PATH SAMPLING FOR PARTICLE FILTERS WITH APPLICATION TO MULTI-TARGET TRACKING

By

Vasileios Maroulas

and

Panos Stinis

IMA Preprint Series # 2336

(September 2010)

INSTITUTE FOR MATHEMATICS AND ITS APPLICATIONS

UNIVERSITY OF MINNESOTA

400 Lind Hall

207 Church Street S.E.

Minneapolis, Minnesota 55455–0436

Phone: 612-624-6066 Fax: 612-626-7370

URL: http://www.ima.umn.edu

Path sampling for particle filters with application to multi-target tracking

Vasileios Maroulas
Department of Mathematics
University of Tennessee
Knoxville, TN 37996
and
Panos Stinis
Department of Mathematics
University of Minnesota
Minneapolis, MN 55455

Abstract

In recent work [15], we have presented a novel approach for improving particle filters for multi-target tracking. The suggested approach was based on Girsanov's change of measure theorem for stochastic differential equations. Girsanov's theorem was used to design a Markov Chain Monte Carlo step which is appended to the particle filter and aims to bring the particle filter samples closer to the observations. In the current work, we present an alternative way to append a Markov Chain Monte Carlo step to a particle filter to bring the particle filter samples closer to the observations. Both current and previous approaches stem from the general formulation of the filtering problem. We have used the currently proposed approach on the problem of multi-target tracking for both linear and nonlinear observation models. The numerical results show that the suggested approach can improve significantly the performance of a particle filter.

Introduction

Multi-target tracking is a central and difficult problem arising in many scientific and engineering applications including radar and signal processing, air traffic control and GPS navigation [12]. The tracking problem consists of computing the best estimate of the targets' trajectories based on noisy measurements (observations). Several strategies have been developed for addressing the multi-target tracking problem, see e.g. [1, 6, 5, 16, 8, 11, 12, 13, 14, 22].

As in our recent work [15], in this paper we also focus on particle filter techniques [5, 16]. The popularity of the particle filter method has increased due to its flexibility to handle cases where the dynamic and observation models are non-linear and/or non-Gaussian. The particle filter approach is an importance sampling method which approximates the target distribution by a discrete set of weighted samples (particles). The weights of the samples are updated when observations become available in order to incorporate information from the observations.

Despite the particle filter's flexibility, it is often found in practice that most samples will have a negligible weight with respect to the observation, in other words their corresponding contribution to the target distribution will be negligible. Therefore, one may resample the weights to create more copies of the samples with significant weights [8]. However, even with the resampling step, the particle filter might still need a lot of samples in order to approximate accurately the target distribution. Typically, a few samples dominate the weight distribution, while the rest of the samples are in statistically insignificant regions. Thus, some authors (see e.g. [7, 23]) have suggested the use of an extra step, after the resampling step, which can help move more samples in statistically significant regions.

The extra step for the particle filter is a problem of conditional path sampling for stochastic differential equations (SDEs). In [19], a new approach to conditional path sampling based on Girsanov's theorem was presented. In that paper, it was also shown how the algorithm can be used to perform the extra step of a particle filter. In [15], we applied the conditional path sampling algorithm from [19] to perform the extra step of a particle filter for the problem of multi-target tracking. The numerical results in [15] suggested that the approach can improve significantly the performance of a particle filter for multi-target tracking. In the current work, we show yet another way of how to perform the extra step of a particle filter. Both the current approach and the one in [15] stem from the general formulation of the filtering problem. The details of the currently proposed implementation of the extra step for a single target are given in Section 1.3 and for multiple targets in Section 1.4. The relative merits of the proposed approach in this paper and the one proposed in [15] are briefly discussed in Section 3. A more detailed comparison will be presented elsewhere.

To address the target-observation association problem we have used a simple Metropolis Monte Carlo algorithm which first appeared in [15]. This algorithm effects a probabilistic search of the space of possible associations to find the best target-observation association. Of course, one can use more sophisticated association algorithms (see [16] and references therein) but the Monte Carlo algorithm performed very well in the numerical experiments.

The paper is organized as follows. Sections 1.1 and 1.2 provide a brief presentation of particle filters for single and multiple targets (more details can be found in [8, 5, 10, 16]), which will serve to highlight the versatility and drawbacks of this popular filtering method. Sections 1.3 and 1.4 demonstrate how one can use an extra step to improve the performance of particle filters for single and multiple targets. Section 2 presents numerical results for multi-target tracking for the cases of linear and nonlinear observation models. Finally, Section 3 contains a discussion of the results as well as directions for future work.

1 Particle filtering

Particle filters are a special case of sequential importance sampling methods. In Sections 1.1 and 1.2 we discuss the generic particle filter for a single and multiple targets respectively. In Sections 1.3 and 1.4 we discuss the addition of an extra step to the generic particle filter for the cases of a single and multiple targets respectively.

1.1 Generic particle filter for a single target

Suppose that we are given an SDE system and that we also have access to noisy observations $Z_{T_1}, \ldots, Z_{T_K}$ of the state of the system at specified instants $T_1, \ldots, T_K$. The observations are functions of the state of the system, say given by $Z_{T_k} = G(X_{T_k}, \xi_k)$, where $\xi_k, k = 1, \ldots, K$ are mutually independent random variables. For simplicity, let us assume that the distribution of the observations admits a density $g(X_{T_k}, Z_{T_k})$, i.e., $p(Z_{T_k}|X_{T_k}) \propto g(X_{T_k}, Z_{T_k})$.

The filtering problem consists of computing estimates of the conditional expectation $E[f(X_{T_k})|\{Z_{T_j}\}_{j=1}^{k}]$, i.e., the conditional expectation of the state of the system given the (noisy) observations. Equivalently, we are looking to compute the conditional density of the state of the system given the observations $p(X_{T_k}|\{Z_{T_j}\}_{j=1}^{k})$. There are several ways to compute this conditional density and the associated conditional expectation but for practical applications they are rather expensive.

Particle filters fall in the category of importance sampling methods. Because computing averages with respect to the conditional density involves the sampling of the conditional density which can be difficult, importance sampling methods proceed by sampling a reference density $q(X_{T_k}|\{Z_{T_j}\}_{j=1}^k)$ which can be easily sampled and then compute the weighted sample mean

$$E[f(X_{T_k})|\{Z_{T_j}\}_{j=1}^k] \approx \frac{1}{N}\sum_{n=1}^{N} f(X_{T_k}^n)\frac{p(X_{T_k}^n|\{Z_{T_j}\}_{j=1}^k)}{q(X_{T_k}^n|\{Z_{T_j}\}_{j=1}^k)}$$

or the related estimate

$$E[f(X_{T_k})|\{Z_{T_j}\}_{j=1}^k] \approx \frac{\sum_{n=1}^{N} f(X_{T_k}^n)\frac{p(X_{T_k}^n|\{Z_{T_j}\}_{j=1}^k)}{q(X_{T_k}^n|\{Z_{T_j}\}_{j=1}^k)}}{\sum_{n=1}^{N}\frac{p(X_{T_k}^n|\{Z_{T_j}\}_{j=1}^k)}{q(X_{T_k}^n|\{Z_{T_j}\}_{j=1}^k)}}, \quad (1)$$

where N has been replaced by the approximation

$$N \approx \sum_{n=1}^{N}\frac{p(X_{T_k}^n|\{Z_{T_j}\}_{j=1}^k)}{q(X_{T_k}^n|\{Z_{T_j}\}_{j=1}^k)}.$$

Particle filtering is a recursive implementation of the importance sampling approach. It is based on the recursion

$$p(X_{T_k}|\{Z_{T_j}\}_{j=1}^k) \propto g(X_{T_k}, Z_{T_k})p(X_{T_k}|\{Z_{T_j}\}_{j=1}^{k-1}), \quad (2)$$

$$\text{where } p(X_{T_k}|\{Z_{T_j}\}_{j=1}^{k-1}) = \int p(X_{T_k}|X_{T_{k-1}})p(X_{T_{k-1}}|\{Z_{T_j}\}_{j=1}^{k-1})dX_{T_{k-1}}. \quad (3)$$

If we set

$$q(X_{T_k}|\{Z_{T_j}\}_{j=1}^k) = p(X_{T_k}|\{Z_{T_j}\}_{j=1}^{k-1}),$$

then from (2) we get

$$\frac{p(X_{T_k}|\{Z_{T_j}\}_{j=1}^k)}{q(X_{T_k}|\{Z_{T_j}\}_{j=1}^k)} \propto g(X_{T_k}, Z_{T_k}).$$

The approximation in expression (1) becomes

$$E[f(X_{T_i})|\{Z_{T_j}\}_{j=1}^k] \approx \frac{\sum_{n=1}^{N} f(X_{T_k}^n)g(X_{T_k}^n, Z_{T_k})}{\sum_{n=1}^{N} g(X_{T_k}^n, Z_{T_k})} \quad (4)$$

From (4) we see that if we can construct samples from the predictive distribution $p(X_{T_k}|\{Z_{T_j}\}_{j=1}^{k-1})$ then we can define the (normalized) weights $W_{T_k}^n =$

$\frac{g(X^n_{T_k}, Z_{T_k})}{\sum_{n=1}^N g(X^n_{T_k}, Z_{T_k})}$, use them to weigh the samples and the weighted samples will be distributed according to the posterior distribution $p(X_{T_k}|\{Z_{T_j}\}_{j=1}^k)$.

