Analysis of Stability, Local Convergence, and Transformation Sensitivity of a Variant of the Particle Swarm Optimization Algorithm

Mohammad Reza Bonyadi and Zbigniew Michalewicz

***Abstract*—In this paper, we investigate three important properties (stability, local convergence, and transformation invariance) of a variant of particle swarm optimization (PSO) called standard PSO 2011 (SPSO2011). Through some experiments, we identify boundaries of coefficients for this algorithm that ensure particles converge to their equilibrium. Our experiments show that these convergence boundaries for this algorithm are: 1) dependent on the number of dimensions of the problem; 2) different from that of some other PSO variants; and 3) not affected by the stagnation assumption. We also determine boundaries for coefficients associated with different behaviors, e.g., nonoscillatory and zigzagging, of particles before convergence through analysis of particle positions in the frequency domain. In addition, we investigate the local convergence property of this algorithm and we prove that it is not locally convergent. We provide a sufficient condition and related proofs for local convergence for a formulation that represents updating rules of a large class of PSO variants. We modify the SPSO2011 in such a way that it satisfies that sufficient condition; hence, the modified algorithm is locally convergent. Also, we prove that the original standard PSO algorithm is not sensitive to rotation, scaling, and translation of the search space.**

***Index Terms*—Local convergence, particle swarm optimization (PSO), stability analysis, transformation invariance.**

I. Introduction

PARTICLE swarm optimization (PSO) is a stochastic population-based optimization algorithm developed by Kennedy and Eberhart [1]. PSO has been applied to many optimization problems such as artificial neural network training and pattern classification [2], [3], to name a few. Since 1995, different aspects of the original version of PSO have been investigated and many variants of the algorithm have been proposed. Due to the emergence of many PSO variants, standard versions for the algorithm [4]–[6] were introduced that were updated according to new advances every few years. Standard PSO 2011 (SPSO2011) is the most recent standard version for PSO [6]. A variant of SPSO2011 was applied to some continuous space benchmark problems and its results were reported to set a baseline for further research [7]. Most existing studies on SPSO2011 are related to its application [7] to continuous space optimization benchmarks and there is only one article [8] (presented by the authors of this paper) that investigates theoretical aspects of SPSO2011 that include stability of particles [9], [10], local convergence [11], [12], and transformation sensitivity [13].

In our earlier paper [8], the stability of particles in a SPSO2011 variant proposed by [7] was investigated under the stagnation assumptions through an experimental approach called estimation of convergence boundaries (ECB). Also, it was proven that the algorithm is not locally convergent and a modification was proposed to resolve the issue. However, no proof was provided to illustrate that the modified algorithm is locally convergent. In addition, the transformation invariance of the algorithm was investigated in [8].

In this paper, stability of particles, local convergence, and transformation sensitivity of SPSO2011 (proposed in [6]) are analyzed in more detail. We modify the ECB algorithm [the new algorithm is called estimation of variance convergence boundaries (EVCB)] and apply it to a PSO variant for which these boundaries are known. Results indicate that the estimated convergence boundaries are in good agreement with those that were found through theoretical analyses. EVCB is applied to estimate the convergence boundaries for SPSO2011 under stagnation as well as general conditions. The behavior of the positions of particles before convergence is also analyzed and the boundaries associated with different behaviors such as harmonic and zigzagging are identified for the algorithm. Furthermore, the local convergence property of SPSO2011 is investigated and it is proven that this algorithm is not locally convergent. We introduce a sufficient condition together with related proofs for local convergence of a recursive equation that formulates a large class of PSO algorithms. We propose a simple modification to SPSO2011 and prove that the modified version satisfies the sufficient condition to address the local convergence issue. Finally, it is proven that the algorithm is invariant from rotation, scaling, and translation (RST). Note that the aim of this paper is to understand

Manuscript received December 10, 2014; revised March 6, 2015, May 13, 2015, July 5, 2015, and July 21, 2015; accepted July 22, 2015. Date of publication July 24, 2015; date of current version May 26, 2016. This work was supported by ARC Discovery Grant DP130104395 and Grant N N519 5788038 from the Polish Ministry of Science and Higher Education (MNiSW).

M. R. Bonyadi is with the Optimisation and Logistics Group, University of Adelaide, Adelaide, SA 5005, Australia, and also with the Data Science Group, Complexica Pty Ltd, Adelaide, SA 5021, Australia (e-mail: mrbonyadi@cs.adelaide.edu.au).

Z. Michalewicz is with the Optimisation and Logistics Group, University of Adelaide, Adelaide, SA 5005, Australia; also with the Data Science Group, Complexica Pty Ltd, Adelaide, SA 5021, Australia; also with the Institute of Computer Science, Polish Academy of Sciences, ul. Ordona 21, 01-237 Warsaw, Poland; and also with the Polish–Japanese Institute of Information Technology, ul. Koszykowa 86, 02-008 Warsaw, Poland (e-mail: rezabny@gmail.com).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TEVC.2015.2460753

different behaviors of particles and to address potential theoretical issues (e.g., local convergence) in SPSO2011 and it does not try to improve the algorithm to find better solutions.

This paper is an extension of our earlier paper [8]. There are significant differences between these two papers, as this paper investigates:

1) A different variant of SPSO2011 than the one investigated in [8].
2) The algorithm with and without stagnation while our earlier paper only investigated the algorithm with stagnation.
3) The behavior of particles before they converge that was not done in our earlier paper.
4) A recursion equation that represents a large class of PSO algorithms and provides sufficient conditions for local convergence of that recursion.
5) The local convergence of SPSO2011 in more detail than it was done in [8].
6) The transformation invariance of SPSO2011 through more formal definitions and analyses from what was done in [8].

Without loss of generality, this paper considers only minimization problems defined as

$$\text{find } \vec{x} \in S \subseteq R^d \text{ such that } \forall \vec{y} \in S, F(\vec{x}) \le F(\vec{y}) \tag{1}$$

where S is the search space defined by $\{x | l_i \le x_i \le u_i \text{ for all } i\}$, l_i and u_i are the lower bound and upper bound of the values of the ith dimension of S, d is the number of dimensions, and $F(.)$ is the objective function. The set of points that are generated by $F(x)$ for all $x \in S$ is called the landscape.

The rest of this paper is organized as follows. We briefly discuss earlier variants of PSO in Section II. Section III reviews existing approaches for stability analysis of PSO variants and provides analysis of stability for SPSO2011. Local convergence properties for some PSO variants are outlined in Section IV and this property for SPSO2011 is investigated. In Section V, an overview of existing transformation sensitivity analyses for different PSO variants is given. Also, we prove that SPSO2011 is invariant from RST of the search space. Section VI concludes this paper.

II. Particle Swarm Optimization

Each particle in the original PSO (OPSO) [1], [14] contains three vectors.

1) *Position ($\vec{x}_t^i$):* This is the position of the ith particle in the tth iteration. This is used to evaluate the particle quality.
2) *Velocity ($\vec{V}_t^i$):* This is the direction and length of movement of the ith particle in the tth iteration.
3) *Personal Best ($\vec{p}_t^i$):* This is the best position (in terms of objective value) that the particle i has visited until iteration t. The role of this vector is to store the knowledge of best found solutions [14].

In OPSO, the velocity of each particle is updated for the next iteration $(t+1)$ by

$$\vec{V}_{t+1}^i = \vec{V}_t^i + \phi_1 R_{1t}\left(\vec{p}_t^i - \vec{x}_t^i\right) + \phi_2 R_{2t}\left(\vec{g}_t - \vec{x}_t^i\right) \tag{2}$$

where ϕ_1 and ϕ_2 are the two real numbers called acceleration coefficients, $\vec{p}_t^i$ is the personal best of the particle i at iteration t, and $\vec{g}_t$ is the best personal best in the swarm.

