Deterministic resampling: Unbiased sampling to avoid sample impoverishment in particle filters

Tiancheng Li a,b,*, Tariq Pervez Sattar a, Shudong Sun b

a Center for Automated and Robotics NDT, London South Bank University, London SE1 0AA, UK
b School of Mechatronics, Northwestern Polytechnical University, Xi’an 710072, China

A novel resampling algorithm (called Deterministic Resampling) is proposed, which avoids uncensored discarding of low weighted particles thereby avoiding sample impoverishment. The diversity of particles is maintained by deterministically sampling support particles to improve the residual resampling. A proof is given that our approach can be strictly unbiased and maintains the original state density distribution. Additionally, it is practically simple to implement in low dimensional state space applications. The core idea behind our approach is that it is important to (re)sample based on both the weight of particles and their state values, especially when the sample size is small. Our approach, verified by simulations, indicates that estimation accuracy is better than traditional methods with an affordable computation burden.

1. Introduction

Particle filters (PF), utilizing the Sequential Monte Carlo (SMC) approach to implement Bayesian estimation, have the ability to carry multiple hypotheses and to relax the linearity and Gaussian assumptions. Since the importance resampling was proposed in [1,2], particle filters have been applied with great success to a variety of dynamical state estimation problems but they still suffer from some difficulties [3]. In particular, the widely used resampling methods that aim to solve the inherent sample degeneracy of Sequential Importance Sampling (SIS) can cause another notorious problem, sample impoverishment, which is the failure to maintain the diversity of particles. This paper is concerned with designing an efficient (re)sampling method, which has the ability to maintain sufficient diversity of particles, more specifically to solve sample impoverishment, without increasing the sample size [4].

To circumvent sample impoverishment caused by the resampling operation, one general characteristic of some of the solutions is to increase the state noise covariance or introduce an additional noise to the samples, namely jittering [5] or roughening [2]. Another solution is to construct advanced kernel, such as the Markov Chain Monte Carlo (MCMC) step [6], the regularized kernel, etc. For example, the Regularized Particle Filter (RPF) [7] convolves each particle with a diffusion kernel before resampling to prevent multiple copies of a few particles and therefore sample impoverishment. Recently, the Risk-Sensitive Particle Filter (RSPF) [8] constructs explicit risk functions from a general class of factorizable functions to mitigate sample impoverishment. The weight of particle is compared by stages and quasi-Monte Carlo method is used in [9] to overcome the sample impoverishment. There are also some hybrid methods, which integrate PF with the genetic algorithm [10], ant colony optimization [11], etc. to improve the sample diversity and filtering performance. In all these approaches, the basic Sequential

A R T I C L E   I N F O

Article history:
Received 23 September 2011
Received in revised form
21 November 2011
Accepted 20 December 2011
Available online 3 January 2012

Keywords:
Particle filter
Sample impoverishment
Residual resampling
Deterministic sampling
Unbiased sampling

A B S T R A C T

A novel resampling algorithm (called Deterministic Resampling) is proposed, which avoids uncensored discarding of low weighted particles thereby avoiding sample impoverishment. The diversity of particles is maintained by deterministically sampling support particles to improve the residual resampling. A proof is given that our approach can be strictly unbiased and maintains the original state density distribution. Additionally, it is practically simple to implement in low dimensional state space applications. The core idea behind our approach is that it is important to (re)sample based on both the weight of particles and their state values, especially when the sample size is small. Our approach, verified by simulations, indicates that estimation accuracy is better than traditional methods with an affordable computation burden.

© 2011 Elsevier B.V. All rights reserved.
Monte Carlo (SMC) is improved using some optimization techniques, which tend to put particles in more “important” regions. The question is how to define “important”, which makes a real difference.