In many applications, most samples will have a negligible weight with respect to the observation, so carrying them along does not contribute significantly to the conditional expectation estimate (this is the problem of degeneracy [10]). To create larger diversity one can resample the weights to create more copies of the samples with significant weights. The particle filter with resampling is summarized in the following algorithm due to Gordon *et al.* [8].

Particle filter for a single target

1. Begin with N unweighted samples $X^n_{T_{k-1}}$ from $p(X_{T_{k-1}}|\{Z_{T_j}\}_{j=1}^{k-1})$.
2. **Prediction**: Generate N samples $X'^n_{T_k}$ from $p(X_{T_k}|X_{T_{k-1}})$.
3. **Update**: Evaluate the weights
$$W^n_{T_k} = \frac{g(X'^n_{T_k}, Z_{T_k})}{\sum_{n=1}^N g(X'^n_{T_k}, Z_{T_k})}.$$
4. **Resampling**: Generate N independent uniform random variables $\{\theta^n\}_{n=1}^N$ in $(0,1)$. For $n = 1, \ldots, N$ let $X^n_{T_k} = X'^j_{T_k}$ where
$$\sum_{l=1}^{j-1} W^l_{T_k} \leq \theta^j < \sum_{l=1}^{j} W^l_{T_k}$$
where j can range from 1 to N.
5. Set $k = k+1$ and proceed to Step 1.

The particle filter algorithm is easy to implement and adapt for different problems since the only part of the algorithm that depends on the specific dynamics of the problem is the prediction step. This has led to the particle filter algorithm's increased popularity [5]. However, even with the resampling step, the particle filter can still need a lot of samples in order to describe accurately the conditional density $p(X_{T_k}|\{Z_{T_j}\}_{j=1}^k)$. Snyder *et al.* [18] have shown how the particle filter can fail in simple high dimensional problems because one sample dominates the weight distribution. The rest of the samples are not in statistically significant regions. Even worse, as we will show in the numerical results section, there are simple examples where not even one sample is in a statistically significant region. In the next subsection we present how an extra step can be used to push samples closer to statistically significant regions.

1.2 Generic particle filter for multiple targets

Suppose that we have $\lambda = 1, \ldots, \Lambda$ targets. Also, for notational simplicity, assume that the λth target comes from the λth observation. Even when this is not the case, we can relabel the observations to satisfy this assumption. The targets are assumed to evolve independently so that the observation weight of a sample of the vector of targets is the product of the individual observation weights of the targets [16]. The same is true for the transition density of the vector of targets between observations. We denote the vector of targets at observation T_k by

$$X_{T_k} = (X_{1,T_k}, \ldots, X_{\Lambda,T_k})$$

and the observation vector at T_k by

$$Z_{T_k} = (Z_{1,T_k}, \ldots, Z_{\Lambda,T_k}).$$

Also, we can have different observation weight densities $g_\lambda,\ \lambda = 1, \ldots, \Lambda$ for different targets. However, in the numerical examples we have chosen the same observation weight density for all targets.

Following [16] we can write the particle filter for the case of multiple targets as

Particle filter for multiple targets

1. Begin with N unweighted samples $X^n_{T_{k-1}}$ from $p(X_{T_{k-1}}|\{Z_{T_j}\}_{j=1}^{k-1}) = \prod_{\lambda=1}^{\Lambda} p(X_{\lambda,T_{k-1}}|\{Z_{\lambda,T_j}\}_{j=1}^{k-1})$.
2. **Prediction**: Generate N samples $X'^n_{T_k}$ from
$$p(X_{T_k}|X_{T_{k-1}}) = \prod_{\lambda=1}^{\Lambda} p(X_{\lambda,T_k}|X_{\lambda,T_{k-1}}).$$
3. **Update**: Evaluate the weights
$$W^n_{T_k} = \frac{\prod_{\lambda=1}^{\Lambda} g_\lambda(X'^n_{\lambda,T_k}, Z_{\lambda,T_k})}{\sum_{n=1}^{N} \prod_{\lambda=1}^{\Lambda} g_\lambda(X'^n_{\lambda,T_k}, Z_{\lambda,T_k})}.$$

4. **Resampling**: Generate N independent uniform random variables $\{\theta^n\}_{n=1}^N$ in $(0,1)$. For $n = 1, \ldots, N$ let $X_{T_k}^n = X_{T_k}'^j$ where

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

where j can range from 1 to N.

5. Set $k = k + 1$ and proceed to Step 1.

1.3 Particle filter with MCMC step for a single target

Several authors (see e.g. [7, 23]) have suggested the use of a MCMC step after the resampling step (Step 4) in order to move samples away from statistically insignificant regions. There are many possible ways to append an MCMC step after the resampling step in order to achieve that objective. The important point is that the MCMC step must preserve the conditional density $p(X_{T_k}|\{Z_{T_j}\}_{j=1}^k)$. In the current section we show that the MCMC step constitutes a case of conditional path sampling.

We begin by noting that one can use the resampling step (Step 4) in the particle filter algorithm to create more copies not only of the good samples according to the observation, but also of the values (initial conditions) of the samples at the previous observation. These values are the ones who have evolved into good samples for the current observation (see more details in [23]). The motivation behind producing more copies of the pairs of initial and final conditions is to use the good initial conditions as starting points to produce statistically more significant samples according to the current observation. This process can be accomplished in two steps. First, Step 4 of the particle filter algorithm is replaced by

Resampling: Generate N independent uniform random variables $\{\theta^n\}_{n=1}^N$ in $(0,1)$. For $n = 1, \ldots, N$ let $(X_{T_{k-1}}^n, X_{T_k}^n) = (X_{T_{k-1}}'^j, X_{T_k}'^j)$where

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

To motivate the second step we use the recursive particle filter formulas (2),

(3) to write an update equation for conditional expectations. We have

$$E[f(X_{T_k})|\{Z_{T_j}\}_{j=1}^{k}] = \int f(X_{T_k}) \frac{g(X_{T_k}, Z_{T_k}) p(X_{T_{k-1}}|\{Z_{T_j}\}_{j=1}^{k-1}) dX_{T_{k-1}} dP}{\int g(X_{T_k}, Z_{T_k}) p(X_{T_{k-1}}|\{Z_{T_j}\}_{j=1}^{k-1}) dX_{T_{k-1}} dP}, \quad (5)$$

where P is the Wiener measure. Formula (5) can be approximated (using the initial conditions of the samples produced by the modified resampling step above) as follows:

$$E[f(X_{T_k})|\{Z_{T_j}\}_{j=1}^{k}] \approx \frac{1}{N} \sum_{n=1}^{N} \int f(X_{T_k}^n) \frac{g(X_{T_k}^n, Z_{T_k}) dP}{\int g(X_{T_k}, Z_{T_k}) dP}, \quad (6)$$

i.e., we have approximated the integration over the initial conditions $X_{T_{k-1}}$ appearing in (5) by an average over the initial conditions produced by the modified resampling step. Note that we have not approximated the integration over $X_{T_{k-1}}$ appearing in the denominator because the denominator is a normalization constant which will not be needed in the MCMC sampling.

