We are IntechOpen, the world’s leading publisher of Open Access books
Built by scientists, for scientists

3,700
Open access books available

108,500
International authors and editors

1.7 M
Downloads

154
Countries delivered to

TOP 1%
Our authors are among the most cited scientists

12.2%
Contributors from top 500 universities

WEB OF SCIENCE™
Selection of our books indexed in the Book Citation Index in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com
Individual Parameter Selection Strategy for Particle Swarm Optimization

Xingjuan Cai, Zhihua Cui, Jianchao Zeng and Ying Tan
Division of System Simulation and Computer Application, Taiyuan University of Science and Technology
P.R.China

1. Brief Survey of Particle Swarm Optimization

With the industrial and scientific developments, many new optimization problems are needed to be solved. Several of them are complex multi-modal, high dimensional, non-differential problems. Therefore, some new optimization techniques have been designed, such as genetic algorithm (Holland, 1992), ant colony optimization (Dorigo & Gambardella, 1997), etc. However, due to the large linkage and correlation among different variables, these algorithms are easily trapped to a local optimum and failed to obtain the reasonable solution.

Particle swarm optimization (PSO) (Eberhart & Kennedy, 1995; Kennedy & Eberhart, 1995) is a population-based, self-adaptive search optimization method motivated by the observation of simplified animal social behaviors such as fish schooling, bird flocking, etc. It is becoming very popular due to its simplicity of implementation and ability to quickly converge to a reasonably good solution (Shen et al., 2005; Eberhart & Shi, 1998; Li et al., 2005).

In a PSO system, multiple candidate solutions coexist and collaborate simultaneously. Each solution called a "particle", flies in the problem search space looking for the optimal position to land. A particle, as time passes through its quest, adjusts its position according to its own "experience" as well as the experience of neighboring particles. Tracking and memorizing the best position encountered build particle's experience. For that reason, PSO possesses a memory (i.e. every particle remembers the best position it reached during the past). PSO system combines local search method (through self experience) with global search methods (through neighboring experience), attempting to balance exploration and exploitation.

A particle status on the search space is characterized by two factors: its position and velocity, which are updated by following equations:

\[
\mathbf{v}_j(t+1) = w\mathbf{v}_j(t) + c_1r_1(\mathbf{p}_j(t) - \mathbf{x}_j(t)) + c_2r_2(\mathbf{p}_b(t) - \mathbf{x}_j(t))
\]

\[
\mathbf{x}_j(t+1) = \mathbf{x}_j(t) + \mathbf{v}_j(t+1)
\]

where \( \mathbf{v}_j(t) \) and \( \mathbf{x}_j(t) \) represent the velocity and position vectors of particle \( j \) at time \( t \), respectively. \( \mathbf{p}_j(t) \) means the best position vector which particle \( j \) had been found, as well as \( \mathbf{p}_b(t) \) denotes the corresponding best position found by the whole swarm. Cognitive
coefficient $c_1$ and social coefficient $c_2$ are constants known as acceleration coefficients, and $r_1$ and $r_2$ are two separately generated uniformly distributed random numbers in the range $[0, 1]$. To keep the moving stability, a limited coefficient $v_{\text{max}}$ is introduced to restrict the size of velocity.

$$\left| v_{jk}(t + 1) \right| \leq v_{\text{max}}$$ (3)

The first part of (1) represents the previous velocity, which provides the necessary momentum for particles to roam across the search space. The second part, known as the "cognitive" component, represents the personal thinking of each particle. The cognitive component encourages the particles to move toward their own best positions found so far. The third part is known as the "social" component, which represents the collaborative effect of the particles, in finding the global optimal solution. The social component always pulls the particles toward the global best particle found so far.

Since particle swarm optimization is a new swarm intelligent technique, many researchers focus their attentions to this new area. One famous improvement is the introduction of the inertia weight (Shi & Eberhart, 1998a), similarly with temperature schedule in the simulated annealing algorithm. Empirical results showed the linearly decreased setting of inertia weight can give a better performance, such as from 1.4 to 0 (Shi & Eberhart, 1998a), and 0.9 to 0.4 (Shi & Eberhart, 1998b, Shi & Eberhart, 1999). In 1999, Suganthan (Suganthan,1999) proposed a time-varying acceleration coefficients automation strategy in which both $c_1$ and $c_2$ are linearly decreased during the course of run. Simulation results show the fixed acceleration coefficients at 2.0 generate better solutions. Following Suganthan's method, Venter (Venter, 2002) found that the small cognitive coefficient and large social coefficient could improve the performance significantly. Further, Ratnaweera (Ratnaweera et al., 2004) investigated a time-varying acceleration coefficients. In this automation strategy, the cognitive coefficient is linearly decreased during the course of run, however, the social coefficient is linearly increased inversely.

Hybrid with Kalman filter, Monson designed a new Kalman filter particle swarm optimization algorithm (Monson & Seppi, 2004). Similarly, Sun proposed a new quantum particle swarm optimization (Sun et al., 2004) in 2004. From the convergence point, Cui designed a global convergence algorithm — stochastic particle swarm optimization (Cui & Zeng, 2004). There are still many other modified methods, such as fast PSO (Cui et al., 2006a), predicted PSO (Cui et al.,2006b), etc. The details of these algorithms can be found in corresponding references.

The PSO algorithm has been empirically shown to perform well on many optimization problems. However, it may easily get trapped in a local optimum for high dimensional multi-modal problems. With respect to the PSO model, several papers have been written on the subject to deal with premature convergence, such as the addition of a queen particle (Mendes et al., 2004), the alternation of the neighborhood topology (Kennedy, 1999), the introduction of subpopulation and giving the particles a physical extension (Lovbjerg et al., 2001), etc. In this paper, an individual parameter selection strategy is designed to improve the performance when solving high dimensional multi-modal problems.

The rest of this chapter is organized as follows: the section 2 analyzes the disadvantages of the standard particle swarm optimization parameter selection strategies; the individual inertia weight selection strategy is designed in section 3; whereas section 4 provides the cognitive parameter selection strategy. In section 5, the individual social parameter selection strategies is designed. Finally, conclusion and future research are discussed.
2. The Disadvantages of Standard Particle Swarm Optimization

Partly due to the differences among individuals, swarm collective behaviors are complex processes. Fig.1 and Fig.2 provide an insight of the special swarm behaviors about birds flocking and fish schooling. For a group of birds or fish families, there exist many differences. Firstly, in nature, there are many internal differences among birds (or fish), such as ages, catching skills, flying experiences, and muscles' stretching, etc. Furthermore, the lying positions also provide an important influence on individuals. For example, individuals, lying in the side of the swarm, can make several choices differing from center others. Both of these differences mentioned above provide a marked contribution to the swarm complex behaviors.

Figure 1. Fish's Swimming Process

Figure 2. Birds' Flying Process
For standard particle swarm optimization, each particle maintains the same flying (or swimming) rules according to (1), (2) and (3). At each iteration, the inertia weight \( w \), cognitive learning factor \( c_1 \) and social learning factor \( c_2 \) are the same values within the whole swarm, thus the differences among particles are omitted. Since the complex swarm behaviors can emerge the adaptation, a more precise model, incorporated with the differences, can provide a deeper insight of swarm intelligence, and the corresponding algorithm may be more effective and efficient. Inspired with this method, we propose a new algorithm in which each particle maintains personal controlled parameter selection setting.

3. Individual Inertia Weight Selection Strategy

Without loss of generality, this paper consider the following problem:

\[
\min f(\mathbf{X}) \quad \mathbf{X} \in D \subseteq \mathbb{R}^n
\]  

From the above analysis, the new variant of PSO in this section will incorporate the personal differences into inertia weight of each particle (called PSO-IIWSS, in briefly) (Cai et al., 2008), providing a more precise model simulating the swarm behaviors. However, as a new modified PSO, PSO-IIWSS should consider two problems listed as follows:

1. **How to define the characteristic differences of each particle?**
2. **How to use the characteristic difference to control inertia weight, so as to affect its behaviors?**

### 3.1 How to define the characteristic differences?

If the fitness value of particle \( u \) is better than which of particle \( m \), the probability that global optima falls into \( u \)'s neighborhood is larger than that of particle \( m \). In this manner, the particle \( u \) should pay more attentions to exploit its neighborhood. On the contrary, it may tend to explore other region with a larger probability than exploitation. Thus the information index is defined as follows:

The information index - score of particle \( u \) at time \( t \) is defined as

\[
\text{Score}_u(t) = \frac{f(x_{\text{worst}}(t)) - f(x_u(t))}{f(x_{\text{worst}}(t)) - f(x_{\text{best}}(t))}
\]

where \( x_{\text{worst}}(t) \) and \( x_{\text{best}}(t) \) are the worst and best particles' position vectors at time \( t \), respectively.