To our knowledge, the Sequential Importance Sampling (SIS) and Resampling (SIR) [12] on which these improvements rely are focused on the likelihood/weight of samples but ignore their spatial distribution, i.e. state values. Put differently, classical Monte Carlo (sampling) procedures entirely ignore the state values of particles when forming the estimate. A procedure that weights points, regardless of their spatial distribution, ignores relevant information and would be serious with entirely arbitrary choice of the sampling distribution (see also [13]). Sample impoverishment is just a failure to guarantee the rationality of estimate in the state space, which will be more serious when the sample size is small. Our work is aimed at improving the performance of particle filters through allocating particles with respect to not only the weight of particles but also to their spatial distribution (state values). In our previous study [14,15], we merged adjacent particles in the state space to reduce the sample size in robot localization applications. In this paper, we propose a novel resampling method to solve the degeneracy problem without the side effect of causing sample impoverishment. We name it Deterministic Resampling, which is implemented on a theoretically simple partitioning of the particle set in the spatial domain.

The remainder of this paper is organized as follows. We begin in Section 2 with a brief review of the traditional resampling algorithms for particle filters. In Section 3, we give details of our solution, Deterministic Resampling (DR) algorithm. This is the core contribution of this paper. Simulations are presented in Section 4 before we conclude in Section 5.

2. Sequential Importance Resampling

In Bayesian terms, the PF approximates the state (density) using a set of random particles with associated non-negative weights:

$$P(x_t) \approx \sum_{i=1}^{N_t} w_{i}^{(t)} \delta_{x_t}(x_i), \quad \sum_{i=1}^{N_t} w_{i}^{(t)} = 1, \quad w_{i}^{(t)} \geq 0$$

where $(x_i^{(t)}, w_i^{(t)})_{i=1,2,...,N_t}$ represent, respectively, the states and weights of particles at time $t$ and $N_t$ is the total number of particles. $\delta_{x_t}(\cdot)$ denotes the delta-Dirac mass located in the $x$. The weights $w_i$ are chosen using the principle of Sequential Importance Sampling (SIS), which relies on

$$w_{i} \propto w_{i-1} \frac{p(y_t|x_t)p(x_t|x_{t-1})}{q(x_t|x_{t-1},y_t)}$$

where $q(\cdot)$ is a proposal importance density.

It is well known and verified that after a few iterations in the particle propagation process, the weight will concentrate on a few particles only and most particles will have negligible weight, namely sample degeneracy (see [5,12]) will result. This is one of the inherent faults of SIS. A very intuitive idea to counteract this problem is Resampling, which eliminates particles that have small weights and concentrates on particles with large weights as described by the equation:

$$P(x_t) = \sum_{i=1}^{N} \frac{N(i)}{N} \delta_{x_t}(x_i), \quad \sum_{i=1}^{N} N(i) = N$$

where $N(i)$ is the number of times the $i$th particle is sampled. Most resampling algorithms that have been proposed [2,16,17] are only based on the weight of particles that the expectation of the number of times each particle is sampled satisfies the equation:

$$E(N(i)|w_{i}^{1,N}) = Nw_{i}^{(t)}$$

The traditional Sequential Importance Resampling (SIR) algorithms, referred to as the only-weight-based resampling in this paper, can be depicted as in Fig. 1, in which the size of the circles represent the weight of particles. Resampling has the effect of “resetting” the particle system during its application thus partially solving the problem of the variance increasing exponentially with the iteration step $t$ in SIS. This resetting process affords a good opportunity for sample size adjustment [18] and particle redistribution [19] as well.

However, sample impoverishment may arise by very few particles having significant weight while most other particles with small weight being abandoned during the resampling process, as shown in Fig. 1 (see also [20,9]). Especially when the state noise variance is very small, the

Fig. 1. Only-weight-based resampling.
particles would remain in a very small region leading to erroneous state estimation.

**Remark 1.** Sample impoverishment and degeneracy are similar inherent defects of *only-weight-based sampling*. They both lead to effects that can be loosely referred to as premature convergence [10,21]. The only difference between them is that the computing resource converges prematurely by weights (that is degeneracy) in the former case and by samples (that is impoverishment) in the latter case.