Formula (6) allows us to perform the MCMC step which will bring the samples closer to the current observation. In particular, starting from the initial conditions at the previous observation which were picked by the resampling step, we can sample the density $g(X_{T_k}^n, Z_{T_k})$ with respect to the Wiener measure dP and produce samples $X_{T_k}^n$ which are more significant with respect to the current observation. The sampling of the density $g(X_{T_k}^n, Z_{T_k})$ with respect to the Wiener measure dP corresponds to the sampling of the noise process (Wiener measure) so that the point $X_{T_k}^n$ has high probability with respect to $g(X_{T_k}^n, Z_{T_k})$. In other words, sampling the density $g(X_{T_k}^n, Z_{T_k})$ given the initial conditions at the previous observation is nothing else but a case of conditional sampling of a path that starts from some given initial condition $X_{T_{k-1}}^n$ and ends at a point $X_{T_k}^n$ which has high probability with respect to $g(X_{T_k}^n, Z_{T_k})$. Of course, the sampled path must also obey the dynamic model (see also [23] for a related approach for improving particle filters).

It is obvious that this sampling procedure preserves the conditional density $p(X_{T_k}|\{Z_{T_j}\}_{j=1}^{k})$ since there was no approximation involved (other than the necessary approximation of the integration over the initial conditions by a discrete summation). The details of how to perform the density sampling will be explained in Section 2.1.

Finally, note that the approximation in equation (6) involves an integration over the Wiener measure. This is because for each of the initial conditions at the previous observation one can create different paths depending on the choice of the Brownian path. The integration over dP can be approximated by an average over appropriately weighted Brownian paths and we find

$$E[f(X_{T_k})|\{Z_{T_j}\}_{j=1}^{k}] \approx \frac{1}{NM}\sum_{n=1}^{N}\sum_{m=1}^{M} f(Y_{T_k}^{n,m}), \qquad (7)$$

The approximation in (7) has an error of order $O(\frac{1}{\sqrt{NM}})$. This means that even if M is kept finite, the approximation still converges in the infinite limit for N. In the numerical implementation we pick $M = 1$ and focus our computational resources on the averaging over initial conditions.

We are now in a position to present the particle filter with MCMC step algorithm

Particle filter with MCMC step for a single target

1. Begin with N unweighted samples $X_{T_{k-1}}^n$ from $p(X_{T_{k-1}}|\{Z_{T_j}\}_{j=1}^{k-1})$.
2. **Prediction**: Generate N samples $X'^n_{T_k}$ from $p(X_{T_k}|X_{T_{k-1}})$.
3. **Update**: Evaluate the weights
$$W_{T_k}^n = \frac{g(X'^n_{T_k}, Z_{T_k})}{\sum_{n=1}^N g(X'^n_{T_k}, Z_{T_k})}.$$
4. **Resampling**: Generate N independent uniform random variables $\{\theta^n\}_{n=1}^N$ in $(0,1)$. For $n = 1,\dots,N$ let $(X_{T_{k-1}}^n, X_{T_k}^n) = (X'^j_{T_{k-1}}, X'^j_{T_k})$ where
$$\sum_{l=1}^{j-1} W_{T_k}^l \le \theta^j < \sum_{l=1}^{j} W_{T_k}^l$$
where j can range from 1 to N.
5. **MCMC step**: Construct a Markov chain $\{Y_{T_k}^{n,l}\}_{l=0}^{\Lambda}$ with initial value $Y_{T_k}^{n,0} = X_{T_k}^n$ and stationary distribution
$$\frac{g(Y, Z_{T_k})dP}{\int g(X_{T_k}, Z_{T_k})dP}.$$
6. Set $X_{T_k}^n = Y_{T_k}^{n,\Lambda}$.

7. Set $k = k + 1$ and proceed to Step 1.

Since the samples $X^n_{T_k} = Y^{n,\Lambda}_{T_k}$ are constructed by starting from different sample paths, they are independent. Also, note that the samples $X^n_{T_k}$ are unweighted. However, we can still measure how well these samples approximate the posterior density by comparing the effective sample sizes of the particle filter with and without the MCMC step. For a collection of N samples the effective sample size $ess(T_k)$ is defined by

$$ess(T_k) = \frac{N}{1 + C_k^2}$$

where

$$C_k = \frac{1}{W_k}\sqrt{\frac{1}{N}\sum_{n=1}^{N}(g(X^n_{T_k}, Z_{T_k}) - W_k)^2} \text{ and } W_k = \frac{1}{N}\sum_{n=1}^{N} g(X^n_{T_k}, Z_{T_k}).$$

The effective sample size can be interpreted as that the N weighted samples are worth of $ess(T_k) = \frac{N}{1+C_k^2}$ i.i.d. samples drawn from the target density, which in our case is the posterior density. By definition, $ess(T_k) \leq N$. If the samples have uniform weights, then $ess(T_k) = N$. On the other hand, if all samples but one have zero weights, then $ess(T_k) = 1$.

1.4 Particle filter with MCMC step for multiple targets

We discuss now the case of multiple, say Λ, targets. Instead of the observations for a single target now we have a collection of observations for all the targets $\{Z_{T_j}\}_{j=1}^k = \{(Z^1_{T_j}, \ldots, Z^\lambda_{T_j})\}_{j=1}^k$. There are two cases which we now examine. First, the function whose conditional expectation we want to compute depends only on one of the targets. Second, the function whose conditional expectation we want to compute depends at least on two targets.

We begin with the first case. Recall that the targets evolve independently and after the appropriate association between a target and an observation, we have for the conditional expectation estimate for the λth target ($\lambda =$

$1, \ldots, \Lambda)$

$$
\begin{aligned}
& E[f(X_{\lambda,T_k})|\{Z_{T_j}\}_{j=1}^{k}] \\
&= \int f(X_{\lambda,T_k}) \frac{\prod_{\lambda'=1}^{\Lambda} g_{\lambda'}(X_{\lambda',T_k}, Z_{\lambda',T_k}) p(X_{\lambda',T_{k-1}}|\{Z_{\lambda',T_j}\}_{j=1}^{k-1}) dX_{\lambda',T_{k-1}} dP_{\lambda'}}{\int \prod_{\lambda'=1}^{\Lambda} g_{\lambda'}(X_{\lambda',T_k}, Z_{\lambda',T_k}) p(X_{\lambda',T_{k-1}}|\{Z_{\lambda',T_j}\}_{j=1}^{k-1}) dX_{\lambda',T_{k-1}} dP_{\lambda'}} \\
&\quad = \int f(X_{\lambda,T_k}) \frac{g_{\lambda}(X_{\lambda,T_k}, Z_{\lambda,T_k}) p(X_{\lambda,T_{k-1}}|\{Z_{\lambda,T_j}\}_{j=1}^{k-1}) dX_{\lambda,T_{k-1}} dP_{\lambda}}{\int g_{\lambda}(X_{\lambda,T_k}, Z_{\lambda,T_k}) p(X_{\lambda,T_{k-1}}|\{Z_{\lambda,T_j}\}_{j=1}^{k-1}) dX_{\lambda,T_{k-1}} dP_{\lambda}}
\end{aligned}
\tag{8}
$$

where the second identity comes from integrating out all the targets except for the λth target. Due to this simplification we are back at the case of a single target as in equation (5). So, we can perform the MCMC step *individually* for each target. The particle filter with MCMC step for the case of multiple targets is

Particle filter with MCMC step for multiple targets

1. Begin with N unweighted samples $X_{T_{k-1}}^n$ from $p(X_{T_{k-1}}|\{Z_{T_j}\}_{j=1}^{k-1}) = \prod_{\lambda=1}^{\Lambda} p(X_{\lambda,T_{k-1}}|\{Z_{\lambda,T_j}\}_{j=1}^{k-1})$.

2. **Prediction**: Generate N samples $X'^n_{T_k}$ from

$$p(X_{T_k}|X_{T_{k-1}}) = \prod_{\lambda=1}^{\Lambda} p(X_{\lambda,T_k}|X_{\lambda,T_{k-1}}).$$

3. **Update**: Evaluate the weights

$$W_{T_k}^n = \frac{\prod_{\lambda=1}^{\Lambda} g_{\lambda}(X'^n_{\lambda,T_k}, Z_{\lambda,T_k})}{\sum_{n=1}^{N} \prod_{\lambda=1}^{\Lambda} g_{\lambda}(X'^n_{\lambda,T_k}, Z_{\lambda,T_k})}.$$

4. **Resampling**: Generate N independent uniform random variables $\{\theta^n\}_{n=1}^{N}$ in $(0,1)$. For $n = 1, \ldots, N$ let $(X_{T_{k-1}}^n, X_{T_k}^n) = (X'^j_{T_{k-1}}, X'^j_{T_k})$ where

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

where j can range from 1 to N.