The vector $\vec{p}_t^i$ for each particle i is updated by

$$\vec{p}_{t+1}^i = \begin{cases} \vec{x}_{t+1}^i & F\left(\vec{x}_{t+1}^i\right) < F\left(\vec{p}_t^i\right) - \epsilon_0 \text{ and } \vec{x}_{t+1}^i \in S \\ \vec{p}_t^i & \text{otherwise} \end{cases} \tag{3}$$

where ϵ_0 is an arbitrarily small real value that represents the precision of the calculations. This constant can be set to the smallest possible value in the simulations (see Section IV).

Particles are attracted by $\text{PI} = \vec{p}_t^i - \vec{x}_t^i$ (personal influence) and $\text{SI} = \vec{g}_t - \vec{x}_t^i$ (social influence) to move toward known quality solutions found until iteration t, i.e., $\vec{p}_t^i$ and $\vec{g}_t$. Furthermore, R_{1t} and R_{2t} are two randomly (a uniform distribution in the interval [0, 1]) generated $d \times d$ diagonal matrices [15], [16]. These two matrices are generated for each particle i at every iteration t separately. The position of the particles is updated by

$$\vec{x}_{t+1}^i = \vec{x}_t^i + \vec{V}_{t+1}^i. \tag{4}$$

OPSO was studied by many researchers since 1995 and many new variants were proposed. As an example, it was proposed [17] to multiply the previous velocity ($\vec{V}_t^i$) by an inertia weight (ω) to control the impact of $\vec{V}_t^i$ on the movement of particles (we call this PSO variant as inertia PSO, IPSO, in this paper). The velocity update rule for IPSO was written as

$$\vec{V}_{t+1}^i = \omega\vec{V}_t^i + \phi_1 R_{1t}\left(\vec{p}_t^i - \vec{x}_t^i\right) + \phi_2 R_{2t}\left(\vec{g}_t - \vec{x}_t^i\right) \tag{5}$$

where ω is the inertia weight.

Because of the fast growth of number of PSO variants (see [18] for a review paper on some variants), the lack of a standard version to compare the results with became more apparent. Therefore, some researchers set a standard version for PSO and updated that frequently (every couple of years).[1] New findings in the PSO area, including new values for coefficients, topology,[2] velocity update rule, and adaptations, were incorporated into the standard PSOs to keep the standard version up-to-date (see for example [5]).

The most recent standard PSO algorithm is called SPSO2011 [6]. The algorithm was applied to some standard continuous optimization benchmarks [7]. The velocity updating rule for SPSO2011 was written as

$$\vec{V}_{t+1}^i = \omega\vec{V}_t^i + H\left(\vec{G}_t^i, \left\|\vec{G}_t^i - \vec{x}_t^i\right\|\right) - \vec{x}_t^i \tag{6}$$

where $H(\vec{G}_t^i, \|\vec{G}_t^i - \vec{x}_t^i\|)$ is a spherical distribution with the center $\vec{G}_t^i$ and radius $\|\vec{G}_t^i - \vec{x}_t^i\|$ and $\|.\|$ is the Euclidean norm. Also

$$\vec{G}_t^i = \frac{\vec{L}_t^i + \vec{P}_t^i + \vec{x}_t^i}{3} \tag{7}$$

where $\vec{P}_t^i = \vec{x}_t^i + c_1(\vec{p}_t^i - \vec{x}_t^i)$, $\vec{L}_t^i = \vec{x}_t^i + c_2(\vec{l}_t^i - \vec{x}_t^i)$, and $\vec{l}_t^i$ is the best personal best among all particles connected to particle i

[1] Some source codes and documentation for these standard PSOs can be found in http://www.particleswarm.info/Programs.html.

[2] If a particle i uses the personal best position of a particle j to update its velocity then we say these two particles are connected. The way that all particles are connected to each other in the swarm is called topology.

Algorithm 1 Generate $H(\vec{a}, b)$

```
1: Input $\vec{a}, b$
2: Output $\vec{y}$
3: $\vec{r} =< N_1(0,1), N_2(0,1), \ldots, N_d(0,1) >$ where $N_i(0,1)$ (for all $i$) generates a random value according to the normal distribution with mean equal to 0 and standard deviation equal to 1
4: $\vec{r'} = \frac{\vec{r}}{\|\vec{r}\|}$, where $||.||$ is the Euclidean norm operator
5: $\vec{y} = \vec{a} + \vec{r'}U(0,b)$ where $U(0,b)$ generates a uniform random scalar in [0, b]
```

(known as local best). Also, in [6], it was suggested that if $\vec{l}_t^i = \vec{p}_t^i$ then calculate

$$\vec{G}_t^i = \frac{\vec{P}_t^i + \vec{x}_t^i}{2}. \tag{8}$$

Algorithm 1 was used[3] to generate the spherical distribution $H(\vec{a}, b)$. Step 4 of the algorithm, inspired by [19], generates a point that is uniformly distributed on the surface of the unit hypersphere. The radius of the hypersphere is then altered in Step 5 randomly so that the point has the chance to be generated on any hypersphere inside the hypersphere with the radius b (this algorithm was also used in [6]). Step 5 also shifts the generated point by the vector $\vec{a}$. One should note that the distribution of the points generated by this method is more dense toward the center of the hypersphere, hence, not uniform in the hypersphere. However, we have used this setting to be consistent with the original SPSO2011 proposed in [6].[4]

In this paper, we assume that all particles are connected (a global best topology) and, consequently, replace l_t^i by g_t. The values for c_1 and c_2 were considered as constants equal to ϕ_1 and ϕ_2, respectively. The term SPSO2011 refers to the variant which was described in [6] throughout this paper unless specified. It was claimed that SPSO2011 is rotation invariant; however, no proper proof for this claim was provided.

In [7], SPSO2011 was modified in a way the values of c_1 and c_2 were set to $\phi_1 R_{1t}$ and $\phi_2 R_{2t}$, respectively, where R_{1t} and R_{2t} are random diagonal matrices and ϕ_1 and ϕ_2 are constant values.

III. Stability Analysis

Perhaps one of the first analyses of convergence for stochastic optimization algorithms was conducted by [20], which was later followed by [21] and [22]. An iterative stochastic optimization algorithm (optimization algorithm in short) is said to converge in probability (to converge in short)[5] if

$$\forall \epsilon > 0, \lim_{t\to\infty} P\left(\left\|\vec{x}_t - \vec{X}\right\| < \epsilon\right) = 1 \tag{9}$$

where P is the probability measure, $\vec{x}_t$ is a generated solution by the optimization algorithm (a point in the search space) at iteration t, ϵ is a small positive value, and $\vec{X}$ is a point in the search space. This type of convergence is usually investigated for an iterative stochastic optimization algorithm to find the boundaries for coefficients in the algorithm so that the sequence of the generated solutions by the algorithm is convergent. This analysis is also known as stability analysis.

In this section, we analyze the stability and behavior of particles in IPSO and SPSO2011.

A. Stability Analysis for IPSO

Stability analysis for IPSO was performed by many researchers [9], [23]–[25], to name a few. One of the aims of the stability analysis for IPSO was to find boundaries for the coefficients of velocity update rule (i.e., inertia weight and acceleration) so that positions of particles converge to a point in the search space. The set of all boundaries for all coefficients of a PSO variant that guarantee convergence to a point is called convergence boundaries in this paper.