In brief, the immediate reason for sample impoverishment is that some particles are discarded that have low weight but that may be potentially important from the perspective of spatial distribution. Alternative selection strategies to resampling include branching [20] and reallocation [12], but are also faced with the confliction between the need of diversity and the need of focus. In [12], a power (less than 1) of the weights is used to give preference to lower-weighted particles. Some other advanced resampling strategies have been proposed to modify the only-weight-based uncensored discarding, e.g., only adopt resampling when it is necessary in selective resampling [22], compare the weights of particles by stages [9], introduce deterministic sampling in residual resampling [16] and similarly in optimal resampling [23], and aided by means of roughening or jittering [2,5]. However, according to our knowledge, these resampling are still only based on particle weights and are unable to avoid lower-weighted particles discarding. In the following section, we use particle spatial distribution information (e.g., position) to overcome the side effect of the resampling method that only uses particle weights.

### 3. Deterministic Resampling

In this section, a scheme for Deterministic Resampling is developed. The state space of weighted particles is partitioned into variable-size countable units, named grids, based on their state values. Since the particles are generally spatially distributed in the state space depending on state dynamics, the spatial distribution therefore represents the diversity and uncertainty of the state estimate. Thus, one of the most important functions of the grids, as in our case, is to quantify the diversity of particles and the uncertainty of the state estimate. To facilitate the description, we make the following definitions:

**Definition 1.** The discrete partition of the state space is named grid. Subscript *p* denotes partition in all dimensions, the grid cell *gₚ* with *dₚ* particles in it is defined as

\[ gₚ = \{(x_p^{(k)}, w_p^{(k)}) | k = 1, 2, \ldots, dₚ\} \quad (5) \]

**Definition 2.** The number of particles distributed in a grid is defined as its particle density. Particles distributed in the same grid are considered to have the same contribution to the diversity of the particle set. If there is no particle in a grid, this grid is called an empty grid; otherwise it is a non-empty grid.

![Fig. 2. Self-fission partition of particles into variable size grids in the 2-D state space (Lstart/Lmin=4, x = 10).](image)

#### 3.1. Density-sensitive space partitioning of the particle set

Partition particles into variable-size grids using a stepwise “self-fission” method as described in Algorithm 1. An example partitioning is shown in Fig. 2 for a 2-dimensional system. The algorithm starts by performing boundary analysis of the scatter particles and choosing a starting grid size *Lstart*, which does not have more dimensions than the state space. All particles are first partitioned into grids of size *Lstart*. Then the particle density of each nonempty grid is detected: if it is more than a threshold *x*, then the grid is subdivided to half size in all dividable dimensions; otherwise, the grid is considered to be an independent one. The detection and division are repeated until the particle density of all grids is under the threshold *x* or the grid size is lower than the threshold bound *Lmin*. Here, *Lmin* is pre-selected to prevent division of grids into too small sizes. At the end of partitioning, the particles are stored in a tree structure, which is easy for data access and is in accordance with the spatial distribution of particles and their density.

