PARTICLE FLOW PARTICLE FILTER FOR GAUSSIAN MIXTURE NOISE MODELS

Soumyasundar Pal and Mark Coates
Dept. of Electrical and Computer Engineering, McGill University, Montreal, Quebec, Canada

ABSTRACT

Particle filters have become a standard tool for state estimation in nonlinear systems. However, their performance usually deteriorates if the dimension of state space is high or the measurements are highly informative. A major challenge is to construct a proposal density that is well matched to the posterior distribution. Particle flow methods are a promising option for addressing this task. In this paper, we develop a particle flow particle filter algorithm to address the case where both the process noise and the measurement noise are distributed as mixtures of Gaussians. Numerical experiments are performed to explore when the proposed method offers advantages compared to existing techniques.

Index Terms— non-linear sequential state estimation, particle flow, Daum-Huang filter, particle filter, Gaussian mixture model, high-dimensional filtering.

I. INTRODUCTION

Particle filters [1] perform sequential importance sampling to solve the discrete-time nonlinear filtering task in a Bayesian framework. However, their performance can be poor if the measurements are highly informative or the state dimension is high [2], [3]. Although there have been multiple proposals to address this issue [2]–[7], the methods rely on proposals to address this issue [2]–[7], the methods rely on

dynamic and measurement models, this approximation is the prior and posterior for construction of the flow. For many

ally expensive.

proposals to address this issue [2]–[7], the methods rely on

dimension is high [2], [3]. Although there have been multiple

poor if the measurements are highly informative or the state

deteriorates if the dimension of state space is high or the

measurements are highly informative. A major challenge is to construct a proposal density that is well matched to the posterior distribution. Particle flow methods are a promising option for addressing this task. In this paper, we develop a particle flow particle filter algorithm to address the case where both the process noise and the measurement noise are distributed as mixtures of Gaussians. Numerical experiments are performed to explore when the proposed method offers advantages compared to existing techniques.

Particle flow filters [8]–[12] exhibit improved performance. Instead of sampling, particles are migrated from the prior to the posterior by identifying and solving differential equations that link these two distributions. The approximations needed to implement these filters can lead to particles not being a genuine sample from the posterior. To address this, several recent algorithms combine particle flow and particle filtering [12]–[16]. In particular, [16] introduces the particle flow particle filter (PFPPF), which uses particle flow to construct the proposal distribution in a particle filtering framework. The theoretical guarantees for any particle filter still apply, while the flow ensures that a good proposal has been constructed.

The PFPPF of [16] relies on a Gaussian approximation to the prior and posterior for construction of the flow. For many dynamic and measurement models, this approximation is poor. Two efforts have been made to expand the applicability of the particle flow filters to mixture models [17], [18], but neither of these approaches leads to a particle filter. The potentially mismatched modelling assumptions and approximations needed to construct and compute the particle flow can negatively impact performance. In this paper we develop a particle flow particle filter that addresses settings where the dynamic and measurement models can be approximated by a nonlinearity with additive noise distributed according to a Gaussian mixture.

The paper is organized as follows. Section II states the filtering task that we address. Section III reviews the exact particle flow filter [10], [19] and Section IV reviews the particle flow particle filters of [16] . Section V introduces the proposed particle flow particle filter for Gaussian mixture models and Section VI presents and discusses the results of numerical simulation experiments. The conclusion is provided in Section VII.

II. PROBLEM STATEMENT

We consider the discrete-time filtering problem where our goal is to track the marginal posterior distribution \( p(x_k|z_{1:k}) \) recursively with time \( k \), starting from an initial probability density function \( p(x_0) \). The dynamic and measurement models are specified as:

\[
\begin{align*}
  x_k &= g_k(x_{k-1}) + v_k \quad \text{for } k \geq 1, \\
  z_k &= h_k(x_k) + w_k \quad \text{for } k \geq 1.
\end{align*}
\]

Here \( g_k : \mathbb{R}^d \to \mathbb{R}^d \) is the state-transition function of the unobserved state \( x_k \in \mathbb{R}^d \), the measurement \( z_k \in \mathbb{R}^S \) is generated conditioned on the current state \( x_k \) through a potentially nonlinear measurement model \( h_k : \mathbb{R}^d \to \mathbb{R}^S \). The process and measurement noises are denoted by \( v_k \in \mathbb{R}^d \) and \( w_k \in \mathbb{R}^S \) respectively. We assume that \( p(x_0) = \mathcal{N}(x_0|\mu_0, P_0) \) is Gaussian whereas \( v_k \sim \sum_{m=1}^M \alpha_{k,m} \mathcal{N}(v_k|m, Q_{k,m}) \) and \( w_k \sim \sum_{n=1}^N \beta_{k,n} \mathcal{N}(w_k|\zeta_{k,n}, R_{k,n}) \) are distributed according to Gaussian mixtures.

