Turkish Journal of Electrical Engineering and Computer Sciences

Volume 24 | Number 2 Article 22

1-1-2016

A generalized particle swarm optimization using reverse direction information

EMRE ÇOMAK

Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2016) 24: 639 – 655

doi:10.3906/elk-1306-39

A generalized particle swarm optimization using reverse direction information

Emre ÇOMAK*
Department of Computer Engineering, Pamukkale University, Denizli, Turkey

Received: 05.06.2013 • **Accepted/Published Online:** 30.12.2013 • **Final Version:** 05.02.2016

Abstract: In this study, an improved particle swarm optimization (PSO) algorithm, including 4 types of new velocity updating formulae (each is equal to the traditional PSO), was introduced. This algorithm was called the reverse direction supported particle swarm optimization (RDS-PSO) algorithm. The RDS-PSO algorithm has the potential to extend the diversity and generalization of traditional PSO by regulating the reverse direction information adaptively. To implement this extension, 2 new constants were added to the velocity update equation of the traditional PSO, and these constants were regulated through 2 alternative procedures, i.e. max–min-based and cosine amplitude-based diversity-evaluating procedures. The 4 most commonly used benchmark functions were used to test the general optimization performances of the RDS-PSO algorithm with 3 different velocity updates, RDS-PSO without a regulating procedure, and the traditional PSO with linearly increasing/decreasing inertia weight. All PSO algorithms were also implemented in 4 modes, and their experimental results were compared. According to the experimental results, RDS-PSO 3 showed the best optimization performance.

Key words: Particle swarm optimization, diversity regulation, cosine amplitude, max–min

1. Introduction

Particle swarm optimization (PSO) is a biologically and sociologically inspired swarm intelligence method. The authors of [1] introduced it as a search and optimization method. The modeling of PSO is based on the simulation of certain social animal behaviors.

PSO begins the algorithm with a randomly created population, similar to many evolutionary computation methods. According to the information shared between particles, the next positions of each particle are computed iteratively. Unlike other mathematical algorithms, PSO does not require gradient information. It has acquired increasing popularity due to its superior properties, such as convergence to optimal solutions quickly, adaptation to different problems easily, and its basic implementation. Therefore, many PSO algorithms and many hybrid methods including PSO have been implemented in numerous fields such as power systems [2,3], field effect transistor design [4], and team-orienteering problems [5].

Despite these superior properties, the general performance of PSO is not as good as what other optimization methods can accomplish. Since the PSO algorithm is conducted on the particles by considering only their personal best position and the best position of the population, PSO could not converge to the global optimal solution in optimization problems involving many local optimal solutions. This directs all particles to a local optimal solution [6]. Besides, PSO cannot enhance its convergence ability, especially in its last iterations, and

*Correspondence: ecomak@pau.edu.tr

for this reason it converges to the local optimal solution prematurely, i.e. by conducting a way of PSO for the particles.

In recent years, PSO has also been implemented for solving dynamic and multiobjective optimization problems. PSO used in dynamic optimization problems can adapt to dynamically changing optimization conditions [7]. To add this property to PSO, new approaches were integrated into the traditional method [8]. Traditional PSO cannot solve a multiobjective optimization problem such as dynamic optimization problems. Two improvements must be integrated into the traditional PSO to solve such problems. The first is related to the selection of the global and local best particles for velocity updating. The second concerns how to maintain information about these particles [9].

There are numerous studies in the literature about how to design improved PSO approaches that solve different types of optimization problems. The authors of [10] suggested integrating genetic operators with improved binary particle swarm optimization (IBPSO) to attain solutions with more alternatives and thus reduce the risk of premature convergence. In addition, IBPSO was carried out in a maintenance scheduling problem. A modified binary particle swarm optimization (MBPSO) algorithm was demonstrated for solving knapsack problems (KPs) in [11]. Unlike traditional BPSO, it was based on a probability function, prolonging the diversity in the swarm and supporting PSO to be more explorative, effective, and efficient in the KPs. When the cost functions of the searching spaces are not known well, finding the most suitable PSO model subjects the searching process to many computational burdens. To avoid this demerit, [12] introduced a self-adaptive learning-based PSO, and [13] proposed a hybrid approach to integrating PSO and the differential evolution algorithm so as to efficiently conduct the positions of the next generations and improve convergence.

In order to reduce loss of diversity in the first generations of PSO, diversity augmentation and neighborhood-based search strategies were developed in [14]. These strategies supply a trade-off between the exploration and exploitation capacities of the population. In [15], the authors handled every particle's dimension individually instead of handling it as a whole. To prevent premature convergence and loss of diversity in PSO, a new idea based on an analysis of visual modeling and 2 parameters (particle distribution degree and particle dimension distance) was asserted. A rank-based particle swarm optimizer, PSO_{rank}, changed the selection strategy of the global best particle in PSO [16]. It determined a group of best particles according to a strength function for updating the velocities of the remaining particles.

In addition to the studies discussed in the above paragraphs, several other studies were dedicated to regulating the trade-off between the global and local searching ability of PSO. For example, [17] attached an inertia weight parameter to PSO to regulate this trade-off. As the dynamic adjustment of the inertia weight is very difficult, [18] used linearly decreasing inertia weight, whereas [19,20] used linearly increasing inertia weight. According to the results of these studies, the general performance of PSO depends not only on changing the inertia weight, but also on the type of optimization problem. A similar study of reverse direction supported particle swarm optimization (RDS-PSO) introduced antipredatory activity to the original PSO and was proposed to solve economic dispatch problems in [21]. However, both the cognitive and social parts of the velocity update equation were divided into 2 parts, and there was no procedure controlling the diversity of the population in that study.

Most studies in the literature are dedicated to avoiding the loss of diversity in progressive iterations and thus to improving the convergence ability of PSO. This study aims to improve the same targets and handles this subject as a whole set of the population distribution, changing the effects of the inertia weight and sharing types of information among particles. Although the traditional PSO algorithm controls the movement of the

particles by depending only on their personal best and the population's best positions, proposed algorithms such as RDS-PSO control it by depending on their personal best and worst, and also on the population's best and worst positions. By adjusting the trade-off between these 4 positions, an RDS-PSO algorithm with 4 different formulas was created. In addition, 2 alternative procedures were used to implement the RDS-PSO algorithm adaptively. Finally, the effects of linearly changed inertia weight on RDS-PSO were analyzed through experimental results.

