Invertible particle flow based Gaussian and Gaussian sum particle filters

Thesis submitted in partial fulfillment
of the requirements for the degree of

*(Master of Science in **Electronics and Communication Engineering** by Research*

by

Comandur Rajasekhar Karthik
2019702017
karthik.comandur@research.iiit.ac.in

International Institute of Information Technology
Hyderabad - 500 032, INDIA
June 2023

International Institute of Information Technology
Hyderabad, India

CERTIFICATE

It is certified that the work contained in this thesis, titled " **Invertible particle flow based Gaussian and Gaussian sum particle filters** " by Comandur Rajasekhar Karthik, has been carried out under my supervision and is not submitted elsewhere for a degree.

_____ _____

Date Adviser: Dr. Santosh Nannuru.

To the Past, Present and Future

Acknowledgments

I want to start by expressing my sincere gratitude to Dr. Santosh Nannuru for serving as my research adviser. After my first semester, I began working as a research assistant with him. Throughout my studies, he gave me much support and guidance, for which I shall always be grateful. I am incredibly grateful for the opportunity to work with him. This opportunity to work under him has greatly aided in my skill development and research experience.

Second, I want to thank Dr. Yunpeng Li, Senior Lecturer in Artificial Intelligence, Department of Computer Science, University of Surrey, UK, for his guidance and support throughout my research. Like my adviser, he served as a mentor to me by imparting his knowledge and research experience, which assisted me in finding an answer to my research question. I will always be appreciative of his help and advice during my research.

Lastly, I would like to express my gratitude to my family, friends, and lab colleagues for their support and encouragement during the stressful time of COVID-19, which inspired and motivated me to finish my research.

Abstract

The estimation of the state of dynamic systems using measurements is a frequently arising challenge. State estimation is used in applications like robotics, industrial manufacturing, weather forecasting, target tracking, etc. The hidden state in the dynamic systems can be estimated using the recursive Bayesian filters. These filters estimate the state by tracking the posterior distribution. These filters include two steps, prediction based on solving the equation modeling the state evolution and update of the state using the measurements. Most real-world systems have non-linear dynamic and measurement models. Additionally, both models can be affected by arbitrary noise sources, which have either Gaussian or non-Gaussian characteristics. For state estimation in linear Gaussian models, Kalman filters are employed. Extended Kalman filters (EKF) are utilized for state estimation in non-linear Gaussian models. For state estimation in non-linear non-Gaussian models, particle filters are generally employed. Particle filters, however, are ineffective in high-dimensional non-linear non-Gaussian models because they experience weight degeneracy.

In this thesis, we propose incorporating the invertible particle flow methods in the Gaussian particle filter (GPF) framework to generate a proposal distribution. These methods are derived under Gaussian assumptions for the flow equation. The resulting particle flow Gaussian particle filter (PFGPF) method preserves the asymptotic characteristics of Gaussian particle filters and has the potential to perform better at state estimation in high-dimensional spaces.

Secondly, we construct a particle flow Gaussian sum particle filter (PFGSPF), which roughly approximates the predictive and posterior as Gaussian mixture models, using a bank of PFGPF filters. In complicated estimating issues where a simple Gaussian approximation is insufficient, this approximation is helpful. We compare the performance of the proposed filters with the existing particle flow filters and particle flow particle filters (PFPF) in high dimensional complex numerical simulations.

Contents

Chapter Page

1 Introduction . . . 1
1.1 Key Contributions . . . 3
1.2 Thesis Organisation . . . 3

2 Background . . . 5
2.1 Notations . . . 5
2.2 Problem statement . . . 6
2.3 Recursive Bayesian method . . . 7
2.4 Particle filter . . . 7
2.4.1 Weight Degeneracy . . . 8
2.4.2 Optimal importance density . . . 9
2.4.3 Resampling . . . 9
2.5 Particle filter with MCMC methods . . . 10
2.6 Particle flow filters . . . 11
2.7 Particle flow particle filters . . . 11

3 Particle filter with invertible particle flow . . . 12
3.1 Particle flow . . . 12
3.1.1 Exact Daum and Huang filter [1, 2] . . . 13
3.1.2 Localized exact Daum and Huang filter [3] . . . 14
3.1.3 Numerical Implementation [3] . . . 14
3.2 Particle flow with Invertible Mapping [4] . . . 16
3.2.1 Invertible particle flow using LEDH . . . 17
3.2.2 Invertible particle flow using EDH . . . 18

4 Particle flow Gaussian particle filter . . . 22
4.1 Gaussian filters . . . 22
4.2 Gaussian particle filter . . . 22
4.2.1 Prediction . . . 23
4.2.2 Update . . . 23
4.3 Particle flow Gaussian particle filter . . . 24
4.3.1 The prediction step . . . 25
4.3.2 The update step . . . 25
4.4 Simulations and Results . . . 27
4.4.1 Multi-target acoustic tracking . . . 27

4.4.1.1 Numerical Simulation Setup 27
4.4.1.2 Filter Implementation 28
4.4.1.3 Simulation Results 29
4.4.2 Large spatial sensor networks: Skewed-t dynamic model and count measurements 29
4.4.2.1 Numerical Simulation Setup 29
4.4.2.2 Filter Implementation 30
4.4.2.3 Simulation Results 30

5 Particle flow Gaussian Sum particle filter 32
5.1 Gaussian Sum filters 32
5.2 Gaussian Sum particle filter 34
5.2.1 The prediction step 34
5.2.2 The update step 34
5.3 Particle flow Gaussian sum particle filter 36
5.3.1 The prediction step 36
5.3.2 The update step 37
5.3.3 Effective number of Gaussians 38
5.4 Simulations and Results 38
5.4.1 Multi-target acoustic tracking 38
5.4.1.1 Numerical simulation setup 38
5.4.1.2 Filter Implementation and Results 38
5.4.2 Large spatial sensor networks: Skewed-t dynamic model and count measurements 39
5.4.2.1 Numerical Simulation Setup 39
5.4.2.2 Filter Implementation and Results 40

6 Conclusions and Future Work 42

Bibliography 44

List of Figures

Figure Page

1.1 Real-world applications of dynamic state estimation. (a) Global positioning systems (GPS), (b) Advanced driver-assisted systems (ADAS), (c) Weather estimation, (d) Multi-target tracking Systems 1
1.2 Flowchart of the filters discussed in the thesis 3

2.1 Weight degeneracy. 2.1a shows particle sampled from the proposal density and 2.1b shows the particle approximation of posterior density with weight degeneracy. 9
2.2 Resampling of particles 10

4.1 A sample true trajectory and estimated trajectory using the PFGPF. 29

5.1 Average MSE vs N_p in the large spatial sensor networks simulation. For PFGSPF, $N_p^* = 200$ is fixed and G is varied. Error bars indicate standard deviation. 40
5.2 Average G_{eff} of PFGSPF. (Acoustic: 500 simulations, large spatial sensor network: 100 simulations). 41

List of Tables

Table Page

4.1 [Acoustic model, 16 dimensions] Average OMAT error (m) and average execution time (s) for various algorithms. 29
4.2 Average MSE in the large spatial sensor networks simulation. State dimension $d = 144$ and $N_p = 200$ particles. 31

5.1 OMAT errors (m) recorded for the acoustic tracking simulation, excluding lost tracks (LT) defined as the estimated trajectories with average tracking error $> 2\,\mathrm{m}$. 39
5.2 Average MSE in the large spatial sensor networks simulation. 40

Chapter 1

Introduction

In complex dynamic systems, accurate estimation of the state vector that describes the system's parameters using sensor data is a challenging task. The dynamic state estimation process is crucial for safe and effective control, identifying potential system failures, and replacing them. However, real-world scenarios involve uncertain sensor data, making it necessary to have a state estimate that captures this uncertainty. State estimation has become an essential aspect of various real-world applications, including Global positioning systems (GPS), Advanced driver-assisted systems (ADAS), Weather estimation, and Multi-target tracking Systems.

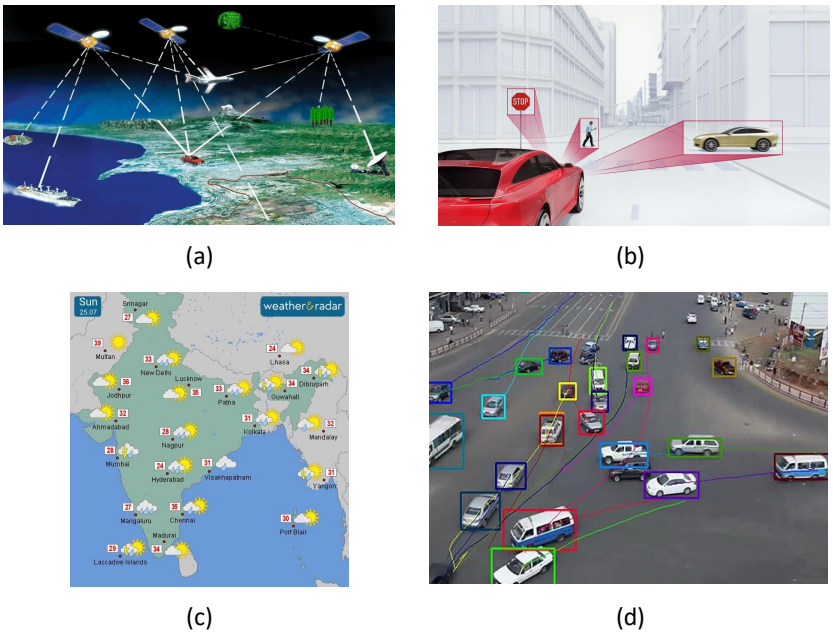


Figure 1.1: Real-world applications of dynamic state estimation. (a) Global positioning systems (GPS), (b) Advanced driver-assisted systems (ADAS), (c) Weather estimation, (d) Multi-target tracking Systems

For linear Gaussian models, the Kalman filter [5] is the optimal filter that has the lowest mean square error (MSE). Kalman filter is based on Gaussian assumption and is easy to implement. Most real-world scenarios are non-linear systems, so the Kalman filter cannot be applied for state estimation.

In non-linear systems, it is difficult to obtain an exact closed-form filter, so approximate methods are often used. The SMC filter [6], also known as the particle filter, is a popular approach for state estimation in non-linear and non-Gaussian scenarios. However, as the system's dimensionality increases, weight degeneracy can occur, which can degrade the filter's performance. Various techniques have been suggested to address this problem in particle-based filters, as outlined in Chapter 2.

Particle flow filters [1, 7, 8, 9, 10, 11] are one of the many solutions proposed to tackle weight degeneracy in particle-based filters. These filters utilize a flow equation to migrate the particles from the prior to the posterior distribution, eliminating the need for importance sampling and avoiding weight degeneracy. However, the practical implementation of many particle flow methods involves numerical approximations and assumptions, which may lead to statistical inconsistency. To address this challenge, the particle flow particle filter (PFPF) [12, 13, 4] framework integrates particle flow methods into a particle filter framework, retaining the particle filter's statistical consistency while also taking advantage of the favorable performance of particle flow techniques.

The Gaussian particle filter (GPF) [14] is a variant of the particle filter that approximates the predictive and posterior distributions as Gaussian densities, eliminating the need for resampling. A new set of particles is drawn from the Gaussian approximation of the posterior at the beginning of each iteration. Gaussian Sum Particle Filters (GSPF) utilize multiple banks of GPF to represent both the prediction and posterior distributions as a sum of Gaussian distributions. GPF can be considered as a specific instance of GSPF [15].

In this thesis, we explore the advantages of combining the invertible particle flow method with the Gaussian particle filter to create the particle flow Gaussian particle filter (PFGPF), considering their common Gaussianity assumptions and approximations. Incorporating the invertible particle flow into the GPF eliminates the need for particle resampling to prevent weight degeneracy and utilizes the encompassing importance sampling method to capture the data from non-linear and non-Gaussian models. Additionally, we introduce the particle flow Gaussian sum particle flow filter (PFGSPF), which employs a set of PFGPF to approximate the predictive and posterior distributions as a Gaussian mixture density. In complex, high-dimensional filtering situations, a Gaussian mixture density can provide a more precise approximation of the posterior than a single Gaussian approximation.

The figure 1.2 shows the flowchart of different filters discussed in this thesis. The connecting lines and arrows indicate the relationship among various filters and how they are developed. The filters Particle flow Gaussian particle filter (PFGPF) and Particle flow Gaussian sum particle filters are proposed in this thesis .

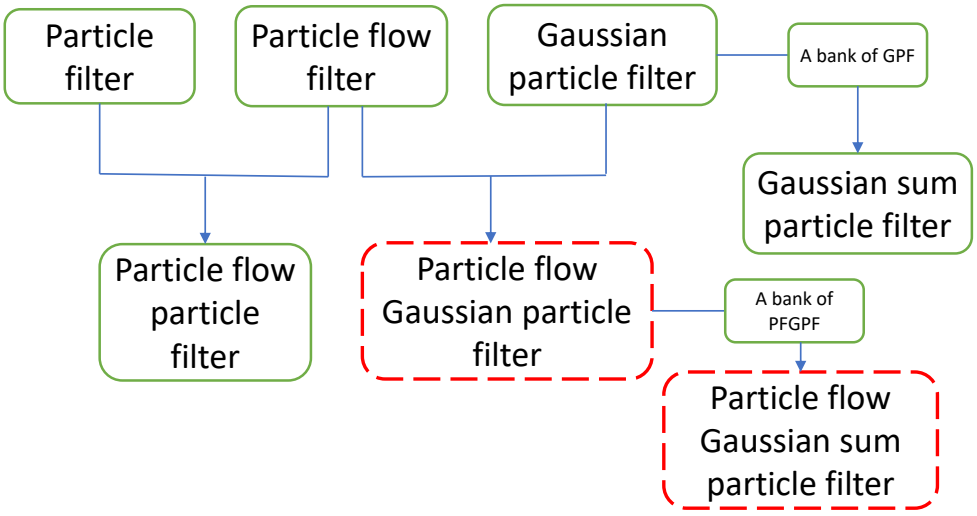


Figure 1.2: Flowchart of the filters discussed in the thesis

1.1 Key Contributions

The key contributions in this thesis are as follows:

- We incorporate an invertible particle flow into a Gaussian particle filter to construct an effective proposal density.
- We derive the modified importance sampling weight due to the incorporation of particle flow into the GPF.
- We incorporate the effectiveness of invertible particle flow into an encompassing Gaussian sum particle filtering framework.
- We introduce an effective number of Gaussians (G_{eff}) to find the number of Gaussian densities significant in the Gaussian mixture distribution.
- We evaluate and compare the performance of the proposed filters in high-dimensional challenging numerical simulation setups.

1.2 Thesis Organisation

Chapter 2 outlines the problem and provides a brief summary of previously proposed methods. Chapter 3 discusses the particle filter with invertible particle flow (PFPF) and the deterministic particle flow

employed to construct it. Chapter 4 focuses on Gaussian filters, specifically the Gaussian particle filter (GPF), and introduces the particle flow Gaussian particle filter (PFGPF), which combines the benefits of both methods. Chapter 5 examines the Gaussian sum filter (GSF), the Gaussian sum particle filter (GSPF), and proposes the particle flow Gaussian sum particle filter (PFGSPF), which utilizes a bank of PFGPF to represent the predictive and posterior distributions as a Gaussian mixture density. Lastly, Chapter 6 concludes the thesis and suggests possible direction for future research.

Chapter 2

Background

In this chapter, we outline the problem statement and provide a brief overview of the previous methods proposed prior to this work.