### 3.2 How to use the characteristic differences to guide its behaviors?

Since the coefficients setting can control the particles' behaviors, the differences may be incorporated into the controlled coefficients setting to guide each particle's behavior. The allowed controlled coefficients contain inertia weight \( w \), two accelerators \( c_1 \) and \( c_2 \). In this section, inertia weight \( w \) is selected as a controlled parameter to reflect the personal characters. Since \( w \) is dependent with each particle, we use \( w_u(t) \) representing the inertia weight of particle \( u \) at time \( t \).

Now, let us consider the adaptive adjustment strategy of inertia weight \( w_u(t) \). The following part illustrates three different adaptive adjustment strategies.

Inspired by the ranking selection mechanism of genetic algorithm (Michalewicz, 1992), the first adaptive adjustment of inertia weight is provided as follows:
The inertia weight $w_u(t)$ of particle $u$ at time $t$ is computed by

$$w_u(t) = w_{low}(t) + (w_{high}(t) - w_{low}(t)) \times (1 - Score_j(t))$$

(6)

where $w_{low}(t)$ and $w_{high}(t)$ are the lower and upper bounds of the swarm at time $t$.

This adaptive adjustment strategy states the better particles should tend to exploit its neighbors, as well as the worse particles prefer to explore other region. This strategy implies the determination of inertia weight of each particle, may provide a large selection pressure.

Compared with ranking selection, fitness uniform selection scheme (FUSS) is a new selection strategy measuring the diversity in phenotype space. FUSS works by focusing the selection intensity on individuals which have uncommon fitness values rather than on those with highest fitness as is usually done, and the more details can be found in (Marcus, 2002).

Inspired by FUSS, the adaptive adjustment strategy two aims to provide a more chance to balance exploration and exploitation capabilities.

The inertia weight $w_u(t)$ of particle $u$ at time $t$ is computed by

$$w_u(t) = w_{low}(t) + (w_{high}(t) - w_{low}(t)) \times (1 - Score_{rand}(t))$$

(7)

where $w_{low}(t)$ and $w_{high}(t)$ are the lower and upper bounds of the swarm at time $t$. $Score_{rand}(t)$ is defined as follows.

$$Score_{rand}(t) = \{Score_i(t) | r - f(x_i(t))| \leq |r - f(x_{best}(t))|, i = 1, ..., s - 1, s + 1, ..., n\}$$

(8)

where $r$ is a random number sampling uniformly between $f(x_{best}(t))$ and $f(x_{worst}(t))$.

Different from ranking selection and FUSS strategies which need to order the whole swarm, tournament strategy (Blickle & Thiele, 1995) is another type of selection strategy, it only uses several particles to determine one particle’s selection probability. Analogized with tournament strategy, the adaptive adjustment strategy three is designed with local competition, and defined as follows:

The inertia weight $w_u(t)$ of particle $u$ at time $t$ is computed by

$$w_u(t) = \begin{cases} w_{low}(t) + (w_{high}(t) - w_{low}(t)) \times (1 - Score_{r_1}(t)), & \text{if } f(x_{r_1}) < f(x_{r_2}) \\ w_{low}(t) + (w_{high}(t) - w_{low}(t)) \times (1 - Score_{r_2}(t)), & \text{otherwise} \end{cases}$$

(9)

where $w_{low}(t)$ and $w_{high}(t)$ are the lower and upper bounds of the swarm at time $t$. $x_{r_1}(t)$ and $x_{r_2}(t)$ are two random selected particles uniformly.

### 3.3 The Step of PSO-IIWSS

The step of PSO-IIWSS is listed as follows.

- **Step 1.** Initializing each coordinate $x_{jk}(0)$ to a value drawn from the uniform random distribution on the interval $[x_{min}, x_{max}]$, for $j = 1, 2, ..., s$ and $k = 1, 2, ..., n$. This distributes the initial position of the particles throughout the search space. Where $s$ is the value of the swarm, $n$ is the value of dimension. Initializing each $v_{jk}(0)$ to a value drawn from the uniform random distribution on the interval $[-v_{max}, v_{max}]$, for all $j$ and $k$. This distributes the initial velocity of the particles.
- **Step 2.** Computing the fitness of each particle.
- **Step 3.** Updating the personal historical best positions for each particle and the swarm;
• Step 4. Determining the best and worst particles at time \( t \), then, calculate the score of each particle at time \( t \).

• Step 5. Computing the inertia weight value of each particle according to corresponding adaptive adjustment strategy one, two and three (section 3.2, respectively).

• Step 6. Updating the velocity and position vectors with equation (1), (2) and (3) in which the inertia \( w \) is changed with \( w(t) \).

• Step 7. If the stop criteria is satisfied, output the best solution; otherwise, go step 2.

3.4 Simulation Results

3.4.1 Selected Benchmark Functions

In order to certify the efficiency of the PSO-IIWSS, we select five famous benchmark functions to testify the performance, and compare PSO-IIWSS with standard PSO (SPSO) and Modified PSO with time-varying accelerator coefficients (MPSO_TVAC) (Ratnaweera et al, 2004). Combined with different adaptive adjustment strategy of inertia weight one, two and three, the corresponding versions of PSO-IIWSS are called PSO-IIWSS1, PSO-IIWSS2, PSO-IIWSS3, respectively.

Sphere Modal:

\[
 f_1(x) = \sum_{j=1}^{n} x_j^2
\]

where \( |x_j| \leq 100.0 \), and

Schwefel Problem 2.22:

\[
 f_2(x) = \sum_{j=1}^{n} |x_j| + \prod_{k=1}^{n} |x_k|
\]

where \( |x_j| \leq 10.0 \), and

Schwefel Problem 2.26:

\[
 f_3(x) = -\sum_{j=1}^{n} (x_j \sin(\sqrt{|x_j|}))
\]

where \( |x_j| \leq 500.0 \), and

Ackley Function:

\[
 f_4(x) = -20 \exp(-0.2 \sqrt{\frac{1}{n} \sum_{j=1}^{n} x_j^2}) - \exp\left(\frac{1}{n} \sum_{k=1}^{n} \cos(2\pi x_k)\right) + 20 + c
\]

www.intechopen.com
where \( |x_j| \leq 32.0 \), and

\[
f_4(x^*) = f_4(0,0,...,0) = 0.0
\]

Hartman Family:

\[
f_5(x) = -\sum_{i=1}^{4} c_i \exp \left[ -\sum_{j=1}^{3} a_{ij}(x_j - p_{ij})^2 \right]
\]

where \( x_j \in [0,0.1] \), and \( a_{ij} \) is satisfied with the following matrix.

\[
\begin{pmatrix}
3 & 10 & 30 \\
0.1 & 10 & 35 \\
3 & 10 & 30 \\
0.1 & 10 & 35
\end{pmatrix}
\]

\( p_{ij} \) is satisfied with the following matrix.

\[
\begin{pmatrix}
0.3687 & 0.1170 & 0.2673 \\
0.4699 & 0.4387 & 0.7470 \\
0.1091 & 0.8732 & 0.5547 \\
0.03815 & 0.5743 & 0.8828
\end{pmatrix}
\]

\( c_i \) is satisfied with the following matrix.

\[
\begin{pmatrix}
1 \\
1.2 \\
3 \\
3.2
\end{pmatrix}
\]

\[
f_5(x^*) = f_5(0.114, 0.556, 0.852) = -3.86
\]

Sphere Model and Schwefel Problem 2.22 are unimodel functions. Schwefel Problem 2.26 and Ackley function are multi-model functions with many local minima, as well as Hartman Family with only several local minima.