We observe that this model can be alternatively expressed as a switching state space model. We introduce two unobserved scalar valued discrete random variables \( d_k \in \{1, 2, ..., M\} \) and \( c_k \in \{1, 2, ..., N\} \) such that \( P(d_k = m) = \alpha_{k,m} \) and \( P(c_k = n) = \beta_{k,n} \). The \( d_k \) and \( c_k \) variables are independent for different \( k \) and independent of each other. Let \( p(x_k|x_{k-1}, d_k = m) = \mathcal{N}(x_k|g_k(x_{k-1}) + \psi_{k,m}, Q_{k,m}), \forall 1 \leq m \leq M \) and \( p(z_k|x_k, c_k = n) = \mathcal{N}(z_k|h_k(x_k) + \eta_{k,n}, R_{k,n}) \).
\[ N(z_k|h_k(x_k) + \zeta_{k,n}, R_{k,n}), \forall 1 \leq n \leq N. \] The state transition density is then:

\[ p(x_k|x_{k-1}) = \sum_{m=1}^{M} \alpha_{k,m} \mathcal{N}(x_k|g_k(x_{k-1}) + \psi_{k,m}, Q_{k,m}) \]

(3)

Similarly, the likelihood is

\[ p(z_k|x_k) = \sum_{n=1}^{N} \beta_{k,n} \mathcal{N}(z_k|h_k(x_k) + \zeta_{k,n}, R_{k,n}) \]

(4)

III. EXACT PARTICLE FLOW (SINGLE GAUSSIAN)

Suppose the posterior at time \( k-1 \) is approximated by a set of \( N_p \) unweighted particles \( \{\tilde{x}_k^i\}_{i=1}^{N_p} \). Propagating the particles through the dynamic model yields \( \{\tilde{x}_k^i\}_{i=1}^{N_p} \), which represent the predictive posterior at time \( k \). The objective of particle flow methods is then to gradually migrate the particles towards the correct regions of state-space so that they approximate the posterior at time \( k \), when the flow is complete. Particle flow can be modelled as a background stochastic process \( \eta_k \) in a pseudo time interval \( \lambda \in [0, 1] \). The time index \( k \) is temporarily omitted to simplify notation, because the particle flow only concerns migration of particles within a single time step. \( \eta_k^i \) denotes the stochastic process’s \( i \)-th realization, and we initialize the flow by setting \( \eta_0^i = \tilde{x}_k^i \), for \( i = 1, 2, \ldots, N_p \).

The zero diffusion particle flow filters [9], [10] involve deterministic flows of particles. The trajectory of \( \eta_k^i \) for realization \( i \) follows the ordinary differential equation (ODE):

\[ \frac{d\eta_k^i}{d\lambda} = f(\eta_k^i, \lambda), \]

(5)

where \( f : \mathbb{R}^d \to \mathbb{R}^d \) is governed by the Fokker-Planck equation and additional flow constraints [10].

An analytically tractable solution of equation (5) exists when the predictive posterior and the likelihood distributions are both Gaussian and the measurement model is linear, i.e., \( \eta_k^i \sim \mathcal{N}(\bar{\eta}_0^i, P), z = H\eta_k^i + w \sim \mathcal{N}(H\eta_k^i, R) \).

