{pmatrix} 1 & \Delta & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \Delta \\ 0 & 0 & 0 & 1 \end{pmatrix}, \mathbf{A}_2 = \begin{pmatrix} \Delta^2/2 & 0 \\ \Delta & 0 \\ 0 & \Delta^2/2 \\ 0 & \Delta \end{pmatrix}$$

and $\Delta = 1$ denotes the time between observations. The noise $\mathbf{u}_k^m$ is distributed according to a 2 dimensional Gaussian distribution with mean 0 and covariance,

$$\Sigma_u^m = \begin{pmatrix} 0.7 & 0 \\ 0 & 0.7 \end{pmatrix}.$$

In our simulation, a linear observation model is considered

$$\mathbf{y}_k^n = \tilde{\mathbf{x}}_k^m + \mathbf{v}_k^n, \tag{4.12}$$

where $\tilde{\mathbf{x}}_k^m = (x_k^m, y_k^m)^T$ is the position of the m^{th} target at time k and $\mathbf{v}_k^n$ is a Gaussian noise with covariance

$$\Sigma_v^n = \begin{pmatrix} 0.004 & 0 \\ 0 & 0.004 \end{pmatrix}.$$

We do not have prior knowledge of target-to-observation association and therefore we use the Munkres algorithm Burkard et al. (2009) to match the observations with targets.

4.3.3 Case 2: Nonlinear non-Gaussian model

In this numerical experiment, a nonlinear non-Gaussian observation model is considered which consists of the measurements of bearing θ and the range r of a target. Let $\mathbf{y}_k^n$ be the nth observation from the m^{th} target at time k, the observation model is defined below

$$\mathbf{y}_k^n = \left(arctan(\frac{y_k^m}{x_k^m}), \sqrt{(x_k^m)^2 + (y_k^m)^2} \right)' + \mathbf{v}_k^n. \tag{4.13}$$

where $\mathbf{v}_k^n$, is distributed according to a suitable Gaussian Mixture Model (GMM) with probability density

$$p(\mathbf{v}_k^n) = \sum_{\ell=1}^{2} w_\ell^v \mathcal{N}(\mu_{\ell,v}^n, \Sigma_{\ell,v}^n), \tag{4.14}$$

where $w_1^v = 0.8$, $w_2^v = 0.2$ and $\mu_{1,v}^n = -0.01$, $\mu_{2,v}^n = 0.01$ and the covariance matrices are defined as follows

$$\Sigma_{1,v}^n = \begin{pmatrix} 0.004 & 0 \\ 0 & 0.004 \end{pmatrix}, \Sigma_{2,v}^n = \begin{pmatrix} 0.0001 & 0 \\ 0 & 0.0001 \end{pmatrix}.$$

Also in this case, the driving noise in the dynamics model is considered to be a suitable GMM with two Gaussian mixands defined as follows

$$p(\mathbf{u}_k^n) = \sum_{\ell=1}^{2} w_\ell^u \mathcal{N}(\mu_{\ell,u}^n, \Sigma_{\ell,u}^n), \tag{4.15}$$

The means of the two Gaussians are also ± 0.01 and the covariance matrices are given in the following

$$\Sigma_{1,u}^n = \begin{pmatrix} 0.7 & 0 \\ 0 & 0.7 \end{pmatrix}, \Sigma_{2,u}^n = \begin{pmatrix} 0.1 & 0 \\ 0 & 0.1 \end{pmatrix}.$$

with weights $w_1^u = 0.8$, $w_2^u = 0.2$ respectively.

In this experiment, the generalized hybrid Monte Carlo Alexander et al. (2005b) is employed as the MCMC sampler. The numerical results in Table (4.3) show that the algorithm needs much less levels for some targets at some time steps to obtain good tracking results. Also, the table specifies that the DHLB-PF with learning parameter is able to automatically choose the terminating level and does not compromise the tracking performance as shown in Fig. 4.10. For the nonlinear and Non-Gaussian case, the tracking result is shown in Fig.4.11. Given that a surveillance distributed sensor network operates under limited power constraints and a quick detection and accurate tracking of targets is needed, the the threshold of the ESS has been chosen to a lower value, precisely, 50%. However, a close examination on the RMSE comparison presented in Fig. 4.12 implies that the learning drift homotopy algorithm estimates accurately the states of the targets while at the same time decreases the computational time by not using all levels as in the drift homotopy. Also, its performance is by far superior in comparison to particle filtering as showing in Fig. 4.

