New clone particle swarm optimization-based particle filter algorithm and its application

Yuming Bo, Zhimin Chen, Jie Zhang, Jianliang Zhu
School of Automation, Nanjing University of Science and Technology, Nanjing, 210094, China
Email: 13675133070@189.com

Received: 7 Jun. 2012; Revised 15 Sep. 2012; Accepted 22 Sep. 2012

Abstract: A new particle filter algorithm based on new Clone particle swarm optimization (NCP-PSO-PF) is presented in this paper in order to solve the problem of low precision and complicated calculation of particle filter based on particle swarm optimization algorithm (PSO-PF). The algorithm enables the particles to fit the environment better and then reach the goal of global optimization through orthogonal initialization, clonal selection and local searching of self-learning. Finally different models are used for simulation experiment and the results indicate that this new algorithm improves the operation speed and precision of practical engineering application.

Keywords: particle filter; particle swarm optimization; Clone; orthogonal.

1 Introduction

Particle filtering is a sequential Monte Carlo methodology where the basic idea is the recursive computation of relevant probability distributions using the concepts of importance sampling and approximation of probability distributions with discrete random measures [1]. It is widely applied to positioning and navigation of non-Gaussian noise and non-linear system, fault diagnosis, image segmentation algorithm [2], target tracking and mode identification field [3] as its state function and observation function has no non-linear hypothesis. Nevertheless, PF may confront with the problem of weight degradation [4] which if solved by resampling method may result unavoidable particle impoverishment [5-6]. As a result, the filtering precision is influenced.

PF based on Intelligent Optimization Algorithms conduces to significant improvement of particle degradation in PF and great enhancement of precision [7]. Particle swarm optimization-based PF (PSO-PF) is a typical representative of Intelligent Optimization Algorithm which introduces PSO algorithm into PF. Through introduction of the latest measurement value to the sample distribution, along with the utilization of PF algorithm for sampling process optimization and constantly update of particle speed, the sample distribution is inclined to move to the area with higher posterior probability [8]. PSO-PF improves the particle degradation of PF and is easier for actualization. Unfortunately PSO-PF is a process of iterative optimization which will prolong the calculation time because of the high iterative frequency [9]. Moreover, PSO-PF may be easily trapped into local optimization, influencing the precision and stability of practical engineering application [10].

A new particle filter algorithm based on new Clone particle swarm optimization (NCP-PSO-PF) is suggested in this paper. This algorithm utilizes orthogonal strategy and clone particle swarm algorithm for optimization, according enhancing the quality and diversity of particles. The experimental results prove that NCP-PSO-PF improves the efficiency of particle filter.

2 PSO-PF algorithm

2.1 Basic PSO algorithm

PSO is an intelligence optimization algorithm imitating birds’ clustering movement. It is widely applied to target tracking, positioning and navigation, mode identification etc. by virtue of its advantages of simple concept, ease in actualization, fewer parameters, and effectiveness in solving complicated optimization and so on. PSO algorithm can be expressed as follows [11]: to randomly initialize a particle swarm whose number is m and dimension is n, in which the ith particle’s position is \( X_i = (x_{i1}, x_{i2}, \ldots, x_{in}) \) and its speed is \( V_i = (v_{i1}, v_{i2}, \ldots, v_{in}) \). During each iteration, the
particles can renew their own speed and position through partial extremum and global extremum so as to reach optimization. The update formula is:

\[ V_t = w \times V_{t-1} + c_1 \times \text{Rand}() \times (P_t - X_t) + c_2 \times \text{Rand}() \times (G - X_t) \]  

(1)

\[ X_{t+1} = X_t + V_t \]  

(2)

where Rand is a random number within interval (0,1), \( w \) is the inertia coefficient (A larger inertia coefficient means stronger global search ability, while a smaller one stands for stronger local search ability), \( c_1 \) and \( c_2 \) are learning factors.

2.2 Description of standard PSO-PF algorithm

The importance sampling process of conventional PF is suboptimal, whereas the incorporation of PSO algorithm will optimize the sampling process of PF, allow the weight of particle sets are more inclined to high likelihood region, accordingly solving the problem of particle impoverishment, and conducing to reduction of particle numbers required by PF. PSO method is fused with PF and the key lies in utilizing the optimal state value \( P_{\text{pbest}} \) experienced by the particles and the state value \( P_{\text{gbest}} \) of the maximum particle with the greatest objective function value, and updating the speed and position of each particle on a real-time base through equation (5) and (6), accordingly forcing the particles to be closer to the real state.

\[ V_t = |\text{Rand} n| \times (P_{\text{pbest}} - X_{t-1}) + |\text{rand} n| \times (P_{\text{gbest}} - X_{t-1}) \]  

(3)

\[ X_{t+1} = X_{t-1} + V_{t+1} \]  

(4)

Where \(|\text{Rand} n| \) and \(|\text{rand} n| \) are positive Gaussian distribution random numbers[12].