III-B. Numerical Implementation

The approximate solution of the ODE (5) is obtained numerically using Euler’s method. We discretize the pseudo-time \( \lambda \) at \( N_\lambda \) positions \( \lambda_1, \lambda_2, \ldots, \lambda_{N_\lambda} \), where \( 0 = \lambda_0 < \lambda_1 < \ldots < \lambda_{N_\lambda} = 1 \). The step sizes \( \epsilon_j = \lambda_j - \lambda_{j-1} \) for \( j = 1, \ldots, N_\lambda \) satisfy \( \sum_{j=1}^{N_\lambda} \epsilon_j = \lambda_{N_\lambda} - \lambda_0 = 1 \). In the LEDH filter, we need to linearize \( H^i(\lambda_j) \) to compute \( A^i(\lambda_j) \) and this is performed at \( \eta_{j-1}^i \). The Euler update rule is

\[ \eta_{j}^i = \eta_{j-1}^i + \epsilon_j (A^i(\lambda_j)\eta_{j-1}^i + b^i(\lambda_j)) \]

IV. PARTICLE FLOW PARTICLE FILTER

Due to approximations in the flow implementation and mismatched modelling assumptions, the migrated particles after the particle flow process are not exactly distributed according to the posterior. A particle filter requires us to generate samples of the state from an importance distribution \( q \) and then calculate importance weights so that we can maintain a weighted particle approximation to the posterior as time evolves. \( \eta_0^i \) can be viewed as being drawn from a proposal distribution \( q(\eta_0^i|x_{k-1}^i, z_{k}) \), which is possibly well matched to the posterior, because of the flow procedure. If the flow parameters, \( A^i(\lambda_p), b^i(\lambda_p) \) are computed based on linearization of the measurement function \( h \) at an auxiliary particle location \( \bar{\eta}_{0,p}^i \), starting from \( \bar{\eta}_{0,p}^i = g_k(x_{k-1}^i) \), then under a mild smoothness condition on the measurement function \( h \) and with small enough step sizes \( \epsilon_j \), the discretized particle flow process introduces an invertible mapping \( \eta_0^i = T(\eta_0^i; x_{k-1}^i, z_{k}) \), as shown in [16]. This property enables efficient evaluation of the importance density:

\[ q(\eta_0^i|x_{k-1}^i, z_{k}) = \frac{p(\eta_0^i|x_{k-1}^i) - \det(T(\eta_0^i; x_{k-1}^i, z_{k}))}{|\det(T(\eta_0^i; x_{k-1}^i, z_{k}))|}, \]

(8)

where \( T(\cdot) \in \mathbb{R}^{d \times d} \) is the Jacobian function of the mapping \( T(\cdot) \) and its determinant can be computed as

\[ \det(T(\eta_0^i; x_{k-1}^i, z_{k})) = \prod_{p=1}^{N_\lambda} \det(I + \epsilon_p A^i(\lambda_p)) \]

(9)

V. PARTICLE FLOW PARTICLE FILTER FOR GAUSSIAN MIXTURE MODELS

In this section we develop the novel particle flow particle filter. We consider the switching state representation of the Gaussian mixture model (1)-(2). We augment \( x_k \) with the unobserved discrete variables \( d_k \) and \( c_k \) and consider the target joint density to be \( p(x_{0:k}, d_{1:k}, c_{1:k}|z_{1:k}) \). We require that the importance distribution \( q \) factorizes:

\[ q(x_{0:k}, d_{1:k}, c_{1:k}, z_{1:k}) = q(x_{0:k-1}, \{d_{1:k-1}, c_{1:k-1}\}, z_{1:k-1}) \]

\[ q(x_k, d_k, c_k|x_{0:k-1}, d_{1:k-1}, c_{1:k-1}, z_{1:k-1}) \]

(10)
Samples \(\{x^i_{0:k}, d^i_{1:k}, c^i_{1:k}\}_{i=1}^{N_p}\) are obtained by augmenting each existing sample, \(\{x^i_{0:k-1}, d^i_{1:k-1}, c^i_{1:k-1}\}\sim q(x_{0:k-1}, d_{1:k-1}, c_{1:k-1}|z_{1:k-1})\) with the new state
\(\{x^i_k, d^i_k, c^i_k\}\sim q(x_k, d_k, c_k|x^i_{0:k-1}, d^i_{1:k-1}, c^i_{1:k-1}, z_{1:k})\).