The remaining of the study is organized as follows. A superficial description of the PSO, the cosine amplitude method, and the max–min method is presented in Section 2. Both the proposed RDS-PSO algorithm and its adaptive regulation procedure are explained in Section 3. Section 4 gives the experimental results of all algorithms, including the RDS-PSO algorithm and the traditional PSO with linearly increasing/decreasing inertia weight in 4 modes. Section 5 discusses and concludes the study.

2. Materials and methods

Two alternative regularization methods (the cosine amplitude and max–min methods) that are used to control the values of the alpha and beta in RDS-PSO and the traditional PSO are clarified in this section.

Many techniques are available for computing numerical values that describe a relationship between 2 or more data points. Such techniques are generally based on Cartesian products, linguistic rules, classification, and data manipulation. In this study, 2 alternative methods, the cosine amplitude and max–min methods, which do not incur much computational cost and belong to the data manipulation category, were used to regulate the alpha and beta constants of RDS-PSO.

2.1. Cosine amplitude

The cosine amplitude method is manipulated on a collection of n data samples. The collected data set is represented as follows:

$$\mathbf{X} = \{\mathbf{x_1}, \mathbf{x_2}, ..., \mathbf{x_n}\} \tag{1}$$

Each of the n samples is represented as a vector with m dimensions:

$$\mathbf{x_i} = \{\mathbf{x_{i1}}, \mathbf{x_{i2}}, ..., \mathbf{x_{im}}\} \tag{2}$$

The position of each datum in space is represented by m feature values. Relation value r_{ij} reflects a similarity relationship between x_i and x_j data. For n data samples, the size of the relation matrix will be n $\times$ n. The relation matrix always obeys the rules of reflexivity and symmetricity, and thus it is a tolerance relation. All r_{ij} values are always in the interval of [0,1] in this method, and they are calculated through the following equation:

$$r_{ij} = \frac{\left|\sum_{k=1}^{m} x_{ik}x_{jk}\right|}{\sqrt{\left(\sum_{k=1}^{m} x_{ik}^2\right)\left(\sum_{k=1}^{m} x_{jk}^2\right)}}, \quad i,j = 1,2,...,n. \tag{3}$$

If x_i and x_j are very similar to each other, r_{ij} becomes close to one. Unlike this situation, if they are very dissimilar to each other, r_{ij} becomes close to zero.

2.2. Max–min method

Another method that is an alternative to the cosine amplitude method is the max–min method. The max–min method is preferred to the cosine amplitude method due to computational cost. Except for the calculation of r_{ij}, this method is similar to the cosine amplitude method. In this method, r_{ij} is calculated through the following equation:

$$r_{ij} = \frac{\sum_{k=1}^{m} \min(x_{ik}, x_{jk})}{\sum_{k=1}^{m} \max(x_{ik}, x_{jk})}, \quad i, j = 1, 2, ..., n. \tag{4}$$

2.3. PSO

PSO is a modeling of sociological and biological populations [1]. In PSO, a population comprises N particles and each particle is a step to close in on the target solution. Since the searching space is the D-dimensional coordinate axis, each particle is represented as a vector with the number of D parts. The actual position of the ith particle is represented by $x_i = (x_{i1}, x_{i2}, ..., x_{iD})$, and its velocity is represented by $v_i = (v_{i1}, v_{i2}, ..., v_{iD})$. Moreover, $p_i = (p_{i1}, p_{i2}, ..., p_{iD})$ indicates the best visited position for the ith particle until time t, and $p_g = (p_{g1}, p_{g2}, ..., p_{gD})$ indicates the best visited position among N particles or local neighbors of p_i (depending on the type of information-sharing between particles) until time t. This time index can also be accepted as an iteration number. The movement of the particles in the search space is conducted by Eqs. (5) and (6), iteratively:

$$v_{id} = w * v_{id} + c_1 * rand1(\,) * (p_{id} - x_{id}) + c_2 * rand2(\,) * (p_{gd} - x_{id}) \tag{5}$$

$$x_{id} = x_{id} + v_{id} \tag{6}$$

Here, $i = 1, 2, ..., N$ and $d = 1, 2, ..., D$. Parameters c_1 and c_2 are positive constants denoting cognitive and social impacts in the PSO, respectively. The functions of rand1 and rand2 generate random numbers in the interval of [0,1] uniformly. Positive parameter w is the inertia weight. As discussed in Section 1, this parameter regulates the trade-off between the global and local searching abilities of the PSO. Moreover, variables p_{id}, p_{gd}, and x_{id} indicate the personal best, global best, and present position of the particles, respectively.

PSO starts the solution with a randomly created population and velocities, as in many population-based methods. The velocities and next positions of the particles are determined iteratively by using Eqs. (5) and (6). Iteration number, improvement, and stability-based criteria have been used to terminate the PSO algorithm. As an exceptional cases, V_{max} value is determined in PSO to prevent particles from exceeding the search space.

3. RDS-PSO algorithms

RDS-PSO was structured according to the generalization idea. Two constants, alpha and beta, were added to the traditional PSO to provide such a generalization. The parameter of alpha controls the trade-off between the effects of the global best and global worst particles on velocity updating. On the other hand, beta controls the trade-off between the effects of the personal best and personal worst particles on velocity updating. When RDS-PSO is implemented with alpha = 1 and beta = 1, the traditional PSO is created. Unlike this situation, when RDS-PSO is implemented with alpha = 0 and beta = 0, the velocities of all particles are controlled only by their global worst and personal worst particles (but in the reverse direction).

Such a generalization offers a controllable diversity management to PSO. If changing the values of these constants is managed properly, the diversity and thus the convergence ability of PSO might be improved.

Four types of RDS-PSO were designed based on the different organization of the alpha and beta constants:

- RDS-PSO 1: This is the type of RDS-PSO with alpha = beta = 1; it can be called traditional PSO.
- RDS-PSO 2: This is the type of RDS-PSO with alpha [0,1] and beta = 1.
- RDS-PSO 3: This is the type of RDS-PSO with alpha = 1 and beta [0,1].
- RDS-PSO 4: This is the type of RDS-PSO with alpha [0,1] and beta [0,1].