3.4.2 Parameter Setting

The coefficients of SPSO, MPSO_TVAC and PSO-IIWSS are set as follows: The inertia weight \( w \) is decreased linearly from 0.9 to 0.4 with SPSO and MPSO_TVAC, while the inertia weight lower bounds of PSO-IIWSS is set 0.4, and the upper bound of PSO-IIWSS is set linearly from 0.9 to 0.4. Two accelerator coefficients \( c_1 \) and \( c_2 \) are both set to 2.0 with SPSO and PSO-IIWSS, as well as in MPSO_TVAC, \( c_1 \) decreases from 2.5 to 0.5, while \( c_2 \)
increases from 0.5 to 2.5. Total individuals are 100 except Hartman Family with 20, and $v_{\text{max}}$ is set to the upper bound of domain. The dimensions of Sphere Model, Schwefel Problem 2.22, Schwefel Problem 2.26, and Ackley Function are set to 30, while Hartman Family’s is 3. Each experiment the simulation runs 30 times while each time the largest evolutionary generation is 1000 for Sphere Model, Schwefel Problem 2.22, Schwefel Problem 2.26, and Ackley Function, and due to small dimensionality, Hartman Family is set to 100.

### 3.4.3 Performance Analysis

Table 1 to 5 are the comparison results of five benchmark functions under the same evolution generations respectively. The average mean value and average standard deviation of each algorithm are computed with 30 runs and listed as follows.

From the Tables, PSO-IIWSSI maintains a better performance than SPSO and MPSO_TVAC with the average mean value. For unimodel functions, PSO-IIWSS3 shows preferable convergence capability than PSO-IIWSS2, while vice versa for the multi-model functions. From Figure 1 and 2, PSO-IIWSSI and PSO-IIWSS3 can find the global optima with nearly a line track, while PSO-IIWSS2 shows the fast search capability during the whole course of simulation for Figure 3 and 4. PSO-IIWSS2 shows the better search performance with the increase of generations. In one word, PSO-IIWSSI owns a better performance within the convergence speed for all functions nearly.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Average Mean Value</th>
<th>Average Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPSO</td>
<td>$9.512e-006$</td>
<td>$1.4809e-005$</td>
</tr>
<tr>
<td>MPSO_TVAC</td>
<td>$4.5945e-018$</td>
<td>$1.9379e-017$</td>
</tr>
<tr>
<td>PSO-IIWSSI</td>
<td>$1.4251e-023$</td>
<td>$1.8342e-023$</td>
</tr>
<tr>
<td>PSO-IIWSS2</td>
<td>$1.2429e-012$</td>
<td>$2.8122e-012$</td>
</tr>
<tr>
<td>PSO-IIWSS3</td>
<td>$1.3374e-019$</td>
<td>$6.0570e-019$</td>
</tr>
</tbody>
</table>

Table 1. Simulation Results of Sphere Model

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Average Mean Value</th>
<th>Average Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPSO</td>
<td>$7.7829e-005$</td>
<td>$7.5821e-005$</td>
</tr>
<tr>
<td>MPSO_TVAC</td>
<td>$3.0710e-007$</td>
<td>$1.0386e-006$</td>
</tr>
<tr>
<td>PSO-IIWSSI</td>
<td>$2.4668e-015$</td>
<td>$2.0972e-015$</td>
</tr>
<tr>
<td>PSO-IIWSS2</td>
<td>$1.9800e-009$</td>
<td>$1.5506e-009$</td>
</tr>
<tr>
<td>PSO-IIWSS3</td>
<td>$3.2359e-012$</td>
<td>$4.1253e-012$</td>
</tr>
</tbody>
</table>

Table 2. Simulation Results of Schwefel Problem 2.22

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Average Mean Value</th>
<th>Average Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPSO</td>
<td>$-6.2474e+003$</td>
<td>$9.2131e+002$</td>
</tr>
<tr>
<td>MPSO_TVAC</td>
<td>$-6.6502e+003$</td>
<td>$6.0927e+002$</td>
</tr>
<tr>
<td>PSO-IIWSSI</td>
<td>$-7.7455e+003$</td>
<td>$8.0910e+002$</td>
</tr>
<tr>
<td>PSO-IIWSS2</td>
<td>$-6.3898e+003$</td>
<td>$9.2699e+002$</td>
</tr>
<tr>
<td>PSO-IIWSS3</td>
<td>$-6.1469e+003$</td>
<td>$9.1679e+002$</td>
</tr>
</tbody>
</table>

Table 3. Simulation Results of Schwefel Problem 2.26
### Table 4. Simulation Results of Ackley Function

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Average Mean Value</th>
<th>Average Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPSO</td>
<td>8.8178e-004</td>
<td>6.8799e-004</td>
</tr>
<tr>
<td>MPSO_TVAC</td>
<td>1.8651e-005</td>
<td>1.0176e-004</td>
</tr>
<tr>
<td>PSO-IIWSS1</td>
<td>2.9940e-011</td>
<td>4.7552e-011</td>
</tr>
<tr>
<td>PSO-IIWSS2</td>
<td>3.8672e-007</td>
<td>5.6462e-007</td>
</tr>
<tr>
<td>PSO-IIWSS3</td>
<td>3.3699e-007</td>
<td>5.8155e-007</td>
</tr>
</tbody>
</table>

Table 4. Simulation Results of Ackley Function

### Table 5. Simulation Results of Hartman Family

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Average Mean Value</th>
<th>Average Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPSO</td>
<td>-3.7507e+000</td>
<td>1.0095e-001</td>
</tr>
<tr>
<td>MPSO_TVAC</td>
<td>-3.8437e+000</td>
<td>2.9505e-002</td>
</tr>
<tr>
<td>PSO-IIWSS1</td>
<td>-3.8562e+000</td>
<td>1.0311e-002</td>
</tr>
<tr>
<td>PSO-IIWSS2</td>
<td>-3.8511e+000</td>
<td>1.6755e-002</td>
</tr>
<tr>
<td>PSO-IIWSS3</td>
<td>-3.8130e+000</td>
<td>5.2168e-002</td>
</tr>
</tbody>
</table>

Table 5. Simulation Results of Hartman Family

### 3.5 Individual non-linear inertia weight selection strategy (Cui et al., 2008)

#### 3.5.1 PSO-IIWSS with Different Score Strategies (PSO-INLIWSS)

As mentioned above, the linearly decreased score strategy cannot reflect the truly complicated search process of PSO. To make a deep insight of action for score, three non-linear score strategies are designed in this paper. These three strategies are unified to a power function, which is set to the following equation:

\[
\text{Score}_u(t) = \left(1 - \left(\frac{f(x_{\text{worst}}(t)) - f(x_u(t))}{f(x_{\text{worst}}(t)) - f(x_{\text{best}}(t))}\right)^{k_1}\right)\cdot k_2
\]

where \(k_1\) and \(k_2\) are two integer numbers.

Figure 3 shows the trace of linear and three non-linear score strategies, respectively. In Figure 1, the value \(f(x_{\text{best}}(t))\) is set 1, as well as \(f(x_{\text{worst}}(t))\) is 100. When \(k_1\) and \(k_2\) are both set to 1, it is just the score strategy proposed in [7], which is also called strategy one in this paper. While \(k_1 > 1\) and \(k_2 = 1\), this non-linear score strategy is called strategy two here. And strategy three corresponds to \(k_1 = 1\) and \(k_2 > 1\), strategy four corresponds to \(k_1 > k_2 > 1\). Description of these three non-linear score strategies are listed as follows: Strategy two: the curve \(k_1 = 2\) and \(k_2 = 1\) in Figure 3 is an example of strategy two. It can be seen this strategy has a lower score value than strategy one. However, the increased ratio of score is not a constant value. For those particles with small fitness values, the corresponding score values are smaller than strategy one, and they pay more attention to exploit the region near the current position. However, the particles tend to make a local search is larger than strategy one due to the lower score values. Therefore, strategy two enhances the local search capability.

Strategy three: the curve \(k_1 = 1\) and \(k_2 = 2\) in Figure 3 is an example of strategy three. As we can see, it is a reversed curve compared with strategy two. Therefore, it enhances the global search capability.

Strategy four: the curve \(k_1 = 2\) and \(k_2 = 5\) in Figure 3 is an example of strategy four. The first part of this strategy is similar with strategy two, as well as the later part is similar with strategy three. Therefore, it augments both the local and global search capabilities.
The step of PSO-INLIWSS with different score strategies are listed as follows.