5. **MCMC step**: For $\lambda = 1, \dots, \Lambda$, construct a Markov chain $\{Y^{n,l}_{\lambda,T_k}\}_{l=0}^{L}$ with initial value $Y^{n,0}_{\lambda,T_k} = X^n_{\lambda,T_k}$ and stationary distribution

$$\frac{g_\lambda(Y_\lambda, Z_{T_k})dP_\lambda}{\int g_\lambda(X_{\lambda,T_k}, Z_{\lambda,T_k})dP_\lambda}. \tag{9}$$

6. Set $X^n_{\lambda,T_k} = Y^{n,L}_{\lambda,T_k}$.

7. Set $k = k + 1$ and proceed to Step 1.

For a collection of N samples the effective sample size $ess_\Lambda(T_k)$ for Λ targets is

$$ess_\Lambda(T_k) = \frac{N}{1 + C^2_{\Lambda,k}}$$

where

$$C_{\Lambda,k} = \frac{1}{W_{\Lambda,k}} \sqrt{\frac{1}{N} \sum_{n=1}^{N} (\prod_{\lambda=1}^{\Lambda} g_\lambda(X^n_{\lambda,T_k}, Z_{\lambda,T_k}) - W_{\Lambda,k})^2}$$

$$\text{and } \; W_{\Lambda,k} = \frac{1}{N} \sum_{n=1}^{N} \prod_{\lambda=1}^{\Lambda} g_\lambda(X^n_{\lambda,T_k}, Z_{\lambda,T_k}).$$

Finally, we mention the necessary modifications to compute conditional expectation estimates of a function $h(X_{1,T_k}, \dots, X_{\Lambda,T_k}) = h(X_{T_k})$ that depends, in general, on all targets.

$$\begin{aligned}
&E[h(X_{T_k})|\{Z_{T_j}\}_{j=1}^{k}] \\
&= \int h(X_{T_k}) \frac{\prod_{\lambda'=1}^{\Lambda} g_{\lambda'}(X_{\lambda',T_k}, Z_{\lambda',T_k}) p(X_{\lambda',T_{k-1}}|\{Z_{\lambda',T_j}\}_{j=1}^{k-1}) dX_{\lambda',T_{k-1}} dP_{\lambda'}}{\int \prod_{\lambda'=1}^{\Lambda} g_{\lambda'}(X_{\lambda',T_k}, Z_{\lambda',T_k}) p(X_{\lambda',T_{k-1}}|\{Z_{\lambda',T_j}\}_{j=1}^{k-1}) dX_{\lambda',T_{k-1}} dP_{\lambda'}} \\
&\approx \frac{1}{N} \sum_{n=1}^{N} \int h(X^n_{T_k}) \frac{\prod_{\lambda'=1}^{\Lambda} g_{\lambda'}(X^n_{\lambda',T_k}, Z_{\lambda',T_k}) dP_{\lambda'}}{\int \prod_{\lambda'=1}^{\Lambda} g_{\lambda'}(X_{\lambda',T_k}, Z_{\lambda',T_k}) dP_{\lambda'}}.
\end{aligned}$$

In order to carry out the MCMC step in this case one has to construct a Markov chain with stationary density

$$\rho(Y) = \frac{\prod_{\lambda'=1}^{\Lambda} g_{\lambda'}(Y, Z_{\lambda',T_k}) dP_{\lambda'}}{\int \prod_{\lambda'=1}^{\Lambda} g_{\lambda'}(X_{\lambda',T_k}, Z_{\lambda',T_k}) dP_{\lambda'}} \tag{10}$$

for each sample $n = 1, \dots, N$. The algorithm described above needs to be modified only in the MCMC step where one needs to construct a Markov chain with stationary density given by (10).

2 Numerical results

We present numerical results for multi-target tracking using the particle filter with an MCMC step. We have synthesized tracks of targets moving on the xy plane using a $2D$ near constant velocity model [1]. At each time t we have a total of K_t targets and the evolution of the kth target ($k = 1, \ldots, K_t$) is given by

$$\begin{aligned} \mathbf{x}_{k,t} &= \mathbf{A}\mathbf{x}_{k,t-1} + \mathbf{B}\mathbf{v}_{k,t} \\ &= [x_{k,t}, \dot{x}_{k,t}, y_{k,t}, \dot{y}_{k,t}]^T, \end{aligned} \tag{11}$$

where $(x_{k,t}, \dot{x}_{k,t})$ and $(y_{k,t}, \dot{y}_{k,t})$ are the xy position and velocity of the kth target at time t. The matrices $\mathbf{A}$ and $\mathbf{B}$ are given by

$$\mathbf{A} = \begin{bmatrix} 1 & T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \text{and} \quad \mathbf{B} = \begin{bmatrix} T^2/2 & 0 \\ T & 0 \\ 0 & T^2/2 \\ 0 & T \end{bmatrix}, \tag{12}$$

where T is the time between observations. For the experiments we have set $T = 1$, i.e., noisy observations of the model are obtained at every step of the model (11). The model noise $\mathbf{v}_{k,t}$ is a collection of independent Gaussian random variables with covariance matrix $\mathbf{\Sigma}_v$ defined as

$$\mathbf{\Sigma}_v = \begin{bmatrix} \sigma_x^2 & 0 \\ 0 & \sigma_y^2 \end{bmatrix}. \tag{13}$$

In the experiments we have $\sigma_x^2 = \sigma_y^2 = 1$. Also, we have considered two possible cases for the observation model, one linear and one nonlinear. Due to the different possible combinations of targets to observations we use a different index m to denote the obsevations. Since we do not assume any clutter we have $m = 1, \ldots, K_t$. If the mth observation $\mathbf{z}_{m,t}$ at time t comes from the kth target we have

$$\mathbf{z}_{m,t} = \begin{bmatrix} x_{k,t} \\ y_{k,t} \end{bmatrix} + \mathbf{w}_{m,t} \tag{14}$$

for the linear observation model and

$$\mathbf{z}_{m,t} = \begin{bmatrix} \arctan(\frac{y_{k,t}}{x_{k,t}}) \\ (x_{k,t}^2 + y_{k,t}^2)^{1/2} \end{bmatrix} + \mathbf{w}_{m,t} \tag{15}$$

for the nonlinear observation model. As is usual in the literature, the nonlinear observation model consists of the bearing θ and range r of a target. The observation noise $\mathbf{w}_{m,t}$ is white and Gaussian with covariance matrix

$$\mathbf{\Sigma}_w = \begin{bmatrix} \sigma^2_{obs,x} & 0 \\ 0 & \sigma^2_{obs,y} \end{bmatrix} \tag{16}$$

for the linear observation model and

$$\mathbf{\Sigma}_w = \begin{bmatrix} \sigma^2_{\theta} & 0 \\ 0 & \sigma^2_{r} \end{bmatrix} \tag{17}$$

for the nonlinear observation model. For the numerical experiments with the linear observation model we chose $\sigma^2_{obs,x} = \sigma^2_{obs,y} = 1$. For the numerical experiments with the nonlinear observation model we chose $\sigma^2_{\theta} = 10^{-4}$ and $\sigma^2_r = 1$. These values make our example comparable in difficulty to examples appearing in the literature (see e.g. [16, 21, 22]).

The synthesized target tracks were created by specifying a certain scenario, to be detailed below, of surviving, newborn and disappearing targets. According to this scenario we evolved the appropriate number of targets according to (11) and recorded the state of each target at each step. For the surviving targets we created an observation by using the state of the target in the observation model. Thus, for the linear observation model, the observations were created directly in xy space by perturbing the xy position of the target by (14). For the nonlinear observation model, the observations were created in bearing and range space θ, r by using (15). The perturbed bearing and range were transformed to xy space by the transformation $x = r\cos\theta,\ y = r\sin\theta$ to create a position for the target in xy space.

The newborn targets for the linear model were created in xy space directly by sampling uniformly in $[-100, 100]$. Afterwards, the observations of the newborn targets were constructed by perturbing the x, y positions using (14). The newborn targets for the nonlinear model were created in xy space by sampling uniformly in $[-100, 100]$. Afterwards, we transformed the x, y positions to the bearing and range space θ, r and perturbed the bearing and range according to (15). The perturbed bearing and range were again transformed back to xy space to create the position of the newborn target. Note that both observation models do *not* involve the velocities. The newborn target velocities were sampled uniformly in $[-1, 1]$.

The number of targets at each observation instant is: $K_0 = 2$, $K_1 = 2$, $K_2 = 1$, $K_3 = 2$, $K_4 = 3$, $K_5 = K_6 = \ldots = K_{200} = 4$. So, for the majority of the steps we have 4 targets which makes the problem of tracking rather difficult.