By designing 4 types of RDS-PSO algorithms, the effects of the personal worst and the global worst on the general convergence ability of PSO are investigated completely. The fundamental difference between RDS-PSO and PSO lies in the way in which the velocity update is achieved. The velocity updating equation for Algorithm 4 can be presented as follows:

$$v_{id} = w * v_{id} + beta * c_1 * rand1(\,) * (p_{id} - x_{id}) + (1 - beta) * c_1 * rand1(\,) * (x_{id} - p_{iwd}) + alpha * c_2 * rand2(\,) * (p_{gd} - x_{id}) + (1 - alpha) * c_2 * rand2(\,) * (x_{id} - p_{gwd}) \quad (7)$$

RDS-PSO uses Eq. (7) instead of Eq. (5). Some parameters that are not presented in Eq. (5) are available in Eq. (7). For instance, p_{gwd} and p_{iwd} represent the global worst particle in the population and the personal worst position of the ith particle, respectively. Alpha and beta are also new added constants to Eq. (7). Their functions were described in the above paragraph. In fact, p_{gwd} and p_{iwd} affect the next positions of all particles in the reverse direction. Figure 1 illustrates the velocity update of the traditional PSO, and Figure 2 illustrates the velocity update for the RDS-PSO with alpha = 0.5 (without considering the personal best and worst particles). When the diversity of the population starts to decrease significantly, a regularization procedure decreasing the value of alpha can be executed to increase diversity.

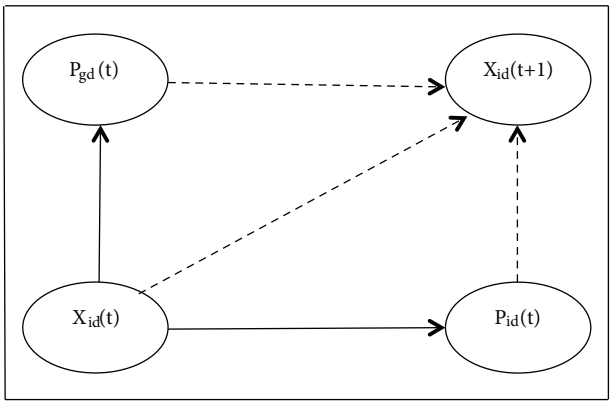

Figure 1. Velocity update of the original PSO.

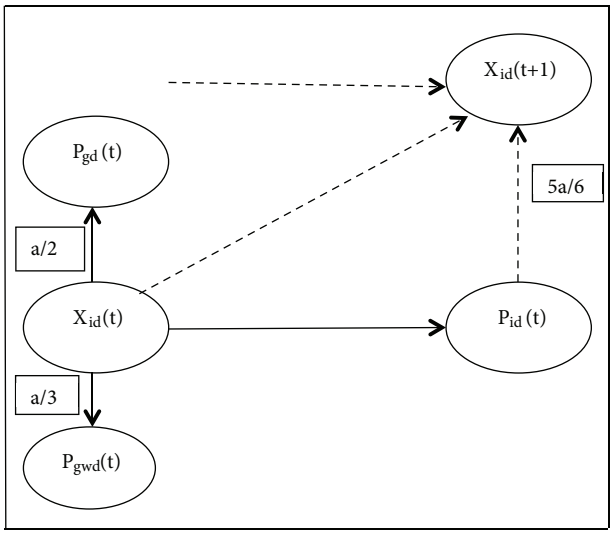

Figure 2. Velocity update of the RDS-PSO.

In this study, the constants of RDS-PSO were also changed adaptively by using the cosine amplitude and max–min methods. Without these methods, which provide adaptability, the computational cost of RDS-PSO

is almost identical to that of the traditional PSO. These adaptive methods bring a little more computational cost to RDS-PSO. The pseudocode of RDS-PSO is illustrated in Figure 3.

RDS-PSO Algorithm with Adaptive Regulation

```
// Initialization
  for i=1 to population size
    set Xi particle in interval of [Xmin, Xmax] randomly
    set Vi velocity in interval of [Vmin, Vmax] randomly
    Pi = Xi
  endfor
  Evaluate all population
  Determine the best particle pg and the worst particle pw
  alpha = beta = 1
// PSO iteration loop
  while (termination criterion is not satisfied & iteration < maximum iteration)
    iteration = iteration + 1
    compute the diversity matrix of population by cosine amplitude or max-min
    compute average diversity value and save it
    if (iteration > 1)
      if (average diversity (iteration) – average diversity (iteration - 1)) > C
        alpha = alpha – ssize
        beta = beta – ssize
      elseif (average diversity (iteration) – average diversity (iteration - 1)) < -C
        alpha = alpha + ssize
        beta = beta + ssize
      else
        alpha = alpha
        beta = beta
      endif
    endif
    if (alpha > 1)
      alpha = 1
    elseif (alpha < 0.6)
      alpha = 0.6
    endif
    if (beta > 1)
      beta = 1
    elseif (beta <0.6)
      beta = 0.6
    endif
    update inertia weight (decrease or increase linearly)
    update velocity according to new equation, update new position of particle
    Evaluate all population
    update pg, pi and pw
  endwhile
```

Figure 3. Pseudocode of the RDS-PSO with adaptive regulation procedure.

3.1. Regulation procedure for the constants of alpha and beta

As distinct from the original PSO, a few constants were added to RDS-PSO. Two of these constants are alpha and beta, regulating the trade-off between the global best/worst particles and personal best/worst particles, respectively. For instance, when alpha = 0 and beta = 1, only the global worst and personal best particles affect the next positions. Another pair of constants are ssize and C, representing step size and threshold, respectively. Step size is a measure of change in alpha and beta in each iteration. According to the C constant, ssize is subtracted from or added to alpha and beta.

A schedule to conduct the values of the alpha and beta constants is introduced in this part in order to set these constants adaptively and improve the general performance of RDS-PSO. The pseudocode of the schedule was illustrated in the lines between 15 and 36 in Figure 3.

According to the author's previous experiments, RDS-PSO without such a schedule showed its best performance with the alpha and beta constants in the range of [0.6,1.0]. Therefore, the values for alpha and beta outside this range could not be allocated to these constants [22]. Average diversity value in pseudocode was computed by 2 alternative methods explained in Sections 2.1 and 2.2.