2.1 Notations

This thesis utilizes the following notations to represent their respective quantities:

$\mathbf{x}$	- $d \times 1$ state vector of the system
$\mathbf{x}_0$	- initial state of the system
$\mathbf{x}_t$	- state of the system at time step t
x_i	- i-th entry of state vector $\mathbf{x}$
$\mathbf{x}_t^i$	- i-th particle of the state $\mathbf{x}_t$
$\mathbf{z}_t$	- observation at time step t
$g_t(\cdot)$	- state transition model
$h_t(\cdot)$	- observation model
v_t	- process noise
u_t	- observation noise
w_t^i	- weight of the i-th sample at time step t
N_p	- Number of particles/samples
w_t^{*i}	- true weight of the i-th sample at time step t
N_{eff}	- effective number of samples
λ	- psuedo time interval
$f(\mathbf{x}, \lambda)$	- drift term of the SDE that describes the particle flow
$\sigma(\mathbf{x}, \lambda)$	- diffusion term of the SDE that describes the particle flow
$Q(\mathbf{x}, \lambda)$	- diffusion matrix
$g^{'}(\mathbf{x})$	- prior density
$h^{'}(\mathbf{x})$	- likelihood

$K(\cdot)$	- normalising constant for the posterior distribution
$A(\cdot)$	- flow parameter (EDH)
$b(\cdot)$	- flow parameter (EDH)
P	- predicted covariance matrix
$H(\cdot)$	- observation matrix for linear model or Jacobian matrix for non-linear models
R	- observation covariance matrix
$\overline{\mathbf{x}}_0$	- predicted mean
$\overline{\mathbf{x}}_\lambda$	- mean of the intermediate distribution at pseudo time step λ
$A^i(\cdot)$	- flow parameter of i-th particle (LEDH)
$b^i(\cdot)$	- flow parameter of i-th particle (LEDH)
$\mathbf{x}^i{}_\lambda$	- i-th particle of the intermediate distribution at pseudo time step λ
$\{\eta_0^i\}_{i=1}^{N_p}$	- set of predicted particles ($\lambda = 0$)
$\{\eta_{\lambda_j}^i\}_{i=1}^{N_p}$	- set of intermediate particles ($\lambda = \lambda_j$)
$\{\eta_1^i\}_{i=1}^{N_p}$	- set of migrated particles ($\lambda = 1$)
N_λ	- number of pseudo discretized steps
ϵ_j	- step size of pseudo time discretisation
$T(\cdot)$	- deterministic transport map
$\lvert\dot{T}(\cdot)\rvert$	- absolute value of the determinant of the Jacobian of $T(\cdot)$
$\{\bar{\eta}^i\}_{i=1}^{N_p}$	- set of deterministic particles
$\bar{\eta}_0$	- mean of the deterministic particles $\{\bar{\eta}^i\}_{i=1}^{N_p}$
$\overline{\mu}_t$	- mean of the Gaussian approximation of predictive distribution at time step t
$\overline{\Sigma}_t$	- covariance of the Gaussian approximation of predictive distribution at time step t
μ_t	- mean of the Gaussian approximation of posterior distribution at time step t
Σ_t	- covariance of the Gaussian approximation of posterior distribution at time step t
G	- number of Gaussian densities in the Gaussian mixture
$\tilde{\alpha}_t^j$	- unnormalised mixing proportion of the j-th Gaussian density in the Gaussian sum approximation of the posterior at time step t
α_t^j	- normalised mixing proportion of the j-th Gaussian density in the Gaussian sum approximation of the posterior at time step t
G_{eff}	- effective number of Gaussian densities in the Gaussian mixture

2.2 Problem statement

In non-linear filtering, the unobserved state $\mathbf{x}_t$ of a system at a particular time step t is estimated by tracking the posterior density $p(\mathbf{x}_t|\mathbf{z}_{1:t})$ over that time step. Here $\mathbf{z}_{1:t} = \{\mathbf{z}_1, ..., \mathbf{z}_t\}$ refers to the observations collected up to the time step t. Both the state dynamic model and observation model are non-linear and can be expressed as follows: the state dynamic model is defined by (2.2), while the

measurement model is defined by (2.3).

$$\mathbf{x}_0 \sim p_0(\mathbf{x}) \ , \tag{2.1}$$

$$\mathbf{x}_t = g_t(\mathbf{x}_{t-1}, v_t) \ , \quad t = 1, 2, \ldots \tag{2.2}$$

$$\mathbf{z}_t = h_t(\mathbf{x}_t, u_t) \ , \quad t = 1, 2, \ldots \tag{2.3}$$

where $p_0(\mathbf{x})$ is the initial state distribution, g_t and h_t are the state transition and observation models, respectively.$\mathbf{z}_t$ is the observation generated by the observation model. The process noise and observation noise are represented by v_t and u_t, respectively. It is assumed that $g_t(\cdot, 0)$ is bounded, and $h_t(\cdot, 0)$ is a C^1 function, meaning it is differentiable at all points, and its derivatives are continuous [16]. The posterior distribution can be tracked using the recursive Bayesian methods [17].

2.3 Recursive Bayesian method

The above state space model can also be represented alternatively using conditional probability densities as given by (2.4) and (2.5).

$$\mathbf{x}_t \sim p(\mathbf{x}_t|\mathbf{x}_{t-1}) \ , \tag{2.4}$$

$$\mathbf{z}_t \sim p(\mathbf{z}_t|\mathbf{x}_t) \tag{2.5}$$

where $p(\mathbf{x}_t|\mathbf{x}_{t-1})$ and $p(\mathbf{z}_t|\mathbf{x}_t)$ represent the transition kernel and likelihood respectively. The recursive Bayesian method [17] consists of two main steps: prediction and update. The predictive density is calculated using equation (2.6) during the prediction step. In the update step, the posterior density is determined by incorporating the measurement obtained at time step t, using equation (2.7).

$$p(\mathbf{x}_t|\mathbf{z}_{1:t-1}) = \int p(\mathbf{x}_t|\mathbf{x}_{t-1}) p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1}) d\mathbf{x}_{t-1} \ , \tag{2.6}$$

$$p(\mathbf{x}_t|\mathbf{z}_{1:t}) = \frac{p(\mathbf{z}_t|\mathbf{x}_t) p(\mathbf{x}_t|\mathbf{z}_{1:t-1})}{p(\mathbf{z}_t|\mathbf{z}_{1:t})} \tag{2.7}$$

where, $p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1})$ represent the posterior distribution at $t-1$. $p(\mathbf{z}_t|\mathbf{z}_{1:t})$ represents the probability distribution of measurement at time t, given all previous measurements upto t. This appears as a normalizing constant in the measurement update step.

2.4 Particle filter

The particle filter [6] utilizes a collection of particles (or samples) and their corresponding weights to represent a probability distribution $p(\mathbf{x})$ as follows:

$$p(\mathbf{x}) = \Sigma_{i=1}^{N_p} w^i \delta(\mathbf{x} - \mathbf{x}^i) \tag{2.8}$$

where $\mathbf{x}^i$ denotes the i-th sample drawn from the distribution $p(\mathbf{x})$, N_p represents the number of samples. w^i is the weight of the particle $\mathbf{x}^i$, $\delta(\cdot)$ is the Dirac delta function. The sequential importance sampling (SIS) algorithm [6] is the fundamental approach used in particle filters, also known as bootstrap particle filter. This algorithm generates estimates based on a set of random particles and their corresponding weights to approximate the required posterior density function. The weighted approximation of posterior density at time t is given by (2.9).

$$p(\mathbf{x}_t|\mathbf{z}_{1:t}) = \Sigma_{i=1}^{N_p} w_t^i \delta(\mathbf{x}_t - \mathbf{x}_t^i) \tag{2.9}$$

where the weight w_t^i is computed using importance sampling [18, 19] as shown in (2.10).

$$w_t^i \propto w_{t-1}^i \frac{p(\mathbf{z}_t|\mathbf{x}_t^i)p(\mathbf{x}_t^i|\mathbf{x}_{t-1}^i)}{q(\mathbf{x}_t^i|\mathbf{x}_{t-1}^i, \mathbf{z}_t)} \tag{2.10}$$

where $q(.)$ is the importance density or proposal density , w_{t-1}^i denotes the importance weight of the particle $\mathbf{x}_{t-1}^i$ and $\Sigma_{i=1}^{N_p} w_t^i = 1$.

As the number of particles increases in the particle filter, the accuracy of the posterior approximation improves. When the number of particles reaches infinity, the posterior approximation of the particle filter converges to the true posterior. This is known as the asymptotic property of the particle filter.

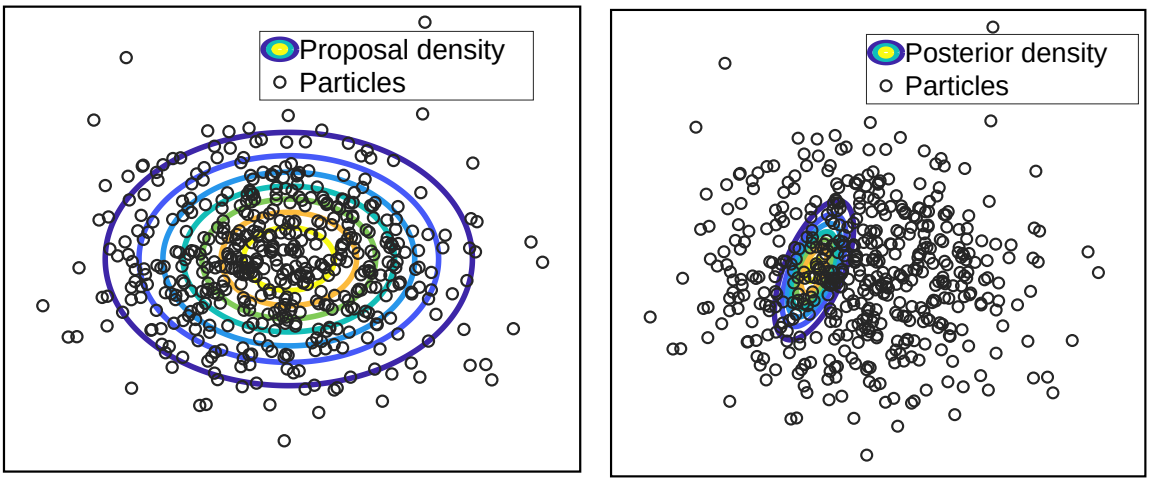
2.4.1 Weight Degeneracy

The most common drawback of the particle filter is weight degeneracy [20, 21, 22]. Weight degeneracy occurs when a large number of particles, used to approximate the posterior density, have insignificant weights after a few iterations. Due to this, the approximation of the posterior density becomes poor, as shown in Figure 2.1, thus affecting the filter performance. The weight degeneracy can be measured by the effective sample size N_{eff} [18, 23], given by the (2.11).

$$N_{\text{eff}} = \frac{N_p}{1 + \text{Var}(w_t^*)} \tag{2.11}$$

where $w_t^{*i} = \frac{p(\mathbf{x}_t^i|\mathbf{z}_{1:t})}{q(\mathbf{x}_t^i|\mathbf{x}_{t-1}^i, \mathbf{z}_t)}$ is the true weight of the i-th particle. This cannot be computed exactly, but an estimate $\hat{N}_{\text{eff}}$ can be obtained, given by (2.12). If the value of $\hat{N}_{\text{eff}}$ is small, then the effect of weight degeneracy is severe [24].

$$\hat{N}_{\text{eff}} = \frac{1}{\sum_{i=1}^{N_p} (w_t^i)^2} \tag{2.12}$$



(a) Particles drawn from proposal density

(b) Particles approximating the posterior density

Figure 2.1: Weight degeneracy. 2.1a shows particle sampled from the proposal density and 2.1b shows the particle approximation of posterior density with weight degeneracy.

2.4.2 Optimal importance density

The optimal importance density is selected to minimize the variance in importance weights. The low variance in importance weights ensures that a larger number of particles have significant weights, resulting in a higher effective sample size N_{eff}. This reduces the effect of weight degeneracy. The optimal importance density is given by (2.14).

$$\pi(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{z}_t)_{\text{opt}} = p(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{z}_t) \tag{2.13}$$

$$= \frac{p(\mathbf{z}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{x}_{t-1})}{p(\mathbf{z}_t|\mathbf{x}_{t-1})} \tag{2.14}$$

In general, drawing particles from the optimal importance density is not easy. Thus suboptimal approximations of the optimal importance density are constructed in most models. The most commonly considered importance density is the prior distribution [24].

2.4.3 Resampling

Resampling is another way that is used to alleviate the impact of weight degeneracy by eliminating particles with small weights and replacing them with particles of larger weights. While resampling can be effective in mitigating weight degeneracy, it can result in the loss of diversity among the samples, known as sample impoverishment [24], as shown in Figure 2.2.

Sequential Importance Resampling (SIR) [24] filter is a variant of SIS filter that uses prior $p(\mathbf{x}_t|\mathbf{x}_{t-1}^i)$ as the importance density and uses the resampling step at the end of every iteration. The Auxiliary Par-

ticle Filter (APF) [25] is a variant of SIR that incorporates information from the new measurement to sample particles more efficiently. The Regularised particle filter [26] is another variation of SIR that resamples particles from a continuous approximation of the posterior density to avoid sample impoverishment. The Rao-Blackwellised particle filters [27] marginalize some states analytically to minimize the variance of Monte Carlo estimations. Using the unscented transformation, the unscented particle filter [28] approaches the ideal proposal distribution.

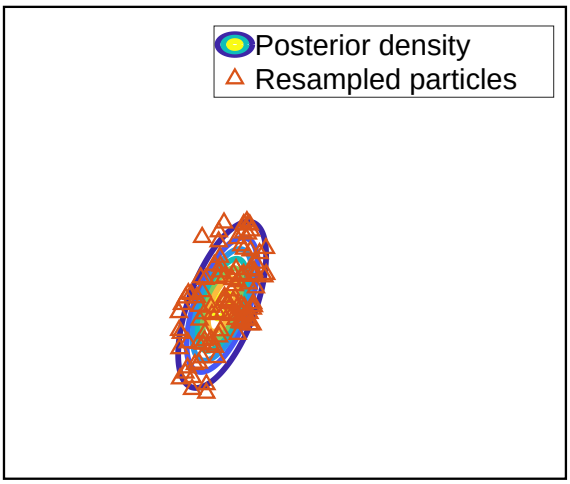


Figure 2.2: Resampling of particles

Although these particle filters are effective in many scenarios, their performance degrades in high-dimensional state spaces. Several different approaches that involve combining MCMC methods with particle filters are proposed to improve the performance in high-dimensional filtering and in settings where the measurements are highly informative.

2.5 Particle filter with MCMC methods

An alternative approach to avoid weight degeneracy in high dimensional filtering is incorporating MCMC methods within a particle filter [26, 29, 30, 31, 32, 33]. The resample-move algorithm [30], incorporates MCMC after the resampling step to diversify the particles. Additional methods have been proposed with a similar approach [34, 35], which modify the stochastic differential equation (SDE) representing state dynamics to simplify and improve MCMC steps. However, the significant limitation of these methods is that it requires a large number of MCMC steps after resampling to produce a set of particles that accurately represents an independent sample of the target posterior. Particle filters incorporating sequential MCMC (SMCMC) [29, 32, 33, 36] methods avoid the resampling step and utilize MCMC iterations to sample directly from the importance density. The major limitation of this method is identifying the effective MCMC kernels.

2.6 Particle flow filters

Particle flow filters [1, 7, 8, 9, 10, 11] use a transport map or flow equation to move particles from the prior to posterior distribution, eliminating the need for importance sampling and preventing weight degeneracy. In [7], a homotopy links the log prior and log posterior, and a PDE is used to transport particles from prior to posterior distribution. However, most particle flow methods developed are analytically intractable. A significant exception is the exact particle flow filter [1], which assumes a linear measurement model and Gaussian prior and posterior distributions. Despite their advantages, these filters lack statistical consistency due to numerical approximation and requires assumptions in practical implementation. To fight the statistical inconsistency an alternative approach of incorporating particle flow or transport maps within a particle filter is proposed.

2.7 Particle flow particle filters

Particle flow algorithms display very promising characteristics, but there is still a lack of good theoretical or practical knowledge of how approximations must be considered while developing the algorithms. An alternate approach is to generate a proposal or importance sample distribution within a particle filter using particle flow concepts. This approach uses the particle flow to move particles to a significant portion of the posterior distribution and also consists of the theoretical understanding of that of a particle filter.

The Gaussian particle flow importance sampling (GPFIS) [37] filter uses approximations of Gaussian flows to sample from non-Gaussian models and updates weights after each particle flow iteration. The stochastic particle flow [38] method approximates the posterior with Gaussian mixtures using stationary solutions to the Fokker-Planck equation. The Gibbs flow [39] approach numerically integrates probability densities in state updates at each intermediate flow step, but it is computationally expensive. The guided sequential Monte Carlo (GSMC) [40] methods generate the proposal density using deterministic transport maps and require calculating the determinant of the Jacobian matrix of the transport map for importance weight evaluation. Although these filters show impressive performance, they are computationally expensive, particularly in high-dimensional filtering.

In PF-APF [12], a simpler approach is proposed where particle flow is used to generate the proposal distribution within an auxiliary particle filter. The flows are implemented to sample auxiliary variables that construct an importance sampling distribution that is close to the posterior. In [41], particle flows are used to construct the proposal distribution within a standard particle filter. However, in high dimensions, it can be challenging to construct an appropriate proposal distribution. In the following chapter, we discuss the method where an importance sample distribution is constructed using invertible particle flows.

Chapter 3

Particle filter with invertible particle flow

In Chapter 2, we discussed briefly the different filters that were proposed prior to our work. Our work is similar to the particle flow with invertible mapping [4]. In this chapter, we will discuss mainly the deterministic particle flows [1, 2, 3] (3.1.1,3.1.2), their numerical implementation [3] (3.1.3), and the integration of an invertible particle flow, (constructed using the deterministic transport map) into a particle filter framework (3.2).