The target joint density can be expressed as follows:
\[
p(x_0:k, d_1:k, c_1:k|z_{1:k}) \propto p(x_k, d_k, c_k|x_0:k-1, d_{1:k-1}, c_{1:k-1})
\]
\[
p(z_k|x_k, d_k, c_k)p(x_0:k-1, d_1:k-1, c_1:k-1|z_{1:k-1})
\]

We can calculate unnormalized importance weights as:
\[
\omega_k^i = \frac{p(x^i_k, d^i_k, c^i_k|x^i_{0:k-1}, d^i_{1:k-1}, c^i_{1:k-1})p(z_k|x^i_k, d^i_k, c^i_k)}{q(x^i_k, d^i_k, c^i_k|x^i_{0:k-1}, d^i_{1:k-1}, c^i_{1:k-1}, z_{1:k})},
\]
\[
\omega_{i-1}^i \propto \omega_k^i \prod_{i=1}^{N_p} \omega_k^i q(x^i_k, d^i_k, c^i_k|x^i_{0:k-1}, d^i_{1:k-1}, c^i_{1:k-1}, z_k)\]
\[
\propto \omega_k^i \sum_{i=1}^{N_p} \omega_k^i q(x^i_k, d^i_k, c^i_k|x^i_{0:k-1}, d^i_{1:k-1}, c^i_{1:k-1}, z_k)
\]
(11)

We design a proposal such that \(q(x_k, d_k, c_k|x_0:k-1, d_{1:k-1}, c_{1:k-1}, z_{1:k}) = q(x_k, d_k, c_k|x_0:k-1, d_{1:k-1}, c_{1:k-1}, z_k)\) is satisfied, so that we only need to store the particles and weights from the previous time step, instead of the full trajectories. For the filtering problem, our focus is the marginal posterior, and we approximate this as
\[
p(x_k|z_{1:k}) \approx \sum_{i=1}^{N_p} \omega_k^i \delta(x_k - x^i_k)
\]
(12)

where \(\delta\) is the Dirac delta-function. From the dynamic model,
\[
p(x^i_k|d^i_k, c^i_k|x^i_{0:k-1}, d^i_{1:k-1}, c^i_{1:k-1}) = P(d^i_k)p(c^i_k)p(x^i_k|x_{0:k-1}, d^i_k, c^i_k).
\]

We design \(q\) such that
\[
q(x^i_k|d^i_k, c^i_k|x^i_{0:k-1}, d^i_{1:k-1}, c^i_{1:k-1}, z_k) = Q(d^i_k)Q(c^i_k)q(x^i_k|x_{0:k-1}, d^i_k, c^i_k, z_k)
\]
(13)
is satisfied. We choose \(Q(d_k) = P(d_k), Q(c_k) = P(c_k)\) and we calculate \(q(x^i_k|x^i_{0:k-1}, d^i_k, c^i_k, z_k)\) based on the invertible mapping established by the flow [16].

\[
q(x^i_k|x^i_{0:k-1}, d^i_k) = m, c^i_k = n, z_k = \frac{p(\eta_0^i|x_{0:k-1}, d^i_k = m)}{|\prod_{p=1}^{N_p} \det(I + \epsilon_p A_n^i(\lambda_p))|}
\]

Then equation (11) can be rewritten as
\[
\omega_k^i \propto \omega_{i-1}^i \frac{p(x^i_k|x^i_{0:k-1}, d^i_k)p(z_k|x^i_k, c^i_k)}{q(x^i_k|x^i_{0:k-1}, d^i_k, c^i_k, z_k)},
\]
(14)

The algorithm is summarized in Algorithm 1. The most computationally demanding parts of the algorithm are the matrix inverse operations needed to calculate \(A_{mn}(\lambda_p)\) and \(b_{mn}(\lambda_p)\). Since individual flow parameters are calculated for each of the \(N_p\) particles at each time step and there are a total of \(N_A\) discrete pseudo time steps, the total computational complexity of the matrix inverse operations is \(O(N_pN_A S^3)\), where \(S\) is the measurement dimension.

Algorithm 1 PFPP-GMM

1. Initialization: Draw \(\{x^i_0\}_{i=1}^{N_p}\) from the prior \(p(x_0)\). Set \(\dot{x}_0\) to be the mean of \(p(x_0)\). Set \(\{\omega^i_0\}_{i=1}^{N_p} = \frac{1}{N_p}\) and \(\{P^i_{0|0}\}_{i=1}^{N_p} = \tilde{P}_0\). Set \(\lambda_0 = 0\).