**Algorithm 1.** Self-fission

**Input:** \( S_t = \{(x^{(i)}, w^{(i)})\}_{i=1}^{N_t}, L_{start}, L_{min}, x \)

**Output:** \( G_t = \{g_p | p \in \mathbb{N}\} \)

**Procedure:**

\[ G_0 = 0 \]
\[ \forall \quad p: \text{set } g_p = 0, \quad d_p = 0 \]
\[ \forall \quad (x^{(i)}, w^{(i)}): \text{for } i = 1 \to N_t \text{ do} \]
\[ \quad \text{if } x^{(i)} \text{ falls into the grid } g_p \text{ do} \]
\[ \quad \quad d_p = d_p + 1 \]
\[ \quad \quad g_p = g_p \cup (x^{(i)}, w^{(i)}): (x_p^{(d_p)}, w_p^{(d_p)}) \to (x_p^{(d_p)}, w_p^{(d_p)}) \]
\[ \quad \quad \text{end if} \]
\[ \text{end if} \]
\[ \forall \quad p: \text{if } d_p \geq x \text{ and } L \geq L_{min} \text{ do} \]
\[ \quad G_t = G_t \cup g_p \]
\[ \text{else} \]
\[ \quad G_t = G_t \cup \text{self-fission}(g_p, L_{start}/2, L_{min}, x) \]
\[ \text{end if} \]

In Algorithm 1, each particle does not take part in more than \( \min\{\log_2(L_{start}/j)/\log_2(L_{min}/j)\}\) grid subdivisions to indentify its attributed grid, where *j* denotes the **The notation \( \lceil \cdot \rceil \) denotes rounding down to the nearest integer.**
coordinate of the grid. The particle “falling into grid” process is the Numerical Inserting calculation in \( J \) independent dimensions. This indicates that each particle needs no more than \( (\min_{i} |\log_2(L_{start,i}/L_{min})|)^J \) times numerical inserting, where \( J \) is the dimensionality of the grid.

**Remark 2.** One limitation of the state space partitioning method is that it suffers from the general curse of high dimension problems so that spatial partitioning becomes computationally expensive and difficult. To overcome this, it will be necessary to partition particles only in partial (significant) dimensions.

**Remark 3.** It is the parameters \( L_{start}, L_{min} \), and \( \pi \) that determine how much the state space will be divided, and in turn how much the state space should be divided determines the parameters. As a rule of thumb, \( (\min_{i} |\log_2(L_{start,i}/L_{min})|) \leq 5 \), \( L_{start} \) is chosen according to the largest approximation error we can endure and \( \pi \) can be chosen according to the expected sample size \( N \). These parameters are tunable depending on the particular situation of the particle filter application.

### 3.2. Particle Space Sampling

As noted earlier, it is necessary to concentrate on important particles when sampling to solve the sample degeneracy problem. Traditional resampling approaches disregard particle spatial distribution information, discard particles in an uncensored way and thereby reduce diversity. We on the other hand introduce Deterministic Resampling method, which compensates some so-called support particles to improve Liu’s residual resampling process [16] (For proof of computational complexity \( O(N) \), \( N \) is the sample size, see [16,17]).

**Algorithm 2.** Deterministic Resampling

1 (partly the same as the residual resampling method)

1.1 All weights of particles are multiplied by the ensemble size \( N_t \) or a fixed sample size.

1.2 Then \( n_t \) copies (later called copy particles) are taken of each particle ((\( x_t^{(p)}, \hat{w}_t^{(p)} \)) and allocated equal new weights (specified in step 4) in which \( n_t \) is the integer part of \( N_t \hat{w}_t^{(p)} \)

\[
 n_t = \left\lfloor N_t \hat{w}_t^{(p)} \right\rfloor
\]

1.3 The integer parts of \( N_t \hat{w}_t^{(p)} \) are subtracted from \( N_t \hat{w}_t^{(p)} \), and then the rest of every particle weight is

\[
 \hat{w}_t^{(p)} = \frac{N_t \hat{w}_t^{(p)} - \left\lfloor N_t \hat{w}_t^{(p)} \right\rfloor}{N_t}
\]

As an option, at this point, particle spreading methods such as roughening [2], MCMC step [6], kernel function method [7], etc. could be used to spread the copy particles better in the state space.