3.1 Particle flow

The process of particle flow involves moving predicted particles toward the posterior distribution. This movement is described by a stochastic differential equation (SDE) that operates within a pseudo time interval $\lambda \in [0, 1]$, as shown in equation (3.1).

$$d\mathbf{x} = f(\mathbf{x}, \lambda)d\lambda + \sigma(\mathbf{x}, \lambda)dW_\lambda \ , \tag{3.1}$$

where $\mathbf{x} = \{x_1, x_2, ..., x_d\}$ represents a d-dimensional vector, $f(\mathbf{x}, \lambda)$ is the drift term, $\sigma(\mathbf{x}, \lambda)$ represent the diffusion term of the SDE. W_λ is the M-dimensional Wiener or Brownian process. The drift term $f(\cdot)$ is governed by the Fokker-Planck equation (FPE) and some additional flow constraints [42]. The FPE describes the transition of the prior to posterior distribution. The one-dimensional FPE is given by (3.2)

$$\frac{\partial p(x, \lambda)}{\partial \lambda} = -\frac{\partial}{\partial x}[f(x, \lambda)p(x, \lambda)] + \frac{1}{2}\frac{\partial^2}{\partial x^2}[\sigma^2(x, \lambda)p(x, \lambda)] \tag{3.2}$$

where x is a one-dimensional random variable, $p(x, \lambda)$ denotes the probability distribution of x at psuedo time λ. The Fokker-Planck equation (FPE) for a d-dimensional system that corresponds to the stochastic process described by the SDE in equation (3.1) is presented in equation (3.3).

$$\frac{\partial p(\mathbf{x}, \lambda)}{\partial \lambda} = -\sum_{i=1}^{d}\frac{\partial}{\partial x_i}[f_i(\mathbf{x}, \lambda)p(\mathbf{x}, \lambda)] + \frac{1}{2}\sum_{i=1}^{d}\sum_{j=1}^{d}\frac{\partial^2}{x_i x_j}[Q_i(\mathbf{x}, \lambda)p(\mathbf{x}, \lambda)] \tag{3.3}$$

$$Q_{i,j}(\mathbf{x},\lambda) = \sum_{l=1}^{M} \sigma_{i,l}(\mathbf{x},\lambda)\sigma_{l,j}(\mathbf{x},\lambda) \tag{3.4}$$

where, $Q(\mathbf{x},\lambda)$ denotes the diffusion matrix, given by (3.4). The FPE can also be expressed as shown in (3.5).

$$\frac{\partial p(\mathbf{x},\lambda)}{\partial \lambda} = -\nabla \cdot (f(\mathbf{x},\lambda)p(\mathbf{x},\lambda)) + \frac{1}{2}\nabla^T Q(\mathbf{x},\lambda)p(\mathbf{x},\lambda)\nabla \tag{3.5}$$

where ∇ represents the spatial vector differentiation operator. In the logarithmic domain, the prior and posterior can be related by a homotopic relationship using the likelihood [7], as shown in (3.6).

$$\log p(\mathbf{x},\lambda) = \log g^{'}(\mathbf{x}) + \lambda \log h^{'}(\mathbf{x}) - \log K(\lambda) \tag{3.6}$$

where, $g^{'}(\mathbf{x})$ and $h^{'}(\mathbf{x})$ represent the prior and the likelihood respectively, $\lambda \in [0,1]$ is the psuedo time interval, $K(\lambda)$ is the normalising constant for the posterior distribution. As λ varies from 0 to 1, $p(\mathbf{x},\lambda)$ transforms from prior to the normalized posterior distribution. The partial derivative of $p(\mathbf{x},\lambda)$ with respect to the pseudo time interval λ is given by (3.7).

$$\frac{\partial p(\mathbf{x},\lambda)}{\partial \lambda} = p(\mathbf{x},\lambda)[\log h(\mathbf{x}) - \frac{\partial \log K(\lambda)}{\partial \lambda}] \tag{3.7}$$

(3.5) and (3.7) are combined together to get (3.8).

$$p(\mathbf{x},\lambda)[\log h^{'}(\mathbf{x}) - \frac{\partial \log K(\lambda)}{\partial \lambda}] = -\nabla \cdot (f(\mathbf{x},\lambda)p(\mathbf{x},\lambda)) + \frac{1}{2}\nabla^T Q(\mathbf{x},\lambda)p(\mathbf{x},\lambda)\nabla \tag{3.8}$$

$$[\log h^{'}(\mathbf{x}) - \frac{\partial \log K(\lambda)}{\partial \lambda}] = -f^T(\mathbf{x},\lambda) \cdot \nabla \log p(\mathbf{x},\lambda) - \nabla \cdot f(\mathbf{x},\lambda) + \frac{1}{2p(\mathbf{x},\lambda)}(\nabla^T Q(\mathbf{x},\lambda)p(\mathbf{x},\lambda)\nabla) \tag{3.9}$$

Using vector calculus (3.8) is further extended as shown in (3.9). Different particle flow solutions are obtained by solving (3.9) under different assumptions as proposed by F. Daum and J. Huang in [7, 8, 9, 1, 10, 11].

3.1.1 Exact Daum and Huang filter [1, 2]

F. Daum and J. Huang developed the exact Daum and Huang (EDH) filter in [1]. Considering the diffusion term $\sigma(\mathbf{x},\lambda)$ to be zero and neglecting the term $\frac{\partial \log K(\lambda)}{\partial \lambda}$, the (3.9) can be reduced to (3.10)

$$\log h^{'}(\mathbf{x}) + f^T(\mathbf{x},\lambda) \cdot \nabla \log p(\mathbf{x},\lambda) = -\nabla \cdot f(\mathbf{x},\lambda) \tag{3.10}$$

The exact particle flow is derived by solving (3.10). Different variants of exact particle flow are derived based on the solution to (3.10) [2]. One special case is considered where the prior and likelihood are assumed to be Gaussian. In this case the flow equation is as follows:

$$f(\mathbf{x},\lambda) = A(\lambda)\mathbf{x} + b(\lambda)\ , \tag{3.11}$$

where

$$A(\lambda) = -\frac{1}{2}PH(\lambda)^T(\lambda H(\lambda)PH(\lambda)^T + R)^{-1}H(\lambda) \ , \tag{3.12}$$

$$b(\lambda) = (I + 2\lambda A(\lambda))\times$$
$$[(I + \lambda A(\lambda))PH(\lambda)^T R^{-1}(\mathbf{z} - e((\lambda)) + A(\lambda)\overline{\mathbf{x}}_0] \ , \tag{3.13}$$

where $\overline{\mathbf{x}}_0$ is the predicted mean, P denotes the predicted covariance and R represents the observation covariance matrix. $H(\lambda)$ represents the observation matrix for linear observation models. But for non-linear observation models, $H(\lambda) = \frac{\partial h(\mathbf{x},0)}{\partial \mathbf{x}}\Big|_{\mathbf{x}=\overline{\mathbf{x}}_\lambda}$ represents the Jacobian matrix, that is obtained by performing linearization at the mean $\overline{\mathbf{x}}_\lambda$ of the intermediate distribution and $e(\lambda) = h(\overline{\mathbf{x}}_\lambda, 0) - H(\lambda)\overline{\mathbf{x}}_\lambda$ The flow parameters $A(\lambda)$ and $b(\lambda)$ are the constant for all the particles in this filter.

3.1.2 Localized exact Daum and Huang filter [3]

In [3], a slight modification is made in EDH filter to derive the localised exact Daum and Huang filter (LEDH). In this filter, the flow parameters are not constant for all the particles, but are derived individually for each particle. The flow parameters for the i-th particle is as follows:

$$A^i(\lambda) = -\frac{1}{2}PH^i(\lambda)^T(\lambda H^i(\lambda)PH^i(\lambda)^T + R)^{-1}H^i(\lambda) \ , \tag{3.14}$$

$$b^i(\lambda) = (I + 2\lambda A^i(\lambda))\times$$
$$[(I + \lambda A^i(\lambda))PH^i(\lambda)^T R^{-1}(\mathbf{z} - e^i(\lambda) + A^i(\lambda)\overline{\mathbf{x}}_0] \ . \tag{3.15}$$

Here the Jacobian matrix for the i-th particle $H^i(\lambda) = \frac{\partial h(\mathbf{x},0)}{\partial \mathbf{x}}\Big|_{\mathbf{x}=\mathbf{x}^i_\lambda}$ is obtained by performing the linearization at each individual particle and $e^i(\lambda) = h(\mathbf{x}^i_\lambda, 0) - H^i(\lambda)\mathbf{x}^i_\lambda$

3.1.3 Numerical Implementation [3]

In [3], numerical implementation of EDH and LEDH filters was proposed. These filters utilize discretized pseudo-time integration to estimate the particle flow. The $\lambda \in [0, 1]$ denotes the pseudo time interval and is divided into N_λ discrete steps $[\lambda_1, \lambda_2, ..., \lambda_{N_\lambda}]$, where $0 = \lambda_0 < \lambda_1 < ..... < \lambda_{N_\lambda} = 1$. The standard Euler's method is used to numerically integrate the particle flow from λ_0 to λ_{N_λ}. Let $\{\eta^i_{\lambda_j}\}_{i=1}^{N_p}$ denote the set of particles at the pseudo time interval λ_j. The functional mapping of the particles for the EDH filter is given by

$$\eta^i_{\lambda_j} = f_{\lambda_j}(\eta^i_{\lambda_{j-1}}) \tag{3.16}$$
$$= \eta^i_{\lambda_{j-1}} + \epsilon_j(A(\lambda_j)\eta^i_{\lambda_{j-1}} + b(\lambda_j)) \tag{3.17}$$

where $\epsilon_j = \lambda_j - \lambda_{j-1}$ is the step size. It can be varying for $j = [1, 2, ..., N_\lambda]$ and should satisfy the condition $\sum_{j=1}^{N_\lambda} \epsilon_j = 1$. The set of particles $\{\eta^i_0\}_{i=1}^{N_p}$ represent the particles at λ_0, before the migration

of particles using particle flow. The particle set $\{\eta_1^i\}_{i=1}^{N_p}$ denotes the particles after migration using the particle flow.

The functional mapping of LEDH is given by (3.19)

$$\eta_{\lambda_j}^i = f_{\lambda_j}^i(\eta_{\lambda_{j-1}}^i) \tag{3.18}$$

$$= \eta_{\lambda_{j-1}}^i + \epsilon_j(A^i(\lambda_j)\eta_{\lambda_{j-1}}^i + b^i(\lambda_j)) \tag{3.19}$$

Algorithm 1 and 2 outlines the pseudocode of EDH and LEDH filters, respectively.

Algorithm 1 Exact Duam and Huang filter (EDH)

Initialization: Draw $\{\mathbf{x}_0^i\}_{i=1}^{N_p}$ from the prior $p_0(\mathbf{x})$. Set $\hat{\mathbf{x}}_0$, m_0 be the mean and $\hat{P}_0$ be covariance of $p_0(\mathbf{x})$, respectively;
for $t = 1$ to T **do**
    for $i = 1, ..., N_p$ **do**
        Propagate particles $\mathbf{x}_t^i = g_t(\mathbf{x}_{t-1}^i, v_t)$;
    end for
    Calculate the ensemble mean $\hat{\mathbf{x}}_t$
    Set $\overline{\mathbf{x}}_0 = \hat{\mathbf{x}}_t$
    Apply EKF/UKF prediction:
        $(m_{t-1|t-1}, P_{t-1|t-1}) \rightarrow (m_{t|t-1}, P_{t|t-1})$;
    Set λ = 0;
    for $j = 1, ..., N_\lambda$ **do**
        Set $\lambda = \lambda + \epsilon_j$;
        for $i = 1, ..., N_p$ **do**
            Calculate $A(\lambda_j)$ and $b(\lambda_j)$ from (3.12) and (3.13)
              with the linearization being performed at $\hat{\mathbf{x}}_t$;
            Migrate particles:
              $\mathbf{x}_t^i = \mathbf{x}_t^i + \epsilon_j(A(\lambda_j)\mathbf{x}_t^i + b(\lambda_j))$;
            Calculate ensemble mean $\hat{\mathbf{x}}_t$
        end for
    end for
    Apply EKF/UKF update:
        $(m_{t|t-1}, P_{t|t-1}) \rightarrow (m_{t|t}, P_{t|t})$;
    Compute the mean $\bar{\mathbf{x}}_t = \frac{1}{N_p}\sum_{i=1}^{N_p}\mathbf{x}_t^i$;
    Optional: Redraw $\{\mathbf{x}_t^i\}_{i=1}^{N_p} \sim \mathcal{N}(\hat{x}_t, P_{t|t})$;
    Replace $m_{t|t} = \bar{\mathbf{x}}_t$
end for

.

Algorithm 2 Localized Exact Duam and Huang filter (LEDH)

Initialization: Draw $\{\mathbf{x}_0^i\}_{i=1}^{N_p}$ from the prior $p_0(\mathbf{x})$. Set $\hat{\mathbf{x}}_0$, m_0 be the mean and $\hat{P}_0$ be covariance of $p_0(\mathbf{x})$, respectively;
for $t = 1$ to T **do**
  for $i = 1, ..., N_p$ **do**
    Propagate particles $\mathbf{x}_t^i = g_t(\mathbf{x}_{t-1}^i, v_t)$;
  end for
  Calculate the ensemble mean $\hat{\mathbf{x}}_t$
  Set $\overline{\mathbf{x}}_0 = \hat{\mathbf{x}}_t$
  Apply EKF/UKF prediction:
    $(m_{t-1|t-1}, P_{t-1|t-1}) \rightarrow (m_{t|t-1}, P_{t|t-1})$;
  Set λ = 0;
  for $j = 1, ..., N_\lambda$ **do**
    Set $\lambda = \lambda + \epsilon_j$;
    for $i = 1, ..., N_p$ **do**
      Calculate $A^i(\lambda_j)$ and $b^i(\lambda_j)$ from (3.14) and (3.15) with the linearization being performed at $\mathbf{x}_t^i$;
      Migrate particles:
        $\mathbf{x}_t^i = \mathbf{x}_t^i + \epsilon_j(A^i(\lambda_j)\mathbf{x}_t^i + b^i(\lambda_j))$;
    end for
  end for
  Apply EKF/UKF update:
    $(m_{t|t-1}, P_{t|t-1}) \rightarrow (m_{t|t}, P_{t|t})$;
  Compute the mean $\bar{\mathbf{x}}_t = \sum_{i=1}^{N_p} \mathbf{x}_t^i$;
  Optional: Redraw $\{\mathbf{x}_t^i\}_{i=1}^{N_p} \sim \mathcal{N}(\hat{\mathbf{x}}_t, P_{t|t})$;
  Replace $m_{t|t} = \bar{\mathbf{x}}_t$
end for

.

3.2 Particle flow with Invertible Mapping [4]

Consider the particles set $\{\mathbf{x}_{t-1}^i\}_{i=1}^{N_p}$ represent the posterior distribution at time step $t-1$. During time step t, this particle set is propagated through the dynamic model $g_t(.)$ to produce the predicted particle set denoted by $\{\eta_0^i\}_{i=1}^{N_p}$ as shown in equation (3.20).

$$\eta_0^i = g_t(\mathbf{x}_{t-1}^i, v_t) \tag{3.20}$$

The predicted particles $\{\eta_0^i\}_{i=1}^{N_p}$ are migrated using the particle flow $f(\eta_\lambda, \lambda)$ to generate the migrated particles $\{\eta_1^i\}_{i=1}^{N_p}$. If the flow process defines a transport map $T(.)$ then particle set $\{\eta_0^i\}_{i=1}^{N_p}$ and $\{\eta_1^i\}_{i=1}^{N_p}$

can be related as shown by (3.21)

$$\eta_1^i = T(\eta_0^i; \mathbf{x}_{t-1}^i, \mathbf{z}_t) \tag{3.21}$$

Let $\pi_{\mathbf{X}}^{prior}(\cdot)$ and $\pi_{\mathbf{X}}^{post}(\cdot)$ represent the prior and posterior densities respectively. If $\mathbf{X} \sim \pi_{\mathbf{X}}^{prior}(\mathbf{x})$ and $\mathbf{Y} \sim \pi_{\mathbf{X}}^{post}(\mathbf{y})$ are two random variables such that $\mathbf{Y} = T(\mathbf{X})$, where $T(\cdot)$ is a deterministic mapping, then $T(\cdot)$ must satisfy the condition shown in (3.22).

$$\pi_{\mathbf{X}}^{post}(T(\mathbf{X}))|\dot{T}(\mathbf{X})| = \pi_{\mathbf{X}}^{prior}(\mathbf{x}) \tag{3.22}$$

where $|\dot{T}(\mathbf{X})|$ denotes the absolute value of the determinant of the Jacobian of the deterministic transport map $T(\cdot)$.

In particle flow particle filter framework, the migrated particles $\{\eta_1^i\}_{i=1}^{N_p}$ are considered to be drawn from the importance sampling density $\pi(\eta_1^i|\mathbf{x}_{t-1}^i, \mathbf{z}_t)$. If the transport mapping $\eta_1^i = T(\eta_0^i; \mathbf{x}_{t-1}^i, \mathbf{z}_t)$, is an invertible mapping, then according to (3.22), the importance sampling density $\pi(\eta_1^i|\mathbf{x}_{t-1}^i, \mathbf{z}_t)$ can be computed as shown in (3.23).

$$\pi(\eta_1^i|\mathbf{x}_{t-1}^i, \mathbf{z}_t) = \frac{p(\eta_0^i; \mathbf{x}_{t-1}^i, \mathbf{z}_t)}{|\dot{T}(\eta_0^i; \mathbf{x}_{t-1}^i, \mathbf{z}_t)|} \tag{3.23}$$

where $p(\eta_0^i; \mathbf{x}_{t-1}^i, \mathbf{z}_t)$ is the predictive distribution. Since the predictive particles $\{\eta_0^i\}_{i=1}^{N_p}$ are generated solely through the dynamic model, (3.23) can be modified as follows:

$$\pi(\eta_1^i|\mathbf{x}_{t-1}^i, \mathbf{z}_t) = \frac{p(\eta_0^i|\mathbf{x}_{t-1}^i)}{|\dot{T}(\eta_0^i; \mathbf{x}_{t-1}^i, \mathbf{z}_t)|} \tag{3.24}$$

.