2. for \(k = 1\) to \(K\) do

3. for \(i = 1, \ldots, N_p\) do

4. Sample \(d^i_k = m \in \{1, 2, \ldots, M\}\) with probability \(\{\alpha^i_{0:k}, \alpha^i_{1:k}, \ldots, \alpha^i_{K-1}\}\)

5. Apply EKF/UKF prediction, \(\{x^i_{k-1}, P^i_{k-1}\} \rightarrow \{m^i_{k|k-1}, m^i_{k|m}, P^i_{k|m}\}\), using \(N(\psi_k, m, Q_k, m)\)

6. Calculate \(\eta_0^i = g_k(x^i_{k-1}) + v_k, m\), where \(v_k, m \sim N(\psi_k, m, Q_k, m)\)

7. Propagate particle \(\hat{\eta}_0^i = g_k(x^i_{k-1}) + v_k, m\), where \(v_k, m \sim N(\psi_k, m, Q_k, m)\)

8. Sample \(c^i_k = n \in \{1, 2, \ldots, N\}\) with probability \(\{\beta_{k,1}, \beta_{k,2}, \ldots, \beta_{k,N}\}\)

9. Set \(\theta^i_{mn} = 1\)

10. for \(p = 1, \ldots, N_m\) do

11. Set \(\lambda_{pm} = \lambda_{p-1}^i + \epsilon_p\)

12. Calculate \(A^i_{mn}(\lambda_p)\) and \(b_{mn}(\lambda_p)\) from (7) with linearization performed at \(\hat{\eta}_{pm}^i\), and with \(z = z_k - \zeta_{kn}, \lambda = R = \eta_0^i, \tilde{\eta}_0^i = \tilde{\eta}_0^i\) and \(P = P^i_{k|m}\)

13. Migrate auxiliary particle: \(\tilde{\eta}_{pm}^i = \tilde{\eta}_{pm}^i + \epsilon_p(A^i_{mn}(\lambda_p))\hat{\eta}_{pm}^i + b_{mn}(\lambda_p)\)

14. Migrate particle: \(\eta_{pm}^i = \eta_{pm}^i + \epsilon_p(A^i_{mn}(\lambda_p))\hat{\eta}_{pm}^i + b_{mn}(\lambda_p)\)

15. \(\theta^i_{mn} = \theta^i_{mn}|\det(I + \epsilon_p A^i_{mn}(\lambda_p))|\)

end for

16. Set \(\dot{x}_k^i = \dot{x}_k^i\)

17. Calculate importance weights:
\[
\omega_k^i \propto \omega_{i-1}^i \frac{p(x^i_k|x^i_{0:k-1}, d^i_k = m)p(z_k|x^i_k, c^i_k = n)}{p(\eta_0^i|x^i_{0:k-1}, d^i_k = m)/\theta^i_{mn}}
\]
(19)

end for

20. for \(i = 1, \ldots, N_p\) do

21. Normalize \(\omega_k^i = \omega_k^i / \sum_{i=1}^{N_p} \omega_k^i\)

22. Apply EKF/UKF update, \(\{m^i_{k|k, m}, P^i_{k|m}\} \rightarrow \{m^i_{k|k}, P^i_{k|m}\}\) using \(N(\zeta_{kn}, R_k, n)\)

23. Estimate \(\hat{x}_k = \sum_{i=1}^{N_p} \omega_k^i x_k^i\)

24. (Optional) resample particles \(\{x^i_k, P^i_k, \omega^i_k\}_{i=1}^{N_p}\) to obtain \(\{x^i_k, P^i_k, \frac{1}{N_p}\}_{i=1}^{N_p}\)

25. end for

end for

VI. NUMERICAL EXPERIMENTS AND RESULTS

We conduct numerical simulations for two scenarios. The first is a linear scenario, which allows us to compare the performance of the proposed filter with an (almost) optimal solution in the form of the Gaussian mixture model Kalman filter. The second nonlinear scenario requires a particle filter for accurate state estimates. We compare with an extended
Kalman filter derived for Gaussian mixture models (EKF-GMM), the particle flow filter for Gaussian mixtures (PF-GMM) [18], the Gaussian Sum Particle Filter (GSPP) [20], an unscented Kalman filter (UKF), the exact Daum Huang (EDH) filter [10] and its localized version (LEDH) [19], the Particle Flow Particle Filters (PFPFs) based on EDH and LEDH [16], and a bootstrap particle filter using 1 million particles. All numerical simulations are executed using an Intel i7-4770K, 3.50GHz CPU and 32GB RAM.