In order to simplify the analysis of stability for IPSO, some researchers [9], [26] assumed that the stochastic values in the velocity update rule are set to constant values (1 for IPSO). This simplification enabled researchers to study particles behaviors through dynamic systems methodologies (analyses with consideration of this simplification is called deterministic model stability analysis). A more realistic view point, however, is to study the algorithm with the presence of stochastic components. Some researchers [10], [23] studied the behavior of particles in expectation, i.e., if $\lim_{t\to\infty} E(x_t)$ is a constant value, where $E(.)$ is the expectation operator (this analysis is called first-order stability analysis). These studies found that if the personal best and global best vectors are not updated during the run then the expectation of the position of each particle converges to a point (first-order stability analysis) between the personal and global best vector $((\varphi_2 g + \varphi_1 p)/(\varphi_1 + \varphi_2))$ if and only if $-1 < \omega < 1$ and $0 < \varphi < 4(1+\omega)$ where $\varphi = \varphi_1 + \varphi_2$. Recently, it has been proven [27] that this relation results in first-order stability under more general conditions, i.e., when personal best and global best can be updated while they can occupy an arbitrarily large finite number of unique positions in the search space.

Although first-order stability analysis is more realistic than deterministic model stability analysis, it is still not the most comprehensive analysis possible to ensure particle convergence. The reason is that, even if the expected position of a particle converges to a single value, it does not mean that the particle is steady at that point as it may oscillate on a line such that its expected value remains constant. Hence, some researchers [24], [25], [28] studied the variance (rather than expectation) of the position of the particles. The aim was to study the behavior of variance of particle positions

[3] In our earlier paper [8], $N(0, b)$ was used rather than $U(0, b)$ in Step 5 of Algorithm 1, that might cause some dissimilarities between the results of that paper and this paper.

[4] One can replace $U(0, b)$ with $U(0, b)^{1/d}$ to ensure that the generated points are uniformly distributed in the hypersphere. The probability that a generated point inside a hypersphere with the radius b and it is also inside a hypersphere with the radius r (i.e., $P(D < r)$, where D is the distance of the point from the center of coordinates system) is calculated by $(\alpha r^d)/(\alpha b^d)$, where αb^d is the volume of a hypersphere with the radius b. Hence, $P(D < r)$ is uniform if $r = bU(0, b)^{1/d}$.

[5] Note that there are many other types of convergence such as almost surely convergence, sure convergence, and nth mean convergence.

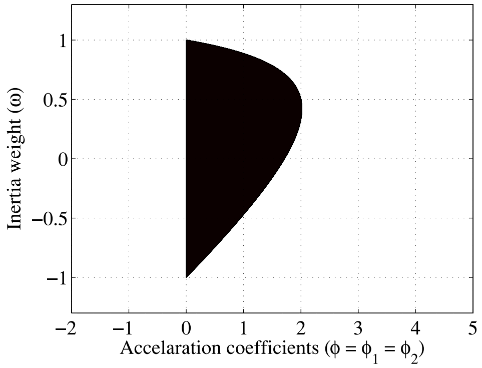


Fig. 1. Convergence boundaries for IPSO (dark area) that were reported in [24] through theoretical analysis.

when iteration number grows to infinity (this analysis is called second-order stability analysis). These studies found that the expectation of the positions of particles converges to a point and the variance of the positions converges to a constant value if $\varphi < (12(\omega^2 - 1))/(5\omega - 7)$ where $\varphi = \varphi_1 = \varphi_2$ and $-1 < \omega < 1$ (see Fig. 1). It was shown [25] that the variance of positions converges to $h(\phi_1, \phi_2, \omega)|g - p|$ where $h(., ., .)$ is a function of inertia weight and acceleration coefficients (see [25] for details on this function). Hence, if $h(\phi_1, \phi_2, \omega) \neq 0$ is guaranteed then particles do not stop moving (nonzero variance) until $p = g$. It has been recently proven [28] that IPSO is second-order stable when $\varphi < (12(\omega^2 - 1))/(5\omega - 7)$ even if personal best is updated during the run.

The convergence boundaries for IPSO were investigated experimentally in [29]. IPSO was applied to a function that its values were generated randomly for each point independently ($F(x) = U(-1000, 1000)$ for all x where $U(a, b)$ is a uniform random number in $[a, b]$). For each $x \in S$, $F(x)$ was generated only once and it was reused afterward if required. Results showed that the convergence boundaries found through experiments are almost exactly the same as the boundaries found through second-order stability analysis. Note that [29] did not simplify the update rules of particles and also allowed personal best and global best positions to be updated during the run.

In these analyses, usually velocity and position update rules are analyzed for an arbitrary particle i in a 1-D space. This assumption is valid for IPSO because, in this algorithm, all calculations (including generation of the random values on the diagonal of R_{1t} and R_{2t}) are performed in each dimension independently. Thus, analyses in a 1-D case is generalizable to the multidimensional case [25].

B. Stability Analysis for SPSO2011

Because each dimension in IPSO is updated separately, the outcome of stability analysis in a 1-D space is generalizable to any number of dimensions. This assumption, however, is not valid for stability analysis of SPSO2011 because the velocity vector in SPSO2011 is not updated for each dimension separately. In SPSO2011, the operator $\vec{c} = H(\vec{a}, b)$ (for an arbitrary vector $\vec{a}$ and scalar value b) is responsible for generating a random point in the space with center $\vec{a}$ and radius b. This random generation process is not readily decomposable to a dimension-wise process. In addition, the value of b is dependent on the (Euclidean) distance of the position and personal best vector of the particle as well as the global best vector. Calculation of this distance is also not easily decomposable to dimension-wise calculations. Therefore, analysis of stability of particles for SPSO2011 should be conducted in the general case of d-dimensional space, i.e., $\|\vec{V}_t^i\| = 0$, rather than considering each dimension separately.

Algorithm 2 EVCB (Estimate Variance Convergence Boundaries)

```
**Input** $\phi_a, \phi_s, \phi_e, \omega_a, \omega_s, \omega_e, d, r, maxIter$
**Output** $Y^d_{\phi,\omega}$
**for** $\phi = \phi_a$ to $\phi_e$ with step size $\phi_s$ **do**
    $\phi_1 = \phi_2 = \phi$
    **for** $\omega = \omega_a$ to $\omega_e$ with step size $\omega_s$ **do**
        Run the PSO algorithm for a predefined number of iterations (maxIter) and number of runs ($r$) for the given $\phi_1 = \phi_2 = \phi$ and $\omega$
        For each run, select a random particle and, for each dimension, calculate the variance of the particle position (a $d \times r$ matrix $M$, where $M_{i,j}$ is the variance of the dimension $i$ for the run $j$),
        For each dimension, calculate the average of variances over all runs (a $d$ dimensional vector $\vec{Z}$, where $Z_i = \sum_{j=1}^{r} M_{i,j}$ for each $i$),
        Calculate $\text{var}_\omega = \|\vec{Z}\|$
    **end for**
    $Y^d_{\phi,\omega} = \text{var}_\omega$ for all $\omega$ in $\omega_a : \omega_s : \omega_e$
**end for**
```

In this paper, we conduct an experimental study (called EVCB), described as Algorithm 2, to estimate the convergence boundaries for PSO variants. As shown in the EVCB algorithm, we study the variance of 1-D of the position (second-order stability) of a randomly selected particle during a long run (20 000 iteration) for different combinations of ω and ϕ.

As EVCB investigates the parameters of PSO variants through experiments, the value of maxIter needs to be a large number in the tests. The reason is that there might be circumstances that particles velocities start shrinking while they start to grow afterward.

We consider the following settings for all experiments.