3 principle of NCPSO-PF

Based on the enlightenments brought by biological immune system and clone mechanism, an orthogonal immune clone partial swarm algorithm is presented here for function solution optimization. First of all, let \( A(k) = \{a_1(k), a_2(k), \cdots, a_{\text{size}}(k)\} \) denotes the \( k \) the generation particle population, \( \text{size} \) denotes the scale of particle swarm. Each particle may actualize particle swarm update through clone selection and self-learning. Corresponding operators are given as follows:

1) Immune clone operator For any antibody \( a_i = (a_{i,1}, a_{i,2}, \cdots, a_{i,n}) \) in particle swarm \( A(k) \) of \( k \) th generation, \( i = 1, 2, \cdots, \text{size} \) performs clone operation with scale \( q_i \), becoming clone \( q_i \) of antibody \( a_i \). The \( q_i \) clone of antibody \( a_i \) is defined as \( \overline{a_i} = I_i \times a_i \), wherein \( \overline{a_i} \) is the antibody group of \( a_i \) after cloning, and \( I_i \) is the \( q_i \)-dimensional vector with element 1.

\[ q_i(k) = \text{Int} \left( n_i \times \frac{f(a_i)}{\sum_{j=1}^{\text{size}} f(a_j)} \right), i = 1, 2, \cdots, \text{size} \]  

(5)

where \( n_i \) denotes population clone scale, \( n_i > \text{size} \), \( \text{Int}(\cdot) \) means rounding up. \( f(a_i) \) is the affinity of antibody \( a_i \), while \( \gamma \) reflects the affinity of antibody \( a_i \) with other particles and is defined as:

\[ \gamma = \min\{D_{ij}\} = \min\{\exp(\|a_i - a_j\|)\}, \]  

(6)

\( i \neq j \); \( i, j = 1, 2, \cdots, \text{size} \)

in which \( \| \cdot \| \) denotes any of the norms, and in this paper Euclidean distance is employed. \( D \) is a symmetric matrix reflecting the diversity of population.

2) Immune gene operation. Immune gene operation mainly consists of genetic recombination and genetic mutation. In order to facilitate the communication of useful information among individuals and to enhance the uniformity of progeny individuals, orthogonal recombination algorithm is employed here to recombine the cloned individuals by probability.

(1) Genetic recombination. Given the genetic recombination operator \( P^r \), and \( C(k) = \{c^1(k), c^2(k), \cdots, c^{\text{size}}(k)\} \) as the population after recombination.

Taking two parental particles \( b_1 = (b_{1,1}, b_{1,2}, \cdots, b_{1,n}) \) and \( b_2 = (b_{2,1}, b_{2,2}, \cdots, b_{2,n}) \) into consideration, a sub-
space \([l,u]\) is determined in accordance with the following equation:

\[
\begin{align*}
[l] &= \min(b_{1,1}, b_{2,1}, \ldots, b_{n,1}) \\
[u] &= \max(b_{1,2}, b_{2,2}, \ldots, b_{n,2})
\end{align*}
\]  

(7)

First of all each of the region in Space \([l,u]\) is quantified into \(Q\) levels, and the difference between two neighboring levels are always the same. \(\beta_i = (\beta_{i,1}, \beta_{i,2}, \ldots, \beta_{i,Q})\) is defined as follows:

\[
\begin{align*}
\min(l_i,u_i), & \quad j = 1 \\
\min(l_i,u_i) + (j-1) \times \frac{|l_i-u_i|}{Q-1}, & \quad 2 \leq j \leq Q-1 \\
\max(l_i,u_i), & \quad j = Q
\end{align*}
\]  

(8)

\(F-1\) integer will be generated, \(k_1, k_2, \ldots, k_{F-1}\), and \(1 < k_1 < k_2 < \cdots < k_{F-1} < n\) is met. Then for any particles \(x = (x_1, x_2, \ldots, x_n)\), the following \(F\) factors will be generated.