Each RDS-PSO type was carried out in 8 modes (4 in the cosine amplitude method and 4 in the max–min method). In other words, each initial population was implemented 32 times (for 4 types and 8 modes). These implementation modes are explained below:

- Mode 1: In this mode, the RDS-PSO algorithm was implemented with 1000 maximum iterations and decreasing inertia weight.
- Mode 2: In this mode, the RDS-PSO algorithm was implemented with 2000 maximum iterations and decreasing inertia weight.
- Mode 3: In this mode, the RDS-PSO algorithm was implemented with 1000 maximum iterations and increasing inertia weight.
- Mode 4: In this mode, the RDS-PSO algorithm was implemented with 2000 maximum iterations and increasing inertia weight.

To clarify the RDS-PSO types and implementation modes, an example can be given. Let us assume that we select alpha and beta constants as alpha [0,1] and beta [0,1] (referring to type 4). In addition to this selection, we implement the algorithm with 1000 maximum iterations and increasing inertia weight (referring to Mode 3). Thus, it can be said that RDS-PSO 4 is carried out in Mode 3.

4. Benchmark functions and experimental results

Benchmark functions used to test the performance of RDS-PSO algorithms are explained in Section 4.1, and experimental results are presented in Section 4.2.

4.1. Benchmark functions

To compare the performances of RDS-PSO algorithms (RDS-PSO 1 is similar to the traditional PSO with linearly decreasing/increasing inertia weight), the 4 most commonly used benchmark functions were preferred. Table 1 summarizes the basic properties of these benchmark functions.

Terms lb and ub in Table 1 represent the lower bound and upper bound, respectively.

Table 1. Properties of the benchmark functions.

Function Name	lb	ub	Optimum point	Modality
Griewangk	–600	600	0	Multimodal
Rastrigin	–5.12	5.12	0	Multimodal
Rosenbrock	–2.048	2.048	0	Unimodal
Sphere	–100	100	0	Unimodal
Ackley	–32.768	32.768	0	Multimodal

4.1.1. Griewangk function

The Griewangk function has many local optima. Consequently, finding its global optimum point is not easy [23]. This function is described in Eq. (8):

$$f(\mathbf{x}) = \sum_{i=1}^{30} \left(\frac{x_i^2}{4000} \right) - \prod_{i=1}^{30} \cos\left(\frac{x_i}{\sqrt{i}} \right) + 1 \tag{8}$$

where $x_i \in [-600, 600]$, the global minimal point of the function is at x = (0, 0, ..., 0), and f(x) = 0.

4.1.2. Rastrigin function

This function is obtained by adding the cosine component to the De Jong function. Such a component makes it highly multimodal [24], and it is defined in Eq. (9):

$$f(\mathbf{x}) = 10 \times 30 + \sum_{i=1}^{30} \left(x_i^2 - 10 \cdot \cos(2\pi x_i) \right) \tag{9}$$

where $x_i \in [-5.12, 5.12]$, the global minimal point of the function is at x = (0, 0, ..., 0), and f(x) = 0.

4.1.3. Rosenbrock function

The Rosenbrock function is also known as the banana function due to its shape. Owing to the difficulty in converging its global optimal, this function is commonly used to test the performances of many optimization algorithms [25]. It is described in Eq. (10):

$$f(\mathbf{x}) = \sum_{i=1}^{30} 100 \left(x_{i+1} - x_i^2 \right)^2 + (1 - x_i)^2 \tag{10}$$

where $x_i \in [-2.048, 2.048]$, the global minimal point of the function is at x = (1, 1, ..., 1), and f(x) = 0.

4.1.4. Ackley function

The Ackley function is also one of the most commonly used test functions in optimization. It is defined in Eq. (11):

$$f(\mathbf{x}) = -a \cdot \exp\left(-b \cdot \sqrt{\frac{1}{n} \sum_{i=1}^{n} x_i^2} \right) - \exp\left(\frac{1}{n} \sum_{i=1}^{n} \cos(cx_i) \right) + a + \exp(1) \tag{11}$$

where $x_i \in [-32.768, 32.768]$, a = 20, b = 0.2, and $c = 2\pi$. The global minimal point of the function is at x = (0, 0, ..., 0), and f(x) = 0.

4.1.5. Sphere function

The sphere function is also one of the most commonly used test functions in optimization. It is defined in Eq. (12):

$$f(\mathbf{x}) = \sum_{i=1}^{10} x_i^2 \tag{12}$$

where $x_i \in [-100.0, 100.0]$. The global minimal point of the function is at x = (0, 0, ..., 0), and f(x) = 0.

4.2. Experimental results

The performances of the RDS-PSO algorithm were compared to the traditional PSO with linearly decreasing/increasing inertia weight, and to RDS-PSO with static alpha value in 4 modes and 4 algorithms. MATLAB was used as the implementation platform. By comparing the results of RDS-PSO with adaptive procedures to RDS-PSO with static alpha value, the efficiency of the regularization methods, i.e. cosine amplitude and max–min methods, was evaluated. The effects of various approaches, such as the linear change of inertia weight value, global/personal best and global/personal worst particles, and cosine amplitude-based or max–min-based regularization for the alpha and beta constants on the general performance of PSO were detected for 4 commonly used benchmark functions. Thus, comprehensive experiments were carried out on the mentioned benchmark functions. Table 2 describes the shared properties of the variables that were used with the same value in all compared algorithms. All compared methods were executed 50 times, i.e. with 50 different initial conditions, but the same 50 for all methods. Therefore, the average and standard deviation data were computed objectively.

Table 2. Variable values for all PSO methods.

Parameter	Value
Population size	25
Maximal iteration	1000/2000
Maximal weight value	1.2
Minimal weight value	0.1
C_1	2.0
C_2	2.0
Dimension	10
Error goal	$1*10^{-6}$

The value of alpha changed from 0.05 to 1.0 with a 0.05 step size, and the best result among these 20 results was determined as a final result in RDS-PSO with static alpha value. In RDS-PSO with adaptive procedures, 20 experimental results (10 with static step constant and variable C constant, and another 10 with the opposite) were also obtained. For changing intervals of the variable constant, the value was set between 0.01 and 0.1, and a value of 0.05 was set for the static constant. The best performance was determined in the RDS-PSO with the static alpha value. The final results, indicated in the relevant figures, were also computed in this way.