2.1 Implementation of the MCMC step

We present now the implementation of the extra (MCMC) step for the particle filter. First, we construct a continuous time model for which (11) is a discrete approximation. This is not necessary. One can work with the discrete model (11). However, we want to show that the proposed approach is more general and can be applied to continuous time models as well. Second, we show how the extra step leads to a sampling problem which we address via MCMC sampling, in particular via Hybrid Monte Carlo (HMC).

2.1.1 Continuous time reformulation of the dynamic model

The $2D$ near constant velocity model (11) is a simplified discrete approximation of the formal linear SDE system

$$\begin{aligned} \ddot{x}_t &= \sigma_x \dot{u}_{x,t} \\ \ddot{y}_t &= \sigma_y \dot{u}_{y,t}, \end{aligned} \tag{18}$$

where $u_{x,t}, u_{y,t}$ are independent Brownian motions and $\sigma_x \dot{u}_{x,t}, \sigma_y \dot{u}_{y,t}$ are Gaussian white noises with covariances $R_x(t) = \sigma_x^2 \delta(t)$ and $R_y(t) = \sigma_y^2 \delta(t)$ respectively and $\delta(t)$ is the delta function. We can define velocities in the x and y directions, $p_{x,t} = \dot{x}_t$ and $p_{y,t} = \dot{y}_t$ and we can rewrite (18) as

$$\begin{aligned} \dot{x}_t &= p_{x,t}, \\ \dot{p}_{x,t} &= \sigma_x \dot{u}_{x,t}, \\ \dot{y}_t &= p_{y,t}, \\ \dot{p}_{y,t} &= \sigma_y \dot{u}_{y,t}. \end{aligned} \tag{19}$$

The formal system (19) can be written rigorously

$$\begin{aligned} dx_t &= p_{x,t} dt, \\ dp_{x,t} &= \sigma_x du_{x,t}, \\ dy_t &= p_{y,t} dt, \\ dp_{y,t} &= \sigma_y du_{y,t}. \end{aligned} \tag{20}$$

The equations for the positions have zero noise and the equations for the velocities have zero drift. We can write the SDE system (20) in matrix form as

$$\begin{bmatrix} dx_t \\ dp_{x,t} \\ dy_t \\ dp_{y,t} \end{bmatrix} = \begin{bmatrix} p_{x,t} \\ 0 \\ p_{y,t} \\ 0 \end{bmatrix} dt + \begin{bmatrix} 0 & 0 \\ \sigma_x & 0 \\ 0 & 0 \\ 0 & \sigma_y \end{bmatrix} \begin{bmatrix} du_{x,t} \\ du_{y,t} \end{bmatrix}. \tag{21}$$

Define the 4×1 vectors $z_t = [z_{1,t}, \dots, z_{4,t}]^T$, $a(z_t) = [a_{1,t}, \dots, a_{4,t}]^T$, the 2×1 vector $w_t = [w_{1,t}, w_{2,t}]^T$ and a 4×2 constant matrix σ by

$$z_t = \begin{bmatrix} x_t \\ p_{x,t} \\ y_t \\ p_{y,t} \end{bmatrix}, \quad a(z_t) = \begin{bmatrix} p_{x,t} \\ 0 \\ p_{y,t} \\ 0 \end{bmatrix}, \quad w_t = \begin{bmatrix} u_{x,t} \\ u_{y,t} \end{bmatrix} \text{ and } \sigma = \begin{bmatrix} 0 & 0 \\ \sigma_x & 0 \\ 0 & 0 \\ 0 & \sigma_y \end{bmatrix}.$$

Note that

$$\begin{bmatrix} z_{1,t} \\ z_{2,t} \\ z_{3,t} \\ z_{4,t} \end{bmatrix} = \begin{bmatrix} x_t \\ p_{x,t} \\ y_t \\ p_{y,t} \end{bmatrix} \quad \text{and} \quad a(z_t) = \begin{bmatrix} z_{2,t} \\ 0 \\ z_{4,t} \\ 0 \end{bmatrix}.$$

With these definitions we can rewrite the SDE system (21) as

$$dz_t = a(z_t)dt + \sigma dw_t \tag{22}$$

We suppose that we are at observation time T and we have N samples. After the resampling step at observation time 0, we have obtained a collection of pairs (z_0^n, z_T^n), for $n = 1, \dots, N$. The system (22) can be solved explicitly. In particular, we have for the position and velocity in the x direction

$$\begin{aligned} z'_{1,T} &= z'_{1,0} + z'_{2,0}T + \int_0^T \sigma_x u_{x,s} ds \\ z'_{2,T} &= z'_{2,0} + \sigma_x u_{x,T}. \end{aligned} \tag{23}$$

where we have used the fact that the Brownian motion $u_{x,t}$ is zero at $t = 0$. For the nth sample, if we use $z'^n_{1,0} = z^n_{1,0}$ and $z'^n_{2,0} = z^n_{2,0}$ we have

$$\begin{aligned} z'^n_{1,T} &= z^n_{1,0} + z^n_{2,0}T + \int_0^T \sigma_x u^n_{x,s} ds \\ z'^n_{2,T} &= z^n_{2,0} + \sigma_x u^n_{x,T}, \end{aligned} \tag{24}$$

where $u^n_{x,s}$, $s \in [0, T]$ is a *new* Brownian path for each sample.

We should note here that if the continuous time dynamic model is non-linear and cannot be solved explicitly we can still use the proposed approach by utilizing a numerical discretization of the dynamic model e.g. the Euler-Maruyama or higher order schemes [9].

Since the targets evolve independently of one another and we are interested in computing conditional expectation estimates of functions that depend only on one target, we need to formulate the MCMC step only for the case of a single target. The conditional density that we have to sample for the linear observation model is given by (see (9) in particle filter algorithm with MCMC step for multiple targets)

$$g_x(Z_{1,T}, z'^n_{1,T}) g_y(Z_{3,T}, z'^n_{3,T}) dP \propto \exp\left[-\frac{(Z_{1,T} - z'^n_{1,T})^2}{2 * \sigma^2_{obs,x}}\right] \exp\left[-\frac{(Z_{3,T} - z'^n_{3,T})^2}{2 * \sigma^2_{obs,y}}\right] dP \quad (25)$$

where $Z_{1,T}$ the observation value of the x position of the target and $Z_{3,T}$ the observation value of the y position of the target. Note that the x, y positions $z'^n_{1,T}, z'^n_{3,T}$ of the nth sample are determined by

$$z'^n_{1,T} \approx z^n_{1,0} + z^n_{2,0} T + T \sigma_x u^n_{x,T} \quad (26)$$

$$z'^n_{3,T} \approx z^n_{3,0} + z^n_{4,0} T + T \sigma_y u^n_{y,T} \quad (27)$$

where we have used the fact that we go in one step of size T from one observation to the next to make the approximations $\int_0^T \sigma_x u^n_{x,s} ds \approx T \sigma_x u^n_{x,T}$ and $\int_0^T \sigma_y u^n_{y,s} ds \approx T \sigma_y u^n_{y,T}$. Due to (26), (27) and since the initial conditions $z^n_{1,0}, \ldots, z^n_{4,0}$ are fixed by the resampling step, the only quantities that we need to sample with the conditional density (25) are the Brownian values $u^n_{x,T}, u^n_{y,T}$. With the help of (26), (27) we can approximate $g_x(Z_{1,T}, z'^n_{1,T}) g_y(Z_{3,T}, z'^n_{3,T}) dP$ as

$$g_x(Z_{1,T}, z'^n_{1,T}) g_y(Z_{3,T}, z'^n_{3,T}) dP \approx \exp\Bigg(-\Bigg\{ \frac{(Z_{1,T} - z^n_{1,0} - z^n_{2,0} T - T \sigma_x u^n_{x,T})^2}{2 * \sigma^2_{obs,x}} + \frac{(Z_{3,T} - z^n_{3,0} - z^n_{4,0} T - T \sigma_y u^n_{y,T})^2}{2 * \sigma^2_{obs,y}} + \frac{(u^n_{x,T})^2}{2} + \frac{(u^n_{y,T})^2}{2} \Bigg\}\Bigg), \quad (28)$$

where the last two terms in the exponential come from approximating the Wiener measure dP. The expression (28) for $g_x(Z_{1,T}, z'^n_{1,T}) g_y(Z_{3,T}, z'^n_{3,T}) dP$ is a Gaussian density for $u^n_{x,T}, u^n_{y,T}$. We do not need HMC to sample it. We can rewrite it as the product of two Gaussian densities, one for $u^n_{x,T}$ and one for

$u^n_{y,T}$ and sample it with standard methods for Gaussian densities. However, we show how HMC is implemented because for the nonlinear observation model, the density to be sampled will no longer be Gaussian.