1) *Setting 1:* $\vec{p}^{\,i}_{t'} = \vec{x}^{\,i}_{t'}$ (the point $\vec{p}^{\,i}_{t'}$ is sampled by $\vec{x}^{\,i}_{t'}$).
2) *Setting 2:* The variance of a randomly selected particle is recorded.
3) *Setting 3:* The search space S is bounded for all dimensions in $[-10, 10]$.
4) *Setting 4:* $\phi = \phi_1 = \phi_2$.
5) *Setting 5:* $t' = 0$.

The boundary (setting 3) has been introduced to make the experiment setups consistent with the general definition mentioned in (1). As the SPSO2011 algorithm is invariant under any scaling and translation (see Section V for the proof),

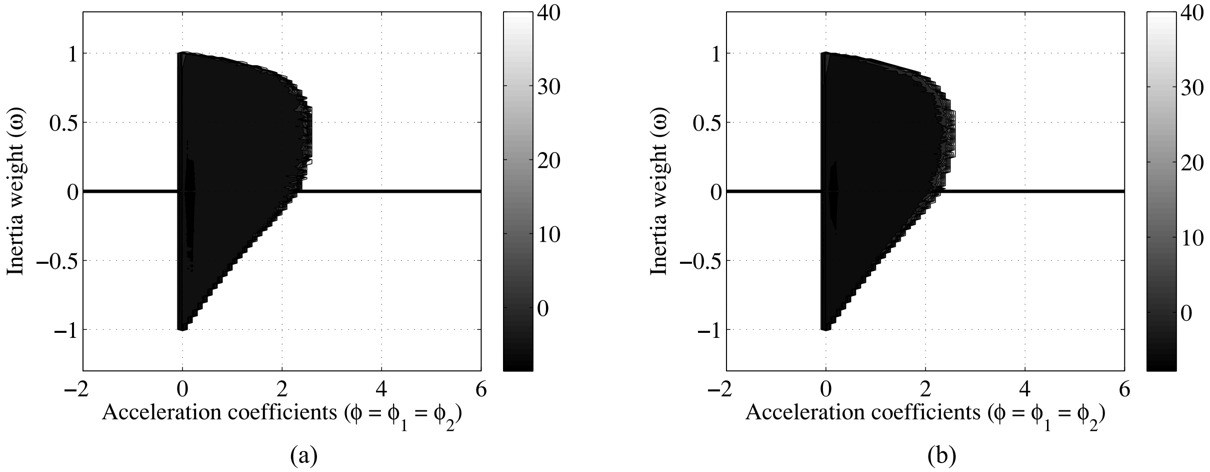


Fig. 2. Boundaries for ω and ϕ so that IPSO are convergent. The darker the area is, the smaller the value of $Y^d_{\phi,\omega}$. Thus, the darker the area is, the more probable that the algorithm is convergent with those coefficients. Experiments were conducted for $d = 1$ (a) and $d = 10$ (b).

resizing or shifting the boundaries do not affect the convergence behavior of particles. Also, if a particle left the search space then its personal best is not updated. Usually, some bound handling techniques are used in PSO variants to prevent particles from leaving the boundaries of the search space [30]. In our experiments, however, the velocities and positions of particles are not prevented from growing/leaving the boundaries, NOR any strategy is applied to bring the particles back inside the boundaries. This indeed is necessary as otherwise it is not possible to claim whether the result of convergence was purely because of the coefficient values or it was also because of the strategy used to prevent particles from leaving the boundaries.

To test the validity of this experimental approach (EVCB algorithm) for identifying convergence boundaries, we applied this approach to find the convergence boundaries for IPSO with $\phi_a = -2$, $\phi_s = 0.1$, $\phi_e = 6$, $\omega_a = -1.3$, $\omega_s = 0.01$, $\omega_e = 1.3$, $r = 10$, maxIter $= 20\,000$ for $d = 1$ and $d = 10$ [see Fig. 2 (a) and (b)]. These parameters were set experimentally in a way that the convergence boundaries are shown completely. Also, we considered the stagnation assumption, i.e., $\vec{p}^i_t = \vec{g}_t$ and g_t is not updated for all t.

The grayscale level in Fig. 2 (a) and (b) represents the value of $Y^d_{\phi,\omega}$ for IPSO for different values of inertia weight and acceleration coefficients. In the boundaries where $Y^d_{\phi,\omega}$ is relatively small (darker areas), the particles have become stable. Also, as it is clear in Fig. 2 (a) and (b), when $\omega = 0$ the stability of the particles is very probable for all values of ϕ and d. This was in fact expected as if $\omega = 0$, and because $\vec{x}^i_0 = \vec{p}^i_0$ and $\vec{p}^i_t = \vec{g}_t$ for all t, the value of $\vec{V}^i_t$ becomes zero for all t, which imposes stability. Of course this case is not interesting from application point of view. Fig. 2(a) and (b) shows that the convergence boundaries for IPSO is not affected by the number of dimensions (the value of $Y^d_{\phi,\omega}$ for $d = 1$ and 10 is very close for the tested ω and ϕ). This indeed confirms the assumption by earlier analyses, e.g., [10], for IPSO that the analysis in 1-D space is generalizable to the analysis in d-dimensional space. Also, these results are very similar to those reported in [24] and [25] for the stability of particles [see also Fig. 1 and compare it with Fig. 2 (a)]. This confirms that EVCB can estimate the convergence boundaries of IPSO.

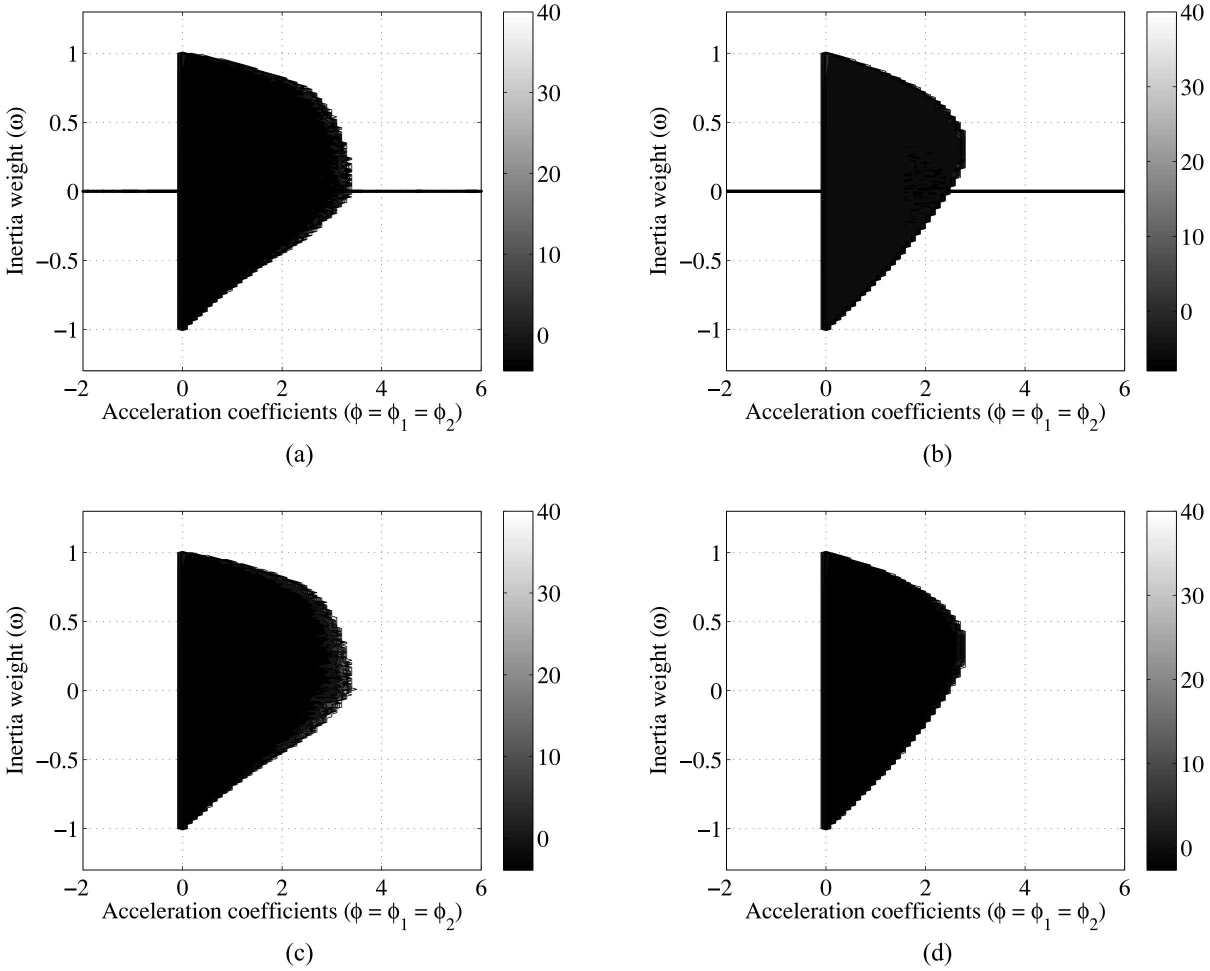
We conducted two experiments to estimate the convergence boundaries for SPSO2011: one without the stagnation assumption and the other with the stagnation assumption ($\vec{p}^i_t = \vec{g}_t$ and g_t is not updated for all t). The aim of these two tests was to understand if the movement of the personal best and global best affects the convergence region. For the experiment without the stagnation assumption, we assumed that there are 10 particles in the swarm and the objective function in [27] and [31] was used to investigate convergence of particles. This objective function prevents particles from stagnating. In the case of the stagnation assumption, because the global best and personal best vectors are not updated, there is no need to consider any objective function. Also, there is no need to consider more than one particle in the swarm as no information is propagated through the swarm (global best vector is not updated).