Table 4.3: This table shows the levels performed before the final level L= 20 at a single time step $k = 10$. The ESS threshold is chosen to be 50% of the sample size. Samples size is set to be 10 in the algorithm.

Target	ℓ_k	MCMC steps	ESS (50%)
1	7	9	7.513377
2	2	16	5.026966
3	0	41	5.016717
4	0	47	7.510589
5	0	33	5.007232
6	0	23	5.012302
7	0	1	9.986915

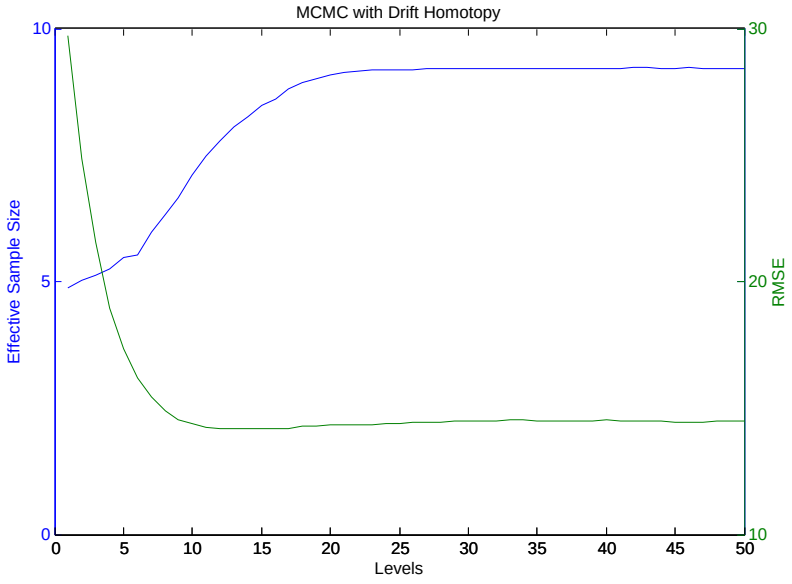

Figure 4.9: This figure shows that as the level increases, the ESS increases and RMSE decreases.

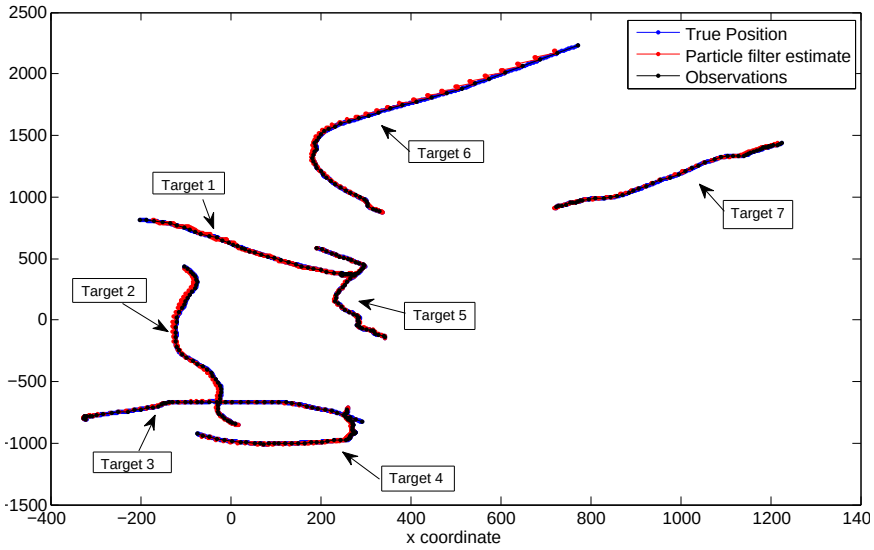

Figure 4.10: Tracking result for Section 4.3.2 with 7 targets.