Tables 3–7 indicate the experimental results for the Rosenbrock, Rastrigin, Griewangk, Ackley, and Sphere benchmark functions, respectively. All methods were implemented in 4 modes. The modes with different iteration numbers and inertia weight-changing strategies were discussed in Section 3.1. In addition, constant alpha values for the best performance were presented in parentheses for RDS-PSO with the constant alpha value method.

RDS-PSO 3, using the max–min method, provided the best results in Modes 1, 3, and 4. However, in Mode 2, RDS-PSO 3, using the cosine amplitude method, was the best for the Rosenbrock function, as detailed in Table 3. The worst results for this function were obtained in the RDS-PSO with constant alpha value and in the traditional PSO with the linearly changing inertia weight methods.

As detailed in Table 4, RDS-PSO 3 using the max–min method provided the best results in Modes 2, 3, and 4. However, in Mode 1, RDS-PSO 3 using the cosine amplitude method provided the best results. Similarly to the Rosenbrock function results, the worst results for the Rastrigin function were obtained in RDS-PSO with constant alpha value and in traditional PSO with linearly changing inertia weight methods. For all methods, the Rosenbrock function results were better than the Rastrigin results.

Table 3. Results of the Rosenbrock function.

Method	Implementation mode	Average best fitness	Standard deviation
RDS-PSO 1 (traditional PSO with linearly dec/inc w)	Mode 1 (5.61 s)	10.271	10.102
	Mode 2 (9.92 s)	6.7966	1.9175
	Mode 3 (7.74 s)	82.034	54.526
	Mode 4 (13.56 s)	91.338	61.71
RDS-PSO with constant alpha	Mode 1 (5.69 s)	23.123 (0.95)	21.315 (0.95)
	Mode 2 (9.88 s)	19.451 (0.9)	18.205 (0.9)
	Mode 3 (7.86 s)	142.65 (0.7)	138.38 (0.7)
	Mode 4 (13.77 s)	95.203 (0.95)	64.71 (0.95)
RDS-PSO 2 (max–min)	Mode 1 (6.61 s)	6.7702	1.8599
	Mode 2 (11.13 s)	5.3516	2.4352
	Mode 3 (7.01 s)	11.835	10.899
	Mode 4 (11.64 s)	8.838	9.4348
RDS-PSO 2 (cosine amplitude)	Mode 1 (8.06 s)	6.0067	1.7905
	Mode 2 (15.77 s)	5.0138	2.2994
	Mode 3 (7.93 s)	66.399	46.244
	Mode 4 (15.44 s)	84.357	61.957
RDS-PSO 3 (max–min)	Mode 1 (6.82 s)	2.1678	1.9446
	Mode 2 (11.52 s)	1.7604	1.4560
	Mode 3 (7.14 s)	4.9949	4.4553
	Mode 4 (12.21 s)	4.8297	5.7400
RDS-PSO 3 (cosine amplitude)	Mode 1 (8.13 s)	2.4372	1.8678
	Mode 2 (16.15 s)	1.3787	1.4498
	Mode 3 (8.19 s)	68.8936	50.2128
	Mode 4 (16.23 s)	88.5538	62.9101
RDS-PSO 4 (max-min)	Mode 1 (7.21 s)	6.0122	1.8446
	Mode 2 (12.08 s)	5.1950	2.7938
	Mode 3 (7.82 s)	11.686	13.991
	Mode 4 (12.67 s)	8.1571	7.6614
RDS-PSO 4 (cosine amplitude)	Mode 1 (8.57 s)	5.9690	1.6221
	Mode 2 (17.44 s)	5.1275	1.8526
	Mode 3 (8.68 s)	64.033	48.338
	Mode 4 (17.89 s)	87.721	62.791
Ladder PSO (LPSO)	—	1.495	1.584
Mutation PSO (MPSO)	—	2.077	1.992
Linearly decreasing inertia weight PSO	—	3.721	3.054

Table 5 presents the results of the Griewangk function. According to this table, RDS-PSO 3 (max–min), RDS-PSO 3 (cosine amplitude), and RDS-PSO 4 (max–min) methods produced the best results in Modes 1, 2, and 3 and 4 together, respectively. RDS-PSO with constant alpha value and the traditional PSO with linearly changing inertia weight methods showed the poorest results in Modes 1-2 and 3-4 together for this function, respectively.

Table 4. Results of the Rastrigin function.

Method	Implementation mode	Average best fitness	Standard deviation
RDS-PSO 1 (traditional PSO with linearly dec/inc w)	Mode 1 (5.12 s)	23.266	8.6419
	Mode 2 (9.55 s)	22.707	13.931
	Mode 3 (7.08 s)	47.313	17.193
	Mode 4 (12.79 s)	45.939	18.918
RDS-PSO with constant alpha	Mode 1 (5.48 s)	45.364 (0.9)	11.061 (0.9)
	Mode 2 (9.68 s)	48.152 (0.9)	13.542 (0.9)
	Mode 3 (7.94 s)	69.287 (0.75)	17.086 (0.75)
	Mode 4 (14.04 s)	69.862 (0.95)	15.113 (0.95)
RDS-PSO 2 (max–min)	Mode 1 (6.17 s)	16.853	4.8218
	Mode 2 (10.78 s)	17.175	6.5709
	Mode 3 (6.77 s)	29.9	9.2769
	Mode 4 (10.84 s)	24.741	8.0494
RDS-PSO 2 (cosine amplitude)	Mode 1 (7.29 s)	13.62	5.3646
	Mode 2 (14.55 s)	13.396	7.0449
	Mode 3 (7.88 s)	42.552	17.552
	Mode 4 (14.93 s)	42.483	16.153
RDS-PSO 3 (max–min)	Mode 1 (6.41 s)	5.1775	1.6570
	Mode 2 (11.49 s)	4.2010	1.5512
	Mode 3 (7.05 s)	13.754	6.6559
	Mode 4 (11.94 s)	13.633	6.7645
RDS-PSO 3 (cosine amplitude)	Mode 1 (8.11 s)	4.9831	1.64
	Mode 2 (16.02 s)	4.5245	1.6398
	Mode 3 (7.88 s)	40.3357	14.251
	Mode 4 (15.85 s)	43.0649	17.9849
RDS-PSO 4 (max–min)	Mode 1 (7.08 s)	16.667	5.5726
	Mode 2 (12.27 s)	16.302	6.3418
	Mode 3 (7.25 s)	25.719	6.6014
	Mode 4 (12.14 s)	22.776	7.5412
RDS-PSO 4 (cosine amplitude)	Mode 1 (8.20 s)	15.776	6.8598
	Mode 2 (16.33 s)	13.603	6.8112
	Mode 3 (8.45 s)	43.617	16.025
	Mode 4 (17.12 s)	43.126	16.88
Ladder PSO (LPSO)	—	4.509	1.79
Mutation PSO (MPSO)	—	7.538	2.33
Linearly decreasing inertia weight PSO	—	6.066	2.46

RDS-PSO 3, using the max–min method, provided the best results in Modes 2, 3, and 4. However, in Mode 1, RDS-PSO 3 using the cosine amplitude method was the best for the Ackley function, as detailed in Table 6. RDS-PSO with constant alpha value again showed the worst results in this function.