The stagnation assumption triggers the velocity update rule in SPSO2011 that is specific for stagnation [i.e., (8)] only. Hence, if we consider the stagnation assumption and run the algorithm, the final convergence region is specific to that formulation while the normal formulation [i.e., (7)] is ignored. This, however, was not the case for IPSO as the update rule was the same for stagnation and nonstagnation. Hence, in the EVCB algorithm for the stagnation assumption for SPSO2011, we set the value of $Y^d_{\phi,\omega}$ to the maximum value generated by (7) and (8) formulations.

The following parameters for EVCB were used: $\phi_a = -2$, $\phi_s = 0.1$, $\phi_e = 6$, $\omega_a = -1.3$, $\omega_s = 0.01$, $\omega_e = 1.3$, $r = 10$, and maxIter $= 20\,000$ for $d = 1$ and $d = 10$ (see Fig. 3). These parameters were set experimentally in a way that the convergence boundaries appear completely in the graph.

Fig. 3 indicates three points about the convergence boundaries for SPSO2011.

1) *Point 1:* The boundaries under the stagnation are very similar to those of without the stagnation assumption [compare Fig. 3 (a) with Fig. 3(c) and compare Fig. 3(b) with Fig. 3(d)].



(a)

(b)

(c)

(d)

Fig. 3. Boundaries for ω and ϕ so that SPSO2011 is convergent under (a) and (b) stagnation and (c) and (d) nonstagnation assumptions. The darker the area is, the smaller the value of $Y^d_{\phi,\omega}$. Thus, the darker the area is, the more probable that the algorithm is convergent with those coefficients. The color spectrum shows the value of the matrix $Y^d_{\phi,\omega}$. Experiments were conducted for (a) and (c) $d = 1$ and (b) and (d) $d = 10$. Notice the similarity between the two graphs.

2) *Point 2:* The boundaries look different from those of IPSO (see Fig. 2 and compare it with Fig. 3).[6]

3) *Point 3:* The boundaries change when the number of dimensions varies.

Based on point 1, it seems that the stagnation assumption is not an unrealistic assumption to study the convergence of particles in SPSO2011. This was also observed for IPSO in [29] and [31].

Also, based on point 2, it seems that the set of coefficients that results in convergence (according to the experiment) is different from that of IPSO. Hence, one should not simply use the same frequently used parameters in the PSO literature for SPSO2011.

Based on point 3, the convergence boundaries for SPSO2011 are affected by the number of dimensions. In order to examine to what extent the convergence boundaries for SPSO2011 change when the number of dimensions grows, we used EVCB to estimate the maximum value of ϕ so that particles are still convergent for $d \in \{1, 2, \ldots, 10\}$ and $\omega \in \{0.5, 0.6, \ldots, 0.9\}$ (see Fig. 4). In this test, the maxIter was set to 50 000 to ensure that particles are stable toward the end of the run.

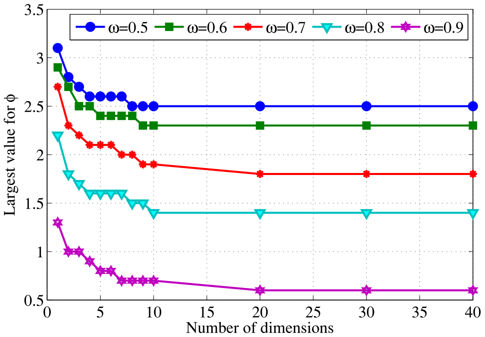
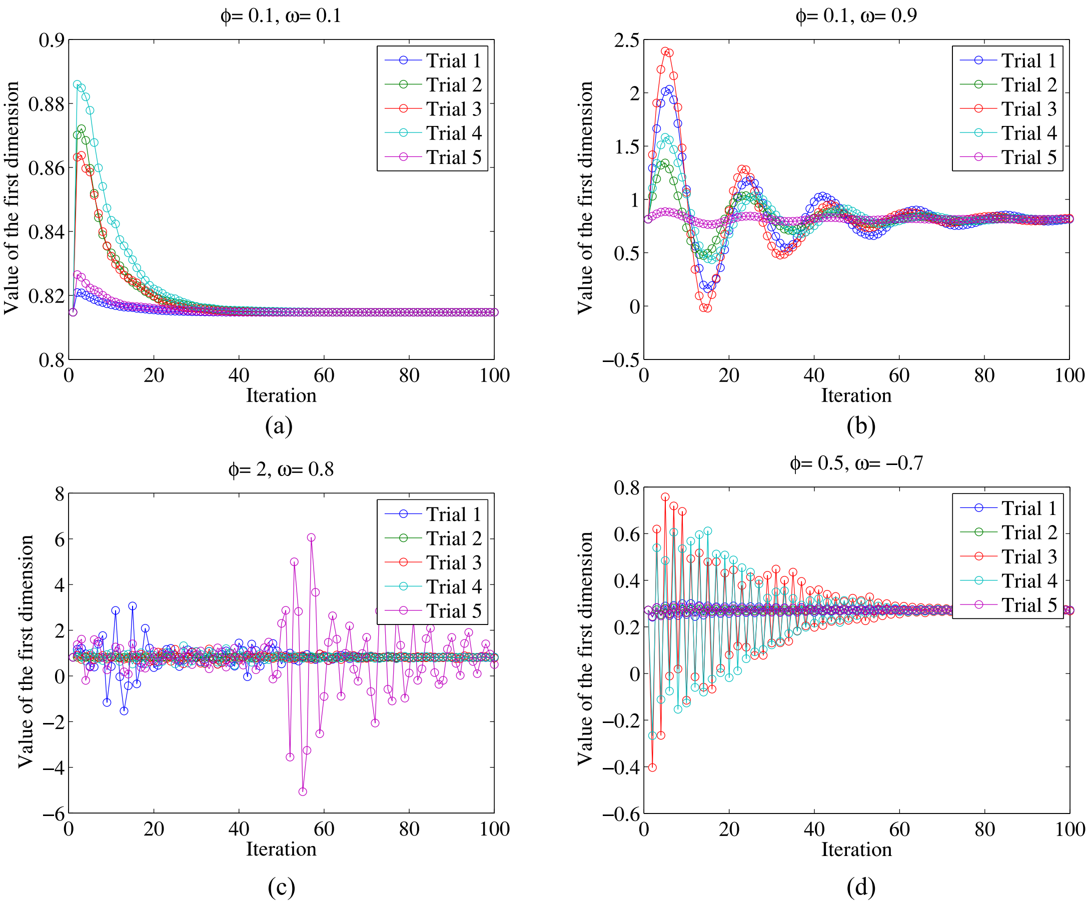