2.1.2 Hybrid Monte Carlo formulation

We present briefly the hybrid Monte Carlo (HMC) formulation that we have used to sample the conditional density for each target. We start with the linear observation model. Define the potential $V_{\epsilon_l}(u^n_{x,T}, u^n_{y,T})$ by

$$V_{\epsilon_l}(u^n_{x,T}, u^n_{y,T}) = \frac{(Z_{1,T} - z^n_{1,0} - z^n_{2,0}T - T\sigma_x u^n_{x,T})^2}{2 * \sigma^2_{obs,x}} + \frac{(Z_{3,T} - z^n_{3,0} - z^n_{4,0}T - T\sigma_y u^n_{y,T})^2}{2 * \sigma^2_{obs,y}} + \frac{(u^n_{x,T})^2}{2} + \frac{(u^n_{y,T})^2}{2} \quad (29)$$

and the density $g_x(Z_{1,T}, z'^n_{1,T})g_y(Z_{3,T}, z'^n_{3,T})dP$ becomes

$$g_x(Z_{1,T}, z'^n_{1,T})g_y(Z_{3,T}, z'^n_{3,T})dP \approx \exp\Big(-V_{\epsilon_l}(u^n_{x,T}, u^n_{y,T})\Big).$$

Consider $u^n_{x,T}, u^n_{y,T}$ as the position variables of a Hamiltonian system. We define the $2D$ position vector $q = [q_1, q_2]^T$ with $q_1 = u^n_{x,T}$ and $q_2 = u^n_{y,T}$. To each of the position variables we associate a momentum variable and we write the Hamiltonian

$$H_{\epsilon_l}(q,p) = V_{\epsilon_l}(q) + \frac{p^T p}{2},$$

where $p = [p_1, p_2]^T$ is the momentum vector. Thus, the momenta variables are Gaussian distributed random variables with mean zero and variance 1. The equations of motion for this Hamiltonian system are given by Hamilton's equations

$$\frac{dq_i}{d\tau} = \frac{\partial H_{\epsilon_l}}{\partial p_i} \text{ and } \frac{dp_i}{d\tau} = -\frac{\partial H_{\epsilon_l}}{\partial q_i} \text{ for } i = 1, \ldots, 2.$$

HMC proceeds by assigning initial conditions to the momenta variables (through sampling from $\exp(-\frac{p^T p}{2})$), evolving the Hamiltonian system in fictitious time τ for a given number of steps of size $\delta\tau$ and then using the solution of the system to perform a Metropolis accept/reject step (more details in [10]). After the Metropolis step, the momenta values are discarded. The most popular method for solving the Hamiltonian system, which is the one we also used, is the Verlet leapfrog scheme. In our numerical implementation, we did not attempt to optimize the performance of the HMC algorithm. For the sampling we used 100 Metropolis accept/reject steps and 1 HMC step of size $\delta\tau = 10^{-1}$ to construct a trial path.

For the nonlinear observation model

$$g_\theta(Z_{\theta,T}, z_T'^n) g_r(Z_{r,T}, z_T'^n) \propto \exp\left[-\frac{(Z_{\theta,T} - \theta(z_T'^n))^2}{2 * \sigma_\theta^2}\right] \exp\left[-\frac{(Z_{r,T} - r(z_T'^n))^2}{2 * \sigma_r^2}\right] \tag{30}$$

where $Z_{\theta,T}, Z_{r,T}$ are the bearing and range observation values for the target and

$$\theta(z_T'^n) = \arctan\left(\frac{z_{3,T}'^n}{z_{1,T}'^n}\right) \text{ and } r(z_T'^n) = ({z_{1,T}'^n}^2 + {z_{3,T}'^n}^2)^{1/2}$$

are the bearing and range values for the nth sample. We can use the same procedure as in the linear observation model to define a Hamiltonian system and its associated equations. We omit the details.

2.2 Linear observations

We start the presentation of our numerical experiments with results for the linear observation model (14). Figures 1 and 2 show the evolution in the xy space of the true targets, the observations as well as the estimates of the improved particle filter. It is obvious from the figures that the improved particle filter follows accurately the targets and there is no ambiguity in the identification of the target tracks.

The performance of the improved particle filter with 100 samples is compared to the performance of the generic particle filter with 120 samples in Figure 3 by monitoring the evolution in time of the RMS error per target. The RMS error per target (RMSE) is defined with reference to the true target tracks by the formula

$$RMSE(t) = \sqrt{\frac{1}{K_t}\sum_{k=1}^{K_t} \|\mathbf{x}_{k,t} - E[\mathbf{x}_{k,t}|Z_1,\dots,Z_t]\|^2} \tag{31}$$

where $\|\cdot\|$ is the norm of the position and velocity vector. Note that the state vector norm involves both positions and velocities even though the observations use information only from the positions of a target. $\mathbf{x}_{k,t}$ is the true

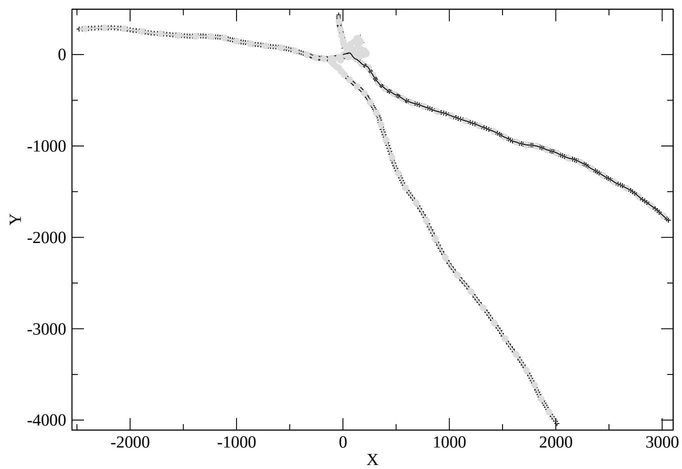

Figure 1: Linear observation model. The solid lines denote the true target tracks, the crosses denote the observations and the dots the conditional expectation estimates from the improved particle filter. We have plotted the conditional expectation estimates every 5 observations to avoid cluttering in the figure.

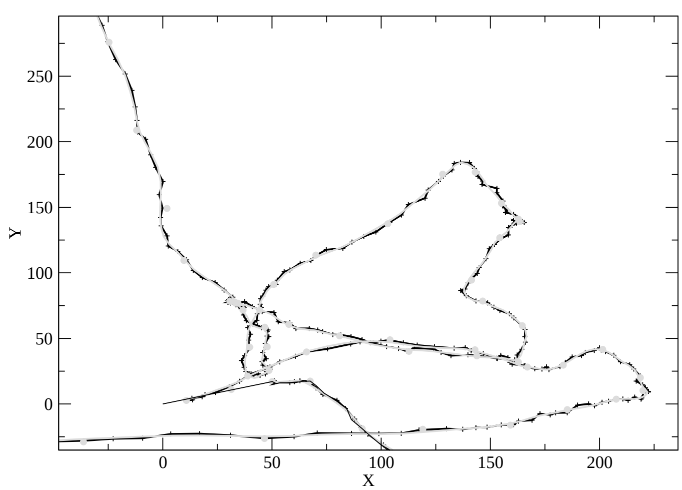

Figure 2: Linear observation model. Detail of Figure 1.

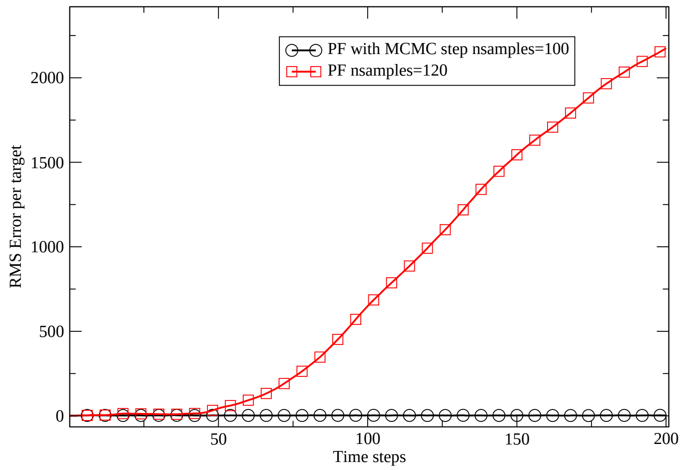

Figure 3: Linear observation model. Comparison of RMS error per target for the improved particle filter and the generic particle filter.

state vector for target k. $E[\mathbf{x}_{k,t}|Z_1, \ldots, Z_t]$ is the conditional expectation estimate calculated with the improved or generic particle filter depending on whose filter's performance we want to calculate.