Table 5. Results of the Griewangk function.

Method	Implementation mode	Average best fitness	Standard deviation
RDS-PSO 1 (traditional PSO with linearly dec/inc w)	Mode 1 (2.13 s)	0.10022	0.36159
	Mode 2 (4.01 s)	0.01184	0.04185
	Mode 3 (3.77 s)	10.77	16.388
	Mode 4 (6.22 s)	11.249	17.383
RDS-PSO with constant alpha	Mode 1 (2.28 s)	0.5438 (0.9)	0.98088 (0.9)
	Mode 2 (4.29 s)	0.234 (0.9)	0.4359 (0.9)
	Mode 3 (3.12 s)	4.9193 (0.7)	7.5322 (0.7)
	Mode 4 (6.17 s)	5.4823 (0.7)	5.1925 (0.7)
RDS-PSO 2 (max–min)	Mode 1 (2.12 s)	0.02978	0.04899
	Mode 2 (3.75 s)	0.00844	0.01336
	Mode 3 (2.31 s)	0.32909	0.7465
	Mode 4 (4.46 s)	0.16767	0.43809
RDS-PSO 2 (cosine amplitude)	Mode 1 (2.29 s)	0.00229	0.01129
	Mode 2 (3.98 s)	0.00156	0.00857
	Mode 3 (2.97 s)	6.6455	10.392
	Mode 4 (5.00 s)	8.644	12.729
RDS-PSO 3 (max–min)	Mode 1 (2.03 s)	6.3408e-7	1.5415e-7
	Mode 2 (3.22 s)	5.5347e-7	1.6521e-7
	Mode 3 (2.08 s)	0.3128	1.3909
	Mode 4 (3.85 s)	0.25	1.2688
RDS-PSO 3 (cosine amplitude)	Mode 1 (2.36 s)	4.0133e-6	2.3107e-5
	Mode 2 (4.00 s)	5.4805e-7	1.79e-7
	Mode 3 (2.79 s)	8.2152	13.178
	Mode 4 (5.03 s)	9.3678	13.6926
RDS-PSO 4 (max–min)	Mode 1 (2.19 s)	0.01159	0.0293
	Mode 2 (3.51 s)	0.00164	0.0044
	Mode 3 (2.17 s)	0.0692	0.1269
	Mode 4 (4.01 s)	0.0551	0.1321
RDS-PSO 4 (cosine amplitude)	Mode 1 (2.58 s)	0.0048	0.0245
	Mode 2 (4.11 s)	3.651e-5	1.5698e-4
	Mode 3 (2.99 s)	6.6461	10.5556
	Mode 4 (5.32 s)	8.112	12.4279
Ladder PSO (LPSO)	—	0.172	0.347
Mutation PSO (MPSO)	—	0.844	0.911
Linearly decreasing inertia weight PSO	—	0.379	0.863

RDS-PSO 3 using the max–min method provided the best results in Modes 3 and 4. However, in Modes 1 and 2, RDS-PSO 3 using the cosine amplitude method was the best for the Sphere function, as detailed in Table 7. Again, RDS-PSO with constant alpha value showed the poorest results in this function.

Table 6. Results of the Ackley function.

Method	Implementation mode	Average best fitness	Standard deviation
RDS-PSO 1 (traditional PSO with linearly dec/inc w)	Mode 1 (5.43 s)	6.688e-7	2.4318e-7
	Mode 2 (9.71 s)	6.26e-7	2.018e-7
	Mode 3 (7.37 s)	0.1599	0.6688
	Mode 4 (12.94 s)	0.1308	0.6418
RDS-PSO with constant alpha	Mode 1 (5.53 s)	2.0897 (0.85)	2.1034 (0.85)
	Mode 2 (9.71 s)	2.1388 (0.85)	2.3734 (0.85)
	Mode 3 (7.95 s)	1.8051 (0.95)	1.4431 (0.95)
	Mode 4 (13.91 s)	1.5324 (0.95)	1.3832 (0.95)
RDS-PSO 2 (max–min)	Mode 1 (6.36 s)	0.1358	0.4985
	Mode 2 (10.94 s)	0.0156	0.0444
	Mode 3 (6.89 s)	0.126	0.4071
	Mode 4 (11.22 s)	0.0845	0.1758
RDS-PSO 2 (cosine amplitude)	Mode 1 (7.38 s)	0.0255	0.0968
	Mode 2 (14.71 s)	0.0684	0.3745
	Mode 3 (7.93 s)	8.5197e-5	5.4861e-4
	Mode 4 (15.19 s)	1.6195e-4	9.4053e-4
RDS-PSO 3 (max–min)	Mode 1 (6.47 s)	2.2347e-7	1.0848e-7
	Mode 2 (11.43 s)	2.0387e-7	8.4127e-8
	Mode 3 (7.00 s)	1.9336e-7	1.0254e-7
	Mode 4 (11.98 s)	1.7601e-7	9.6005e-8
RDS-PSO 3 (cosine amplitude)	Mode 1 (8.15 s)	1.7691e-7	8.9678e-8
	Mode 2 (15.93 s)	2.0747e-7	9.7837e-8
	Mode 3 (8.01 s)	2.0687e-7	1.4653e-7
	Mode 4 (16.11 s)	2.11e-7	1.4937e-7
RDS-PSO 4 (max–min)	Mode 1 (7.61 s)	0.1072	0.2994
	Mode 2 (12.71 s)	0.0835	0.3247
	Mode 3 (7.38 s)	0.0319	0.0689
	Mode 4 (12.69 s)	0.0365	0.0799
RDS-PSO 4 (cosine amplitude)	Mode 1 (8.59 s)	0.0308	0.0749
	Mode 2 (16.69 s)	0.1007	0.4263
	Mode 3 (8.77 s)	0.0011	0.007
	Mode 4 (17.34 s)	8.335e-6	4.9859e-5
Ladder PSO (LPSO)	—	4.337e-7	6.827e-7
Mutation PSO (MPSO)	—	3.519e-7	8.473e-7
Linearly decreasing inertia weight PSO	—	7.834e-7	1.043e-6