Fig. 4. Maximum value of ϕ for different values of ω and number of dimensions where the particles are still convergent.

Results indicate that the most significant changes for the maximum value of ϕ take place between 1-D and 10-D and the maximum value for ϕ remains almost unchanged for larger

[6] Note that these boundaries also look different from that of reported in our earlier paper [8]. The reason is that in that paper we considered the SPSO2011 proposed in [7] while here we experimented with the original version described in [6]. Also, the implementation of the spherical distribution in [8] was different from what we used in this paper.



(a)

(b)

(c)

(d)

Fig. 5. Examples for (a) nonoscillatory convergence, (b) harmonic oscillation, (c) harmonic combined with zigzagging, and (d) zigzagging oscillations of the positions of particles.

number of dimensions. Hence, a parameter setting that is conducted for a 10-D space is most likely applicable to a wide range of number of dimensions.

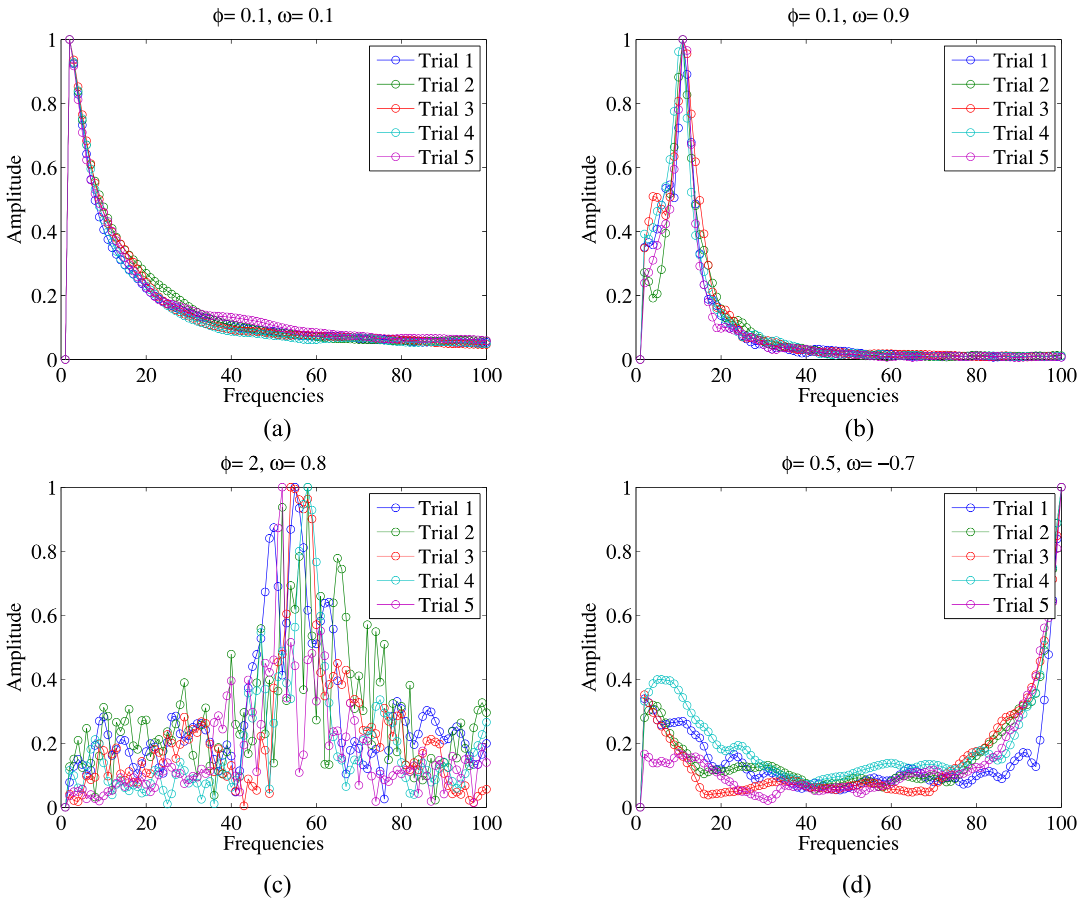
C. Convergence Behavior

If the values of coefficients in a PSO variant are selected in the convergence boundaries then particles converge to their equilibrium. During the run, however, particles oscillate in different ways (e.g., harmonic and zigzagging) around their equilibrium until they collapse on it (converge). These different oscillations are a consequence of different values of coefficients of the velocity update rule [10] and, potentially, impact the final solutions found by the algorithm. Fig. 5 shows examples of these oscillations for IPSO.

These different behaviors potentially affect the quality of final solutions found by the particles. As an example, a nonoscillatory behavior [Fig. 5 (a)] causes particles to search only one side of each dimension of the equilibrium point. This behavior can be beneficial when searching boundaries of the search space (feasible space) is required. Harmonic behavior [Fig. 5 (b)] is beneficial in the exploitation phase as particles smoothly oscillate around their equilibrium point (exploitation) and, potentially, higher quality solutions might be found. This behavior can also be beneficial when the search space is smooth (not rugged). The zigzagging behavior [Fig. 5 (d)] is more beneficial for the exploitation phase as particles jump all around the space to look for better basins of attraction. This behavior can be more useful in rugged search spaces. The combined harmonic with zigzagging behavior [Fig. 5 (c)] can be beneficial for the transition from exploration to exploitation phase. If the boundaries of coefficients associated with these behaviors are known then one can design an adaptive approach that changes the values of coefficients according to the most beneficial behavior. Note, however, that designing such adaptive approach is not a trivial task as it needs strategies to identify the most beneficial behavior in the first place.

These behaviors were studied by [10] and later by [32] for IPSO. In those studies, the position update rule of the particles in IPSO was written as a recursion equation, $x_{t+1} = \omega(x_t - x_{t-1}) + \phi_1 R_{1,t}(p_t - x_t) + \phi_2 R_{2,t}(l_t - x_t)$, and was investigated by ordinary differential equation techniques (this recursion is correspondence with the following second-order differential equation: $x'' = \omega(x' - x) + \phi_1 R_{1,t}(p_t - x') + \phi_2 R_{2,t}(l_t - x')$). In [10], the characteristic equation of this differential equation was used to calculate the boundaries associated with different behaviors [boundaries calculated in [10] are shown in Fig. 7(a)]. Fig. 7(a) shows that the behavior of particles for the frequently used setting of coefficients in IPSO ($\omega = 0.73$ and $\phi = 1.5$) is harmonic.