- Step 1. Initializing the position and velocity vectors of the swarm, and determining the historical best positions of each particle and its neighbors;
- Step 2. Determining the best and worst particles at time \( t \) with the following definitions:

\[
x_{\text{best}}(t) = \arg \min \{ f(x_j(t)), j = 1, 2, ..., s \}
\]  

and

\[
x_{\text{worst}}(t) = \arg \max \{ f(x_j(t)), j = 1, 2, ..., s \}
\]

- Step 3. Calculate the score of each particle at time \( t \) with formula (10) using different strategies.
- Step 4. Calculating the PSO-INLIWSS inertia weight according to formula (6);
- Step 5. Updating the velocity and position vectors according to formula (1), (2) and (3);
- Step 6. Determining the current personal memory (historical best position);
- Step 7. Determining the historical best position of the swarm;
- Step 8. If the stop criteria is satisfied, output the best solution; otherwise, go step 2.

### 3.5.2 Simulation Results

To certify the efficiency of the proposed non-linear score strategy, we select five famous benchmark functions to test the performance, and compared with standard PSO (SPSO), modified PSO with time-varying accelerator coefficients (MPSO-TVAC) (Ratnaweera et al., 2004), and comprehensive learning particle swarm optimization (CLPSO) (Liang et al., 2006). Since we adopt four different score strategies, the proposed methods are called PSO-INLIWSS1 (with strategy one, in other words, the original linearly PSO-IIWSS1), PSO-INLIWSS2 (with strategy two), PSO-INLIWSS3 (with strategy three) and PSO-INLIWSS4 (with strategy four), respectively. The details of the experimental environment and results are explained as follows.
In this paper, five typical unconstraint numerical benchmark functions are used to test. They are: Rosenbrock, Schwefel Problem 2.26, Ackley and two Penalized functions.

Rosenbrock Function:

\[
f_1(x) = \sum_{j=1}^{n-1} [100(x_{j+1} - x_j^2)^2 + (x_j - 1)^2]
\]

where \(|x_j| \leq 30.0\), and 

\[f_1(x^*) = f_1(1,1,...,1) = 0.0\]

Schwefel Problem 2.26:

\[
f_2(x) = -\sum_{j=1}^{n} (x_j \sin(\sqrt{|x_j|}))
\]

where \(|x_j| \leq 500.0\), and 

\[f_2(x^*) = f_2(420.9687, 420.9687, ..., 420.9687) \approx -12569.5\]

Ackley Function:

\[
f_3(x) = -20 \exp\left(-0.2 \sqrt{\frac{1}{n} \sum_{j=1}^{n} x_j^2}\right) - \exp\left(\frac{1}{n} \sum_{k=1}^{n} \cos(2\pi x_k)\right) + 20 + e
\]

where \(|x_j| \leq 32.0\), and 

\[f_3(x^*) = f_3(0,0,...,0) = 0.0\]

Penalized Function 1:

\[
f_4(x) = \frac{\pi}{30} \{10 \sin^2(\pi y_1) + \sum_{i=1}^{n-1} (y_i - 1)^2[1 + 10 \sin^2(\pi y_{i+1})] + (y_n - 1)^2\} + \sum_{i=1}^{n} u(x_i, 10, 100, 4)
\]

where \(|x_j| \leq 50.0\), and

\[u(x_i, a, k, m) = \begin{cases} 
  k(x_i - a)^m, & \text{if } x_i > a \\
  0, & \text{if } -a \leq x_i \leq a \\
  k(-x_i - a)^m, & \text{if } x_i < -a 
\end{cases}
\]

\[y_i = 1 + \frac{1}{4} (x_i + 1)
\]

\[f_4(x^*) = f_4(1,1,...,1) = 0.0\]
Penalized Function 2:

\[ f_5(x) = 0.1 \left( \sin^2(3\pi x_1) + \sum_{i=1}^{n-1} (x_i - 1)^2 [1 + \sin^2(3\pi x_{i+1})] \right) \\
+ (x_n - 1)^2 [1 + \sin^2(2\pi x_n)] + \sum_{i=1}^{n} u(x_i, 5, 100, 4) \]

where \( \|x_j\| \leq 50.0 \), and

\[ f_5(x^*) = f_5(1, 1, \ldots, 1) = 0.0 \]

Generally, Rosenbrock is viewed as a unimodal function, however, in recent literatures, several numerical experiments (Shang & Qiu, 2006) have been made to show Rosenbrock is a multi-modal function with only two local optima when dimensionality between 4 to 30. Schwefel problem 2.26, Ackley, and two penalized functions are multi-modal functions with many local minima.

The coefficients of SPSO, MPSO-TVAC, and PSO-INLIWSS are set as follows: inertia weight \( w \) is decreased linearly from 0.9 to 0.4 with SPSO and MPSO-TVAC, while the inertia weight lower bounds of all version of PSO-INLIWSS are both set to 0.4, and the upper bounds of PSO-INLIWSS are both set linearly decreased from 0.9 to 0.4. Two accelerator coefficients \( c_1 \) and \( c_2 \) are set to 2.0 with SPSO and PSO-INLIWSS, as well as in MPSO-TVAC, \( c_1 \) decreases from 2.5 to 0.5, while \( c_2 \) increases from 0.5 to 2.5. Total individuals are 100, and the velocity threshold \( v_{\text{max}} \) is set to the upper bound of the domain. The dimensionality is 30. In each experiment, the simulation run 30 times, while each time the largest iteration is 50 x dimension.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Mean Value</th>
<th>Std Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPSO</td>
<td>5.6170e+001</td>
<td>4.3584e+001</td>
</tr>
<tr>
<td>MPSO-TVAC</td>
<td>3.3589e+001</td>
<td>4.1940e+001</td>
</tr>
<tr>
<td>CLPSO</td>
<td>5.1948e+001</td>
<td>2.7775e+001</td>
</tr>
<tr>
<td>PSO-INLIWSS1</td>
<td>2.3597e+001</td>
<td>2.3238e+001</td>
</tr>
<tr>
<td>PSO-INLIWSS2</td>
<td>3.4147e+001</td>
<td>2.9811e+001</td>
</tr>
<tr>
<td>PSO-INLIWSS3</td>
<td>4.0342e+001</td>
<td>3.2390e+001</td>
</tr>
<tr>
<td>PSO-INLIWSS4</td>
<td>3.1455e+001</td>
<td>2.4259e+001</td>
</tr>
</tbody>
</table>

Table 6. The Comparison Results for Rosenbrock

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Mean Value</th>
<th>Std Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPSO</td>
<td>-6.2762e+003</td>
<td>1.1354e+003</td>
</tr>
<tr>
<td>MPSO-TVAC</td>
<td>-6.7672e+003</td>
<td>5.7050e+002</td>
</tr>
<tr>
<td>CLPSO</td>
<td>-1.0843e+004</td>
<td>3.6105e+002</td>
</tr>
<tr>
<td>PSO-INLIWSS1</td>
<td>-7.7885e+003</td>
<td>1.1526e+003</td>
</tr>
<tr>
<td>PSO-INLIWSS2</td>
<td>-7.2919e+003</td>
<td>1.1476e+003</td>
</tr>
<tr>
<td>PSO-INLIWSS3</td>
<td>-9.0079e+003</td>
<td>7.1024e+002</td>
</tr>
<tr>
<td>PSO-INLIWSS4</td>
<td>-9.0064e+003</td>
<td>9.6881e+002</td>
</tr>
</tbody>
</table>

Table 7. The Comparison Results for Schwefel Problem 2.26
For Rosenbrock (see Table 6), because there is an additional local optimum near \((-1, 0, 0, ..., 0)\), the performance of the MPSO-TVAC, PSO-INLIWSS1 and PSO-INLIWSS4 are better than others. We also perform several other unimodal and multi-modal functions with only few local optima, the PSO-INLIWSS1 are always the best one within these seven algorithms. For Schwefel problem 2.26 (Table 7) and Ackley (Table 8), the performance of PSO-INLIWSS3 and PDPSO4 are nearly the same. Both of them are better than others. However, for two penalized functions (Table 9 and 10), the performance of PSO-INLIWSS3 is not the same as the previous two, although PSO-INLIWSS4 is still stable and better than others. As we known, both of these two penalized functions has strong linkage among dimensions. This implies PSO-INLIWSS4 is more suit for multi-modal problems.