Generally, RDS-PSO 3 with both max–min and cosine amplitude, and occasionally RDS-PSO 4 with max–min, gave the best results in all implementations. RDS-PSO with constant alpha value showed the worst results in almost all the implementations. These results proved that the regularization methods used to set the values of alpha and beta constants are suitable and reliable for avoiding the diversity reduction problem. In

all implementations using linearly increasing inertia weight, the methods using the max–min procedure achieve better results than the methods using the cosine amplitude procedure. However, in the implementations using the linearly decreasing procedure, these 2 procedures achieved a close performance.

Table 7. Results of the Sphere function.

Method	Implementation mode	Average best fitness	Standard deviation
RDS-PSO 1 (traditional PSO with linearly dec/inc w)	Mode 1 (4.29 s)	0.0335	0.661
	Mode 2 (8.08 s)	0.0084	0.023
	Mode 3 (6.13 s)	18.933	22.174
	Mode 4 (11.54 s)	32.017	38.288
RDS-PSO with constant alpha	Mode 1 (4.77 s)	0.951 (0.85)	1.244 (0.85)
	Mode 2 (8.24 s)	0.046 (0.9)	0.483 (0.9)
	Mode 3 (6.86 s)	26.291 (0.9)	31.104 (0.9)
	Mode 4 (12.77 s)	34.619 (0.75)	41.577 (0.75)
RDS-PSO 2 (max–min)	Mode 1 (4.88 s)	0.0142	0.061
	Mode 2 (7.99 s)	0.00022	0.0094
	Mode 3 (5.35 s)	6.881	5.772
	Mode 4 (8.11 s)	8.229	9.005
RDS-PSO 2 (cosine amplitude)	Mode 1 (4.99 s)	0.0067	0.0029
	Mode 2 (8.28 s)	0.00005	0.00053
	Mode 3 (5.69 s)	127.38	143.401
	Mode 4 (10.03 s)	623.71	736.067
RDS-PSO 3 (max–min)	Mode 1 (4.83 s)	5.5891e-7	3.6429e-7
	Mode 2 (7.43 s)	7.5229e-7	5.8304e-7
	Mode 3 (5.27 s)	2.0684e-7	3.4627e-7
	Mode 4 (8.04 s)	1.8167e-7	9.9213e-8
RDS-PSO 3 (cosine amplitude)	Mode 1 (5.07 s)	3.7354e-7	4.4937e-7
	Mode 2 (8.17 s)	2.8723e-7	3.0572e-7
	Mode 3 (5.47 s)	4.5791e-7	5.8366e-7
	Mode 4 (10.22 s)	4.7361e-7	7.5732e-8
RDS-PSO 4 (max–min)	Mode 1 (5.02 s)	0.0055	0.066
	Mode 2 (7.56 s)	0.00016	0.0097
	Mode 3 (5.38 s)	3.3942	5.662
	Mode 4 (8.29 s)	4.2125	7.0154
RDS-PSO 4 (cosine amplitude)	Mode 1 (5.15 s)	0.00091	0.0024
	Mode 2 (8.31 s)	0.000061	0.00052
	Mode 3 (5.61 s)	68.191	72.058
	Mode 4 (10.39 s)	15.446	13.552
Ladder PSO (LPSO)	—	6.4881e-7	8.371e-7
Mutation PSO (MPSO)	—	8.9221e-5	5.613e-4
Linearly decreasing inertia weight PSO	—	2.6441e-6	4.344e-5

In addition to these experimental studies, the history of the gbest values for RDS-PSO 2, 3, and 4 was investigated in Figure 4. The constants of step size and C were set to 0.05. RDS-PSO 2, 3, and 4 were implemented in Mode 2 (with 2000 maximum iterations and linearly decreasing inertia weight) with the max–min technique for the Rastrigin test function. As indicated in Table 4, the gbest history of RDS-PSO 3 had better convergence ability than the others for 50 trials. The results in Table 4 are more favorable than the results in Figure 4 due to the selection of the constants of step size and C.

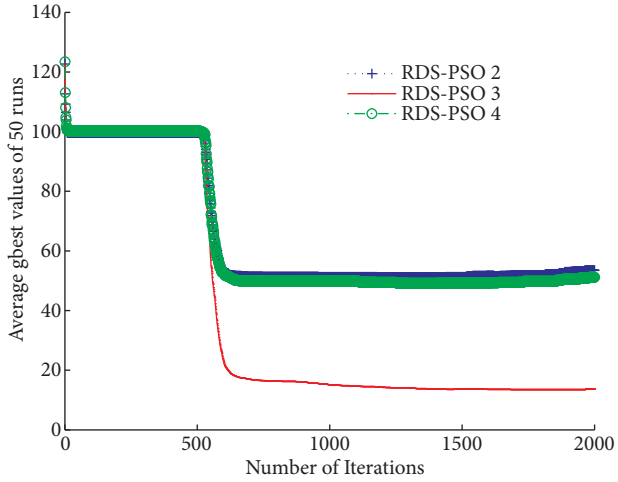

Figure 4. The history of gbest in RDS-PSO 2, 3, and 4 for Mode 2 and the Rastrigin function.

5. Discussion

In this paper, a generalization idea for the PSO method was introduced by using 2 new constants, and the generalized PSO was transformed into an adaptive method through 2 alternative regularization procedures. The alpha constant sets a trade-off between the impact of the global best and global worst particles in the velocity update process. Similarly, the beta constant sets the same regularization for the local best and worst particles. In other words, when the values of the alpha and beta constants are equal to one, the traditional PSO is constituted. Although RDS-PSO with constant alpha value showed worse results than the traditional PSO, by using different values for these constants, generalization and diversity enhancement can be improved. Thus, a procedure regulating the values of these constants is needed. According to the implementation results, the max–min-based procedure has better diversity-representing properties than the cosine amplitude-based procedure.