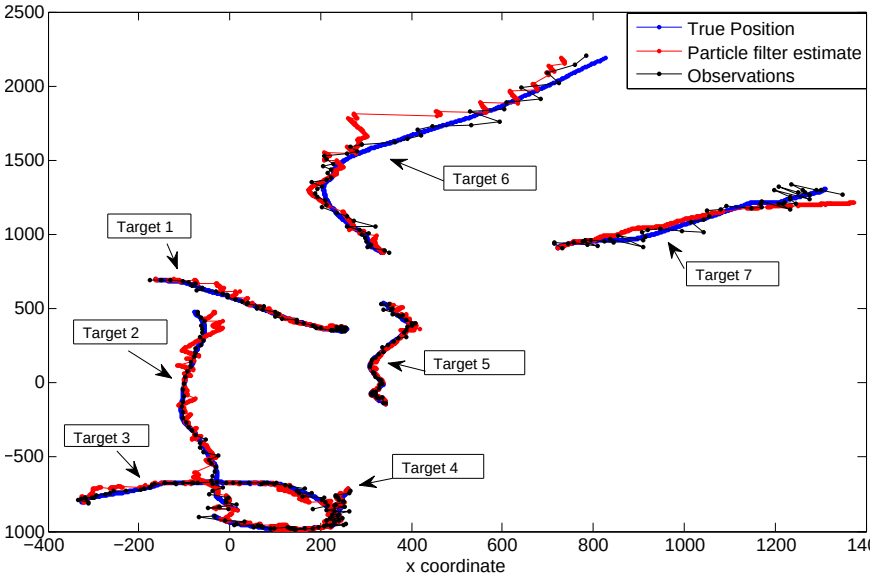

Figure 4.11: Tracking result for Section 4.3.3 with 7 targets.

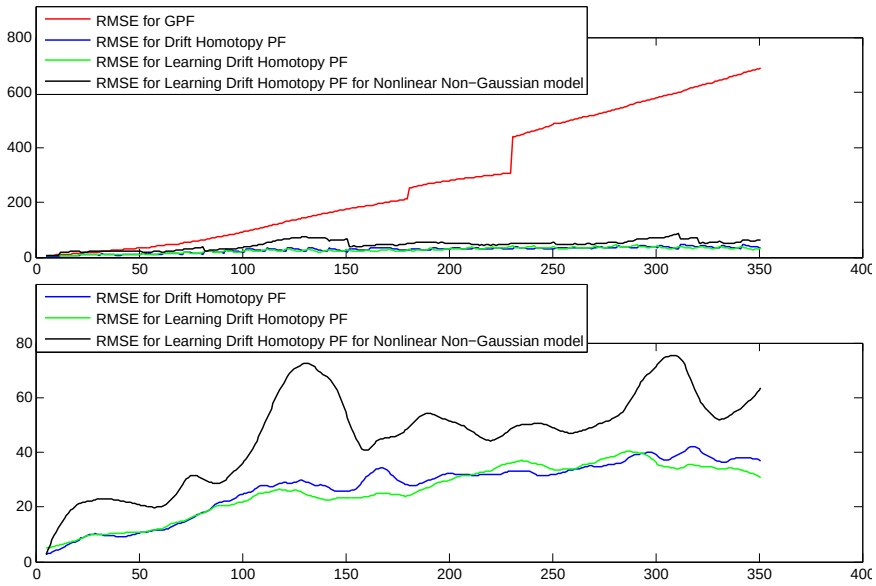

Figure 4.12: This figure shows the RMSE comparison of the generic particle filter (GPF) with 500 samples, the DHLB-PF and the learning drift homotopy particle filtering with 10 samples in both methods. We also include the RMSE of the learning DHLB-PF for a nonlinear non-Gaussian model. The lower panel is a zoomed in and smoothed figure of the upper panel that compares the RMSE of DHLB-PF with and without learning.