The improved particle filter has a computational overhead of the order of a few percent compared to the generic particle filter. We have thus used the generic particle filter with more samples than the improved particle filter. This additional number of samples more than accounts for the computational overhead of the improved particle filter. As can be seen in Figure 3 the generic particle filter's accuracy deteriorates quickly. On the other hand, the improved particle filter maintains an $O(1)$ RMS error per target for the entire tracking interval. The average value of the RMS error over the entire time interval of tracking is about 2.5 with standard deviation of about 0.5. For the generic particle filter, the average of the RMS error over the time interval of tracking is about 800 with standard deviation of about 760.

Figure 4 compares the effective sample size for the generic particle filter and the improved particle filter. Because the number of samples is different for the two filters we have plotted the effective sample size as a percentage of the number of samples. We have to note that, after about 50 steps, the generic particle filter started producing observation weights (before the normalization) which were numerically zero. This makes the normalization impossible. In order to allow the generic particle filter to continue we chose

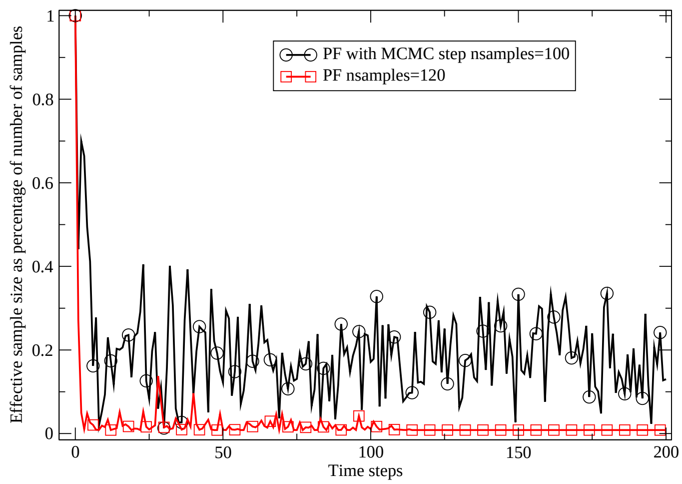

Figure 4: Linear observation model. Comparison of effective sample size for the improved particle filter and the generic particle filter.

at random one of the samples, since all of them are equally bad, and assigned all the weight to this sample. We did that for all the steps for which the observation weights were zero before the normalization. As a result, the effective sample size for the generic particle filter drops down to 1 sample after about 50 steps. Once the generic particle filter deviates from the true target tracks there is no mechanism to correct it. Also, we tried assigning equal weights to all the samples when the observation weights dropped to zero. This did not improve the generic particle's performance either. On the other hand, the improved particle filter maintains an effective sample size which is about 25% of the number of samples.

2.3 Nonlinear observations

We continue with results for the nonlinear observation model (15). Figures 5 and 6 show the evolution in the xy space of the true targets, the observations as well as the estimates of the improved particle filter. Again, as in the case of the linear observation model, the improved particle filter follows accurately the targets and there is no ambiguity in the identification of the target tracks.

The case of the nonlinear observation model is much more difficult than the case of the linear observation model. The reason is that for the nonlinear

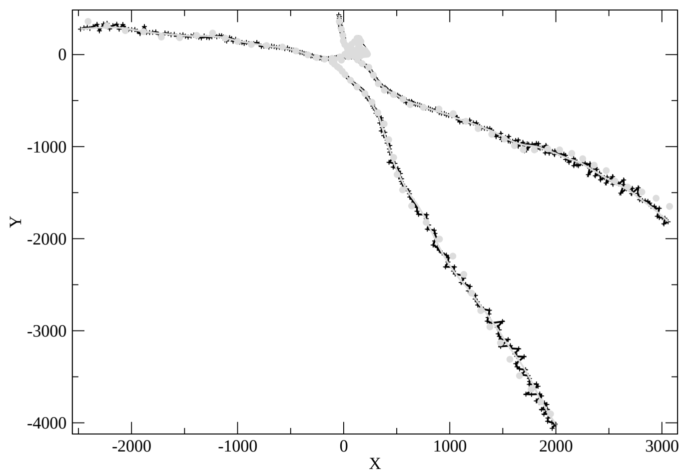

Figure 5: Noninear observation model. The solid lines denote the true target tracks, the crosses denote the observations and the dots the conditional expectation estimates from the improved particle filter. We have plotted the conditional expectation estimates every 5 observations to avoid cluttering in the figure.

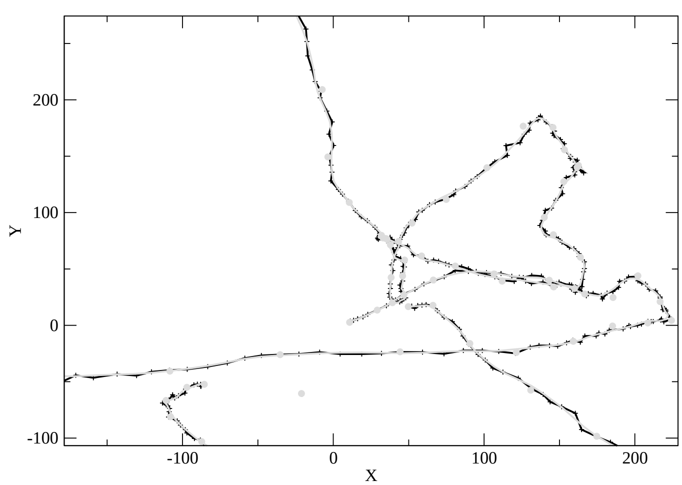

Figure 6: Nonlinear observation model. Detail of Figure 5.

observation model, the observation errors, though constant in bearing and range space, they become position dependent in xy space. In particular, when x and/or y are large, the observation errors can become rather large. This is easy to see by Taylor expanding the nonlinear transformation from bearing and range space to xy space around the true target values. Suppose that the true target bearing and range are θ_0, r_0 and its xy space position is $x_0 = r_0 \cos\theta_0, y_0 = r_0 \sin\theta_0$. Also, assume that the observation error in bearing and range space is, respectively, $\delta\theta$ and δr. The xy position of a target that is perturbed by $\delta\theta$ and δr in bearing and range space is (to first order)

$$\begin{aligned} x &= x_0 - y_0\delta\theta - \delta r \cos\theta_0 \\ y &= y_0 + x_0\delta\theta - \delta r \sin\theta_0. \end{aligned}$$

Thus, the perturbation in xy space can be significant even if $\delta\theta$ and δr are small. In our example we have $\sigma_\theta = 10^{-2}$. So, when the true target x and y values become of the order of 10^3 as happens for some of the targets, the observation value in bearing and range space can be quite misleading as far as the xy space position of the target is concerned. As a result, even if one does a good job in following the observation in bearing and range space, the conditional expectation estimate of the xy space position can be inaccurate.

With this in mind, we have used 200 samples for the improved particle filter and 220 samples for the generic particle filter. Again, the extra samples used for the generic particle filter more than account for the computational overhead of the improved particle filter. The performance of the improved particle filter is compared to the performance of the generic particle filter in Figure 7 by monitoring the evolution in time of the RMS error per target. The generic particle filter's accuracy again deteriorates rather quickly. The error for the improved particle filter is larger than in the linear observation model but never exceeds about 80 even after 200 steps when the targets have reached large values of x and/or y. The average value of the RMS error over the entire time interval of tracking is about 22 with standard deviation of about 21. For the generic particle filter, the average of the RMS error over the time interval of tracking is about 760 with standard deviation of about 770.