2 Partition the particles set as Algorithm 1.

3 The residue of the particles in each previous partitioned grid are summed up separately as follows, namely Particle Space Sampling (PSS):

\[
g_p = (\hat{x}_t, p, \hat{w}_t, p) : \begin{cases} 
\hat{x}_t, p = \sum_{k=1}^{d_p} x_t^{(k)} \frac{w_t^{(k)}}{\hat{w}_t, p} \\
\hat{w}_t, p = \sum_{k=1}^{d_p} w_t^{(k)} 
\end{cases}
\]

s.t. \( \hat{w}_t, p > w_{min} \)

where \( \Rightarrow \) means summing up, \((\hat{x}_t, p, \hat{w}_t, p)\) is called support particle, s.t. means “under the condition that” and \( w_{min} \) is the smallest weight threshold below which weighted particles still need to be discarded.

4 Copy particle weights are specified as the equal division of the complement of the support particle weights:

\[
\left(1 - \sum_{p} \hat{w}_t, p\right) / \sum_{i} n_t
\]

**Theorem.** The computational complexity of Deterministic Resampling is \( O(N_t) \), \( N_t \) is the number of particles to resample.

**Proof.** The proof is straightforward. In step 1, 2 and 4 of Algorithm 2, the computational complexity of grid partitioning and particle coping is \( O(N_t) \), \( N_t \) is the number of particles. The computational complexity of PSS in step 3 is \( O(M) \), \( M \) is the support particles number satisfying \( M < N_t \) and only addition calculation is needed. Thus, the computational complexity of Deterministic Resampling is \( O(N_t) \).

Our approach differs from the residual resampling method at step 3 above because of PSS, visualized as shown in Fig. 3. Fig. 3 shows that particles with small weights and the rest of the big weight particles are summed up in PSS. Contrast this to residual resampling where particles are drawn randomly from this residual weight distribution during which small–weight particles are surely discarded. It makes much more sense to locally sum up residuals since this will maintain the previous state density distribution. There is at least one particle that survives in each grid (when \( w_{min} = 0 \) enabling the particle distribution to cover the state space more widely. From another perspective, the reason to do so is to reserve the possibility of small weighted regions/particles performing better in the future. This is particularly necessary when the state noise variance is comparatively small (then the particles would remain in a very small area) and the sample size is small. In summary, our approach uses the fact that the “importance” in resampling is not limited to the weights of particles but also to their spatial distribution, i.e. state values. This is a distinguishing feature over the traditional resampling.

**Proposition.** Set \( w_{min} = 0 \), then Deterministic Resampling is strictly unbiased.

**Proof.** Dividing the big weight particles into copy particles and support particles will not change the particle’s distribution or its weight. The only change to the state
density distribution in the process of Deterministic Resampling is the local particle summing up in PSS, which indeed performs the local moving of the particles. If \( w_{\min} = 0 \), then no particle (and weight) will be discarded. The mean of the support particles distribution will not change at all. Deterministic Resampling is strictly unbiased.

**Remark 4.** Proposition 1 guarantees global unbiasedness, but local bias happens in the grid summing up operation of PSS and when \( w_{\min} \neq 0 \) (some particles may be discarded). This local bias can be further tested and hence we can gain an insight into how to control it in our approach as described in the next sub-section.

### 3.3. Non-parametric hypothesis testing

In statistics, the two-sample Kolmogorov–Smirnov statistic [24] quantifies a distance between the empirical distribution functions of two samples:

\[
D_{n_1,n_2} = \sup_{-\infty < x < +\infty} |F_{1,n_1}(x) - F_{2,n_2}(x)|,
\]

where \( \sup S \) is the supremum of set \( S \), and \( F_{1,n_1} \) and \( F_{2,n_2} \) are the empirical distribution functions of the first and the second sample, respectively. The empirical distribution function \( F_n \) for \( n \) independent and identically distributed observations \( X_i \) is defined as

\[
F_n(x) = \frac{1}{n} \sum_{i=1}^{n} (I_{X_i} \leq x)
\]

where \( I_{X_i} \leq x \) is the indicator function, if \( X_i \leq x \), it is equal to 1; otherwise, and it is equal to 0.