As investigation of the characteristic equation of PSO variants is not always as easy, we propose an estimation approach



(a)

(b)

(c)

(d)

Fig. 6. (a) Nonoscillatory convergence. (b) Slow harmonic oscillation. (c) Harmonic combined with zigzagging. (d) Zigzagging.

that uses frequency domain information of particle positions in order to specify these boundaries. Let us have a closer look at the behavior of some sample particles in the time (see Fig. 5) and frequency (see Fig. 6) domains (amplitude in the frequency domain). We used the Fourier transform to convert the first 200 samples of the particle positions from time to frequency domain. One may observe the following.

1) In Fig. 5(a), the signal (particle position) converges to its steady point without any oscillation (or with small oscillations). Thus, the maximum amplitude of this signal in the frequency domain belongs to very low frequencies [see Fig. 6(a)].
2) In Fig. 5(b), the signal converges to its steady point with some harmonic oscillations. Thus, the maximum amplitude in the frequency domain belongs to low to mid-range frequencies [see Fig. 6(b)].
3) In Fig. 5(c), the signal converges to its steady point with some zigzag together with harmonic oscillations. Thus, the maximum amplitude in the frequency domain belongs to mid-range to high frequencies [see Fig. 6(c)].
4) In Fig. 5(d), the signal converges to its steady point with some zigzagging oscillations. Thus, the maximum amplitude in the frequency domain belongs to very high frequencies [see Fig. 6(d)].

Thus, the boundaries in Fig. 7(a) can be estimated by drawing the contour of the maximum frequency of the positions of particles for different values of ω and ϕ. This contour has been presented in Fig. 7(b). By comparing Fig. 7(a) and (b), it is clear that the pattern of the maximum amplitude in the frequency domain of particles position is very similar to what was found theoretically [10] for the behavior of particles.

According to Fig. 7(b) small values for ϕ and negative values for ω result in low-frequency oscillations. This is in contradiction with what was observed in Fig. 7(a). The reason is that the oscillation of particles in that area (ω negative and ϕ small) is a combination of zigzagging and nonoscillatory, which results in large amplitudes in both very high and very low frequencies. Note also that, according to Fig. 7(b), the behavior of particles is zigzagging for $\phi = 0$ and $\omega < 0$. The reason is that when the value of ϕ is zero, only the inertia component affects the behavior of movement. As $\omega < 0$, the sign of the value of each dimension of velocity changes in each iteration, which implies zigzagging behavior (high-frequency changes).

We calculated the maximum frequency of particle positions in SPSO2011 for 1-D space (see Fig. 8). It is clear that the spectrum of maximum frequencies in SPSO2011 is different from that of IPSO. For a positive value of ω, the behavior of particles changes from nonoscillatory to harmonic when ϕ increases. If the value of ω is negative, however, particles behave more nonoscillatory for small values of ϕ and more zigzagging for larger values of ϕ. These patterns are almost consistent in both 1-D and 10-D cases.

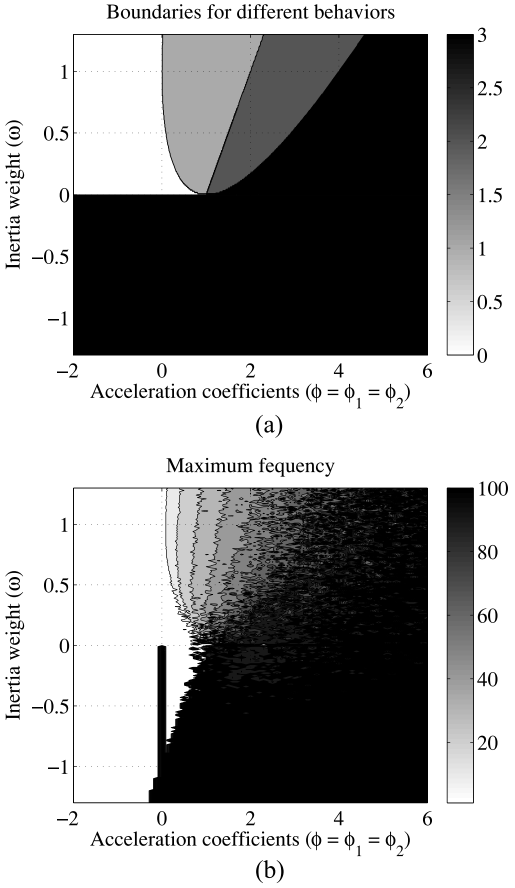


Fig. 7. Different behavior of particles before convergence in IPSO. (a) Spectrum of different behaviors found in [10], where nonoscillatory convergence (Value 0), harmonic oscillation (Value 1), harmonic combined with zigzagging (Value 2), and zigzagging (Value 3) have been represented by integers from 0 to 3, respectively. (b) Maximum frequency of oscillation of particles with different coefficient values.

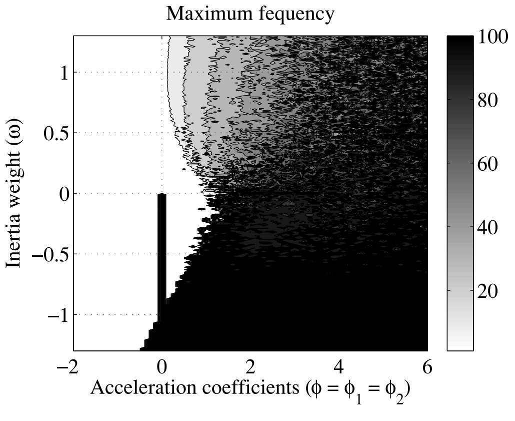


Fig. 8. Different behavior of particles before convergence in SPSO2011. The spectrum shows the maximum frequency of position of a particle in a 1-D space. The results for the 10-D space was similar.

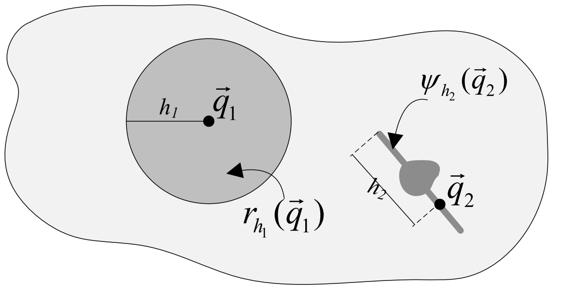


Fig. 9. Light-gray area represents a search space S and the dark-gray circle represents $r_{h_1}(\vec{q_1})$. Also, an example of the set $\psi_{h_2}(\vec{q_2})$ has been shown in the figure. $n_{h_1,h_2}(\vec{q}_1, \vec{q}_2)$ is a random point that is inside the dark-gray areas with nonzero probability.

Fig. 8 shows that for $\omega = 1$ and ϕ from 0 to almost 3, the oscillation of particles is harmonic while this behavior is transformed to zigzagging for larger ϕ. Furthermore, for $\omega < 0$, the behavior is zigzagging (the reason is the same as what was explained for IPSO). Also, for small values of ω (0 to 0.5) and small values of ϕ (0 to 1.5) the particles are nonoscillatory. The behavior is also nonoscillatory if ϕ is negative for any ω.

These observations are useful for practitioners to pick appropriate values for coefficients according to the specifications of the search space.