VI-A. Linear Model

We first consider a linear dynamic and measurement model \( x_k = \alpha x_{k-1} + v_k \) and \( z_k = x_k + w_k \), where \( x_k \in \mathbb{R}^d \) and \( z_k \in \mathbb{R}^d \). We set \( d = 64 \) and \( \alpha = 0.9 \). The noise terms are drawn from GMMs: \( v_k \sim \sum_{m=1}^3 \frac{1}{3} N(\mu_m 1_{d \times 1}, \sigma^2_w I_{d \times d}) \), where \( \mu_1 = -1, \mu_2 = 0, \mu_3 = 1 \) and \( \sigma_w = 1, \). \( w_k \sim \sum_{n=1}^3 \frac{1}{3} N(\delta_n 1_{d \times 1}, \sigma^2_w I_{d \times d}) \), where \( \delta_1 = -5, \delta_2 = 0, \delta_3 = 5 \) and \( \sigma_w = 0.1 \). The true state starts with \( x_0 = 0 \). The experiment is executed 200 times for 50 time steps.

Table I summarizes the results, reporting the mean-squared error (MSE) in the state estimation. The proposed particle flow particle filter for Gaussian mixture models (PFPF-GMM) performs only slightly worse than the EKF-GMM algorithm, which is close to optimal for this case. The PF-GMM also achieves close to the (almost) optimal performance. The PFPF-GMM outperforms the other particle-based methods significantly because they either use inaccurate Gaussian approximations or are ill-suited to the high-dimensional problem (e.g. GSPF, BPF).

VI-B. Nonlinear Model

We now consider a nonlinear dynamical model \( g_k : \mathbb{R}^d \rightarrow \mathbb{R}^d \) and measurement function \( h_k : \mathbb{R}^d \rightarrow \mathbb{R}^d \). The \( c \)-th element of the measurement vector is \( h_k^c(x_k) = \frac{(x_k^c)^2}{20} \). Each element \( c \) of the state vector is defined as follows:

\[
\begin{align*}
g_k^c(x_{k-1}) &= 0.5x_{k-1}^c + 8 \cos(1.2(k - 1)) \\
&+ \begin{cases} 
2.5 \frac{x_{k-1}^c}{1 + (\sigma_w^2)^{-1}} , & \text{if } c = 1 \\
2.5 \frac{x_{k-1}^c}{1 + (\sigma_w^2)^{-1}} , & \text{if } 1 < c < d \\
2.5 \frac{x_{k-1}^c}{1 + (\sigma_w^2)^{-1}} , & \text{if } c = d
\end{cases}
\end{align*}
\]

We set \( d = 64. v_k \sim \sum_{m=1}^3 \frac{1}{3} N(\mu_m 1_{d \times 1}, \sigma^2_v I_{d \times d}) \), where \( \mu_1 = -1, \mu_2 = 0, \mu_3 = 1 \) and \( \sigma_v = 0.5, \). \( w_k \sim \sum_{n=1}^3 \frac{1}{3} N(\delta_n 1_{d \times 1}, \sigma^2_w I_{d \times d}) \), where \( \delta_1 = -3, \delta_2 = 0, \delta_3 = 3 \) and \( \sigma_w = 0.1 \), are drawn from a GMM. The true state starts at \( x_0 = 0 \). The experiment is executed 200 times for 50 time steps. Table II summarizes the results, reporting the mean-squared error (MSE) in the state estimation.

For this challenging nonlinear problem, the EKF-GMM algorithm and the UKF struggle. The BPF and the GSPF suffer from significant weight degeneracy. The PF-GMM algorithm outperforms the LEDH filter as the latter is based on the incorrect approximation of a single Gaussian likelihood. The proposed PFPF-GMM has a considerably smaller MSE than all other filters.

VII. CONCLUSION

In this paper, we developed a particle flow particle filter that can address the scenario when both process and measurement noise are mixtures of Gaussians. The proposed method can be employed in challenging high-dimensional filtering problems with multi-modal posteriors. Future research will investigate ways to reduce the computational overhead and assess performance when the Gaussian mixture models only approximate the true behaviour of the system.

VIII. REFERENCES