In the particle filter framework, the importance sample weights can be estimated using (2.10). Combining (2.10) and (3.24), the importance sample weights can be modified as follows:

$$w_t^i \propto w_{t-1}^i \frac{p(\mathbf{z}_t|\eta_1^i)p(\eta_1^i|\mathbf{x}_{t-1}^i)|\dot{T}(\eta_0^i; \mathbf{x}_{t-1}^i, \mathbf{z}_t)|}{p(\eta_0^i|\mathbf{x}_{t-1}^i)} \tag{3.25}$$

3.2.1 Invertible particle flow using LEDH

If LEDH is used to migrate the predicted particles, then the transport function $T(\cdot)$ is given as shown in (3.27). The absolute value of the Jacobian determinant of the transport map $T(\cdot)$ is given in (3.30).

$$\eta_1^i = T(\eta_0^i; \mathbf{x}_{t-1}^i, \mathbf{z}_t) \tag{3.26}$$

$$= \prod_{j=1}^{N_\lambda} [(I + \epsilon_j A^i(\lambda_j))\eta_0^i + \epsilon_j b^i(\lambda_j)] \tag{3.27}$$

$$|\dot{T}(\eta_0^i; \mathbf{x}_{t-1}^i, \mathbf{z}_t)| = |\det(\frac{d\eta_1^i}{d\eta_0^i})| \tag{3.28}$$

$$= |\det(\frac{\prod_{j=1}^{N_\lambda}[(I + \epsilon_j A^i(\lambda_j))\eta_0^i + \epsilon_j b^i(\lambda_j)]}{d\eta_0^i} \tag{3.29}$$

$$= \prod_{j=1}^{N_\lambda} |\det(I + \epsilon_j A^i(\lambda_j))| \tag{3.30}$$

where $\det(\cdot)$ represents the determinant. The transport map $T(\cdot)$ is invertible if the step size ϵ_j is sufficiently small, i.e. $\epsilon_j < \frac{2\underline{r}(\lambda_j)}{\overline{p}\overline{h}^2}$ for $j \in \{1, ..., N_\lambda\}$ (See Theorem *IV.3* in [4]). Here $\overline{p}$ denotes the largest eigenvalue of P, $\underline{r}(\lambda_j)$ denotes the smallest eigenvalue of $(\lambda H^i(\lambda) P H^i(\lambda)^T + R)$. There exists a $\overline{h} > 0$ such that $\|H^i(\lambda_j)\|_F \leq \overline{h}$ for $\lambda \in [0, 1]$, where $\|\cdot\|_F$ denotes the Frobenius norm [4].

The importance sample weight of the i^{th} migrated particle is estimated as shown in (3.31)

$$w_t^i \propto w_{t-1}^i \frac{p(\mathbf{z}_t|\eta_1^i) p(\eta_1^i|\mathbf{x}_{t-1}^i) \prod_{j=1}^{N_\lambda} |\det(I + \epsilon_j A^i(\lambda_j))|}{p(\eta_0^i|\mathbf{x}_{t-1}^i)} \tag{3.31}$$

For the above choice of ϵ_j, it can be proved that $0 < |\dot{T}(\eta_0^i; \mathbf{x}_{t-1}^i, \mathbf{z}_t)| < \infty$ (See Lemma *IV.4* in [4]) . Thus all the migrated particles $\{\eta_1^i\}_{i=1}^{N_p}$ generated by applying the invertible transport map $T(\cdot)$ have finite importance sample weights.

3.2.2 Invertible particle flow using EDH

The invertible transport map $T(\cdot)$ constructed using EDH flow is given by (3.33). Since the flow parameters $A(\lambda_j)$ and $b(\lambda_j)$ are same for all the particles in EDH, the jacobian determinant $|\dot{T}(\eta_0^i; \mathbf{x}_{t-1}^i, \mathbf{z}_t)|$ is constant for all the particles and is given by (3.34), and the weight update expression is modified as shown in (3.35).

$$\eta_1^i = T(\eta_0^i; \mathbf{x}_{t-1}^i, \mathbf{z}_t) \tag{3.32}$$

$$= \prod_{j=1}^{N_\lambda} [(I + \epsilon_j A(\lambda_j))\eta_0^i + \epsilon_j b(\lambda_j)] \tag{3.33}$$

$$|\dot{T}(\eta_0^i; \mathbf{x}_{t-1}^i, \mathbf{z}_t)| = \prod_{j=1}^{N_\lambda} |\det(I + \epsilon_j A(\lambda_j))| \tag{3.34}$$

$$w_t^i \propto w_{t-1}^i \frac{p(\mathbf{z}_t|\eta_1^i) p(\eta_1^i|\mathbf{x}_{t-1}^i)}{p(\eta_0^i|\mathbf{x}_{t-1}^i)} \tag{3.35}$$

.

Algorithm 3 Particle flow particle filter (LEDH)

Initialization: Draw $\{\mathbf{x}_0^i\}_{i=1}^{N_p}$ from the prior $p_0(\mathbf{x})$. Set $\hat{\mathbf{x}}_0$ be the mean and $\hat{P}_0$ be covariance of $p_0(\mathbf{x})$, respectively;
Set $\{w_0^i\}_{i=1}^{N_p} = \frac{1}{N_p}$
for $t = 1$ to T **do**
    for $i = 1, ..., N_p$ **do**
        Apply EKF/UKF prediction:
        $(\mathbf{x}_{t-1}^i, P_{t-1}^i) \rightarrow (m_{t|t-1}^i, P^i)$;
        Calculate $\bar{\eta}^i = g_t(\mathbf{x}_{t-1}^i, 0)$
        Propagate particles $\eta_0^i = g_t(\mathbf{x}_{t-1}^i, v_t)$;
        Set $\eta_1^i = \eta_0^i$ and $\theta^i = 1$
        Calculate $\bar{\eta}_0^i = g_t(\mathbf{x}_{t-1}^i, 0)$;
    end for
    Set λ_j = 0;
    for $j = 1, ..., N_\lambda$ **do**
        Set $\lambda_j = \lambda_j + \epsilon_j$;
        for $i = 1, ..., N_p$ **do**
            Set $\bar{\eta}_0 = \bar{\eta}_0^i$ and $P = P^i$
            Calculate $A^i(\lambda_j)$ and $b^i(\lambda_j)$ from (3.14) and (3.15)
              with the linearization being performed at $\bar{\eta}^i$;
            Migrate:
              $\bar{\eta}^i = \bar{\eta}^i + \epsilon_j(A^i(\lambda_j)\bar{\eta}^i + b^i(\lambda_j))$;
            Migrate particles:
              $\eta_1^i = \eta_1^i + \epsilon_j(A^i(\lambda_j)\eta_1^i + b_j^i(\lambda_j))$;
            Calculate $\theta_i = \theta_i|\det(I + \epsilon_j A^i(\lambda_j))|$;
        end for
    end for
    for $i = 1, ..., N_p$ **do**
        Set $\mathbf{x}_t^i = \eta_1^i$
        Compute weights $w_t^i = \frac{p(\mathbf{z}_t|\mathbf{x}_t^i)p(\mathbf{x}_t^i|\mathbf{x}_{t-1}^i)\theta^i}{p(\eta_0^i|\mathbf{x}_{t-1}^i)} w_{t-1}^i$
    end for
    for $i = 1, ..., N_p$ **do**
        Normalize $w_t^i = \frac{w_t^i}{\sum_{s=1}^{N_p} w_t^s}$;
    end for
    for $i = 1, ..., N_p$ **do**
        Apply EKF/UKF update:
        $(m_{t|t-1}^i, P^i) \rightarrow (m_{t|t}^i, P_t^i)$;
    end for
    Compute the mean $\hat{\mathbf{x}}_t = \sum_{i=1}^{N_p} w_t^i \mathbf{x}_t^i$;
    Optional: Resample $\{\mathbf{x}_t^i, P_t^i, w_t^i\}_{i=1}^{N_p}$ to obtain $\{\mathbf{x}_t^i, P_t^i, \frac{1}{N_p}\}_{i=1}^{N_p}$
end for

.

Algorithm 4 Particle flow particle filter (EDH)

Initialization: Draw $\{\mathbf{x}_0^i\}_{i=1}^{N_p}$ from the prior $p_0(\mathbf{x})$. Set $\hat{\mathbf{x}}_0$ be the mean and $\hat{P}_0$ be covariance of $p_0(\mathbf{x})$, respectively;
Set $\{w_0^i\}_{i=1}^{N_p} = \frac{1}{N_p}$
for $t = 1$ to T **do**
    Apply EKF/UKF prediction:
        $(\hat{\mathbf{x}}_{t-1}, P_{t-1}) \rightarrow (m_{t|t-1}, P)$;
    for $i = 1, ..., N_p$ **do**
        Propagate particles $\eta_0^i = g_t(\mathbf{x}_{t-1}^i, v_t)$;
        Set $\eta_1^i = \eta_0^i$
    end for
    Calculate $\bar{\eta}_0 = g_t(\hat{\mathbf{x}}_{t-1}, 0)$;
    Set $\bar{\eta} = \bar{\eta}_0$
    Set λ_j = 0;
    for $j = 1, ..., N_\lambda$ **do**
        Set $\lambda_j = \lambda_j + \epsilon_j$;
        Calculate $A(\lambda_j)$ and $b(\lambda_j)$ from (3.12) and (3.13)
            with the linearization being performed at $\bar{\eta}$;
        Migrate:
            $\bar{\eta} = \bar{\eta} + \epsilon_j(A(\lambda_j)\bar{\eta} + b(\lambda_j))$;
        for $i = 1, ..., N_p$ **do**
            Migrate particles:
            $\eta_1^i = \eta_1^i + \epsilon_j(A(\lambda_j)\eta_1^i + b(\lambda_j))$;
        end for
    end for
    for $i = 1, ..., N_p$ **do**
        Set $\mathbf{x}_t^i = \eta_1^i$
        Compute weights $w_t^i = \frac{p(\mathbf{z}_t|\mathbf{x}_t^i)p(\mathbf{x}_t^i|\mathbf{x}_{t-1}^i)}{p(\eta_0^i|\mathbf{x}_{t-1}^i)} w_{t-1}^i$
    end for
    for $i = 1, ..., N_p$ **do**
        Normalize $w_t^i = \frac{w_t^i}{\sum_{s=1}^{N_p} w_t^s}$;
    end for
    Apply EKF/UKF update:
        $(m_{t|t-1}, P) \rightarrow (m_{t|t}, P_t)$;
    Compute the mean $\hat{\mathbf{x}}_t = \sum_{i=1}^{N_p} w_t^i \mathbf{x}_t^i$;
    Optional: Resample $\{\mathbf{x}_t^i, w_t^i\}_{i=1}^{N_p}$ to obtain $\{\mathbf{x}_t^i, \frac{1}{N_p}\}_{i=1}^{N_p}$
end for

.

In [12, 41], the particle flow does not satisfy the invertible mapping property (See Lemma *IV.2* in [4]). So the weight update equation in (3.25) cannot be applied.

In PFPF-EDH the flow parameters are computed using the auxiliary flow, by linearizing at the mean of the deterministic particles $\overline{\eta}_0$, whereas the flow parameters in PFPF-LEDH are computed by linearizing at each individual deterministic particle $\overline{\eta}_0^i$. The most computationally demanding step in both PFPF-EDH and PFPF-LEDH is the inverse operation in calculating the flow parameters [4].

In PFPF-LEDH, since individual flow parameters are calculated for each particle, the computational complexity of the matrix inverse operations is $O(N_p S^3)$, where S is the measurement dimension. The determinant θ_i is calculated seperately for each i-th particle. The weight update step is dependent on the determinant θ, thus the computational complexity of the weight update step in PFPF-LEDH is $O(N_p)$

In PFPF-EDH, the computational complexity of the inverse operation is $O(S^3)$. Since the flow parameters are constant for all the particles in PFPF-EDH, the computational complexity of the inverse operation is independent of N_p. Since the determinant θ_i is constant for each i-th particle, the weight update step is independent of the determinant θ. Thus the computational complexity of the weight update step is negligible compared to that of inverse operations in flow estimation.

Algorithm 3 and 4 outlines the PFPF-LEDH and PFPF-EDH filters, respectively. The predicted covariance matrix P required to compute the flow parameters $A(\lambda)$ and $b(\lambda)$ is estimated using EKF/UKF prediction step. Generally, $N_\lambda = 29$ exponential spaced step sizes are recommended for pseudo-time discretization. The ratio between step sizes is constant and is equal to 1.2 for $j = 2, 3, ...N_\lambda$. The initial step size $\epsilon_1 \approx 0.001$ [4].

Chapter 4

Particle flow Gaussian particle filter

The construction of importance density using an inverted particle flow in a particle filter framework was discussed in the previous chapter. This technique takes advantage of the particle flow method's appealing performance and adds the particle filter's statistical consistency, which the particle flow method lacks. In [43], an invertible particle flow is combined with sequential Markov chain Monte Carlo (SMCMC) methods to improve the performance of the filter. These methods are computationally expensive. This chapter discusses incorporating the invertible particle flow in a Gaussian particle filter (GPF) [14] to construct an effective importance density.

4.1 Gaussian filters

Gaussian filters are a class of filters that approximate the predictive and posterior distribution as Gaussian. The Extended Kalman filter (EKF) [44] and its variations [45, 46, 47, 48] fall under this class of filters. The mean and covariance of the Gaussian density that approximates the posterior are propagated through each iteration. In [49, 50, 51], the filters propagate the approximations of the first two moments of the densities through each iteration.

In EKF and its variants, the dynamic and measurement noise are assumed to be additive Gaussian noise. These filters perform poorly in highly non-linear models or when the posterior is multimodal. To improve the performance compared to EKF, various filters, including the unscented Kalman filter (UKF) [52] and filters proposed in [53], were proposed. These filters use deterministic samples to approximate the mean and covariance more accurately than EKF. These filters can still diverge in some non-linear models, and their improvement is problem-specific. If the assumption of additive Gaussian noise of dynamic and measurement noise is avoided, the filter performance can be improved. In [14], a Gaussian particle filter (GPF), a variant of a Gaussian filter, and a particle filter is proposed.

4.2 Gaussian particle filter

In the Gaussian particle filter (GPF) [14], the predictive and posterior distributions are approximated as Gaussian. Consider a set of particles $\{\mathbf{x}_{t-1}^i\}_{i=1}^{N_p}$ is drawn from the Gaussian density $\mathcal{N}(\cdot\,; \mu_{t-1}, \Sigma_{t-1})$,

that approximates the posterior at time step $t-1$. The μ_{t-1} and Σ_{t-1} are the mean and covariance of this Gaussian distribution. As in particle filters, the GPF consists of prediction and update steps.

4.2.1 Prediction

In the prediction step, the particles $\{\mathbf{x}_{t-1}^i\}_{i=1}^{N_p}$ drawn from the Gaussian density $\mathcal{N}(\cdot\,;\mu_{t-1},\Sigma_{t-1})$ are propagated through the dynamic model to generate the predicted set of particles $\{\mathbf{x}_{t|t-1}^i\}_{i=1}^{N_p}$. These predicted particles are considered to be drawn from the transitional kernel $p(\mathbf{x}_t|\mathbf{x}_{t-1})$ as shown in (4.1).

$$\mathbf{x}_{t|t-1}^i \sim p(\mathbf{x}_t|\mathbf{x}_{t-1}^i) \tag{4.1}$$

Using Monte Carlo approximation, the predictive density can be approximated from the transitional kernel as shown in (4.2)

$$p(\mathbf{x}_t|\mathbf{z}_{1:t-1}) \approx \frac{1}{N_p}\sum_{i=1}^{N_p} p(\mathbf{x}_t|\mathbf{x}_{t-1}^i) \tag{4.2}$$

A Gaussian distribution $\mathcal{N}(\cdot\,;\overline{\mu}_t,\overline{\Sigma}_t)$, constructed from the set of predicted particles $\{\mathbf{x}_{t|t-1}^i\}_{i=1}^{N_p}$ with the mean and covariance given by (4.4) and (4.5) is used to approximate the predictive distribution as shown in (4.3).

$$p(\mathbf{x}_t|\mathbf{z}_{1:t-1}) \approx \mathcal{N}(\cdot\,;\overline{\mu}_t,\overline{\Sigma}_t)\,, \tag{4.3}$$

$$\overline{\mu}_t = \frac{1}{N_p}\sum\nolimits_{i=1}^{N_p}\mathbf{x}_{t|t-1}^i\,, \tag{4.4}$$

$$\overline{\Sigma}_t = \frac{1}{N_p}\sum\nolimits_{i=1}^{N_p}(\mathbf{x}_{t|t-1}^i-\overline{\mu}_t)(\mathbf{x}_{t|t-1}^i-\overline{\mu}_t)^T \tag{4.5}$$

4.2.2 Update

We recall from importance sampling [18, 19] that if samples $\mathbf{x}_t^i$ are drawn from an importance distribution $\pi(\cdot)$, the importance weight computed for each particle is

$$w_t^i \propto \frac{p(\mathbf{x}_t^i|\mathbf{z}_{1:t-1})p(\mathbf{z}_t|\mathbf{x}_t^i)}{\pi(\mathbf{x}_t^i|\mathbf{z}_{0:t})}\,, \tag{4.6}$$

where $p(\mathbf{x}_t|\mathbf{z}_{1:t-1})$ is the predictive distribution, $p(\mathbf{z}_t|\mathbf{x}_t)$ is the likelihood, and $\pi(\mathbf{x}_t|\mathbf{z}_{1:t})$ is the importance sampling distribution. In GPF, the predictive distribution is approximated by the Gaussian $\mathcal{N}(\cdot\,;\overline{\mu}_t,\overline{\Sigma}_t)$, so the importance weight of each particle is modified as shown in (4.7)

$$w_t^i \propto \frac{\mathcal{N}(\mathbf{x}_t^i;\overline{\mu}_t,\overline{\Sigma}_t)p(\mathbf{z}_t|\mathbf{x}_t^i)}{\pi(\mathbf{x}_t^i|\mathbf{z}_{1:t})}. \tag{4.7}$$

The posterior distribution is approximated by the Gaussian $\mathcal{N}(\cdot\,; \mu_t, \Sigma_t)$ with the mean and covariance computed empirically from the particles and normalized weights:

$$\mu_t = \sum_{i=1}^{N_p} w_t^i \mathbf{x}_t^i\,, \tag{4.8}$$

$$\Sigma_t = \sum_{i=1}^{N_p} w_t^i (\mathbf{x}_t^i - \mu_t)(\mathbf{x}_t^i - \mu_t)^T. \tag{4.9}$$

The GPF has asymptotic property i.e. the mean μ_t and covariance Σ_t converge to the minimum mean square error (MMSE) estimates of posterior mean and covariance i.e.,$\mu_t = E[\mathbf{x}_t|\mathbf{z}_{0:t}]$ and Σ_t = $E[(\mathbf{x}_t - \mu_t)(\mathbf{x}_t - \mu_t)^T|\mathbf{z}_{0:t}]$ respectively as $N_p \to \infty$ (see Theorem 1 and Corollary 1 in [14]). This does not hold true for EKF and UKF, thus GPF has better performance compared to these filters.