The null distribution of this statistic is calculated under the null hypothesis that the samples are drawn from the same distribution. According to the two-sample K–S test, the null hypothesis is rejected at level \( \alpha \) if

\[
\sqrt{\frac{n_1 n_2}{n_1 + n_2}} D_{n_1,n_2} > K_{\alpha},
\]

where \( K_{\alpha} \) is found from

\[
\Pr(K \leq K_{\alpha}) = 1 - \alpha
\]

The cumulative distribution function of \( K \) is given by

\[
K(x) = \begin{cases} 
0, & x \leq 0 \\
2 \sum_{k=1}^{\infty} (-1)^{k-1} e^{-2k^2 x^2}, & x > 0
\end{cases}
\]

In practice, the values of \( K_{\alpha} \) for typical values of \( \alpha \) are readily available in standard statistical tables [24], e.g., for \( \alpha = 0.05 \), \( K_{\alpha} = 1.36 \); \( \alpha = 0.01 \), \( K_{\alpha} = 1.63 \).

Note that the two-sample test checks whether the two data samples are subjected to the same distribution. It does not specify what that common distribution is. The distributions considered under the null hypothesis are continuous but are otherwise unrestricted, which is suited to our approach if we consider all the particles with respect to their spatial grid partition. The only difference between the distributions of particles before (assumed as \( F_{1,n_1} \)) and after (\( F_{2,n_2} \)) Deterministic Resampling is just the support particles. The weight sum in each grid is unchanged, and so the greatest distance \( D_{n_1,n_2} \) is the largest weight \( \hat{w}_{\max} \) of support particles. Thus, we have the following restriction if the null hypothesis holds:

\[
\hat{w}_{\max} < K_{\alpha} \sqrt{\frac{n_1 n_2}{n_1 + n_2}}
\]

e.g., for a significance level \( \alpha = 0.05 \) and \( n_1 = n_2 = 200 \), \( K_{\alpha} = 1.36 \), then \( \hat{w}_{\max} < 0.136 \). This is a necessary and sufficient condition for Deterministic Resampling to satisfy the two-sample K–S test. That means at a certain confidence level, we believe Deterministic Resampling operation has maintained the original state density distribution.

### 4. Simulations

#### 4.1. Example 1: one-dimensional financial model

For the sake of evaluating the effectiveness of our approach, we first consider the popular univariate non-stationary growth model with state dynamics given by Eq. (16) and measurement by Eq. (17):

\[
x_t = \frac{x_{t-1}}{2} + \frac{25x_{t-1}}{1+x_{t-1}^2} + 8 \cos(1.2t) + \nu_t,
\]

\[
\nu_t \sim \text{Normal}(0, \sigma^2)
\]

\[
t = 1, 2, \ldots, T
\]

The result of consistency test is depicted in Fig. 4, which shows that the two data samples are drawn from the same distribution.
\( y_t = \frac{x_t^2}{20} + w_t, \)

where Gaussian distribution noise \( v_t \sim \mathcal{N}(0, 10), w_t \sim \mathcal{N}(0, 1). \)

We use Root Mean Square Error (RMSE) to evaluate the estimation accuracy, which is defined as

\[
\text{RMSE} = \left[ \frac{1}{T} \sum_{t=1}^{T} (x_t - \hat{x}_t)^2 \right]^{1/2}.
\]

where \( \hat{x} \) is the mean of particle estimates, \( T \) is the number of iterations, in time steps. A large \( T = 10,000 \) is chosen for more trustworthy results unless otherwise stated in our simulation.

Our approach adopting Deterministic Resampling (DR) in SIR particle filter is named DR-SIR Particle filter. In this simulation, six other basic particle filters are also implemented for comparison. They are Selective resampling PF [22], SIR PF (with residual resampling), Bootstrap PF (with roughening-improved multinomial resampling), Auxiliary Particle filter (APF), RPF, with parameter set as follows.