Bibliography

Ahmed, N., Rutten, M., Bessell, T., Kanhere, S. S., Gordon, N., and Jha, S. (2010). Detection and tracking using particle-filter-based wireless sensor networks. *IEEE Transactions on Mobile Computing*, 9(9):1332–1345. 1

Alexander, F. J., Eyink, G. L., and Restrepo, J. M. (2005a). Accelerated monte carlo for optimal estimation of time series. *Journal of Statistical Physics*, 119(5-6):1331–1345. 44

Alexander, F. J., Eyink, G. L., and Restrepo, J. M. (2005b). Accelerated monte carlo for optimal estimation of time series. *Journal of Statistical Physics*, 119(5-6):1331–1345. 82

Alspach, D. L. and Sorenson, H. W. (1972). Nonlinear bayesian estimation using gaussian sum approximations. *Automatic Control, IEEE Transactions on*, 17(4):439–448. 21

Bajović, D., Sinopoli, B., and Xavier, J. (2011). Sensor selection for event detection in wireless sensor networks. *Signal Processing, IEEE Transactions on*, 59(10):4938–4953. 12

Bar-Shalom, Y., Li, X. R., and Kirubarajan, T. (2004). *Estimation with applications to tracking and navigation: theory algorithms and software*. John Wiley & Sons. 4

Baum, M. and Hanebeck, U. D. (2012). Extended object tracking based on set-theoretic and stochastic fusion. *IEEE Transactions on Aerospace and Electronic Systems*, 48(4):3103–3115. 4

Baum, M. and Hanebeck, U. D. (2013). The kernel-sme filter for multiple target tracking. In *Information Fusion (FUSION), 2013 16th International Conference on*, pages 288–295. IEEE. 4

Berry, M. W., Browne, M., Langville, A. N., Pauca, V. P., and Plemmons, R. J. (2007). Algorithms and applications for approximate nonnegative matrix factorization. *Computational statistics & data analysis*, 52(1):155–173. 12, 13

Berzuini, C. and Gilks, W. (2001). Resample-move filtering with cross-model jumps. In *Sequential Monte Carlo Methods in Practice*, pages 117–138. Springer. 5

Brunet, J.-P., Tamayo, P., Golub, T. R., and Mesirov, J. P. (2004). Metagenes and molecular pattern discovery using matrix factorization. *Proceedings of the national academy of sciences*, 101(12):4164–4169. 12

Bunke, H. and Caelli, T. (2001). *Hidden Markov models: applications in computer vision*, volume 45. World Scientific. 3

Burkard, R. E., Dell'Amico, M., and Martello, S. (2009). *Assignment Problems, Revised Reprint*. Siam. 81

Candès, E. J., Romberg, J., and Tao, T. (2006). Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information. *Information Theory, IEEE Transactions on*, 52(2):489–509. 2

Cappé, O., Moulines, E., and Rydén, T. (2005). *Inference in hidden Markov models*. Springer series in statistics. Springer, New York. 4, 5, 21, 24, 77

Chatzi, E. N. and Smyth, A. W. (2009). The unscented kalman filter and particle filter methods for nonlinear structural system identification with non-collocated heterogeneous sensing. *Structural control and health monitoring*, 16(1):99–123. 21

Chung, K. L. (2001). *A course in probability theory*. Academic press. 23

Cichocki, A., Amari, S.-i., Zdunek, R., Kompass, R., Hori, G., and He, Z. (2006). Extended smart algorithms for non-negative matrix factorization. In *Artificial Intelligence and Soft Computing–ICAISC 2006*, pages 548–562. Springer. 12, 13, 15

Cichocki, A., Zdunek, R., and Amari, S.-i. (2007a). Hierarchical als algorithms for nonnegative matrix and 3d tensor factorization. In *Independent Component Analysis and Signal Separation*, pages 169–176. Springer. 14, 15

Cichocki, A., Zdunek, R., Choi, S., Plemmons, R., and Amari, S.-I. (2007b). Novel multi-layer non-negative tensor factorization with sparsity constraints. In *Adaptive and Natural Computing Algorithms*, pages 271–280. Springer. 14, 15

Coates, M. (2004). Distributed particle filters for sensor networks. In *Proceedings of the 3rd international symposium on Information processing in sensor networks*, pages 99–107. ACM. 1

Crisan, D. and Doucet, A. (2002). A survey of convergence results on particle filtering methods for practitioners. *IEEE Transactions on Signal Processing*, 50(3):736–746. 5

d'Aspremont, A., El Ghaoui, L., Jordan, M. I., and Lanckriet, G. R. (2007). A direct formulation for sparse pca using semidefinite programming. *SIAM review*, 49(3):434–448. 17

Del Moral, P. (1996). Non-linear filtering: interacting particle resolution. *Markov processes and related fields*, 2(4):555–581. 22