The importance sample distribution $\pi(\cdot)$ can be user-specific depending upon the problem. The predictive distribution is a simple choice for $\pi(\cdot)$. Algorithm 5 outlines the pseudocode of GPF.

Algorithm 5 Gaussian particle filter (GPF)

Initialization: Draw $\{\mathbf{x}_0^i\}_{i=1}^{N_p}$ from the prior $p_0(\mathbf{x})$. Set $\hat{\mathbf{x}}_0$ be the mean and $\hat{P}_0$ be covariance of $p_0(\mathbf{x})$, respectively;
Set $\{w_0^i\}_{i=1}^{N_p} = \frac{1}{N_p}$
for $t = 1$ to T **do**
    $\{\mathbf{x}_{t-1}^i\}_{i=1}^{N_p} \sim \mathcal{N}(\cdot\,; \mu_{t-1}, \Sigma_{t-1})$
    for $i = 1, ..., N_p$ **do**
        Propagate particles : $\mathbf{x}_{t|t-1}^i = g_t(\mathbf{x}_{t-1}^i, v_t)$
    end for
    Calculate ensemble mean and covariance of $\{\mathbf{x}_{t|t-1}^i\}_{i=1}^{N_p}$ using (4.4) and (4.5).
    Approximation of predictive density : $p(\mathbf{x}_t|\mathbf{z}_{1:t-1}) \approx \mathcal{N}(\cdot\,; \overline{\mu}_t, \overline{\Sigma}_t)$
    for $t = 1$ to T **do**
        Set $\mathbf{x}_t^i = \mathbf{x}_{t|t-1}^i$
        Compute weights using (4.7)
    end for
    Estimate the weighted mean and covariances by (4.8) and (4.9)
    Posterior approximation : $p(\mathbf{x}_t|\mathbf{z}_{1:t}) \approx \mathcal{N}(\cdot\,; \mu_t, \Sigma_t)$
    Estimated state $\hat{\mathbf{x}}_t = \mu_t$
end for

4.3 Particle flow Gaussian particle filter

The invertible particle flows (EDH and LEDH) are constructed based on Gaussian assumptions and the GPF approximates the predictive and posterior distribution as Gaussian. Considering this, we propose the particle flow Gaussian particle filter (PFGPF), where an invertible particle flow is incorporated into the Gaussian particle filter.

4.3.1 The prediction step

Consider the Gaussian density $\mathcal{N}(\mathbf{x}_{t-1}; \mu_{t-1}, \Sigma_{t-1})$ approximates the posterior distribution at the time step $t-1$. A set of particles $\{\mathbf{x}_{t-1}^i\}_{i=1}^{N_p}$ are drawn from this Gaussian density. A new set of particles known as the predictive particles $\{\eta_0^i\}_{i=1}^{N_p}$ is generated by propagating the particle set $\{\mathbf{x}_{t-1}^i\}_{i=1}^{N_p}$ through the dynamic mode as shown in (4.10)

$$\eta_0^i = g_t(\mathbf{x}_{t-1}^i, v_t). \tag{4.10}$$

The ensemble mean and covariance of the predicted particles are computed to construct the Gaussian density $\mathcal{N}(\cdot\,; \overline{\mu}_{t|t-1}, \overline{\Sigma}_{t|t-1})$, that approximate the predictive distribution $p(\eta_0^i|\mathbf{x}_{t-1}^i, \mathbf{z}_{t-1})$

$$\overline{\mu}_{t|t-1} = \frac{1}{N_p}\sum\nolimits_{i=1}^{N_p} \eta_0^i \ , \tag{4.11}$$

$$\overline{\Sigma}_{t|t-1} = \frac{1}{N_p}\sum\nolimits_{i=1}^{N_p} (\eta_0^i - \overline{\mu}_{t|t-1})(\eta_0^i - \overline{\mu}_{t|t-1})^T \ . \tag{4.12}$$

If LEDH flow is used , then a set of deterministic particles $\{\bar{\eta}_0^i\}_{i=1}^{N_p}$ is generated by propagating the particles $\{\mathbf{x}_{t-1}^i\}_{i=1}^{N_p}$ through the dynamic model with no noise as shown in (4.13). In case of EDH, a deterministic mean $\{\bar{\eta}_0\}$ is computed as shown in (4.14), where $\hat{\mathbf{x}}_{t-1}$ is the mean of the particle set $\{\mathbf{x}_{t-1}^i\}_{i=1}^{N_p}$

$$\bar{\eta}_0^i = g_t(\mathbf{x}_{t-1}^i, 0) \ . \tag{4.13}$$

$$\bar{\eta}_0 = g_t(\hat{\mathbf{x}}_{t-1}, 0) \ . \tag{4.14}$$

The linearisation is performed at $\bar{\eta}^i$ (in LEDH) and $\bar{\eta}$ (in EDH) to compute the flow parameters. The flow parameters are then applied to the predicted particles $\{\eta_0^i\}_{i=1}^{N_p}$ to generate the migrated particles $\{\eta_1^i\}_{i=1}^{N_p}$. This method is followed to construct an invertible transport map $T(\cdot)$ using the particle flows under mild assumptions and numerical implementation as discussed in 3.2.1.

4.3.2 The update step

As the predictive density is approximated by the Gaussian density $\mathcal{N}(\eta_0^i; \overline{\mu}_{t|t-1}, \overline{\Sigma}_{t|t-1})$, the importance sampling distribution $\pi(\cdot)$ is computed as

$$\pi(\eta_1^i|\mathbf{z}_{1:t}) = \frac{\mathcal{N}(\eta_0^i; \overline{\mu}_{t|t-1}, \overline{\Sigma}_{t|t-1})}{|\dot{T}(\eta_0^i; \mathbf{z}_t, \mathbf{x}_{t-1}^i)|} \ , \tag{4.15}$$

where $T(\cdot)$ is the mapping function between the predicted particles and migrated particles. The denominator in (4.15) is the absolute value of the Jacobian determinant of the transport mapping function $T(\cdot)$ given by (3.30) for LEDH and (3.34) for EDH respectively. From (3.31) and (4.15), the importance weight of the i-th migrated particle η_1^i is modified as

$$w_t^i \propto \frac{\mathcal{N}(\eta_1^i; \overline{\mu}_{t|t-1}, \overline{\Sigma}_{t|t-1}) p(\mathbf{z}_t|\eta_1^i) |\dot{T}(\eta_0^i; \mathbf{z}_t, \mathbf{x}_{t-1}^i)|}{\mathcal{N}(\eta_0^i; \overline{\mu}_{t|t-1}, \overline{\Sigma}_{t|t-1})} \ . \tag{4.16}$$

Algorithm 6 Particle flow Gaussian particle filter (LEDH)

Initialization: Draw $\{\mathbf{x}_0^i\}_{i=1}^{N_p}$ from the prior $p_0(\mathbf{x})$. Set $\hat{\mathbf{x}}_0$ and $\hat{P}_0$ to be the mean and covariance of $p_0(\mathbf{x})$, respectively;
for $t = 1$ to T **do**
  for $i = 1, ..., N_p$ **do**
    Propagate particles $\eta_0^i = g_t(\mathbf{x}_{t-1}^i, v_t)$;
    Propagate particles $\overline{\eta}_0^i = g_t(\mathbf{x}_{t-1}^i, 0)$;
  end for
  Calculate ensemble mean of η_0^i: $\overline{\mu}_{t|t-1}$ by (4.11);
  Calculate ensemble covariance matrix of η_0^i: $\overline{\Sigma}_{t|t-1}$ by (4.12);
  Apply EKF/UKF prediction:
    $(\hat{\mathbf{x}}_{t-1}, P_{t-1|t-1}) \rightarrow (m_{t|t-1}, P_{t|t-1})$;
  Set $\eta_1^i = \eta_0^i$ and $\theta_i = 1$, $\overline{\eta}^i = \overline{\eta}_0^i$;
  Set λ_j = 0;
  for $j = 1, ..., N_\lambda$ **do**
    Set $\lambda_j = \lambda_j + \epsilon_j$;
    for $i = 1, ..., N_p$ **do**
      Calculate $A^i(\lambda_j)$ and $b^i(\lambda_j)$ from (3.14) and (3.15)
        with the linearization being performed at $\overline{\eta}^i$;
      Migrate particles:
        $\overline{\eta}^i = \overline{\eta}^i + \epsilon_j(A^i(\lambda_j)\overline{\eta}^i + b^i(\lambda_j))$;
      Migrate particles:
        $\eta_1^i = \eta_1^i + \epsilon_j(A^i(\lambda_j)\eta_1^i + b^i(\lambda_j))$;
      Calculate $\theta_i = \theta_i|det(I + \epsilon_j A^i(\lambda_j))|$;
    end for
  end for
  for $i = 1, ..., N_p$ **do**
    Compute weights w_t^i using (4.16);
  end for
  for $i = 1, ..., N_p$ **do**
    Normalize $w_t^i = \frac{w_t^i}{\sum_{s=1}^{N_p} w_t^s}$;
  end for
  Compute $\mu_t = \sum_{i=1}^{N_p} w_t^i \eta_1^i$;
  Compute $\Sigma_t = \sum_{i=1}^{N_p} w_t^i (\eta_1^i - \mu_t)(\eta_1^i - \mu_t)^T$;
  Apply EKF/UKF update:
    $(m_{t|t-1}, P_{t|t-1}) \rightarrow (m_{t|t}, P_{t|t})$;
  Draw $\{\mathbf{x}_t^i\}_{i=1}^{N_p} \sim \mathcal{N}(\mu_t, \Sigma_t)$;
  State estimate: $\hat{\mathbf{x}}_t = \mu_t$;
end for

The weighted mean μ_t and weighted covariance Σ_t of the migrated particles are estimated as shown in (4.17) and (4.18).

$$\mu_t = \sum_{i=1}^{N_p} w_t^i \eta_1^i \,, \tag{4.17}$$

$$\Sigma_t = \sum_{i=1}^{N_p} w_t^i (\eta_1^i - \mu_t)(\eta_1^i - \mu_t)^T. \tag{4.18}$$

The Gaussian density $\mathcal{N}(\mu_t, \Sigma_t)$ is used to approximate the posterior distribution. Algorithm 6 outlines the PFGPF with LEDH flow.

As discussed in 3.2, EKF/UKF is used to estimate the predicted covariance P, that is used to compute the flow parameters. The exponential pseudo-time discretization is used in this filter. The computation complexity of PFGPF is similar to that of PFPF. Due to the fact that PFGPF is basically a variant of GPF, it inherits the GPF's asymptotic properties. The PFGPF avoids the resampling of the particles, instead performs IID sampling from the Gaussian approximation of the posterior.

4.4 Simulations and Results

We examine performance of the proposed PFGPF algorithm in numerical simulations of multi-target acoustic tracking and high-dimensional filtering problems. The baseline algorithms include particle flow filters such as the EDH filter [1] and the LEDH filter [3], and particle flow particle filters such as the PFPF (EDH) and PFPF (LEDH) filters [4].

4.4.1 Multi-target acoustic tracking

4.4.1.1 Numerical Simulation Setup

We adopt the setup for numerical simulation of multi-target acoustic tracking used in [4, 54]. There are $M = 4$ acoustic targets with the state evolution dynamic given by

$$\mathbf{x}_t^m = F\mathbf{x}_{t-1}^m + v_t^m \,, \tag{4.19}$$

where $\mathbf{x}_t^m = [\mathrm{x}_t^m, \mathrm{y}_t^m, \dot{\mathrm{x}}_t^m, \dot{\mathrm{y}}_t^m]$ is the state of the m-th target, which consists of the position and velocity components of the target. The process noise $v_t^m \sim \mathcal{N}(0, V)$ is Gaussian, and the state transition matrix F is given by

$$F = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} . \tag{4.20}$$

The targets move independently in a region of size $40m \times 40m$. The s-th sensor, located at position r^s, records the superpositional measurement

$$\overline{\mathbf{z}}^s(\mathbf{x}_t) = \sum_{m=1}^{M} \frac{\psi}{\|[\mathrm{x}_t^m, \mathrm{y}_t^m]^T - r^s\|_2 + d_0} \ , \tag{4.21}$$

where $\|\cdot\|_2$ is the Euclidean norm, $\psi = 10$ is the amplitude of the sound emitted by the targets, and $d_0 = 0.1$. There are $N_s = 25$ sensors located in the given region. The measurement sensed by each sensor is affected by a Gaussian Noise $\mathcal{N}(0, \sigma_w^2)$ with variance $\sigma_w^2 = 0.01$, leading to highly informative measurements. The true initial states of the targets are $[12, 6, 0.001, 0.001]^T$, $[32, 32, -0.001, -0.005]^T$, $[20, 13, -0.1, 0.01]^T$ and $[15, 35, 0.002, 0.002]^T$. We have simulated 100 different trajectories with a constant velocity model and process noise covariance matrix given by

$$V = \frac{1}{20} \begin{bmatrix} 1/3 & 0 & 0.5 & 0 \\ 0 & 1/3 & 0 & 0.5 \\ 0.5 & 0 & 1 & 0 \\ 0 & 0.5 & 0 & 1 \end{bmatrix} \ . \tag{4.22}$$

4.4.1.2 Filter Implementation

Measurements are generated for each trajectory and each algorithm runs 5 times on each measurement set with different initial distributions. Each initial distribution $p_0(\mathbf{x})$ has a mean sampled from a Gaussian with true initial states of the targets as mean and standard deviation for position and velocity components set as 10 and 1 respectively. The process noise covariance matrix S for the filters is set as below which assumes that there is more uncertainty during tracking.

$$S = \begin{bmatrix} 3 & 0 & 0.1 & 0 \\ 0 & 3 & 0 & 0.1 \\ 0.1 & 0 & 0.03 & 0 \\ 0 & 0.1 & 0 & 0.03 \end{bmatrix} \ . \tag{4.23}$$

We run simulations with $N_p = 100$ and $N_p = 500$ particles for all the algorithms. Resampling in particle flow particle filter is performed when the effective sample size is less than $\frac{N_p}{2}$.

The error metric used to compare algorithmic performance is the optimal mass transfer (OMAT) metric [55]. The OMAT metric $d_p(\mathbf{X}, \hat{\mathbf{X}})$ is defined as

$$d_p(\mathbf{X}, \hat{\mathbf{X}}) = \frac{1}{C} \left(\min_{\pi \in \Pi} \sum_{c=1}^{C} d(\mathbf{X}_c, \hat{\mathbf{x}}_{\pi(c)})^p \right)^{1/p} \ , \tag{4.24}$$

where $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_C\}$ and $\hat{\mathbf{X}} = \{\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2, \ldots, \hat{\mathbf{x}}_C\}$ are the two sets to be compared, p is a fixed scalar parameter, Π is the set of the possible permutations of $\{1, 2, ..., C\}$, and $d(\mathbf{x}, \hat{\mathbf{x}})$ is the Euclidean distance between $\mathbf{x}$ and $\hat{\mathbf{x}}$. The value of p is set to 1.

4.4.1.3 Simulation Results

The average OMAT error for various algorithms is given in Table 4.1 for $N_p = 100$ and $N_p = 500$ particles, respectively. The proposed PFGPF has the smallest error compared to the other algorithms. 500 particles leads to significantly smaller average OMAT errors for importance sampling based methods, in particular the PFPF (LEDH) and the PFGPF, with the PFGPF leading to the smallest average tracking error. A sample plot of true trajectories and trajectories estimated using the PFGPF is shown in Figure 5.2.