Selective resampling particle filter resamples only when the variance of the non-normalized weights is superior to a pre-specified threshold. A simple implementation is using the so-called Effective Sample Size criterion, given by

\[
\text{ESS} = \frac{1}{N_t} \left( \sum_{t=1}^{N_t} (w_t)^2 \right).
\]

The ESS takes values between 1 and \( N_t \) and the resampling is implemented only when it is below a threshold \( N_T \), typically \( N_T = N_t/2 \), \( N_t \) is the sample size. This is the case in our approach.

**Fig. 4.** Average RMS error when using different starting sample size.

**Table 1**

<table>
<thead>
<tr>
<th>Filters</th>
<th>Selective PF</th>
<th>SIR PF</th>
<th>Bootstrap PF</th>
<th>RPF</th>
<th>APF</th>
<th>DR-SIR PF</th>
</tr>
</thead>
<tbody>
<tr>
<td>Processing time/s</td>
<td>1.5231</td>
<td>1.6607</td>
<td>3.9236</td>
<td>546.096</td>
<td>1.7205</td>
<td>5.703</td>
</tr>
</tbody>
</table>
To implement roughening in Bootstrap PF and our approaches, a random zero mean Gaussian noise is added to each resampled particle, with a variance determined on the largest state difference between different samples:

$$0.2 \times \frac{\max(x_i^t) - \min(x_i^t)}{N}$$

RPF convolves each particle with a diffusion kernel density $K_h(x)$ before resampling as

$$p(x_t|y_{1:t}) = \sum_{i=1}^{N_s} w^i_t K_h(x_t - x_i^t)$$

where bandwidth $h > 0$, and when the underlying density is Gaussian with unit covariance matrix, the optimal choice for the bandwidth:

$$h_{opt} = \left(\frac{8}{c}(d+4)(2\sqrt{\pi})^d N_s\right)^{1/d+4}$$

where $c$ is the volume of the unit sphere of $\mathbb{R}^d$, $N$ is the sample size. $h_{opt}$ denoted above is called the Epanechnikov kernel. In our case, $d=1$, $c=2$.

In our approach, we choose $L_{start}=1$, $L_{min}=0.1$, $\alpha=8$. Furthermore, we set the same starting sample size $N_s$ for all PFs at the beginning of simulations. The sample size in our approach is floating as particle spatial distribution is changeable.

Figs. 4 and 5 show the average RMSE and the average sample size in filtering while using different starting number of particles from 10 to 100. It is clear that the estimation accuracy of DR-SIR PF is the best of all even while using smaller sample size, especially when $N_s < 55$. This is because that it is easier to appear insufficient diversity of particles when the starting number of particles is small, and our approach will benefit more from using the support particles.

The average sample size, RMSE and processing time for 10,000 iterations when using $N_s=50$ is given in Table 1. The results show that, the number of particles adaptively changes in our DR-SIR PF as we set $w_{min}=0.1/N$ to delete some support particles whose weight is smaller than 10% of average. However, Table 1 also shows that our approach has cost a little more processing time compared with Basic SIR, APF and bootstrap PF but less than RPF.

4.2. Example 2: multi-dimensional ballistic object tracking

In this section we apply the DR-SIR PF to estimate state vectors with multiple dimensions by applying it to a vertically falling body example, which is well studied in several filters, such as [25] and some of its references. The altitude $x_1(t)$, velocity $x_2(t)$ and constant ballistic coefficient $x_3(t)$ of a vertically falling body is estimated at discrete points $k$ by measuring the range $r(k)$ of the body using radar in the presence of white, uncorrelated noise $w(k)$. The geometry of the situation is shown in Fig. 6.