Figure 8 compares the effective sample size for the generic particle filter and the improved particle filter. After about 60 steps, the generic particle filter, started producing observation weights (before the normalization) which were numerically zero. This makes the normalization impossible. In order to allow the generic particle filter to continue we chose at random

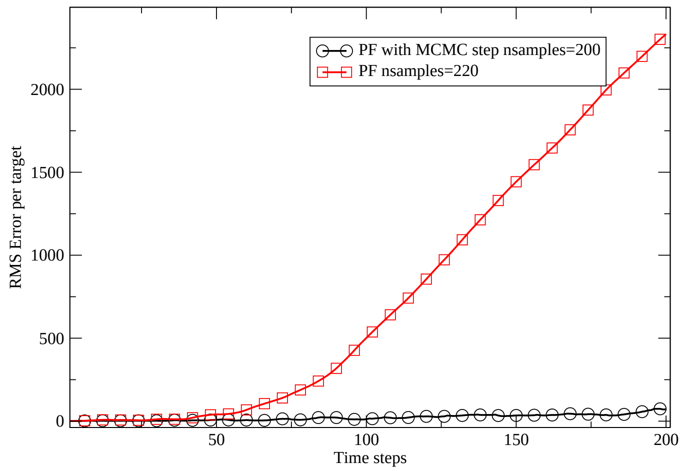

Figure 7: Nonlinear observation model. Comparison of RMS error per target for the improved particle filter and the generic particle filter.

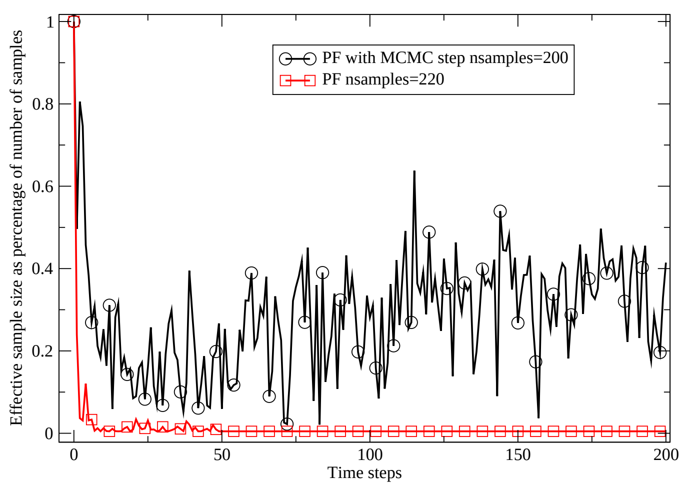

Figure 8: Nonlinear observation model. Comparison of effective sample size for the improved particle filter and the generic particle filter.

one of the samples, since all of them are equally bad, and assigned all the weight to this sample. We did that for all the steps for which the observation weights were zero before the normalization. As a result, the effective sample size for the generic particle filter drops down to 1 sample after about 60 steps. Once the generic particle filter deviates from the true target tracks there is no mechanism to correct it. Also, we tried assigning equal weights to all the samples when the observation weights dropped to zero. This did not improve the generic particle's performance either. On the other hand, the improved particle filter maintains an effective sample size which is about 25% of the number of samples.

3 Discussion

We have presented an algorithm for multi-target tracking which builds on the existing particle filter methodology for multi-target tracking by appending an MCMC step after the particle filter resampling step. The purpose of the addition of the MCMC step is to bring the samples closer to the observation. Even though the addition of an MCMC step for a particle filter has been proposed and used before [7], to the best of our knowledge, the currently proposed implementation of the MCMC step is novel (see also [23] for a related approach).

We have tested the performance of the algorithm on the problem of tracking multiple targets evolving under the near constant velocity model [1]. We have examined two cases of observation models: i) a linear observations model involving the positions of the targets and ii) a nonlinear observation model involving the bearing and range of the targets. For both cases the proposed improved particle filter exhibited a significantly better performance than the generic particle filter. Since the improved particle filter requires more computations than the generic particle filter it is bound to be more expensive. However, the computational overhead of the improved particle filter is rather small, of the order of a few extra samples worth for the generic particle filter.

In [15] we proposed another way of performing the extra MCMC step of a particle filter. That approach was based on modifying the drift of the dynamic model and then accounting for the modification via Girsanov's theorem. In the current work, we use the original drift of the dynamic model without any modification. For the case of multi-target tracking with observations at every time step both algorithms perform equally well. Thus, at first sight it would appear that there is no need for the extra complication of modifying the drift of the dynamic model and then accounting for the modification by Girsanov's theorem as was done in [15]. However, in cases where there are only sparse observations, the sampling of the conditional density needed for the extra step can be much more difficult (and consequently expensive) for the original dynamic model than for the modified dynamic model. With this in mind, the approach in [15] seems to have wider applicability. A detailed comparison of the algorithm proposed in the current work and the one proposed in [15] will be presented in a future publication.

Acknowledgements

We are grateful to Prof. J. Weare for many discussions. Also, we would like to thank the Institute for Mathematics and its Applications in the University of Minnesota for its support.

References

[1] Bar-Shalom Y. and Fortmann T.E., Tracking and Data Association, Academic Press, 1988.

[2] Bar-Shalom Y. and Blair W.D., Eds. Multitarget-Multisensor Tracking: Applications and Advances, vol. III, Norwood, MA, Artech House, 2000.

[3] Blackman S. and Popoli R., Design and Analysis of Modern Tracking Systems, Norwood, MA, Artech House, 1999.

[4] Chorin, A.J. and Tu X., Implicit sampling for particle filters, Proc. Nat. Acad. Sc. USA 106 (2009) pp. 17249-17254.

[5] Doucet A., de Freitas N. and Gordon N. (eds.) , Sequential Monte Carlo Methods in Practice, Springer NY, 2001.

[6] Fortmann T. E., Bar-Shalom Y. and Scheffe M., Sonar tracking of multiple targets using joint probabilistic data association, IEEE J. Ocea. Eng., vol.8 (1983) pp.173-184.

[7] Gilks W. and Berzuini C., Following a moving target. Monte Carlo inference for dynamic Bayesian models, J. Royal Stat. Soc. B 63 (1) (1999) pp. 2124-2137.

[8] Gordon N.J., Salmond D.J. and Smith A.F.M., Novel approach to nonlinear/non-Gaussian Bayesian state estimation, Proc. Inst. Elect. Eng. F 140(2) (1993) pp. 107-113.

[9] Kloeden P. and Platen E., Numerical Solution of Stochastic Differential Equations, Springer Berlin 1992.

[10] Liu J.S., Monte Carlo Strategies in Scientific Computing, Springer NY, 2001.

[11] Liu J.S. and Chen R. Sequential Monte Carlo Methods for Dynamic Systems. Journal of the American Statistical Association, vol.93 no. 443 (1993) pp. 1032-1044.

[12] Mahler R. P., Statistical Multisource-Multitarget Information Fusion, Artech House Publishers MA, 2007.

[13] Mahler R. P. S., Multitarget Bayes filtering via first-order multitarget moments, IEEE Trans. Aero. Elect. Sys., Vol. 39 no. 4 (2003) pp. 1152-1178.

[14] Mahler R.P.S. and Maroulas V. Tracking Spawning Objects, (2010) *submitted*.

[15] Maroulas V. and Stinis P., A Girsanov Monte Carlo approach to particle filtering for multi-target tracking, (2010), arXiv:1006.3100v1 .

[16] Ng W., Li J.F., Godsill S.J. and Vermaak J., A hybrid approach for online joint detection and tracking for multiple targets, Proc. IEEE Aerospace Conference (2005).

[17] Oksendal B., Stochastic Differential Equations, An Introduction with Applications, Sixth Edition Springer Heidelberg 2005.

[18] Snyder C., Bengtsson T., Bickel P. and Anderson J., Obstacles to High-dimensional Particle Filtering, Mon. Wea. Rev., Vol. 136 (2008) pp. 4629-4640.

[19] Stinis P., Conditional path sampling for stochastic differential equations by drift relaxation, (2010), arXiv:1006.2492v1.

[20] Stoer J, and Bulirsch R., Introduction to Numerical Analysis, Third Edition, Springer 2002.

[21] Vermaak J., Godsill S. and Perez P., Monte Carlo filtering for multi-target tracking and data association, IEEE Trans. Aero. Elect. Sys., 41(1) (2005) pp. 309-332.

[22] Vo B-N., Singh S. and Doucet A., Sequential Monte Carlo Methods for Multi-Target Filtering with Random Finite Sets, IEEE Trans. Aero. Elect. Sys., 41(4) (2005) pp. 1224-1245.

[23] Weare J., Particle filtering with path sampling and an application to a bimodal ocean current model, J. Comp. Phys. 228 (2009) pp. 4312-4331.