\[
\begin{align*}
f_i(1) &= (\beta_{k_i+1,1}, \beta_{k_i+2,1}, \ldots, \beta_{k_i,1}) \\
f_i(2) &= (\beta_{k_i+1,2}, \beta_{k_i+2,2}, \ldots, \beta_{k_i,2}) \\
&\quad \vdots \\
f_i(Q) &= (\beta_{k_i+1,Q}, \beta_{k_i+2,Q}, \ldots, \beta_{k_i,Q})
\end{align*}
\]  

(10)

The orthogonal matrix \(L_m(Q^F) = [r_{i,j}]M_2 \times F\) is used to generate the following \(M\) particles:

\[
\begin{align*}
(f_1(r_{1,1}), f_2(r_{1,2}), \ldots, f_F(r_{1,F})) \\
(f_1(r_{2,1}), f_2(r_{2,2}), \ldots, f_F(r_{2,F})) \\
&\quad \vdots \\
(f_1(r_{M,1}), f_2(r_{M,2}), \ldots, f_F(r_{M,F}))
\end{align*}
\]  

(11)

Finally, an optimal particle is selected from \(M\) particles as the clone progeny subject. Given \(Q = 3, F = 4, M = 9\) , and orthogonal matrix is marked as \(L_9(3^4)\).

(2) Genetic mutation. AEA mutation strategy is used, letting \(D(k) = \{d^0_1(k), d^0_2(k), \ldots, d^0_{\text{SIZE}}(k)\}\) the population after mutation, that is,

\[
d^i_j = c^i_j - \frac{\text{fix}(c^i_j \times 10^l \mod (10 - md(10)))}{10^l} 
\]  

\(i = 1, 2, \ldots, \text{SIZE} ; \quad j = 1, 2, \ldots, q\),

where \(t\) is the random number of 0~15, \(\text{fix}(\cdot)\) is rounded down, \(md(10)\) a random integer less than 10, whereas the probability of mutation is usually very small.

(3) Clone selection operator. Different from the selection operation in evolutionary operation, clone selection operator is to select the excellent individual from particle clone and to form new populations. Given \(\forall i = 1, 2, \ldots, \text{SIZE}\), then:

\[
d'_i(k) = \{d'_j(k) \mid \min f(d'_j), j = 1, 2, \ldots, q\}
\]  

For probability \(p^k(d'_i(k) \cup a_i(k) \rightarrow a_i(k+1))\),

\[
p^k(a_i(k+1) = d'_i(k) =
\begin{cases}
1, & f(a_i(k)) > f(d'_i(k)) \\
\exp \left( - \frac{f(a_i(k)) - f(d'_i(k))}{a} \right), & f(a_i(k)) \leq f(d'_i(k)) \text{ and} \\
0, & f(a_i(k)) \leq f(d'_i(k)) \text{ and} \\
& a_i(k) \text{ is the best antibody.}
\end{cases}
\]  

(14)

(4) Self-learning operator. During the clone operation of particle swarm, it searches in a relatively large range, to avoid the loss of optimal solution, self-learning local searching of optimal individuals after clone selection is performed in this paper. For the optimal individual \(\hat{a}(k) = (\hat{a}_1(k), \hat{a}_2(k), \ldots, \hat{a}_n(k))\), first of all a self-learning population \(L\) is generated with size \(L_{\text{SIZE}}\), then \(L\):

\[
L = \begin{cases}
\hat{a}(k), & i = 1 \\
\text{New}_i, & \text{others}
\end{cases}
\]  

(15)

wherein \(\text{New}_i = (e_{i,1}, e_{i,2}, \ldots, e_{i,n})\) is generated according to the following equation:

\[
e_{i,k} = \hat{a}_{i,k} \cdot U(1 - sradius, 1 + sradius)
\]  

(16)

\(k = 1, 2, \ldots, n\), in which \(sradius \in [0, 1]\) denotes searching semi-diameter, \(U(1 - sradius, 1 + sradius)\) denotes the random
number in interval \([1 - sradius, 1 + sradius]\). Given \(Min_i = (m_1, m_2, \ldots, m_n)\) the individual with optimal fitness in \(L\), e.g. \(Min_i \in L\), and for any individual \(L_i \in L\) in a small population, we have \(f(L_i) \geq f(Min_i)\), if \(L_i\) meets \(f(L_i) \leq f(Min_i)\), then it is a winner and will be preserved in the self-learning population, or otherwise a loser that should die. The eliminated individuals will be occupied by new individuals \(New_i = (e_{i,1}, e_{i,2}, \ldots, e_{i,n})\) generated by \(Min_i\), wherein \(P_0\) is the probability given.