Based on the above analysis, we can draw the following two conclusions:

1. PSO-INLIWSS1 (the original version of PSO-IIWSS1) is suit for unimodal and multi-modal functions with a few local optima;

2. PSO-INLIWSS4 is the most stable and effective among three score strategies. It is fit for multi-modal functions with many local optima especially for linkages among dimensions;

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Mean Value</th>
<th>Std Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPSO</td>
<td>5.8161e-006</td>
<td>4.6415e-006</td>
</tr>
<tr>
<td>MPSO-TVAC</td>
<td>7.5381e-007</td>
<td>3.3711e-006</td>
</tr>
<tr>
<td>CLPSO</td>
<td>5.6159e-006</td>
<td>4.9649e-006</td>
</tr>
<tr>
<td>PSO-INLIWSS1</td>
<td>4.2810e-014</td>
<td>4.3890e-014</td>
</tr>
<tr>
<td>PSO-INLIWSS2</td>
<td>1.1696e-011</td>
<td>1.2619e-011</td>
</tr>
<tr>
<td>PSO-INLIWSS3</td>
<td>2.2559e-014</td>
<td>8.7745e-015</td>
</tr>
<tr>
<td>PSO-INLIWSS4</td>
<td>2.1493e-014</td>
<td>7.8195e-015</td>
</tr>
</tbody>
</table>

Table 8. The Comparison Results for Ackley

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Mean Value</th>
<th>Std Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPSO</td>
<td>6.7461e-002</td>
<td>2.3159e-001</td>
</tr>
<tr>
<td>MPSO-TVAC</td>
<td>1.8891e-017</td>
<td>6.9756e-017</td>
</tr>
<tr>
<td>CLPSO</td>
<td>1.0418e-002</td>
<td>3.1898e-002</td>
</tr>
<tr>
<td>PSO-INLIWSS1</td>
<td>1.6477e-025</td>
<td>4.7735e-025</td>
</tr>
<tr>
<td>PSO-INLIWSS2</td>
<td>6.2234e-026</td>
<td>1.6641e-025</td>
</tr>
<tr>
<td>PSO-INLIWSS3</td>
<td>2.4194e-024</td>
<td>7.6487e-024</td>
</tr>
<tr>
<td>PSO-INLIWSS4</td>
<td>2.2684e-027</td>
<td>4.4964e-027</td>
</tr>
</tbody>
</table>

Table 9. The Comparison Results for Penalized Function1

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Mean Value</th>
<th>Std Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SPSO</td>
<td>5.4943e-003</td>
<td>2.4568e-003</td>
</tr>
<tr>
<td>MPSO-TVAC</td>
<td>9.3610e-027</td>
<td>4.1753e-026</td>
</tr>
<tr>
<td>CLPSO</td>
<td>1.1098e-007</td>
<td>2.6748e-007</td>
</tr>
<tr>
<td>PSO-INLIWSS1</td>
<td>4.8692e-027</td>
<td>1.3533e-026</td>
</tr>
<tr>
<td>PSO-INLIWSS2</td>
<td>2.8092e-028</td>
<td>5.6078e-028</td>
</tr>
<tr>
<td>PSO-INLIWSS3</td>
<td>9.0765e-027</td>
<td>2.5940e-026</td>
</tr>
<tr>
<td>PSO-INLIWSS4</td>
<td>8.2794e-028</td>
<td>1.6562e-027</td>
</tr>
</tbody>
</table>

Table 10. The Comparison Results for Penalized Function2
4. Individual Cognitive Selection Strategy

Because each particle maintains two types of performance at time $t$: the fitness value $f(\bar{p}_j(t))$ of historical best position found by particle $j$ and that of current position $f(\bar{x}_j(t))$ respectively. Similarly, two different rewards of environment are also designed associated with $f(\bar{p}_j(t))$ and $f(\bar{x}_j(t))$. For convenience, the reward based upon $f(\bar{p}_j(t))$ is called the self-learning strategy one, and the other one is called the self-learning strategy two. The details of these two strategies are explained as follows.

4.1 Self-learning Strategy One

Let us suppose $\bar{P}(t) = (\bar{p}_1(t), \bar{p}_2(t), ..., \bar{p}_n(t))$ is the historical best position vector of the swarm at time $t$, where $n$ and $\bar{p}_j(t)$ denote the dimensionality and the historical best position found by particle $j$ until time $t$.

The expectation limitation position of particle $j$ of standard version of PSO is

$$E\{\lim_{t \to \infty} \bar{x}_j(t)\} = \frac{c_1 \bar{p}_j + c_2 \bar{p}_g}{c_1 + c_2}$$  \hspace{1cm} (13)

if $c_1$ and $\bar{p}_j(t)$ are constant values. Thus, a large $c_1$ makes the $E\{\lim_{t \to \infty} \bar{x}_j(t)\}$ moving towards $\bar{p}_j(t)$ and exploits near $\bar{p}_j(t)$ with more probability, and vice versa. Combined the better $\bar{p}_j(t)$ implies the more capability of which global optima falls into, the cognitive coefficient is set as follows.

$$c_{1,j}(t) = c_{low} + (c_{high} - c_{low}) \times Reward_{1,j}(t)$$  \hspace{1cm} (14)

where $c_{low}$ and $c_{high}$ are two predefined lower and upper bounds to control this coefficient. $Reward_{1,j}(t)$ is defined

$$Reward_{1,j}(t) = \begin{cases} 1 & \text{if } f_{worst} = f_{best}, \\ \frac{f_{worst} - f(\bar{p}_j(t))}{f_{worst} - f_{best}} & \text{otherwise} \end{cases}$$ \hspace{1cm} (15)

where $f_{worst}$ and $f_{best}$ denote the worst and best values among $f(\bar{P}(t))$.

4.2 Self-learning Strategy Two

Let us suppose $\bar{X}(t) = (\bar{x}_1(t), \bar{x}_2(t), ..., \bar{x}_n(t))$ is the population at time $t$, where $n$, $\bar{x}_j(t)$ denote the dimensionality and the position of particle $j$ at time $t$.

Different from strategy one, if the performance $\bar{x}_j(t)$ is better than $\bar{x}_k(t)$ ($j$ and $k$ are arbitrary chosen from the population), the probability of global optimal fallen near $\bar{x}_j(t)$ is larger than $\bar{x}_k(t)$, thus, particle $j$ should exploit near its current position with a larger probability than particle $k$. It means $c_{1,j}(t)$ should be less than $c_{1,k}(t)$ to provide little affection of historical best position $\bar{p}_j(t)$ and the adjustment is defined as follows

$$c_{1,j}(t) = c_{low} + (c_{high} - c_{low}) \times Reward_{2,j}(t)$$  \hspace{1cm} (16)

where $Reward_{2,j}(t)$ is defined as
where $f_{\text{worst}}$ and $f_{\text{best}}$ denote the worst and best values among $f(\mathbf{x}(t))$.

4.3 Mutation Strategy

To avoid premature convergence, a mutation strategy is introduced to enhance the ability escaping from the local optima. This mutation strategy is designed as follows. At each time, particle $j$ is uniformly random selected within the whole swarm, as well as the dimensionality $k$ is also uniformly random selected, then, the $v_{jk}(t)$ is changed as follows.

\[
R_{2j}(t) = \begin{cases} 
0, & \text{if } f_{\text{worst}} = f_{\text{best}}, \\
\frac{f_j(x_j(t)) - f_{\text{best}}}{f_{\text{worst}} - f_{\text{best}}}, & \text{otherwise.}
\end{cases}
\]  

(17)

where $r_1$ and $r_2$ are two random numbers generated with uniform distribution within 0 and 1.

4.4 The Steps of PSO-ILCSS

For convenience, we call the individual Linear Cognitive Selection Strategy (Cai X.J. et al., 2007; Cai X.J. et al., 2008) as ILCSS, and the corresponding variant is called PSO-ILCSS. The detailed steps of PSO-ILCSS are listed as follows.