Dias, S. S. and Bruno, M. G. (2013). Cooperative target tracking using decentralized particle filtering and rss sensors. *Signal Processing, IEEE Transactions on*, 61(14):3632–3646. 31

Doucet, A., Godsill, S., and Andrieu, C. (2000). On sequential monte carlo sampling methods for bayesian filtering. *Statistics and computing*, 10(3):197–208. 28, 29

Doucet, A., Vo, B.-N., Andrieu, C., and Davy, M. (2002). Particle filtering for multi-target tracking and sensor management. 31, 32

Evangelou, E. and Maroulas, V. (2015). Sequential empirical bayes method for filtering dynamic spatiotemporal processes. *arXiv preprint arXiv:1506.02691*. 30

Frenkel, D. and Smit, B. (2001). *Understanding molecular simulation: from algorithms to applications*, volume 1. Academic press. 44

Fuemmeler, J. A. and Veeravalli, V. V. (2010). Energy efficient multi-object tracking in sensor networks. *Signal Processing, IEEE Transactions on*, 58(7):3742–3750. 11

Gao, Y. and Church, G. (2005). Improving molecular cancer class discovery through sparse non-negative matrix factorization. *Bioinformatics*, 21(21):3970–3975. 16

Gilks, W. R. and Berzuini, C. (2001). Following a moving target—monte carlo inference for dynamic bayesian models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(1):127–146. 5

Gordon, N. J., Salmond, D. J., and Smith, A. F. (1993). Novel approach to nonlinear/non-gaussian bayesian state estimation. In *Radar and Signal Processing, IEE Proceedings F*, volume 140, pages 107–113. IET. 22, 26

Gorji, A., Menhaj, M. B., and Shiry, S. (2009). Multiple target tracking for mobile robots using the jpdaf algorithm. In *Tools and Applications with Artificial Intelligence*, pages 51–68. Springer. 31

Guan, N., Tao, D., Luo, Z., and Yuan, B. (2012). Nenmf: an optimal gradient method for nonnegative matrix factorization. *Signal Processing, IEEE Transactions on*, 60(6):2882–2898. 2

Gupta, V., Chung, T. H., Hassibi, B., and Murray, R. M. (2006). On a stochastic sensor selection algorithm with applications in sensor scheduling and sensor coverage. *Automatica*, 42(2):251–260. 10

Hamilton, J. D. (1989). A new approach to the economic analysis of nonstationary time series and the business cycle. *Econometrica: Journal of the Econometric Society*, pages 357–384. 3

Hamilton, J. D. and Raj, B. (2013). *Advances in Markov-Switching Models: Applications in Business Cycle Research and Finance*. Springer Science & Business Media. 3

Hamilton, J. D. and Susmel, R. (1994). Autoregressive conditional heteroskedasticity and changes in regime. *Journal of Econometrics*, 64(1):307–333. 3

Handschin, J. (1970). Monte carlo techniques for prediction and filtering of non-linear stochastic processes. *Automatica*, 6(4):555–563. 22

Handschin, J. and Mayne, D. Q. (1969). Monte carlo techniques to estimate the conditional expectation in multi-stage non-linear filtering. *International journal of control*, 9(5):547–559. 21

He, H., Xiong, R., Zhang, X., Sun, F., and Fan, J. (2011). State-of-charge estimation of the lithium-ion battery using an adaptive extended kalman filter based on an improved thevenin model. *Vehicular Technology, IEEE Transactions on*, 60(4):1461–1469. 21

Hlinka, O., Hlawatsch, F., and Djuric, P. M. (2013). Distributed particle filtering in agent networks: A survey, classification, and comparison. *Signal Processing Magazine, IEEE*, 30(1):61–81. 1

Horwood, J. T. and Poore, A. B. (2011). Adaptive gaussian sum filters for space surveillance. *Automatic Control, IEEE Transactions on*, 56(8):1777–1790. 21

Hoyer, P. O. (2004a). Non-negative matrix factorization with sparseness constraints. *The Journal of Machine Learning Research*, 5:1457–1469. 2

Hoyer, P. O. (2004b). Non-negative matrix factorization with sparseness constraints. *The Journal of Machine Learning Research*, 5:1457–1469. 15