\[
e_{i,k} = \begin{cases} 
    m_i + d \times (m_i - L_{i,k}), & k = 1, 2, \ldots, n, \text{ if } U(0,1) < P_0 \\
    (m_i, m_2, \ldots, m_{n-1}, m_1, m_{n-1}, \ldots), & \text{ else} 
\end{cases} \tag{17}
\]

where \(d\) is the random variable within range \([0,1]\). Finally, Gaussian mutation of the individuals in self-learning population is carried out, and the learned individual is replaced by optimal individual.

4 Process of NCPSO-PF

(1) Take \(N\) particles \(\{x_{i0}, i = 1, \ldots, N\}\) as samples from importance function at the initial time. The importance density function is expressed in equation(18):

\[
x_i^j = q(x_i^j | x_{i-1}^j, z_k) = p(x_i^j | x_{i-1}^j) \tag{18}
\]

Giving the fitness function:

\[
Y = \exp\left[-\frac{1}{2R_k}(z_{New} - z_{Pred})\right] \tag{19}
\]

Where \(z_{New}\) is the latest observed value, \(z_{Pred}\) is the predictive observed value.

(2) calculate the importance value:

\[
w_i^j = w_{i-1}^j p(z_k | x_{i-1}^j) = w_{i-1}^j p(z_k | x_{i}^j) \tag{20}
\]

(3) Updating the speed of particle swarm \(A(k)\), then we have \(A'(k)\).

(4) Replace \(T\%\) relatively undesirable individuals with \(T\%\) desirable ones in \(A(k)\), and \(A(k) \leftarrow A'(k)\).

(5) Perform clone operation against particle swarm \(A(k)\), including clone recombination and clone mutation, then \(\overline{A}(k)\).

(6) Perform immune gene operation against particle swarm \(\overline{A}(k)\), including clone recombination and clone mutation, then \(D\).

(7) Perform clone selection against \(D\), then \(A^*(k)\).

(8) Force the self-learning operator to take effect on the optimal individual of \(A^*(k)\) and then to upgrade the optimal individual.

(9) Assuming \(x_n\) is the individual extreme value of the current particle, \(pb_i\) the optimal solution at \(t\) time of particle \(i\). \(gbest_k\) is the global optimal solution at \(t\) time. Compare their fitness, update \(pb\) and \(pg\):

\[
pb_i = \begin{cases} p_i, & Y(x_i) < Y(pb_i) \\
x_i, & Y(x_i) > Y(pb_i) \end{cases} \tag{21}
\]

\[
pg_k \in \{x_1^k, x_2^k, x_3^k, \ldots, x_N^k \} \big| Y(x) \tag{22}
\]

(10) When the optimal value of particle complies with the initially-set threshold value \(\varepsilon\) or algorithm reached maximum iteration times \(\lambda\), it is indicated that the particles have been already distributed around the true values. By now particle optimization should be stopped. Else jump to step(3).

(11) Calculate the importance weight of the particles after optimization and perform normalization:

\[
w_k^j = w_{k-1}^j \frac{p(y_k^j | x_k^j) p(x_k^j | x_{k-1}^j)}{q(x_k^j | x_{k-1}^j, y_k)} \tag{23}
\]

\[
w_k^j = w_k^j \sum_{i=1}^N w_k^i \tag{24}
\]

(12) State output:

\[
x = \sum_{i=1}^N w_k^i x_k^i \tag{25}
\]

5 Simulation experiment

5.1 Univariate nonstationary growth model
Choosing a univariate nonstationary growth model (UNGM), and the process model and measurement model of the simulated objects are given as follows:

\[
x(t) = 0.5x(t-1) + \frac{25x(t-1)}{1+[(x(t-1))^2]} + 8\cos[1.2(t-1)] + w(t) \tag{26}
\]

\[
z(t) = \frac{x(t)^2}{20} + v(t) \tag{27}
\]

In which, \(w(t)\) and \(v(t)\) are zero-mean Gaussian noise. Since this system is highly non-linear and the likelihood function presents bimodal, it will be difficult for traditional filtering methods to deal with this system. By using PF, PSO-PF, NCPSO-PF, state estimation and tracking of this non-linear system are performed, and the formula of root-mean-square error is

\[
RMSE = \sqrt{\frac{1}{T} \sum_{t=1}^{T} (\hat{x}_t - x_t)^2} \tag{28}
\]