- Step 1. Initializing each coordinate $x_{jk}(0)$ and $v_{jk}(0)$ sampling within $[x_{\text{min}}, x_{\text{max}}]$ and $[0, v_{\text{max}}]$, respectively, determining the historical best position by each particle and the swarm.
- Step 2. Computing the fitness of each particle.
- Step 3. For each dimension $k$ of particle $j$, the personal historical best position $p_{jk}(t)$ is updated as follows.

\[
p_{jk}(t) = \begin{cases} 
x_{jk}(t), & \text{if } f(x_j(t)) < f(p_j(t-1)), \\
p_{jk}(t-1), & \text{otherwise.}
\end{cases}
\]  

(19)

- Step 4. For each dimension $k$ of particle $j$, the global best position $p_{gk}(t)$ is updated as follows.

\[
p_{gk}(t) = \begin{cases} 
p_{jk}(t), & \text{if } f(p_j(t)) < f(p_{gk}(t-1)), \\
p_{gk}(t-1), & \text{otherwise.}
\end{cases}
\]  

(20)

- Step 5. Selecting the self-learning strategy: if strategy one is selected, computing the cognitive coefficient $c_{1j}(t)$ of each particle according to formula (14) and (15); otherwise, computing cognitive coefficient $c_{1j}(t)$ with formula (16) and (17).
- Step 6. Updating the velocity and position vectors with equations (1)-(3).
- Step 7. Making mutation operator described in section 4.3.
- Step 8. If the criteria is satisfied, output the best solution; otherwise, goto step 2.

4.5 Simulation Results

Five famous benchmark functions are used to test the proposed algorithm's efficiency. They are Schwefel Problem 2.22, 2.26, Ackley, and two different Penalized Functions, the global optima is 0 except Schwefel Problem 2.26 is -12569.5, while Schwefel Problem 2.22 is
unimodel function. Schwefel Problem 2.26, Ackley function and two Penalized Functions are multi-model functions with many local minima.

In order to certify the efficiency, four different versions are used to compare: PSO-ILCSS with self-learning strategy one (PSO-ILCSSI), PSO-ILCSS with self-learning strategy two (PSO-ILCSS2), standard PSO (SPSO) and Modified PSO with time-varying accelerator coefficients (MPSO-TVAC) (Ratnaweera et al., 2004).

The coefficients of SPSO, MPSO-TVAC, PSO-ILCSS1 and PSO-ILCSS2 are set as follows: the inertia weight $w$ is decreased linearly from 0.9 to 0.4. Two accelerator coefficients $c_1$ and $c_2$ are both set to 2.0 with SPSO, and in MPSO-TVAC, $c_1$ decreased from 2.5 to 0.5, while $c_2$ increased from 0.5 to 2.5. In PSO-ILCSS1 and PSO-ILCSS2, the lower bound $c_{low}$ of $c_1$ is set to 1.0, and the upper bound $c_{high}$ set to linearly decreased from 2.0 to 1.0, while $c_2$ is set to 2.0. Total individual is 100, and the dimensionality is 30, and $v_{max}$ is set to the upper bound of domain. In each experiment, the simulation run 30 times, while each time the largest evolutionary generation is 1000.

Table 2 is the comparison results of five benchmark functions under the same evolution generations. The average mean value and average standard deviation of each algorithm are computed with 30 runs and listed as follows.

<table>
<thead>
<tr>
<th>Function</th>
<th>Algorithm</th>
<th>Average Mean Value</th>
<th>Average Standard Deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>F1</td>
<td>SPSO</td>
<td>6.6044e-005</td>
<td>4.7092e-005</td>
</tr>
<tr>
<td></td>
<td>MPSO-TVAC</td>
<td>3.0710e-007</td>
<td>1.0386e-006</td>
</tr>
<tr>
<td></td>
<td>PSO-ILCSS1</td>
<td>2.1542e-007</td>
<td>3.2436e-007</td>
</tr>
<tr>
<td></td>
<td>PSO-ILCSS2</td>
<td>9.0189e-008</td>
<td>1.3398e-007</td>
</tr>
<tr>
<td>F2</td>
<td>SPSO</td>
<td>-6.2474e+003</td>
<td>9.2313e+002</td>
</tr>
<tr>
<td></td>
<td>MPSO-TVAC</td>
<td>-6.6502e+003</td>
<td>6.0927e+002</td>
</tr>
<tr>
<td></td>
<td>PSO-ILCSS1</td>
<td>-8.1386e+003</td>
<td>6.2219e+002</td>
</tr>
<tr>
<td></td>
<td>PSO-ILCSS2</td>
<td>-8.0653e+003</td>
<td>7.2042e+002</td>
</tr>
<tr>
<td>F3</td>
<td>SPSO</td>
<td>1.9864e-003</td>
<td>6.0721e-003</td>
</tr>
<tr>
<td></td>
<td>MPSO-TVAC</td>
<td>1.8651e-005</td>
<td>1.0176e-004</td>
</tr>
<tr>
<td></td>
<td>PSO-ILCSS1</td>
<td>3.8530e-008</td>
<td>3.8205e-008</td>
</tr>
<tr>
<td></td>
<td>PSO-ILCSS2</td>
<td>1.3835e-008</td>
<td>1.0414e-008</td>
</tr>
<tr>
<td>F4</td>
<td>SPSO</td>
<td>4.3043e-002</td>
<td>6.6204e-002</td>
</tr>
<tr>
<td></td>
<td>MPSO-TVAC</td>
<td>1.7278e-002</td>
<td>3.9295e-002</td>
</tr>
<tr>
<td></td>
<td>PSO-ILCSS1</td>
<td>9.1694e-012</td>
<td>3.4561e-011</td>
</tr>
<tr>
<td></td>
<td>PSO-ILCSS2</td>
<td>9.7771e-014</td>
<td>4.9192e-013</td>
</tr>
<tr>
<td>F5</td>
<td>SPSO</td>
<td>4.3662e-003</td>
<td>6.3953e-003</td>
</tr>
<tr>
<td></td>
<td>MPSO-TVAC</td>
<td>3.6624e-004</td>
<td>2.0060e-003</td>
</tr>
<tr>
<td></td>
<td>PSO-ILCSS1</td>
<td>9.4223e-013</td>
<td>3.9129e-012</td>
</tr>
<tr>
<td></td>
<td>PSO-ILCSS2</td>
<td>4.6303e-015</td>
<td>1.0950e-014</td>
</tr>
</tbody>
</table>

Table 11. The Comparison Results of Benchmark Function

From the Table 2, PSO-ILCSSI and PSO-ILCSS2 both maintain better performances than SPSO and MPSO-TVAC no matter the average mean value or standard deviation. The dynamic performances of PSO-ILCSSI and PSO-ILCSS2 are near the same with SPSO and MPSO-TVAC in the first stage, although PSO-ILCSSI and PSO-ILCSS2 maintains quick
global search capability in the last period. In one words, the performances of PSO-ILCSSI and PSO-ILCSS2 surpasses slightly than which of MPSO-TVAC and SPSO a little for unimodel functions, while for the multi-model functions, PSO-ILCSSI and PSO-ILCSS2 show preferable results.

5. Individual Social Selection Strategy

5.1 Individual Linear Social Selection Strategy (ILSSS)
Similarly with cognitive parameter, a dispersed control manner (Cai et al., 2008) is introduced, in which each particle selects its social coefficient value to decide the search direction: $\overrightarrow{p}_j$ or $\overrightarrow{p}_g$. Since the literatures only consider the extreme value $\overrightarrow{p}_g$, however, they neglect the differences between $\overrightarrow{p}_j$ and $\overrightarrow{p}_g$. These settings lose some information maybe useful to find the global optima or escape from a local optima. Thus, we design a new index by introducing the performance differences, and the definition is provided as follows:

$$Grade_u(t) = \frac{f_{\text{worst}}(t) - f(x_u(t))}{f_{\text{worst}}(t) - f_{\text{best}}(t)}$$  \hspace{1cm} (21)$$

where $f_{\text{worst}}(t)$ and $f_{\text{best}}(t)$ are the worst and best fitness values of the swarm at time $t$, respectively. Occasionally, the swarm converges onto one point, that means $f_{\text{worst}}(t) = f_{\text{best}}(t)$. In this case, the value $Grade_u(t)$ of arbitrary particle $u$ is set to 1. $Grade_u(t)$ is an information index to represent the differences of particle $u$ at time $t$, according to its fitness value of the current position. The better the particle is, the larger $Grade_u(t)$ is, and vice versa.