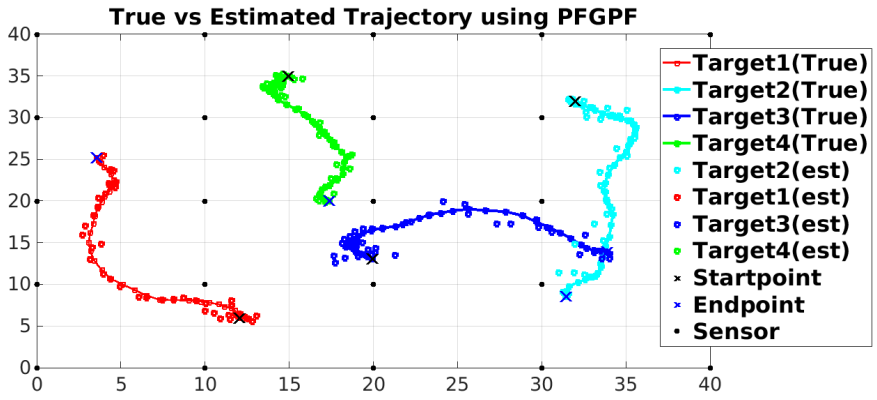


Figure 4.1: A sample true trajectory and estimated trajectory using the PFGPF.

Table 4.1: [Acoustic model, 16 dimensions] Average OMAT error (m) and average execution time (s) for various algorithms.

Algorithm	Average OMAT (m)		Average time (s)	
	$N_p = 100$	$N_p = 500$	$N_p = 100$	$N_p = 500$
EDH	2.53	2.61	0.01	0.01
LEDH	1.77	1.80	0.20	0.80
PFPF (EDH)	2.70	2.63	0.01	0.02
PFPF (LEDH)	1.25	0.76	0.38	1.70
PFGPF	**1.19**	**0.73**	**0.35**	**1.60**

4.4.2 Large spatial sensor networks: Skewed-t dynamic model and count measurements

4.4.2.1 Numerical Simulation Setup

We evaluate the proposed algorithm in a spatial sensor network simulation setup [33] that has been examined with the baseline algorithms [4]. The dynamic model follows Generalized Hyperbolic (GH)

skewed-t distribution, a heavy-tailed distribution that is useful for modelling physical processes and financial markets with extreme behavior and asymmetric data [56]. The transition kernel is given by

$$p(\mathbf{x}_t|\mathbf{x}_{t-1}) = K_{\frac{\nu+d}{2}}\sqrt{(\nu+S(\mathbf{x}_t))(\gamma^T\Sigma^{-1}\gamma)} \times \frac{e^{(\mathbf{x}_t-\mu_t)^T\Sigma^{-1}\gamma}}{\sqrt{(\nu+S(\mathbf{x}_t))(\gamma^T\Sigma^{-1}\gamma)}^{-\frac{\nu+d}{2}}(1+\frac{S(\mathbf{x}_t)}{\nu})^{\frac{\nu+d}{2}}} , \tag{4.25}$$

where d is the number of sensors which are deployed uniformly on a two dimensional grid $\{1, 2, ..., \sqrt{d}\}\times \{1, 2, ..., \sqrt{d}\}$, $K_{\frac{\nu+d}{2}}$ is the modified Bessel function of the second kind of order $\frac{\nu+d}{2}$, $\mu_t = \alpha\mathbf{x}_{t-1}$, $S(\mathbf{x}_t) = (\mathbf{x}_t - \mu_t)^T\Sigma^{-1}(\mathbf{x}_t - \mu_t)$, and the (i, j)-th entry of Σ is

$$\Sigma_{i,j} = \alpha_0 e^{\frac{-||R^i-R^j||_2^2}{\beta}} + \alpha_1\delta_{i,j} , \tag{4.26}$$

where $\|\cdot\|_2$ is the Euclidean norm, R^i is the physical position of sensor i, and $\delta_{i,j}$ is the Kronecker delta symbol. We set $\alpha_0 = 3, \alpha_1 = 0.01, \beta = 20$ following [33, 4]. The shape of the distribution is defined by the parameters γ and ν. The covariance $\hat{\Sigma}$ is given by

$$\hat{\Sigma} = \frac{\nu}{\nu-2}\Sigma + \frac{\nu^2}{(2\nu-8)(\frac{\nu}{2}-1)^2}\gamma\gamma^T . \tag{4.27}$$

The measurements in this setup are count data which follow the Poisson distribution

$$p(\mathbf{z}_t|\mathbf{x}_t) = \prod_{c=1}^{d}\mathcal{P}_0(\mathbf{z}_t^c; m_1 e^{m_2\mathbf{x}_t^c}) , \tag{4.28}$$

where $\mathcal{P}_0(\cdot\,; m)$ is the Poisson(m) distribution, $m_1 = 1$, and $m_2 = \frac{1}{3}$. The value of d is set to 144.

4.4.2.2 Filter Implementation

Each experiment is simulated for 30 time steps and we conduct the simulations 100 times. We report the results for a shorter simulation time of 10 steps as well as done in [4]. $N_p = 200$ particles are used for all the algorithms. The measurement covariance R in this setup depends on the state $\mathbf{x}_t$, thus it is updated in each step of particle flow and before the EKF update in all algorithms. We follow the practice in [33, 4] to set the initial true state as 0 in each state dimension, as used by all the compared algorithms.

4.4.2.3 Simulation Results

Table 4.2 compares the mean squared error (MSE) of the algorithms used in this simulation. It has been observed in [4] that particle flow filters perform better in this simulation setup and results reported in Table 4.2 are consistent with this observation. The challenge for importance sampling-based methods in filtering in such high state dimensions is that the variance of importance samples can be high, leading to decreased performance in state estimation. Still, among all filters with asymptotic

statistical consistency properties of the mean estimator, namely the PFPF (EDH), the PFPF (LEDH) and the PFGPF, the PFGPF exhibits the smallest average MSE. Though the improvement in MSE of PFGPF is small compared to PFPF (EDH) and the PFPF (LEDH) in the case of 10 time steps, the improvement is significant when 30 time steps are considered. The PFGPF avoids the weight degeneracy by using invertible partilce flow to construct an efficient proposal density. In PFGPF, since there is no resampling step, it avoids the sample impoverishment.

Table 4.2: Average MSE in the large spatial sensor networks simulation. State dimension $d = 144$ and $N_p = 200$ particles.

Algorithm	Average MSE (m)		Average time (s)
	10 timesteps	30 timesteps	
EDH	0.69	0.62	0.08
LEDH	0.71	0.64	10.95
PFPF (EDH)	0.98	1.09	0.12
PFPF (LEDH)	0.97	1.08	22.28
PFGPF	0.94	0.87	21.01

Chapter 5

Particle flow Gaussian Sum particle filter

In the previous chapter, we explored the Gaussian particle filter and introduced the particle flow Gaussian particle filter (PFGPF). This filter combines a Gaussian particle filter with an invertible particle flow technique to construct the importance sample density. The PFGPF approximates the predictive and posterior distribution as Gaussian and has no resampling step, thus avoiding weight degeneracy. The Gaussian approximation may not be accurate when the posterior is multimodal. In this chapter; we discuss the filters where the Gaussian mixture distributions are used to approximate the predictive and posterior distribution. We propose the particle flow Gaussian sum particle filter (PFGSPF), which is built using the banks of PFGPF.

5.1 Gaussian Sum filters

Gaussian filters rely on approximating the predictive and posterior distributions using a single Gaussian density, whereas Gaussian sum filter [15] utilizes a Gaussian mixture for the same purpose. The use of a Gaussian mixture approximation can offer superior performance over a single Gaussian approximation, especially in highly complex non-linear and non-Gaussian models. In [57, 58, 59, 60], various approaches based on Gaussian sum filters are discussed. The Gaussian sum filter is built using banks of EKF.

Consider the posterior distribution at time $t-1$ is approximated by a Gaussian mixture as follows

$$p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1}) \approx \sum_{j=1}^{G} \alpha_{t-1}^{j} \mathcal{N}(\mathbf{x}_{t-1}; \mu_{t-1}^{j}, \Sigma_{t-1}^{j}) \tag{5.1}$$

where G is the total number of Gaussian distributions in the mixture, $(\alpha_{t-1}^{j}, \mu_{t-1}^{j}, \Sigma_{t-1}^{j})$ are the mixing proportion, the mean, and covariance of the j^{th} Gaussian distribution respectively.

At time t the predictive distribution $p(\mathbf{x}_t|\mathbf{z}_{1:t-1})$ is given as follows

$$p(\mathbf{x}_t|\mathbf{z}_{1:t-1}) = \int p(\mathbf{x}_t|\mathbf{x}_{t-1})p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1})d\mathbf{x}_{t-1} \tag{5.2}$$

$$= \int p(\mathbf{x}_t|\mathbf{x}_{t-1}) \sum_{j=1}^{G} \alpha_{t-1}^j \mathcal{N}(\mathbf{x}_{t-1}; \mu_{t-1}^j, \Sigma_{t-1}^j)d\mathbf{x}_{t-1} \tag{5.3}$$

$$= \sum_{j=1}^{G} \alpha_{t-1}^j \int p(\mathbf{x}_t|\mathbf{x}_{t-1})\mathcal{N}(\mathbf{x}_{t-1}; \mu_{t-1}^j, \Sigma_{t-1}^j)d\mathbf{x}_{t-1} \tag{5.4}$$

Each integral on the right-hand side of the (5.4) can be approximated as a Gaussian, upon linearization of the dynamic model $g_t(\cdot)$ about μ_{t-1}^j (See Theorem 2 in [15]). This results in the Gaussian mixture approximation of the predictive distribution as follows

$$p(\mathbf{x}_t|\mathbf{z}_{1:t-1}) \approx \sum_{j=1}^{G} \alpha_{t-1}^j \mathcal{N}(\mathbf{x}_t; \overline{\mu}_t^j, \overline{\Sigma}_t^j) \tag{5.5}$$

where $\overline{\mu}_t^j$ and $\overline{\Sigma}_t^j$ are the predicted mean and covariance using EKF. The posterior at time t can be approximated by a Gaussian mixture as follows (See Theorem 1 in [15])

$$p(\mathbf{x}_t|\mathbf{z}_{1:t}) = C_t^{-1} p(\mathbf{z}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{z}_{1:t-1}) \tag{5.6}$$

$$\approx C_t^{-1} \sum_{j=1}^{G} \alpha_{t-1}^j \mathcal{N}(\mathbf{x}_t; \overline{\mu}_t^j, \overline{\Sigma}_t^j) p(\mathbf{z}_t|\mathbf{x}_t) \tag{5.7}$$

$$\approx \sum_{j=1}^{G} \alpha_t^j \mathcal{N}(\mathbf{x}_t; \mu_t^j, \Sigma_t^j) \tag{5.8}$$

where $C_t = \int p(\mathbf{x}_t|\mathbf{z}_{1:t-1})p(\mathbf{z}_t|\mathbf{x}_t)d\mathbf{x}_{t-1}$ is the normalisation constant, $p(\mathbf{z}_t|\mathbf{x}_t)$ is the likelihood. μ_t^j and Σ_t^j are the updated mean and covariance using EKF. The mixing proportion α_t^j is updated as follows

$$\alpha_t^j = \frac{\alpha_{t-1}^j \gamma_t^j}{\sum_{j=1}^{G} \alpha_{t-1}^j \gamma_t^j} \tag{5.9}$$

where $\gamma_t^j = \mathcal{N}(\mathbf{z}_t; h_t(\overline{\mu}_t^j), H_t^{j^T} \overline{\Sigma}_t^j H_t^j + R_t)$, $\mathbf{z}_t$ is the measurement at time t, $h_t(\cdot)$ is the measurement model, $H_t^j = \frac{\partial h_t(\mathbf{x})}{\partial \mathbf{x}}\big|_{\mathbf{x}=\overline{\mu}_t^j}$, R_t is the covariance of the measurement noise. The MMSE estimates for the state and its covariance can be computed easily with the use of Gaussian mixture approximation as follows

$$\hat{\mathbf{x}}_t = \sum_{j=1}^{G} \alpha_t^j \mu_t^j \tag{5.10}$$

$$\hat{\Sigma}_t = \sum_{j=1}^{G} \alpha_t^j (\Sigma_t^j + (\hat{\mathbf{x}}_t - \mu_t^j)(\hat{\mathbf{x}}_t - \mu_t^j)^T) \tag{5.11}$$

In highly non-linear models, the Gaussian sum filter may diverge due to the linearizations in the EKF. This divergence is avoided by the Gaussian sum particle filter (GSPF).

5.2 Gaussian Sum particle filter

The construction of the Gaussian sum filter involves using banks of EKF, whereas the Gaussian sum particle filter (GSPF) [15] is constructed using banks of Gaussian particle filters (GPF). In Gaussian sum particle filters, the predictive and posterior distributions are approximated using a Gaussian mixture model, where the individual Gaussian densities within the mixture are assumed to have a small covariance matrix. At time step $t-1$, the posterior is approximated by a Gaussian mixture as shown in (5.1). For each Gaussian in the mixture, the GPF is applied. The prediction and update steps for the Gaussian components are discussed in the following subsections.

5.2.1 The prediction step

A set of particles $\{\mathbf{x}_{t-1}^{ij}\}_{i=1}^{N_p^*}$ are drawn from the j^{th} Gaussian distribution $\mathcal{N}(\mathbf{x}_{t-1}^{ij}; \mu_{t-1}^j, \Sigma_{t-1}^j)$. This particle set is propagated through the dynamic model $g_t(\cdot)$ to generate the predictive particles, $\mathbf{x}_t^{ij} = g_t(\mathbf{x}_{t-1}^{ij}, v_t)$. The empirical mean and covariance of this set of particles is used for Gaussian approximation, $\mathcal{N}(\cdot\,; \overline{\mu}_t^j, \overline{\Sigma}_t^j)$, of the predictive distribution,

$$\overline{\mu}_t^j = \frac{1}{N_p^*} \sum\nolimits_{i=1}^{N_p^*} \mathbf{x}_t^{ij} \ , \tag{5.12}$$

$$\overline{\Sigma}_t^j = \frac{1}{N_p^*} \sum\nolimits_{i=1}^{N_p^*} (\mathbf{x}_t^{ij} - \overline{\mu}_t^j)(\mathbf{x}_t^{ij} - \overline{\mu}_t^j)^T \tag{5.13}$$

The predictive distribution is approximated as $p(\mathbf{x}_t|\mathbf{z}_{1:t-1}) \approx \sum_{j=1}^{G} \alpha_{t-1}^j \mathcal{N}(\mathbf{x}_t; \overline{\mu}_t^j, \overline{\Sigma}_t^j)$.

5.2.2 The update step

The importance weight of the i^{th} particle in the j^{th} Gaussian is computed as

$$w_t^{ij} \propto \frac{\mathcal{N}(\mathbf{x}_t^{ij}; \overline{\mu}_t^j, \overline{\Sigma}_t^j) p(\mathbf{z}_t|\mathbf{x}_t^{ij})}{\pi(\mathbf{x}_t^{ij}|\mathbf{z}_{1:t})} \ . \tag{5.14}$$

where $\pi(\cdot|\mathbf{z}_{1:t})$ is the proposal density. A Gaussian density $\mathcal{N}(\mathbf{x}_t; \mu_t^j, \Sigma_t^j)$ is approximated from this particle set where, $\mu_t^j = \sum_{i=1}^{N_p^*} \underline{w}_t^{ij} \mathbf{x}_t^{ij}$ and $\Sigma_t^j = \sum_{i=1}^{N_p^*} \underline{w}_t^{ij} (\mathbf{x}_t^{ij} - \mu_t^j)(\mathbf{x}_t^{ij} - \mu_t^j)^T$ are weighted mean and covariance of the j^{th} Gaussian and $\underline{w}_t^{ij}$ are normalized w_t^{ij}, computed as follows

$$\underline{w}_t^{ij} = \frac{w_t^{ij}}{\sum_{i=1}^{N_p^*} w_t^{ij}} \tag{5.15}$$

The posterior distribution is approximated as $p(\mathbf{x}_t|\mathbf{z}_{1:t}) \approx \sum_{j=1}^{G} \alpha_t^j \mathcal{N}(\mathbf{x}_t; \mu_t^j, \Sigma_t^j)$, where the mixing proportion of the j^{th} Gaussian α_t^j are updated and normalized as follows,

$$\tilde{\alpha}_t^j = \alpha_{t-1}^j \frac{\sum_{i=1}^{N_p^*} w_t^{ij}}{\sum_{j=1}^{G} \sum_{i=1}^{N_p^*} w_t^{ij}} \ , \quad \alpha_t^j = \frac{\tilde{\alpha}_t^j}{\sum_{j=1}^{G} \tilde{\alpha}_t^j} \tag{5.16}$$

Algorithm 7 Gaussian Sum particle filter (GSPF)

Initialization: Draw $\{\mathbf{x}_0^i\}_{i=1}^{N_p}$ from the prior $p_0(\mathbf{x})$. Set $\hat{\mathbf{x}}_0$ be the mean and $\hat{P}_0$ be covariance of $p_0(\mathbf{x})$, respectively;
Set $\{\alpha_0^i\}_{i=1}^{G} = \frac{1}{G}$
for $t = 1$ to T **do**
    The posterior at $t-1$: $p(\mathbf{x}_t|\mathbf{z}_{1:t}) \approx \sum_{j=1}^{G} \alpha_{t-1}^j \mathcal{N}(\cdot\,; \mu_{t-1}, \Sigma_{t-1})$
    for $j = 1$ to G **do**
        $\{\mathbf{x}_{t-1}^{ij}\}_{i=1}^{N_p^*} \sim \mathcal{N}(\cdot\,; \mu_{t-1}^j, \Sigma_{t-1}^j)$
        for $i = 1, ..., N_p^*$ **do**
            Propagate particles : $\mathbf{x}_{t|t-1}^{ij} = g_t(\mathbf{x}_{t-1}^{ij}, v_t)$
        end for
        Calculate ensemble mean and covariance of $\{\mathbf{x}_{t|t-1}^{ij}\}_{i=1}^{N_p^*}$ using (5.12) and (5.13).
    end for
    Approximation of predictive density : $p(\mathbf{x}_t|\mathbf{z}_{1:t-1}) \approx \sum_{j=1}^{G} \alpha_{t-1}^j \mathcal{N}(\cdot\,; \overline{\mu}_t^j, \overline{\Sigma}_t^j)$
    for $j = 1$ to G **do**
        for $i = 1, ..., N_p^*$ **do**
            Set $\mathbf{x}_t^{ij} = \mathbf{x}_{t|t-1}^{ij}$
            Compute weights using (5.14)
            Normalise the weights: $\underline{w}_t^{ij} = \frac{w_t^{ij}}{\sum_{j=1}^{N_p^*} w_t^{ij}}$
        end for
        Estimate the weighted mean: $\mu_t^j = \sum_{i=1}^{N_p^*} \underline{w}_t^{ij} \mathbf{x}_t^{ij}$
        Estimate the weighted covariances : $\Sigma_t^j = \sum_{i=1}^{N_p^*} \underline{w}_t^{ij} (\mathbf{x}_t^{ij} - \mu_t^j)(\mathbf{x}_t^{ij} - \mu_t^j)^T$
        Compute $\tilde{\alpha}_t^j$: $\tilde{\alpha}_t^j = \alpha_{t-1}^j \frac{\sum_{i=1}^{N_p^*} w_t^{ij}}{\sum_{j=1}^{G} \sum_{i=1}^{N_p^*} w_t^{ij}}$
        Normalise α_t^j: $\alpha_t^j = \frac{\tilde{\alpha}_t^j}{\sum_{j=1}^{G} \tilde{\alpha}_t^j}$
    end for
    Posterior approximation : $p(\mathbf{x}_t|\mathbf{z}_{1:t}) \approx \sum_{j=1}^{G} \alpha_t^j \mathcal{N}(\cdot\,; \mu_t^j, \Sigma_t^j)$
    Estimated state $\hat{\mathbf{x}}_t = \sum_{j=1}^{G} \alpha_t^j \mu_t^j$
end for

.