Giving the particle number \(N = 100\), and process noise variance \(Q = 10\) or \(20\), measurement noise variance \(R = 1\), initially-set threshold value is \(\epsilon\) and maximum iteration times is \(\lambda\), simulation is conducted. After single-step simulation by using PF, PSO-PF and NCPSO-PF, the simulation result is presented in figure 5.1~5.4. After 500 times of Monte-Carlo simulation, result is given in Table 1.
Tab.1 comparison of simulation parameters by UNGM model

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Algorithm</th>
<th>RMSE</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PF</td>
<td>2.8652</td>
<td>0.3528</td>
</tr>
<tr>
<td>N=100,Q=10,R=1</td>
<td>PSO-PF</td>
<td>1.7063</td>
<td>0.5281</td>
</tr>
<tr>
<td></td>
<td>NCP-SO-PF</td>
<td>1.4682</td>
<td>0.4298</td>
</tr>
<tr>
<td></td>
<td>PF</td>
<td>5.7628</td>
<td>0.3401</td>
</tr>
<tr>
<td>N=100,Q=20,R=1</td>
<td>PSO-PF</td>
<td>2.8860</td>
<td>0.5328</td>
</tr>
<tr>
<td></td>
<td>NCP-SO-PF</td>
<td>2.5645</td>
<td>0.4582</td>
</tr>
</tbody>
</table>

From the experimental result, regardless in precision or speed, NCP-SO-PF has been improved by contrast with PF and PSO-PF. This can be ascribed to the fact that the orthogonal cross strategy can reinforce the uniformity of progeny individuals, avoiding loss of optimal solution in the individual neighborhood. At the same time the precocity of particles can be eliminated, and thereby the speed and precision of the algorithm is enhanced. In the above calculation, with respect to both precision and time, NCP-SO-PF has the most desirable performance.

5.2 simulation model of white noise deconvolution filter

In practical engineering, the estimation of optimal output of deconvolution white noise is a common-seen problem. In this paper, PF, PSO-PF and NCP-SO-PF are used to forecast the white noise of a randomly-selected system that is described as follows:

\[ x(t+1) = \begin{bmatrix} 1 & 0 \\ 0.3 & 0.5 \end{bmatrix} x(t) + \begin{bmatrix} -1 \\ 2 \end{bmatrix} w(t) \]  

\[ z(t) = [1 \ 1]x(t) + v(t) \]  

in which \( w(t) = b(t)g(t) \) is Gaussian white noise, \( b(t) \) is the white noises with values 1 and 2. The probability for value acquisition is:

\[ P(b(t)) = \begin{cases} \lambda, & b(t) = 1 \\ 1 - \lambda, & b(t) = 0 \end{cases} \]

\( \lambda \) is the probability of \( w(t) \) with a non-zero value. \( g(t) \) is the Gaussian white noise independent of \( b(t) \) with mean value 0 and variance \( \sigma_g^2 \). In the simulation, \( \sigma_x^2 = 0.1, \sigma_g^2 = 0.1 \), and the simulation results are shown as following, in which the vertical axis of the solid line end point, and the vertical axis of round dot represents the estimated value.

Fig 5.5: PF estimate  
Fig 5.6: PSO-PF estimate  
Fig 5.7: NCP-SO-PF estimate

Making a comparison between the simulation results shown in the figure, the estimation of white noise by PF displays the lowest precision, and it fails to well predict the value of most of the state points, while the precision of two particle filters by PSO is improved significantly. As for NCP-SO-PF, since it successfully preserves the diversity of population, it has the highest precision and can well estimate the values of white noise.

6 Conclusion

This paper presents a new particle filter algorithm based on clone particle swarm which can enhance the convergence speed and guarantee the population diversity while enlarge the local searching range, thereby improving the quality
of the particles. The experiment results indicate that the algorithm in this paper conduces to enhancement of particle filter precision and speed as well as good robust, thus it is of great application value in practical engineering.

Acknowledgements

This paper was supported by the National Nature Science Foundation of China (No. 61104196), the National defense pre-research fund of China (No. 40405020201), the Specialized Research Fund for the Doctoral Program of Higher Education of China (No. 200802881017), the Special Plan of Independent research of Nanjing University of Science and Technology of China (No. 2010ZYTS051).

References