As we known, if the fitness value of particle $u$ is better than which of particle $m$, the probability that global optima falls into $m$’s neighborhood is larger than that of particle $m$. In this manner, the particle $u$ should pay more attentions to exploit its neighborhood. On the contrary, it may tend to explore other region with a larger probability than exploitation. Thus, for the best solution, it should make complete local search around its historical best position, as well as for the worse solution, it should make global search around $\overrightarrow{p}_g$. Then, the dispersed social coefficient of particle $j$ at time $t$ is set as follows:

$$c_{2,j}(t) = c_{\text{low}} + (c_{\text{up}} - c_{\text{low}}) \times Grade_j(t)$$  \hspace{1cm} (22)$$

where $c_{\text{low}}$ and $c_{\text{up}}$ are two predefined numbers, and $c_{2,j}(t)$ represents the social coefficient of particle $j$ at time $t$.

5.2 Individual Non-linear Social Selection Strategy(INLSSS)
As mentioned before, although the individual linear social parameter selection strategy improves the performance significantly, however, its linear manner can not meet the complex optimization tasks. Therefore, in this section, we introduce four different kinds of non-linear manner, and investigate the affection for the algorithm’s performance. Because there are fruitful results about inertia weight, therefore, an intuitive and simple method is to introduce some effective non-linear manner of inertia weight into the study of social parameter automation. Inspired by the previous literatures (Chen et al., 2006; Jiang & Etorre, 2005), four different kinds of nonlinear manner are designed.
The first non-linear social automation strategy is called parabola opening downwards strategy:

\[ c_{2,j}(t) = c_{\text{start}} - (c_{\text{start}} - c_{\text{end}}) \times \left( \frac{t}{\text{MAX}_\text{ITER}} \right)^2 \times \text{Grade}_j(t) \] (23)

The second non-linear social automation strategy is called parabola opening upwards strategy:

\[ c_{2,j}(t) = c_{\text{start}} + (c_{\text{end}} - c_{\text{start}}) \times \left( \frac{t}{\text{MAX}_\text{ITER}} \right)^2 - (c_{\text{start}} - c_{\text{end}}) \times \frac{2t}{\text{MAX}_\text{ITER}} \times \text{Grade}_j(t) \] (24)

The third non-linear social automation strategy is called exponential curve strategy:

\[ c_{2,j}(t) = c_{\text{end}} \times \left( \frac{c_{\text{start}}}{c_{\text{end}}} \right)^{1 + \frac{t}{\text{MAX}_\text{ITER}}} \times \text{Grade}_j(t) \] (25)

The fourth non-linear social automation strategy is called negative-exponential strategy:

\[ c_{2,j}(t) = c_{\text{start}} \times e^{-\frac{t}{\text{MAX}_\text{ITER}}} \times \text{Grade}_j(t) \] (26)

5.3 The Steps of PSO-INLSSS

The detail steps of PSO-INLSSS are listed as follows:

- Step 1. Initializing each coordinate \( x_j \) and \( v_j \) sampling within \([x_{\text{min}}, x_{\text{max}}]\) and \([-v_{\text{max}}, v_{\text{max}}]\), respectively.
- Step 2. Computing the fitness value of each particle.
- Step 3. For \( k \)’th dimensional value of \( j \)’th particle, the personal historical best position \( p_{j}^{k} \) is updated as follows.

\[ p_{j}^{k} = \begin{cases} x_{j}^{k} & \text{if } f(x_j) < f(p_{j}^{k}) \\ p_{j}^{k} & \text{otherwise.} \end{cases} \] (27)

- Step 4. For \( k \)’th dimensional value of \( j \)’th particle, the global best position \( p_{g}^{k} \) is updated as follows.

\[ p_{g}^{k} = \begin{cases} p_{j}^{k} & \text{if } f(p_{j}) < f(p_{g}^{k}) \\ p_{g}^{k} & \text{otherwise.} \end{cases} \] (28)

- Step 5. Computing the social coefficient \( c_{2,j} \) value of each particle according to formula (23)-(26).
- Step 6. Updating the velocity and position vectors with equation (1)-(3) in which social coefficient \( c_{2} \) is changed with \( c_{2,j} \).
- Step 7. Making mutation operator described in section 4.3.
- Step 8. If the criteria is satisfied, output the best solution; otherwise, goto step 2.
5.4 Simulation Results
To testify the performance of these four proposed non-linear social parameter automation strategies, three famous benchmark functions are chosen to test the performance, and compared with standard PSO (SPSO), modified PSO with time-varying accelerator coefficients (MPSO-TVAC) (Ratnaweera et al., 2004) and individual social selection strategy (PSO-ILSSS). Since we adopt four different non-linear strategies, the proposed methods are called PSO-INLSSS-1 (with strategy one), PSO-INLSSS-2 (with strategy two), PSO-INLSSS-3 (with strategy three) and PSO-INLSSS-4 (with strategy four), respectively. The details of the experimental environment and results are explained as follows.

5.4.1 Benchmarks
In this paper, three typical unconstraint numerical benchmark functions are used to test. 

Rastrigin Function:

\[ f_1(x) = \sum_{j=1}^{n} [x_j^2 - 10\cos(2\pi x_j) + 10] \]

where \( |x_j| \leq 5.12 \), and

Ackley Function:

\[ f_2(x) = -20e^{0.2 \sqrt{\frac{1}{n} \sum_{j=1}^{n} x_j^2}} - e^{\frac{1}{n} \sum_{j=1}^{n} \cos 2\pi x_j} + 20 + e \]

where \( |x_j| \leq 32.0 \), and

Penalized Function:

\[ f_3(x) = 0.1 \left[ \sin^2(3\pi x_1) + \sum_{i=1}^{n-1} (x_i - 1)^2 [1 + \sin^2(3\pi x_{i+1})] \right] 
+ (x_n - 1)^2 [1 + \sin^2(2\pi x_n)] + \sum_{i=1}^{n} u(x_i, 5, 100, 4) \]

where \( |x_j| \leq 50.0 \), and

\[ u(x_i, a, k, m) = \begin{cases} 
  k(x_i - a)^m, & \text{if } x_i > a \\
  0, & \text{if } -a \leq x_i \leq a \\
  k(-x_i - a)^m, & \text{if } x_i < -a 
\end{cases} \]

\[ y_i = 1 + \frac{1}{4} (x_i + 1) \]

\[ f_3(x^*) = f_3(1, 1, ..., 1) = 0.0 \]
5.4.2 Parameter Settings
The coefficients of SPSO, MPSO-TVAC, PSO-ILSSS and PSO-INLSSS are set as follows:
The inertia weight $w$ is decreased linearly from 0.9 to 0.4 within SPSO, MPSO-TVAC, PSO-ILSSS and PSO-INLSSS. Accelerator coefficients $c_1$ and $c_2$ are set to 2.0 within SPSO, as well as in MPSO-TVAC, $c_1$ decreases from 2.5 to 0.5, while $c_2$ increases from 0.5 to 2.5. For PSO-ILSSS and PSO-INLSSS, cognitive parameter $c_1$ is fixed to 2.0, while social parameter $c_2$ is decreased, whereas the lower bounds of $c_2$ is set to 1.0, and the upper bounds is set from 2.0 decreased to 1.0. Total individuals are 100, and the velocity threshold $v_{max}$ is set to the upper bound of the domain. The dimensionality is 30 and 50. In each experiment, the simulation run 30 times, while each time the largest iteration is $50 \times \text{dimension}$.

5.4.3 Performance Analysis
The comparison results of these three famous benchmarks are listed as Table 12-14, in which Dim. represents the dimension, Alg. represents the corresponding algorithm, Mean denotes the average mean value, while STD denotes the standard variance.

For Rastrigin Function (Table 12), the performances of all non-linear PSO-INLSSS algorithms are worse than PSO-ILSSS when dimension is 30, although they are better than SPSO and MPSO-TVAC. However, with the increased dimensionality, the performance of non-linear modified variant PSO-INLSSS surpasses that of PSO-ILSSS, for example, the best performance is achieved by PSO-INLSSS-3. This phenomenon implies that non-linear strategies can exactly affect the performance.

For Ackley Function (Table 13) and Penalized Function (Table 14), the performance of PSO-INLSSS-3 always wins. Based on the above analysis, we can draw the following two conclusions:
PSO-INLSSS-3 is the most stable and effective among four non-linear strategies. It is especially suit for multi-modal functions with many local optima especially.