The MMSE of the state $\mathbf{x}_t$ and its error covariance are computed as shown in (5.10) and (5.11), respectively. Algorithm 7 outlines the pseudocode of the GSPF filter.

The approximations in GSF and GSPF are valid only for small covariances (See Theorem 1 and 2 in [15]). As the filtering process continues, the covariance of the mixture components may increase. This can lead to the Gaussian mixture (GM) approximations becoming less accurate. Additionally, this can cause the mixture components to collapse, resulting in only one remaining distinct component. When the mixture components collapse, it becomes necessary to represent the posterior distributions as Gaussian mixtures with smaller covariances [15, 45]. This could be a difficult problem, particularly in

higher dimensions. Additionally, it is not desirable and can be computationally expensive. To mitigate this problem, the noise models are modeled as Gaussian mixture [57]. This leads to an exponential growth in the number of mixture components. Different methods have been proposed to reduce the number of mixture components [48]. These methods are difficult to implement and may not be adequate for many practical problems.

An alternative approach includes combining particle flow methods with Gaussian sum filtering. In [61], Gaussian mixture particle flow is proposed, where particle flow is derived by assuming the prior to be a Gaussian mixture and the measurement model to be linear Gaussian. To compute the flow, matrix inversion is necessary. However, as the state dimension increases, the size of the matrix also grows rapidly, making the computation of the flow more challenging.

In [62], similar approach is proposed where an approximate particle flow is constructed when both prior and likelihood are considered to be Gaussian mixture. A particle filter with invertible particle flow is constructed when the process noise and measurement noise are modelled as Gaussian mixture in [63]. A Gaussian mixture based particle flow filter is proposed in [43] for multimodal distributions. In the following section, we introduce particle flow Gaussian sum particle filter (PFGSPF) which is constructed using banks of PFGPF.

5.3 Particle flow Gaussian sum particle filter

The PFGSPF algorithm approximates the posterior and predictive distributions in nonlinear and non-Gaussian state-space models using a mixture of Gaussian distributions. The posterior distribution at $t-1$ is approximated by a weighted Gaussian sum distribution as shown in (5.1). Particle flow Gaussian particle filter (PFGPF) is applied to each Gaussian in the Gaussian mixture to generate a new Gaussian mixture that approximates the posterior distribution.

5.3.1 The prediction step

The prediction step in PFGSPF is similar to that of GSPF, The set of predictive particles $\{\eta_0^{ij}\}_{i=1}^{N_p^*}$ are generated by propagating the the particles $\{\mathbf{x}_{t-1}^{ij}\}_{i=1}^{N_p^*}$ sampled from the j^{th} Gaussian distribution $\mathcal{N}(\mathbf{x}_{t-1}^{ij}; \mu_{t-1}^j, \Sigma_{t-1}^j)$ through the dynamic model $g_t(\cdot)$ as shown in (5.17).

$$\eta_0^{ij} = g_t(\mathbf{x}_{t-1}^{ij}, v_t) \tag{5.17}$$

The Gaussian distribution $\mathcal{N}(\cdot\,; \overline{\mu}_t^j, \overline{\Sigma}_t^j)$ that approximates the predictive density is constructed from the empirical mean and covariance of the predicted particles computed as shown in (5.12) and (5.13), with $\mathbf{x}_t^{ij}$ replaced by η_0^{ij}.

The predicted particles are subjected to the invertible LEDH particle flow to produce the migrated particles $\{\eta_1^{ij}\}_{i=1}^{N_p^*}$. The flow parameters of the invertible flow are computed in the same way as mentioned in 4.3.1.

Algorithm 8 Particle flow Gaussian sum particle filter

Input: The posterior at time $t-1$
$p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1}) = \sum_{j=1}^{G} \alpha_{t-1}^{j} \mathcal{N}(\mathbf{x}_{t-1}; \mu_{t-1}^{j}, \Sigma_{t-1}^{j})$

1: **for** $j = 1$ to G **do**
2: Apply PFGPF to $\mathcal{N}(\mathbf{x}_{t-1}; \mu_{t-1}^{j}, \Sigma_{t-1}^{j})$; (Algorithm 0)
3: $\mathcal{N}(\mathbf{x}_{t-1}; \mu_{t-1}^{j}, \Sigma_{t-1}^{j}) \xrightarrow{\text{PFGPF}} \mathcal{N}(\mathbf{x}_{t}; \mu_{t}^{j}, \Sigma_{t}^{j})$;
4: **end for**
5: **for** $j = 1$ to G **do**
6: Update the mixing proportions using (5.16);
7: **end for**
8: Normalize the mixing proportions using (5.16);

Outputs: Posterior: $p(\mathbf{x}_t|\mathbf{z}_{1:t}) \approx \sum_{j=1}^{G} \alpha_t^j \mathcal{N}(\mathbf{x}_t; \mu_t^j, \Sigma_t^j)$ State estimation: $\hat{\mathbf{x}}_t = \sum_{j=1}^{G} \alpha_t^j \mu_t^j$

5.3.2 The update step

The proposal density $\pi(\cdot)$ determined according to the PFPF and PFGPF framework is given as follows

$$\pi(\eta_1^{ij}|\mathbf{z}_{0:t}) = \frac{\mathcal{N}(\eta_0^{ij}; \overline{\mu}_t^j, \overline{\Sigma}_t^j)}{|\dot{T}(\eta_0^{ij}; \mathbf{z}_t, \mathbf{x}_{t-1}^i)|} \tag{5.18}$$

where $|\dot{T}(\cdot)|$ is the Jacobian determinant of the mapping function $T(\cdot)$ between the predicted particles and migrated particles, estimated by (3.27). The importance weight of the migrated particle η_1^{ij} is computed as

$$w_t^{ij} \propto \frac{\mathcal{N}(\eta_1^{ij}; \overline{\mu}_t^j, \overline{\Sigma}_t^j) p(\mathbf{z}_t|\eta_1^{ij}) |\dot{T}(\eta_0^{ij}; \mathbf{z}_t, \mathbf{x}_{t-1}^{ij})|}{\mathcal{N}(\eta_0^{ij}; \overline{\mu}_t^j, \overline{\Sigma}_t^j)} \tag{5.19}$$

Following the PFGPF framework, after the weight update step, the Gaussian distribution $\mathcal{N}(\cdot\ \mu_t^j, \Sigma_t^j)$ is constructed where $\mu_t^j = \sum_{i=1}^{N_p^*} \underline{w}_t^{ij} \eta_1^{ij}$ and $\Sigma_t^j = \sum_{i=1}^{N_p^*} \underline{w}_t^{ij} (\eta_1^{ij} - \mu_t^j)(\eta_1^{ij} - \mu_t^j)^T$, and $\underline{w}_t^{ij}$ is the normalized w_t^{ij}.

All the G Gaussians constructed are now combined to form a Gaussian mixture $\sum_{j=1}^{G} \alpha_t^j \mathcal{N}(\mathbf{x}_t; \mu_t^j, \Sigma_t^j)$ that approximates the posterior. The mixing proportions are updated following (5.16) and (**??**). The estimated state $\hat{\mathbf{x}}_t$ is estimated as follows,

$$\hat{\mathbf{x}}_t = \sum_{j=1}^{G} \alpha_t^j \mu_t^j \tag{5.20}$$

In PFGSPF, N_p^* particles are sampled from each Gaussian distribution in the Gaussian mixture. Thus the total number of particles used in the implementation of PFGSPF is $N_p = N_p^* \times G$ particles. Algorithm 8 outlines the proposed filter PFGSPF, built using a bank of PFGPF.

5.3.3 Effective number of Gaussians

Though the number of Gaussians is fixed (G) over an entire simulation, their mixing proportions change over time reflecting the complexity of the distribution as time progresses. Some of the mixing proportions may have very low value indicating that the particular Gaussian has negligible contribution to the overall posterior density. As a measure of multi-modal complexity of the distribution, we propose the *effective number* of Gaussians $G_{\text{eff},t}$ at time step t defined as follows,

$$G_{\text{eff},t} = \frac{1}{\sum_{j=1}^{G} (\alpha_t^j)^2} \tag{5.21}$$

It is trivial to show that when only one Gaussian is signficiant (with $\alpha \approx 1$), the G_{eff} is close to 1 and when all the G Gaussians have equal weights, the G_{eff} is equal to G.

5.4 Simulations and Results

We evaluate the proposed algorithm in two challenging numerical simulations – multi-target tracking using acoustic sensors and a large spatial sensor network model. The performance of the proposed PFGSPF is compared with a number of related algorithms including the EDH [1], the LEDH [3], the PFPF (EDH) and the PFPF (LEDH) [4], and the PFGPF [16].

5.4.1 Multi-target acoustic tracking

5.4.1.1 Numerical simulation setup

The simulated acoustic tracking setup of [4, 54] involves $M = 4$ acoustic targets with state dynamics $\mathbf{x}_t^m = F\mathbf{x}_{t-1}^m + v_t^m$, where the state of the m^{th} target $\mathbf{x}_t^m = [\mathrm{x}_t^m, \mathrm{y}_t^m, \dot{\mathrm{x}}_t^m, \dot{\mathrm{y}}_t^m]$ consists of the position and velocity components. The process noise $v_t^m \sim \mathcal{N}(0, V)$. The s^{th} sensor, located at position r^s, records the measurement

$$\overline{\mathbf{z}}^s(\mathbf{x}_t) = \sum_{m=1}^{M} \frac{\psi}{\|[\mathrm{x}_t^m, \mathrm{y}_t^m]^T - r^s\|_2 + d_0} \tag{5.22}$$

where $\|\cdot\|_2$ is the Euclidean norm, ψ is the amplitude of the sound emitted by the targets, and $d_0 = 0.1$. There are $N_s = 25$ sensors located in the given region. The measurement noise follows $\mathcal{N}(0, \sigma_w^2)$ with variance $\sigma_w^2 = 0.01$. We simulate 100 trajectories with a constant velocity model and the corresponding measurements.

5.4.1.2 Filter Implementation and Results

Each algorithm runs 5 times with different initial distribution. All other parameter values are the same as those mentioned in 4.4.1. The numerical simulations are carried out for $N_p = 500$ and $N_p =$

2500 particles. For the PFGSPF, $N_p = N_p^* \times G$ and we vary G from 1 to 5. We set $N_p^* = 100$ and $N_p^* = 500$. Table 5.1 reports the average optimal mass transfer (OMAT) error and standard deviation of various filters. The error is estimated for tracks excluding the lost tracks (i.e., tracks which have an OMAT error $> 2\,\mathrm{m}$). The number of lost tracks (LT) is also shown in the table. We observe that at $N_p = 2500$, the proposed filter PFGSPF outperforms all the other filters with the smallest mean OMAT values and the lowest standard deviation compared to the other filters. We also observe that when N_p^* is small, the performance of the PFGSPF is not much affected with increase in number of Gaussians G, but as we increase N_p^* , the performance of the PFGSPF improves as we increase the number of Gaussians G.

Algorithm	$N_p = 500$		$N_p = 2500$	
	mean $\pm$ sd	#LT	mean $\pm$ sd	#LT
EDH	0.92$\pm$ 0.39	212	0.90 $\pm$ 0.37	212
LEDH	1.28$\pm$ 0.32	111	1.29$\pm$ 0.33	103
PFPF (EDH)	0.97$\pm$ 0.37	211	0.98$\pm$ 0.39	213
PFPF (LEDH)	0.73$\pm$ 0.19	5	0.74$\pm$ 0.22	4
PFGPF	0.72$\pm$ 0.16	2	0.77$\pm$ 0.19	0
	N_p=$N_p^* \times G$		N_p=$N_p^* \times G$	
PFGSPF	($N_p^* = 100$)	#LT	($N_p^* = 500$)	#LT
$G = 1$	0.76 $\pm$ 0.23	42	0.72 $\pm$ 0.16	4
$G = 2$	0.75 $\pm$ 0.25	40	0.70 $\pm$ 0.16	2
$G = 3$	0.76 $\pm$ 0.26	22	0.69 $\pm$ 0.13	1
$G = 4$	0.76 $\pm$ 0.26	33	0.69 $\pm$ 0.16	1
$G = 5$	0.75 $\pm$ 0.24	29	0.68 $\pm$ 0.13	3

Table 5.1: OMAT errors (m) recorded for the acoustic tracking simulation, excluding lost tracks (LT) defined as the estimated trajectories with average tracking error $> 2\,\mathrm{m}$.

5.4.2 Large spatial sensor networks: Skewed-t dynamic model and count measurements

5.4.2.1 Numerical Simulation Setup

We also examine the filters' performance in 144-dimensional large spatial sensor networks used in 4.4.2. The dynamic model follows Generalized Hyperbolic (GH) skewed-t distribution, a heavy-tailed distribution [56], where the model equations and parameter values used in the simulation can be found in 4.4.2.

5.4.2.2 Filter Implementation and Results

Each experiment is simulated for 30 time steps and we conduct the simulations for 100 times. The simulations are carried out for $N_p = [200, 400, 600, 800, 1000, 1400, 2000]$. For PFGSPF, simulations are carried out for both different G with constant N_p^* and different N_p^* with different G as shown in Table 5.2.

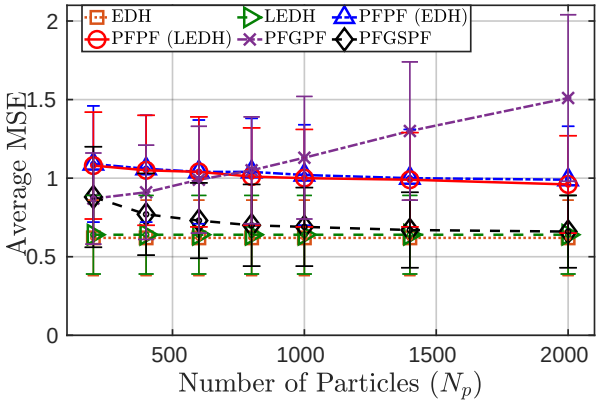


Figure 5.1: Average MSE vs N_p in the large spatial sensor networks simulation. For PFGSPF, $N_p^* = 200$ is fixed and G is varied. Error bars indicate standard deviation.

N_p	Average MSE [avg $\pm$ sd]	
	PFGSPF ($N_p^* \times G$)	PFGSPF ($N_p^* \times G$)
200	0.88 $\pm$ 0.32 (200x1)	-
400	0.77 $\pm$ 0.26 (200x2)	0.77 $\pm$ 0.26 (200x2)
600	0.73 $\pm$ 0.24 (200x3)	0.79 $\pm$ 0.28 (300x2)
800	0.70 $\pm$ 0.26 (200x4)	0.81 $\pm$ 0.29 (400x2)
1000	0.69 $\pm$ 0.25 (200x5)	0.84 $\pm$ 0.29 (500x2)
1400	0.67 $\pm$ 0.24 (200x7)	0.73 $\pm$ 0.27 (350x4)
2000	0.66 $\pm$ 0.23 (200x10)	0.77 $\pm$ 0.28 (500x4)

Table 5.2: Average MSE in the large spatial sensor networks simulation.