Hue, C., Cadre, J. L., and Pérez, P. (2002). Tracking multiple objects with particle filtering. *Aerospace and Electronic Systems, IEEE Transactions on*, 38(3):791–812. 31

Janke, W. (2007). *Rugged free energy landscapes: Common computational approaches to spin glasses, structural glasses and biological macromolecules*, volume 736. Springer. 63

Jelinek, F. (1997). *Statistical methods for speech recognition*. MIT press. 3

Joshi, S. and Boyd, S. (2009). Sensor selection via convex optimization. *Signal Processing, IEEE Transactions on*, 57(2):451–462. 11

Julier, S. J., Uhlmann, J. K., and Durrant-Whyte, H. F. (1995). A new approach for filtering nonlinear systems. In *American Control Conference, Proceedings of the 1995*, volume 3, pages 1628–1632. IEEE. 4

Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Journal of basic Engineering*, 82(1):35–45. 3

Kang, K. and Maroulas, V. (2013). Drift homotopy methods for a non-gaussian filter. In *Information Fusion (FUSION), 2013 16th International Conference on*, pages 1088–1094. IEEE. 3, 76

Kang, K., Maroulas, V., and Schizas, I. D. (2014). Drift homotopy particle filter for non-gaussian multi-target tracking. In *Information Fusion (FUSION), 2014 17th International Conference on*, pages 1–7. IEEE. 3, 4, 76

Kim, C.-J., Nelson, C. R., et al. (1999). *State-space models with regime switching: classical and Gibbs-sampling approaches with applications*, volume 2. MIT press Cambridge, MA. 3

Kim, H. and Park, H. (2007). Sparse non-negative matrix factorizations via alternating non-negativity-constrained least squares for microarray data analysis. *Bioinformatics*, 23(12):1495–1502. 16

Kim, S., Shephard, N., and Chib, S. (1998). Stochastic volatility: likelihood inference and comparison with arch models. *The Review of Economic Studies*, 65(3):361–393. 3

Kitagawa, G. (1996). Monte carlo filter and smoother for non-gaussian nonlinear state space models. *Journal of computational and graphical statistics*, 5(1):1–25. 22

Koski, T. (2001). *Hidden Markov models for bioinformatics*, volume 2. Springer Science & Business Media. 3

Krishnamurthy, V., Maskery, M., and Yin, G. (2008). Decentralized adaptive filtering algorithms for sensor activation in an unattended ground sensor network. *Signal Processing, IEEE Transactions on*, 56(12):6086–6101. 10

Kulhavỳ, R. (1990). Recursive nonlinear estimation: A geometric approach. *Automatica*, 26(3):545–555. 21

Law, K., Stuart, A., and Zygalakis, K. (2015). *Data assimilation: a mathematical introduction*, volume 62. Springer. 3

Lee, D. D. and Seung, H. S. (1999). Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791. 12, 13

Lee, D. D. and Seung, H. S. (2001). Algorithms for non-negative matrix factorization. In *Advances in neural information processing systems*, pages 556–562. 2

Lin, C.-J. (2007a). Projected gradient methods for nonnegative matrix factorization. *Neural computation*, 19(10):2756–2779. 2

Lin, C.-J. (2007b). Projected gradient methods for nonnegative matrix factorization. *Neural computation*, 19(10):2756–2779. 14

Lin, J., Xiao, W., Lewis, F. L., and Xie, L. (2009). Energy-efficient distributed adaptive multisensor scheduling for target tracking in wireless sensor networks. *Instrumentation and Measurement, IEEE Transactions on*, 58(6):1886–1896. 31

LIU, H., SO, H. C., CHAN, K. W. F., and LUI, W. K. (2009). Distributed particle filter for target tracking in sensor networks. *Progress In Electromagnetics Research C*. 31

Liu, J. S. (2008). *Monte Carlo strategies in scientific computing*. Springer Science & Business Media. 3, 77

Liu, J. S. and Chen, R. (1998). Sequential monte carlo methods for dynamic systems. *Journal of the American statistical association*, 93(443):1032–1044. 4, 22

Mahler, R. and Maroulas, V. (2013). Tracking spawning objects. *IET Radar, Sonar and Navigation*, 7(3):321–331. 3, 4