<table>
<thead>
<tr>
<th>Dim.</th>
<th>Alg.</th>
<th>Mean</th>
<th>STD</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>SPSO</td>
<td>1.7961e+001</td>
<td>4.2276e+000</td>
</tr>
<tr>
<td>30</td>
<td>MPSO-TVAC</td>
<td>1.5471e+001</td>
<td>4.2023e+000</td>
</tr>
<tr>
<td></td>
<td>PSO-ILSSS</td>
<td>6.4012e+000</td>
<td>5.0712e+000</td>
</tr>
<tr>
<td></td>
<td>PSO-INLSSS-1</td>
<td>6.8676e+000</td>
<td>3.1269e+000</td>
</tr>
<tr>
<td></td>
<td>PSO-INLSSS-2</td>
<td>8.2583e+000</td>
<td>2.3475e+000</td>
</tr>
<tr>
<td></td>
<td>PSO-INLSSS-3</td>
<td>8.6868e+000</td>
<td>1.7600e+000</td>
</tr>
<tr>
<td></td>
<td>PSO-INLSSS-4</td>
<td>1.0755e+001</td>
<td>4.2686e+000</td>
</tr>
<tr>
<td>50</td>
<td>SPSO</td>
<td>3.9958e+001</td>
<td>7.9258e+000</td>
</tr>
<tr>
<td></td>
<td>MPSO-TVAC</td>
<td>3.8007e+001</td>
<td>7.0472e+000</td>
</tr>
<tr>
<td></td>
<td>PSO-ILSSS</td>
<td>1.5380e+001</td>
<td>5.8287e+000</td>
</tr>
<tr>
<td></td>
<td>PSO-INLSSS-1</td>
<td>1.4329e+001</td>
<td>4.7199e+000</td>
</tr>
<tr>
<td></td>
<td>PSO-INLSSS-2</td>
<td>1.5623e+001</td>
<td>4.4020e+000</td>
</tr>
<tr>
<td></td>
<td>PSO-INLSSS-3</td>
<td>1.3740e+001</td>
<td>4.3426e+000</td>
</tr>
<tr>
<td></td>
<td>PSO-INLSSS-4</td>
<td>2.1975e+001</td>
<td>5.6844e+000</td>
</tr>
</tbody>
</table>

Table 12. Comparison Results for Rastrigin Function
6. Conclusion and Future Research

This chapter proposes a new model incorporated with the characteristic differences for each particle, and the individual selection strategy for inertia weight, cognitive learning factor and social learning factor are discussed, respectively. Simulation results show the individual selection strategy maintains a fast search speed and robust. Further research should be made on individual structure for particle swarm optimization.

<table>
<thead>
<tr>
<th>Dim.</th>
<th>Alg.</th>
<th>Mean</th>
<th>STD</th>
</tr>
</thead>
<tbody>
<tr>
<td>30</td>
<td>SPSO</td>
<td>5.8161e-006</td>
<td>4.6415e-006</td>
</tr>
<tr>
<td></td>
<td>MPSO-TVAC</td>
<td>7.5381e-007</td>
<td>3.3711e-006</td>
</tr>
<tr>
<td></td>
<td>PSO-INLSSS-1</td>
<td>1.8094e-011</td>
<td>1.8533e-011</td>
</tr>
<tr>
<td></td>
<td>PSO-INLSSS-2</td>
<td>1.1870e-011</td>
<td>2.0876e-011</td>
</tr>
<tr>
<td></td>
<td>PSO-INLSSS-3</td>
<td>5.2100e-013</td>
<td>5.5185e-013</td>
</tr>
<tr>
<td></td>
<td>PSO-INLSSS-4</td>
<td>3.2118e-010</td>
<td>2.2272e-010</td>
</tr>
<tr>
<td>50</td>
<td>SPSO</td>
<td>1.7008e-004</td>
<td>1.2781e-004</td>
</tr>
<tr>
<td></td>
<td>MPSO-TVAC</td>
<td>4.4132e-002</td>
<td>1.9651e-001</td>
</tr>
<tr>
<td></td>
<td>PSO-ILSSS</td>
<td>1.5870e-008</td>
<td>1.7852e-008</td>
</tr>
<tr>
<td></td>
<td>PSO-INLSSS-1</td>
<td>2.3084e-008</td>
<td>3.6903e-008</td>
</tr>
<tr>
<td></td>
<td>PSO-INLSSS-2</td>
<td>1.1767e-008</td>
<td>1.3027e-008</td>
</tr>
<tr>
<td></td>
<td>PSO-INLSSS-3</td>
<td>4.7619e-010</td>
<td>1.4337e-009</td>
</tr>
<tr>
<td></td>
<td>PSO-INLSSS-4</td>
<td>3.4499e-008</td>
<td>4.7674e-008</td>
</tr>
</tbody>
</table>

Table 13. Comparison Results for Ackley Function

<table>
<thead>
<tr>
<th>Dim.</th>
<th>Alg.</th>
<th>Mean</th>
<th>STD</th>
</tr>
</thead>
<tbody>
<tr>
<td>30</td>
<td>SPSO</td>
<td>5.4943e-004</td>
<td>2.4568e-003</td>
</tr>
<tr>
<td></td>
<td>MPSO-TVAC</td>
<td>9.3610e-027</td>
<td>4.1753e-026</td>
</tr>
<tr>
<td></td>
<td>PSO-ILSSS</td>
<td>5.1601e-023</td>
<td>1.7430e-022</td>
</tr>
<tr>
<td></td>
<td>PSO-INLSSS-1</td>
<td>6.0108e-020</td>
<td>1.5299e-019</td>
</tr>
<tr>
<td></td>
<td>PSO-INLSSS-2</td>
<td>4.5940e-021</td>
<td>6.2276e-021</td>
</tr>
<tr>
<td></td>
<td>PSO-INLSSS-3</td>
<td>9.7927e-024</td>
<td>1.6162e-023</td>
</tr>
<tr>
<td></td>
<td>PSO-INLSSS-4</td>
<td>1.0051e-016</td>
<td>1.9198e-016</td>
</tr>
<tr>
<td>50</td>
<td>SPSO</td>
<td>6.4279e-003</td>
<td>1.0769e-002</td>
</tr>
<tr>
<td></td>
<td>MPSO-TVAC</td>
<td>4.9270e-002</td>
<td>2.0248e-001</td>
</tr>
<tr>
<td></td>
<td>PSO-ILSSS</td>
<td>1.6229e-017</td>
<td>3.9301e-017</td>
</tr>
<tr>
<td></td>
<td>PSO-INLSSS-1</td>
<td>6.2574e-015</td>
<td>1.3106e-014</td>
</tr>
<tr>
<td></td>
<td>PSO-INLSSS-2</td>
<td>1.6869e-014</td>
<td>3.3596e-014</td>
</tr>
<tr>
<td></td>
<td>PSO-INLSSS-3</td>
<td>6.2959e-018</td>
<td>5.6981e-018</td>
</tr>
<tr>
<td></td>
<td>PSO-INLSSS-4</td>
<td>8.0886e-013</td>
<td>3.7972e-013</td>
</tr>
</tbody>
</table>

Table 14. Comparison Results for Penalized Function
7. Acknowledgement

This chapter is supported by National Natural Science Foundation of China under Grant No.60674104, and also supported by Doctoral Scientific Research Starting Foundation of Taiyuan University of Science and Technology under Grant No.20082010.

8. References


Particle swarm optimization (PSO) is a population based stochastic optimization technique influenced by the social behavior of bird flocking or fish schooling. PSO shares many similarities with evolutionary computation techniques such as Genetic Algorithms (GA). The system is initialized with a population of random solutions and searches for optima by updating generations. However, unlike GA, PSO has no evolution operators such as crossover and mutation. In PSO, the potential solutions, called particles, fly through the problem space by following the current optimum particles. This book represents the contributions of the top researchers in this field and will serve as a valuable tool for professionals in this interdisciplinary field.

How to reference
In order to correctly reference this scholarly work, feel free to copy and paste the following:

© 2009 The Author(s). Licensee IntechOpen. This chapter is distributed under the terms of the Creative Commons Attribution-NonCommercial-ShareAlike-3.0 License, which permits use, distribution and reproduction for non-commercial purposes, provided the original is properly cited and derivative works building on this content are distributed under the same license.