Figure 5.1 shows the average MSE of all the filters for different N_p over 100 simulations. We observe that the EDH has the lowest average MSE and standard deviation compared to other filters. As we increase the number of Gaussians G, the performance of the PFGSPF improves and approaches that of the EDH.

The PFGSPF is implemented with different values of N_p^* and G as shown in Table 5.2. We observe that, for fixed N_p^*, an increase in the number of Gaussians G improves its performance. Meanwhile,

the filter performance degrades as we increase N_p^* while keeping G constant. The same behaviour is observed for the PFGPF in Figure 5.1. We speculate that as N_p^* increases, more particles may fall into the regions of posterior with less probability, resulting in poorer approximation.

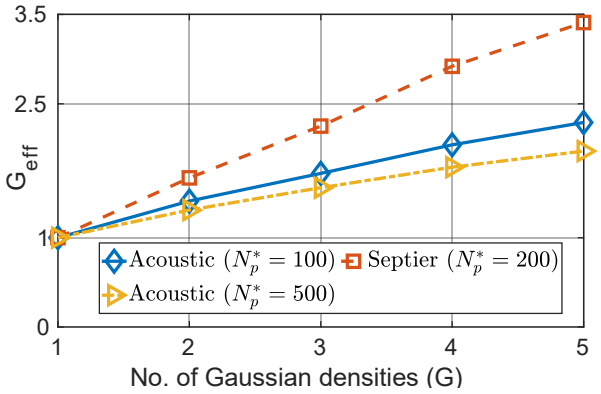


Figure 5.2: Average G_{eff} of PFGSPF. (Acoustic: 500 simulations, large spatial sensor network: 100 simulations).

Figure 5.2 shows the average effective number of Gaussians in the two numerical simulations for different values of G in the PFGSPF. In acoustic simulation, the average effective number of Gaussians is 2 when $G = 5$ are available. In the large spatial sensor network simulation, the average effective number of Gaussians is 3.5 when $G = 5$ are available. Thus the target distribution in the large spatial sensor network is considered to be more complex than the target distribution in acoustic simulation.

Chapter 6

Conclusions and Future Work

In this thesis, we have proposed the particle flow Gaussian particle filter (PFGPF) algorithm. It embeds invertible particle flow to generate the proposal distribution within the Gaussian particle filtering framework. We explore the capacity of the (local) Gaussian approximations, introduced from both the adopted localized invertible particle flow procedure and the Gaussian particle filter model, in high-dimensional non-linear state estimation tasks. Empirical results in two challenging state estimation tasks show encouraging results compared with several particle flow filters and particle flow particle filters.

We also propose the particle flow Gaussian sum particle filter (PFGSPF) constructed using a bank of particle flow Gaussian particle filters. It leverages particle flow to construct more effective proposals and Gaussian sum particle filters to approximate multi-modal high-dimensional distributions. We apply the filter in complex high-dimensional numerical simulations to demonstrate the effectiveness of the proposed method. In particular, we observe that, for fixed number of particles per Gaussian, an increase in the number of Gaussian components results in improved performance.

Future work may include incorporating stochastic particle flows in the Gaussian sum particle filter framework.

Related Publications

[1] K. Comandur, Y. Li, and S. Nannuru, "Particle flow Gaussian particle filter," in 25th Int. Conf. Inf. Fusion (FUSION), Linkoping, Sweden, Jul 2022, pp. 1–6

[2] K. Comandur, Y. Li and S. Nannuru, "Particle Flow Gaussian Sum Particle Filter," in 2023 IEEE Int. Conf. Acoust. Speech and Signal Process. (ICASSP), Rhodes Island, Greece, June 2023.

Bibliography

[1] F. Daum, J. Huang, and A. Noushin, "Exact particle flow for nonlinear filters," in *Proc. SPIE Signal Process. Sensor Fusion Target Recognit.*, Orlando, FL, USA, Apr 2010, p. 769704.

[2] F. Daum and J. Huang, "Exact particle flow for nonlinear filters: seventeen dubious solutions to a first order linear underdetermined PDE," in *Proc. Asilomar Conf. Signals, Syst. Comput*, Nov 2010, pp. 64–71.

[3] T. Ding and M. J. Coates, "Implementation of the Daum-Huang exact-flow particle filter," in *Proc. IEEE Statist. Signal Process. Workshop*, Ann Arbor, MI, USA, Aug 2012, pp. 257–260.

[4] Y. Li and M. J. Coates, "Particle filtering with invertible particle flow," *IEEE Trans. Signal Process.*, vol. 65, no. 15, pp. 4102–4116, Aug 2017.

[5] G. Welch, G. Bishop, et al., "An introduction to the kalman filter," 1995.

[6] N. Gordon, D. Salmond, and A. Smith, "Novel approach to nonlinear/non-Gaussian Bayesian state estimation," in *IEE Proc. F Radar Signal Process.*, Apr 1993, vol. 140, pp. 107–113.

[7] F. Daum and J. Huang, "Nonlinear filters with log-homotopy," in *Proc. SPIE Signal Data Proc. Small Targets*, San Diego, CA, Sep. 2007, pp. 423–437.

[8] E. Daum and J. Huang, "Particle flow for nonlinear filters with log-homotopy," in *Proc. SPIE Signal Data Process. Small Targets*, Orlando, FL, USA, Apr 2008, p. 696918.

[9] F. Daum, J. Huang, A. Noushin, and M. Krichman, "Gradient estimation for particle flow induced by log-homotopy for nonlinear filters," in *Proc. SPIE Signal Process. Sensor Fusion Target Recognit.*, Orlando, FL, USA, Apr 2009, pp. 65–75.

[10] F. Daum and J. Huang, "Particle flow with non-zero diffusion for nonlinear filters," in *Proc. SPIE Signal Process. Sensor Fusion Target Recognit.*, Baltimore, MD, USA, May 2013, p. 87450P.

[11] F. Daum and J. Huang, "Seven dubious methods to mitigate stiffness in particle flow with non-zero diffusion for nonlinear filters, Bayesian decisions, and transport," in *Proc. SPIE Signal Data Process. Small Targets*, Baltimore, MD, USA, May 2014, pp. 72–82.

[12] Y. Li, L. Zhao, and M. J. Coates, "Particle flow auxiliary particle filter," in *Proc. Comput. Adv. Multi-Sensor Adapt. Process.*, Dec 2015, pp. 157–160.

[13] Y. Li, L. Zhao, and M. J. Coates, "Particle flow auxiliary particle filter," in *Proc. Comput. Adv. Multi-Sensor Adapt. Process.*, Dec 2015, pp. 157–160.

[14] J. Kotecha and P. Djuric, "Gaussian particle filtering," *IEEE Trans. Signal Process.*, vol. 51, pp. 2592 – 2601, Nov 2003.

[15] J.H. Kotecha and P.M. Djuric, "Gaussian sum particle filtering for dynamic state space models," in *Proc. Int. Conf. Acoust. Speech Signal Process.*, Salt Lake City, UT, USA, May 2001, vol. 6, pp. 3465–3468 vol.6.

[16] K. Comandur, Y. Li, and S. Nannuru, "Particle flow Gaussian particle filter," in *25th Int. Conf. Inf. Fusion (FUSION)*, Linköping, Sweden, Jul 2022, pp. 1–6.

[17] Z. Chen et al., "Bayesian filtering: From kalman filters to particle filters, and beyond," *Statistics*, vol. 182, no. 1, pp. 1–69, 2003.

[18] N. Bergman, *Recursive Bayesian estimation: Navigation and tracking applications*, Ph.D. thesis, Linköping University, 1999.

[19] A Doucet, "On sequential monte-carlo methods for bayesian filtering," Tech. Rep., Dept. Eng., Univ. Cambridge, Cambridge, UK, 1999.

[20] T. Bengtsson, P. Bickel, and B. Li, "Curse-of-dimensionality revisited: Collapse of the particle filter in very large scale systems," in *Probability and statistics: Essays in honor of David A. Freedman*, pp. 316–334. Institute of Mathematical Statist., Beachwood, OH, USA, 2008.

[21] P. Bickel, B. Li, and T. Bengtsson, "Sharp failure rates for the bootstrap particle filter in high dimensions," in *Pushing the limits of contemporary statistics: Contributions in honor of Jayanta K. Ghosh*, pp. 318–329. Institute of Mathematical Statist., Beachwood, OH, USA, 2008.

[22] C. Snyder, T. Bengtsson, P. Bickel, and J. Anderson, "Obstacles to high-dimensional particle filtering," *Mon. Weather Rev.*, vol. 136, no. 12, pp. 4629–4640, Dec 2008.

[23] J. S. Liu and R. Chen, "Sequential monte carlo methods for dynamic systems," *J. Amer. Statist. Assoc.*, vol. 93, no. 443, pp. 1032–1044, 1998.

[24] M.S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," *IEEE Trans. Signal Process.*, vol. 50, no. 2, pp. 174–188, Feb 2002.

[25] M. Pitt and N. Shephard, "Filtering via simulation: Auxiliary particle filters," *J Ameri. Statist. Assoc.*, vol. 94, no. 446, pp. 590–599, Jun 1999.

[26] C. Musso, N. Oudjane, and F. Le Gland, "Improving regularised particle filters," in *Sequential Monte Carlo methods in practice*, pp. 247–271. New York, NY, USA, 2001.

[27] A. Doucet, N. d. Freitas, K. P. Murphy, and S. J. Russell, "Rao-Blackwellised particle filtering for dynamic Bayesian networks," in *Proc. Conf. Uncertainty Artif. Intell.*, San Francisco, CA, USA, 2000, p. 176.

[28] R. Van Der Merwe, A. Doucet, N. De Freitas, and E. Wan, "The unscented particle filter," *Adv. Neural Inf. Process. Syst.*, vol. 13, 2000.

[29] C. Berzuini, N. G. Best, W. R. Gilks, and C. Larizza, "Dynamic conditional independence models and Markov chain Monte Carlo methods," *J. Amer. Statist. Assoc.*, vol. 92, no. 440, pp. 1403–1412, 1997.

[30] W. R. Gilks and C. Berzuini, "Following a moving target — Monte Carlo inference for dynamic Bayesian models," *J. Roy. Statist. Soc. B,*, vol. 63, no. 1, pp. 127–146, Jan 2001.

[31] S. Godsill and T. Clapp, "Improvement strategies for Monte Carlo particle filters," in *Sequential Monte Carlo methods in practice*, pp. 139–158. New York, NY, USA, 2001.

[32] A. Brockwell, P. D. Moral, and A. Doucet, "Sequentially interacting Markov chain Monte Carlo methods," *Ann. Statist.*, vol. 38, no. 6, pp. 3387–3411, Dec 2010.

[33] F. Septier and G. W. Peters, "Langevin and Hamiltonian based sequential MCMC for efficient Bayesian filtering in high-dimensional spaces," *IEEE J. Sel. Topics Signal Process.*, vol. 10, no. 2, pp. 312–327, March 2016.

[34] V. Maroulas and P. Stinis, "Improved particle filters for multi-target tracking," *J. Comput. Phys.*, vol. 231, no. 2, pp. 602–611, Jan 2012.

[35] K. Kang, V. Maroulas, and I. D. Schizas, "Drift homotopy particle filter for non-Gaussian multi-target tracking," in *Proc. Int. Conf. Inf. Fusion*, Salamanca, Spain, 2014, pp. 1–7.

[36] A. Golightly and D. Wilkinson, "Bayesian sequential inference for nonlinear multivariate diffusions," *Statist. Comput.*, vol. 16, no. 4, pp. 323–338, Dec 2006.

[37] P. Bunch and S. Godsill, "Approximations of the optimal importance density using gaussian particle flow importance sampling," *J. Amer. Statist. Assoc.*, vol. 111, no. 514, pp. 748–762, 2016.

[38] F. E. De Melo, S. Maskell, M. Fasiolo, and F. Daum, "Stochastic particle flow for nonlinear high-dimensional filtering problems," *arXiv preprint arXiv:1511.01448*, 2015.

[39] J. Heng, A. Doucet, and Y. Pokern, "Gibbs flow for approximate transport with applications to bayesian computation," *arXiv preprint arXiv:1509.08787*, 2015.

[40] S Reich, "A guided sequential monte carlo method for the assimilation of data into stochastic dynamical systems," in *Recent Trends in Dynamical Systems: Proceedings of a Conference in Honor of Jürgen Scheurle*. Springer, 2013, pp. 205–220.

[41] Y. Li, L. Zhao, and M. J. Coates, "Particle flow for particle filtering," in *Proc. Int. Conf. Acoust. Speech Signal Process.*, 2016, pp. 3979–3983.

[42] F. Daum and J. Huang, "Exact particle flow for nonlinear filters: seventeen dubious solutions to a first order linear underdetermined PDE," in *Proc. Asilomar Conf. Signals, Syst. Comput*, Nov 2010, pp. 64–71.

[43] Y. Li, S. Pal, and M. J. Coates, "Invertible particle-flow-based sequential MCMC with extension to Gaussian mixture noise models," *IEEE Trans. Signal Process.*, vol. 67, no. 9, pp. 2499–2512, May 2019.

[44] J. H. Lee and N. L. Ricker, "Extended Kalman filter based nonlinear model predictive control," *Industrial & Engineering Chemistry Research*, vol. 33, no. 6, pp. 1530–1541, Jun 1994.

[45] B. D. Anderson and J. B Moore, *Optimal filtering*, Courier Corporation, 2012.

[46] M. S. Grewal, A. P. Andrews, and C. G. Bartone, "Kalman filtering," 2001.

[47] A. H. Jazwinski, *Stochastic processes and filtering theory*, Courier Corporation, 2007.

[48] H.W. Sorenson, "Recursive estimation for nonlinear dynamic systems," *Bayesian Anal. Time Ser. Dyn. Models*, vol. 94, pp. 127–165, 1988.

[49] S. Frühwirth-Schnatter, "Applied state space modelling of non-gaussian time series using integration-based kalman filtering," *Statist. and Comput.*, vol. 4, pp. 259–269, 1994.

[50] C. Masreliez, "Approximate non-gaussian filtering with linear state and observation relations," *IEEE. Trans. Autom. Control*, vol. 20, no. 1, pp. 107–110, 1975.

[51] M. West, P. J. Harrison, and H. S. Migon, "Dynamic generalized linear models and bayesian forecasting," *J. Amer. Statist. Assoc.*, vol. 80, no. 389, pp. 73–83, 1985.

[52] S. J. Julier, J. K. Uhlmann, and H. F. Durrant-Whyte, "A new approach for filtering nonlinear systems," in *Proc. 1995 Amer. Control Conf. ACC'95*. IEEE, 1995, vol. 3, pp. 1628–1632.

[53] K. Ito and K. Xiong, "Gaussian filters for nonlinear filtering problems," *IEEE. Trans. Autom. Control*, vol. 45, no. 5, pp. 910–927, 2000.

[54] O. Hlinka, O. Slučiak, F. Hlawatsch, P. M. Djurić, and M. Rupp, "Distributed Gaussian particle filtering using likelihood consensus," in *Proc. Int. Conf. Acoust. Speech Signal Process.*, Prague, Czech Republic, May 2011, pp. 3756–3759.

[55] D. Schuhmacher, B. T. Vo, and B. N. Vo, "A consistent metric for performance evaluation of multi-object filters," *IEEE Trans. Signal Process.*, vol. 56, no. 8, pp. 3447–3457, Aug 2008.

[56] D. Zhu and J. W. Galbraith, "A generalized asymmetric Student-t distribution with application to financial econometrics," *Journal of Econometrics*, vol. 157, no. 2, pp. 297–305, Aug 2010.

[57] D. Alspach and H. Sorenson, "Nonlinear bayesian estimation using gaussian sum approximations," *IEEE Trans. Autom. Control*, vol. 17, no. 4, pp. 439–448, 1972.

[58] R. A. Redner and H. F. Walker, "Mixture densities, maximum likelihood and the em algorithm," *SIAM review*, vol. 26, no. 2, pp. 195–239, 1984.

[59] D. Sengupta and S. Kay, "Efficient estimation of parameters for non-gaussian autoregressive processes," *IEEE Trans. Acoust., Speech and Signal Process.*, vol. 37, no. 6, pp. 785–794, 1989.

[60] R. S. Blum, R. J. Kozick, and B. M. Sadler, "An adaptive spatial diversity receiver for non-gaussian interference and noise," *IEEE Trans. on Signal Process.*, vol. 47, no. 8, pp. 2100–2111, 1999.

[61] M. A. Khan, M. Ulmke, and W. Koch, "A log homotopy based particle flow solution for mixture of gaussian prior densities," in *2016 IEEE Int. Conf. Multisensor Fusion and Integr. Intell. Syst.*, Kongresshaus Baden-Baden, Germany, Sept 2016, IEEE, pp. 546–551.

[62] S. Pal and M. J. Coates, "Gaussian sum particle flow filter," in *7th Int. Workshop on Comput. Adv. in Multi-Sensor Adaptive Process. (CAMSAP)*, CW, AN, Dec 2017, pp. 1–5.

[63] S. Pal and M. J. Coates, "Particle flow particle filter for Gaussian mixture noise models," in *Proc. Int. Conf. Acoust. Speech and Signal Process.*, Calgary, AB, Canada, Apr 2018, pp. 4249–4253.