Mahler, R. P. (2007). *Statistical multisource-multitarget information fusion*. Artech House, Inc. 4

Maroulas, V., Kang, K., Schizas, I. D., and Berry, M. W. (2015). A learning drift homotopy particle filter. In *Information Fusion (Fusion), 2015 18th International Conference on*, pages 1930–1937. 3, 4

Maroulas, V. and Nebenfuhr, A. (2015). Tracking rapid intracellular movements: a bayesian random set approach. *Annals of Applied Statistics*, 9(2):926–949. 3, 4

Maroulas, V. and Stinis, P. (2012). Improved particle filters for multi-target tracking. *Journal of Computational Physics*, 231(2):602–611. 4, 76

Ng, W., Li, J., Godsill, S., and Vermaak, J. (2005). A hybrid approach for online joint detection and tracking for multiple targets. In *Aerospace Conference, 2005 IEEE*, pages 2126–2141. IEEE. 31

Oh, S. (2012). A scalable multi-target tracking algorithm for wireless sensor networks. *International Journal of Distributed Sensor Networks*, 2012. 31

Olfati-Saber, R. (2005). Distributed kalman filter with embedded consensus filters. In *Decision and Control, 2005 and 2005 European Control Conference. CDC-ECC'05. 44th IEEE Conference on*, pages 8179–8184. IEEE. 1

Owen, A. B. (2013). *Monte Carlo theory, methods and examples.* 22, 24

Ozdemir, O., Niu, R., and Varshney, P. K. (2009). Tracking in wireless sensor networks using particle filtering: Physical layer considerations. *Signal Processing, IEEE Transactions on*, 57(5):1987–1999. 1

Pascual-Montano, A., Carazo, J. M., Kochi, K., Lehmann, D., and Pascual-Marqui, R. D. (2006). Nonsmooth nonnegative matrix factorization (nsnmf). *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(3):403–415. 15

Pauca, V. P., Piper, J., and Plemmons, R. J. (2006). Nonnegative matrix factorization for spectral data analysis. *Linear algebra and its applications*, 416(1):29–47. 15

Pauca, V. P., Shahnaz, F., Berry, M. W., and Plemmons, R. J. (2004). Text mining using non-negative matrix factorizations. In *SDM*, volume 4, pages 452–456. SIAM. 16

Pitt, M. K. and Shephard, N. (1999). Filtering via simulation: Auxiliary particle filters. *Journal of the American statistical association*, 94(446):590–599. 26

Rabiner, L. and Juang, B.-H. (1993). Fundamentals of speech recognition. 3

Ren, G., Maroulas, V., and Schizas, I. (2015). Distributed spatio-temporal association and tracking of multiple targets using multiple sensors. *Aerospace and Electronic Systems, IEEE Transactions on*, 51(4):2570–2589. 8, 32, 34, 76

Ren, G. and Schizas, I. D. (2013). Distributed sensor-informative tracking of targets. In *Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP), 2013 IEEE 5th International Workshop on*, pages 81–84. IEEE. 31

Ristic, B., Arulampalam, M., and Gordon, A. (2004). Beyond kalman filters: Particle filters for target tracking. *Artech House*, 66. 4, 21

Roberts, G. O., Rosenthal, J. S., et al. (2004). General state space markov chains and mcmc algorithms. *Probability Surveys*, 1:20–71. 57, 58

Sajda, P., Du, S., Brown, T. R., Stoyanova, R., Shungu, D. C., Mao, X., and Parra, L. C. (2004). Nonnegative matrix factorization for rapid recovery of constituent spectra in magnetic resonance chemical shift imaging of the brain. *Medical Imaging, IEEE Transactions on*, 23(12):1453–1465. 12

Sandell, N. F. and Olfati-Saber, R. (2008). Distributed data association for multi-target tracking in sensor networks. In *Decision and Control, 2008. CDC 2008. 47th IEEE Conference on*, pages 1085–1090. IEEE. 32

Schizas, I. D. (2013). Distributed informative-sensor identification via sparsity-aware matrix decomposition. *Signal Processing, IEEE Transactions on*, 61(18):4610–4624. 2

Smith, A., Doucet, A., de Freitas, N., and Gordon, N. (2013). *Sequential Monte Carlo methods in practice*. Springer Science & Business Media. 4, 5