The continuous time dynamics of the system are

$$\dot{x}_1(t) = -x_2(t)$$

$$\dot{x}_2(t) = -e^{-5x_1(t)}x_2(t)^2x_3(t)$$

$$\dot{x}_3(t) = 0$$

In accordance with [25], a fourth order Runge-Kutta method with 64 steps taken between each observation is used to implement the dynamics integration and no process noise is introduced. The discrete observation
The model is
\[
y(k) = r(k) + w(k) = \sqrt{M^2 + (x_1(k) - H)^2} + w(k).
\]
with parameter setting
\[
M = 10^5 \text{ ft}, \quad H = 10^5 \text{ ft}, \quad \gamma = 5 \times 10^{-5},
\]
\[
\mathbb{E}[w(k)^2] = 10^4 \text{ ft}^2.
\]

The initial state of the system is
\[
\begin{align*}
x_1(0) &= 3 \times 10^5 \text{ ft}, \quad x_2(0) = 2 \times 10^5 \text{ ft/s}, \\
x_3(0) &= 10^{-3} \text{ ft}^{-1},
\end{align*}
\]
and the initial estimates of these state are set as
\[
\begin{align*}
\hat{x}_1(0) &= 3 \times 10^5 \text{ ft}, \quad \hat{x}_2(0) = 2 \times 10^5 \text{ ft/s}, \\
\hat{x}_3(0) &= 3 \times 10^{-5} \text{ ft}^{-1},
\end{align*}
\]
with initial covariance matrix
\[
\hat{P}(0) = \begin{bmatrix} 10^6 & 0 & 0 \\ 0 & 4 \times 10^6 & 0 \\ 0 & 0 & 10^{-4} \end{bmatrix}
\]

In our approach, the particle set is partitioned in the altitude and velocity 2-dimensional space using a Minimum Bounding Rectangle (MBR) method. The starting grid size is set as
\[
L_{\text{start}} = \frac{\max(x_1(t)) - \min(x_1(t))}{10} \quad \frac{\max(x_2(t)) - \min(x_2(t))}{10}
\]
and \(L_{\text{min}} = L_{\text{start}}/4\) (no more than twice subdivision), \(\varepsilon = 8\), \(w_{\text{min}} = 0.1/N_s\). Comparison particle filters are SIR PF (residual resampling) and bootstrap PF (with roughening-improved multinomial resampling, satisfying (20)), all with starting sample size \(N_s = 100\). These filters are implemented in discrete time and the observation is taken once per second.

![Fig. 8. Absolute estimate error of altitude, velocity and ballistic coefficient estimation of particle filters using starting sample size 100.](image)

<table>
<thead>
<tr>
<th>Table 2</th>
<th>Processing time of particle filters ((N_s=100)).</th>
</tr>
</thead>
<tbody>
<tr>
<td>Particle filters</td>
<td>SIR PF</td>
</tr>
<tr>
<td>Processing time/s</td>
<td>4.908</td>
</tr>
</tbody>
</table>

The results in Figs. 7 and 8 show that Deterministic Resampling is efficient in multiple dimensional cases and obtains better estimate accuracy than the basic resampling and roughing methods. Table 2 shows that our approach has not increased processing time but on the contrary, DR-SIR PF saves time since it reduces the sample size during the filtering.

5. Conclusion

A novel resampling algorithm called Deterministic Resampling for particle filtering is presented. Deterministic Resampling can strictly maintain the original state density and hence maintain the diversity of particles without the side effect of sample impoverishment. Practically, this will be quite useful when a small sample size is required in the particle filter. Simulations of lower dimensional problems show that our Deterministic Resampling can get better estimation accuracy with an affordable computation burden.

Allocating particles more sensibly is still an art in particle filtering. There is room for further study to improve the performance of particle filters from particle distribution optimization, especially to cope with large dimension problems.

Acknowledgments

The authors would like to thank the National Natural Science Foundation of China (Grant no. 51075337) for the
partial support of this research and thank Simon J. Godsill with University of Cambridge for his kind comments on this paper. Tiancheng Li would like to thank China Scholarship Council for its support of his research in the United Kingdom.

References