Generally, RDS-PSO 3, which regulates beta values while alpha is equal to one as an invariable parameter, showed the best results among all methods. Nevertheless, RDS-PSO 4, which regulates both alpha and beta constants, sometimes showed the best results. As the personal best and worst particles provide more diversity than the global particles, regulating the beta constant properly is more important than alpha regulations. In addition, the average performance of the max–min-based procedure is better than the cosine amplitude-based procedure.

6. Conclusion

In this study, 4 types of RDS-PSO were designed based on a different organization of the alpha and beta constants. The results of these types were also compared with the original PSO, ladder particle swarm

optimization (LPSO), mutation particle swarm optimization (MPSO), linearly decreasing inertia weight particle swarm optimization (LDWPSO), and linearly increasing inertia weight particle swarm optimization (LIWPSO).

According to the experimental results, RDS-PSO 3 showed the highest performance among almost all benchmark functions. However, LPSO showed almost the same results as RDS-PSO 3. MPSO and LDWPSO also showed better results than the other RDS-PSO modes.

References

[1] Kennedy J, Eberhart RC. Particle swarm optimization. In: IEEE 1995 Neural Networks Conference; 27 November–1 December 1995; Perth, Australia. New York, NY, USA: IEEE. pp. 1942–1948.

[2] Ajami A, Armaghan M. A comparative study in power oscillation damping by STATCOM and SSSC based on the multiobjective PSO algorithm. Turk J Electr Eng Co 2013; 21: 213–224.

[3] Pandian SMV, Thanushkodi K. Considering transmission loss for an economic dispatch problem without valve-point loading using an EP-EPSO algorithm. Turk J Electr Eng Co 2012; 20: 1259–1267.

[4] Güneş F, Özkaya U. Multiobjective FET modeling using particle swarm optimization based on scattering parameters with Pareto optimal analysis. Turk J Electr Eng Co 2012; 20: 353–365.

[5] Şevkli AZ, Sevilgen FE. Discrete particle swarm optimization for the team orienteering problem. Turk J Electr Eng Co 2012; 20: 231–239.

[6] Angeline P. Evolutionary optimization versus particle swarm optimization: philosophy and performance difference. In: Proceedings of the 1998 Evolutionary Programming Conference; 25–27 March 1998; San Diego, CA, USA. London, UK: Springer-Verlag. pp. 601–610.

[7] Du WL, Li B. Multi-strategy ensemble particle swarm optimization for dynamic optimization. Inform Sciences 2008; 178: 3096–3109.

[8] Carlisle A, Dozier G. Tracking changing extrema with particle swarm optimizer. In: IEEE 2002 World Automation Congress; 9–13 June 2002; Orlando, FL, USA. New York, NY, USA: IEEE. pp. 265–270.

[9] Mousa AA, El-Shorbagy MA, Abd-El-Wahed WF. Local search based hybrid particle swarm optimization algorithm for multiobjective optimization. Swarm Evol Comput 2012; 3: 1–14.

[10] Suresh K, Kumarappan N. Hybrid improved binary particle swarm optimization approach for generation maintenance scheduling problem. Swarm Evol Comput 2013; 9: 69–89.

[11] Bansal JC, Deep K. A modified binary particle swarm optimization for Knapsack problems. Appl Math Comput 2012; 218: 11042–11061.

[12] Wang Y, Li B, Weise T, Wang J, Yuan B, Tian Q. Self-adaptive learning based particle swarm optimization. Inform Sciences 2011; 181: 4515–4538.

[13] Epitropakis MG, Plagianakos VP, Vrahatis MN. Evolving cognitive and social experience in particle swarm optimization through differential evolution: a hybrid approach. Inform Sciences 2012; 216: 50–92.

[14] Wang H, Sun H, Li C, Rahnamayan S, Pan J. Diversity enhanced particle swarm optimization with neighborhood search. Inform Sciences 2013; 223: 119–135.

[15] Zhao Y, Zu W, Zeng H. A modified particle swarm optimization via particle visual modeling analysis. Comput Math Appl 2009; 57: 2022–2029.

[16] Akbari R, Ziarati K. A rank based particle swarm optimization algorithm with dynamic adaptation. J Comput Appl Math 2011; 235: 2694–2714.

[17] Shi Y, Eberhart RC. Fuzzy adaptive particle swarm optimization. In: IEEE 2001 Evolutionary Computation Congress; 27–30 May 2001; Seoul, Korea. New York, NY, USA: IEEE. pp. 101–106.

[18] Jiao B, Lian Z, Gu X. A dynamic inertia weight particle swarm optimization algorithm. Chaos Soliton Fract 2008; 37: 698–705.

[19] Zheng YL, Ma LH, Zhang LY, Qian JX. On the convergence analysis and parameter selection in particle swarm optimization. In: IEEE 2003 Machine Learning and Cybernetics International Conference; 2–5 November 2003; Piscataway, NJ, USA. New York, NY, USA: IEEE. pp. 1802–1807.

[20] Zheng YL, Ma LH, Zhang LY, Qian JX. 2003b. Empirical study of particle swarm optimizer with an increasing inertia weight. In: IEEE 2003 Evolutionary Computation Congress; 8–12 December 2003; Canberra, Australia. New York, NY, USA: IEEE. pp. 221–226.

[21] Selvakumar AI, Thanushkodi K. Anti-predatory particle swarm optimization: solution to nonconvex economic dispatch problems. Electr Pow Syst Res 2008; 78: 2–10.

[22] Çomak E. A flexible particle swarm optimization based on global best and global worst information. In: INSTICC 2013 Pattern Recognition Application and Methods International Conference; 15–18 February 2013; Barcelona, Spain. Lisbon, Portugal: INSTICC. pp. 255–262.

[23] Griewangk AO. Generalized descent of global optimization. J Optimiz Theory App 1981; 34: 11–39.

[24] Rastrigin LA. External Control Systems. Theoretical Foundations of Engineering Cybernetics Series. Moscow, Russia: Nauka, 1974.

[25] De Jong K. An analysis of the behavior of a class of genetic adaptive systems. PhD, University of Michigan, Ann Arbor, MI, USA, 1975.