Snyder, C., Bengtsson, T., Bickel, P., and Anderson, J. (2008). Obstacles to high-dimensional particle filtering. *Monthly Weather Review*, 136(12):4629–4640. 5, 39

Sorenson, H. W. and Alspach, D. L. (1971). Recursive bayesian estimation using gaussian sums. *Automatica*, 7(4):465–479. 4

Tharmarasa, R., Kirubarajan, T., Sinha, A., and Lang, T. (2011). Decentralized sensor selection for large-scale multisensor-multitarget tracking. *Aerospace and Electronic Systems, IEEE Transactions on*, 47(2):1307–1324. 32

Thatte, G. and Mitra, U. (2008). Sensor selection and power allocation for distributed estimation in sensor networks: Beyond the star topology. *Signal Processing, IEEE Transactions on*, 56(7):2649–2661. 10, 11

Tibshirani, R. (1996a). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288. 2, 33

Tibshirani, R. (1996b). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288. 16

Toral, R. and Ferreira, A. (1994). A general class of hybrid monte carlo methods. In *Proceedings of Physics Computing*, volume 94, pages 265–268. 41

Ulfarsson, M. O. and Solo, V. (2008). Sparse variable pca using geodesic steepest descent. *Signal Processing, IEEE Transactions on*, 56(12):5823–5832. 2

Valverde, G. and Terzija, V. (2011). Unscented kalman filter for power system dynamic state estimation. *Generation, Transmission & Distribution, IET*, 5(1):29–37. 21

Vermaak, J., Godsill, S. J., and Perez, P. (2005). Monte carlo filtering for multi target tracking and data association. *Aerospace and Electronic Systems, IEEE Transactions on*, 41(1):309–332. 32

Vo, B.-N., Singh, S., and Doucet, A. (2005). Sequential monte carlo methods for multitarget filtering with random finite sets. *Aerospace and Electronic Systems, IEEE Transactions on*, 41(4):1224–1245. 4

Wan, E. A. and Van Der Merwe, R. (2000). The unscented kalman filter for nonlinear estimation. In *Adaptive Systems for Signal Processing, Communications, and Control Symposium 2000. AS-SPCC. The IEEE 2000*, pages 153–158. Ieee. 4

Weare, J. (2009). Particle filtering with path sampling and an application to a bimodal ocean current model. *Journal of Computational Physics*, 228(12):4312–4331. 5, 39

Welch, G. and Bishop, G. (2006). An introduction to the kalman filter. department of computer science, university of north carolina. 3

Zdunek, R. and Cichocki, A. (2007). Nonnegative matrix factorization with constrained second-order optimization. *Signal Processing*, 87(8):1904–1916. 15

Zhou, G., Cichocki, A., and Xie, S. (2012). Fast nonnegative matrix/tensor factorization based on low-rank approximation. *Signal Processing, IEEE Transactions on*, 60(6):2928–2940. 17

Zhu, H., Schizas, I. D., and Giannakis, G. B. (2009). Power-efficient dimensionality reduction for distributed channel-aware kalman tracking using wsns. *Signal Processing, IEEE Transactions on*, 57(8):3193–3207. 1

Zou, H. (2006). The adaptive lasso and its oracle properties. *Journal of the American statistical association*, 101(476):1418–1429. 33

Zou, H., Hastie, T., and Tibshirani, R. (2006). Sparse principal component analysis. *Journal of computational and graphical statistics*, 15(2):265–286. 2

Zwanzig, R. (1988). Diffusion in a rough potential. *Proceedings of the National Academy of Sciences*, 85(7):2029–2030. 63, 64

Vita

Kai Kang was born on July 14th, 1983 in Shenyang, China. In 2002, he joined Northeastern University in China in the Department of Mathematics. He received his B.S. degree in July 2006. He worked as a programmer after graduation.

In August 2009, he started to pursue his Master's degree at the University of Tennessee, Knoxville in the Department of Mathematics with concentration in numerical analysis. He obtained his M.S. degree in 2012. He continued his study in the Department of Mathematics at the University of Tennessee, Knoxville as a Ph.D. student with concentration in computational statistics.

He will join the National Institutes of Health as a statistician in August, 2016 in North Carolina.