IV. Local Convergence

If the point $\vec{X}$ in (9) is a local optimum and a stochastic algorithm guarantees the satisfaction of (9) then that algorithm is said to be locally convergent. In the context of PSO, the local convergence condition is written as follows [33]:

$$\forall \epsilon > 0 \ \forall i, \lim_{t\to\infty} P\left(\left|\left|\vec{p}_t^{\,i} - \vec{X}\right|\right| < \epsilon\right) = 1. \tag{10}$$

If the personal bests of all particles in a PSO variant converge to a local optimum[7] then that PSO variant is locally convergent. Alternatively, as the final output of a PSO variant is the best personal best over the swarm, a PSO variant is locally convergent if [33]

$$\forall \epsilon > 0, \lim_{t\to\infty} P\left(\left\|\vec{g}_t - \vec{X}\right\| < \epsilon\right) = 1 \tag{11}$$

where g_t is the best found solution over the swarm.

Local convergence is an important characteristic of an optimization algorithm. If an optimization algorithm is not locally convergent, the final solution of the algorithm might be a point that can be improved further, i.e., the gradient at the final point is nonzero, while the algorithm has stopped searching. There might be different approaches to solve this issue. At the meta-algorithmic level, one can hybridize the algorithm with another algorithm that is locally convergent and ensure that the locally convergent algorithm is applied at appropriate time (see [34] for example). Another approach at the meta-algorithmic level is to restart some (or all) of the particles whenever they stagnate and initialize them randomly (see [35], [36] for example). The issue can also be addressed at the algorithmic level, i.e., guarantee that the generated solutions by the algorithm converge to a local optimum (see [12], [33], [37] for examples).

In this section, some earlier analyses on the local convergence property of IPSO at the algorithmic level are described and the local convergence for SPSO2011 is investigated.

[7] c_i is a local minimum of an objective function F over the search space S if there exists an open interval $I_i \subseteq S$ such that $F(c_i) \leq F(x)$ for all $x \in I_i$ and $c_i \in I_i$.

A. Local Convergence for IPSO

Local convergence for IPSO has been investigated in [12], [33], and [37]. A PSO variant is locally convergent if it guarantees finding a local optimum in the search space. In IPSO, if $\vec{p}_t^i = \vec{g}_t = \vec{x}_t^i$, then $\vec{x}_t^i$ is moved in the next step only if $\omega\vec{V}_t^i$ is nonzero. However, if $\vec{V}_t^i$ becomes zero then $\vec{V}_{t+1}^i = 0$ and, consequently, no movement takes place in the next step [12]. Also, because $\vec{V}_{t+1}^i = 0$ and $\vec{p}_t^i = \vec{g}_t = \vec{x}_t^i = \vec{x}_{t+1}^i$, particles do not move even during further steps, i.e., the particle is in its equilibrium. Note that there is no guarantee that the equilibrium point is a high quality point (e.g., a local optimum); hence, there is no guarantee that IPSO is locally convergent.

It was proposed [12] to mutate the position of the global best particle (the particle whose personal best is $\vec{g}_t$) to a random point (with some distribution) "around" the current global best vector to guarantee local convergence. Hence, by using this strategy, the global best particle continues movement randomly around $\vec{g}_t$ even if all particles stop moving. It was proven that the algorithm is locally convergent. The general idea behind the proof was that if there is any better solution around the global best vector then that solution is found by the perturbation with nonzero probability. Thus, the algorithm will converge to a local optimum eventually.

The local convergence for IPSO was investigated in [37] where the authors proved that IPSO is locally convergent in a 1-D space if $n > 1$ and coefficients are set in the convergence boundaries. However, this conclusion is not generalizable to a multidimensional spaces. To address this issue, it was proposed to regenerate the velocity vector if $\|\vec{V}_t^i\| + \|\vec{g}_t - \vec{x}_t^i\|$ is smaller than a constant δ. It was proven that this approach guarantees local convergence.

A PSO variant called Locally convergent Rotationally invariant Particle Swarm Optimization (LcRiPSO) was proposed in [33]. The velocity update rule for LcRiPSO was written as

$$\vec{V}_{t+1}^i = \omega\vec{V}_t^i + \sum_{j\in E_t^i} r_{jt}^i \phi_j \left(f_{jt}^i\left(\vec{p}_t^i\right) - \vec{x}_t^i\right) \tag{12}$$

where r_{jt}^i is a uniform random scalar in the interval [0, 1] and f_{jt}^i is a function and $f : R^d \to R^d$, E_t^i is the set of all neighbors of the particle i that contribute into its velocity update rule. It was proven that LcRiPSO is locally convergent if the function f satisfies the following condition:

$$\forall\vec{y} \in S\ \exists A_y \subseteq S\ \forall z \in A_y\ \forall\delta > 0, P(||f(\vec{y}) - \vec{z}|| < \delta) > 0$$

where $\vec{y}$ is an arbitrary point in the search space S, A_y is an open set that contains $\vec{y}$, $\vec{z}$ is an arbitrary point in A_y, and δ is a positive value. In other words, LcRiPSO is locally convergent if the function f is designed in such a way that for any input vector $\vec{y}$ in the search space, there exists an open region A which contains $\vec{y}$ and $f(\vec{y})$ can be located anywhere in A. To the best of our knowledge, this variant is the only PSO variant that guarantees both local convergence and rotation invariance at the same time.

B. Local Convergence for SPSO2011

SPSO2011 in its original form is not locally convergent. Assume that $\vec{p}_t^i = \vec{l}_t^i = \vec{x}_t^i$ in SPSO2011. The value of $\vec{G}_t^i$ in (6) is then calculated by

$$\vec{G}_t^i = \frac{\vec{x}_t^i + \phi_1\left(\vec{p}_t^i - \vec{x}_t^i\right) + \vec{x}_t^i}{2} = \vec{x}_t^i. \tag{13}$$

Thus, in this case, $\|\vec{G}_t^i - \vec{x}_t^i\|$ is zero, which implies that $H(\vec{G}_t^i, \|\vec{G}_t^i - \vec{x}_t^i\|) = \vec{G}_t^i = \vec{x}_t^i$ for all t. Hence, the value of velocity for $t + 1$ is $\vec{V}_{t+1}^i = \omega\vec{V}_t^i$. This is exactly the same as the case we discussed in Section IV-A for IPSO (i.e., if $\vec{V}_t^i = 0$, the particle does not move in further iterations). Using the same analysis as in Section IV-A, SPSO2011 is not locally convergent in general.

There is also another condition for which SPSO2011 is not locally convergent that can be found in [8].

C. Locally Convergent SPSO2011

A simple modification of SPSO2011 to guarantee local convergence is to bound $\|\vec{G}_t^i - \vec{x}_t^i\|$ to a small nonzero value $\delta > 0$ [8]. Before we prove that this strategy actually fixes the local convergence issue, we introduce some definitions, notations, and lemmas.

We define three notations, $r_h(\vec{q})$, $\psi_h(\vec{q})$, and $n_{h_1,h_2}(\vec{q}_1, \vec{q}_2)$ as follows.

Notation 1: $r_h(\vec{q})$ is the set of all points in a hyper-ball with the center $\vec{q}$ and radius $h > 0$.

Notation 2: $\psi_h(\vec{q})$ is a connected set such that $\psi_h(\vec{q}) \subseteq S$, $\vec{q} \in \psi_h(\vec{q})$, $\psi_h(\vec{q}) - \{\vec{q}\}$ is nonempty, and

$$\forall\vec{z} \in \psi_h(\vec{q}), ||\vec{z} - \vec{q}|| \leq h$$

i.e., any point in $\psi_h(\vec{q})$ is closer (Euclidean distance) than or as far as h to $\vec{q}$.

Notation 3: $n_{h_1,h_2}(\vec{q}_1, \vec{q}_2)$ is a random point generated by a probability distribution in $r_{h_1}(\vec{q}_1) \cup \psi_{h_2}(\vec{q}_2)$ and

$$\forall\vec{q}_1, \vec{q}_2 \in S\ \exists h_1, h_2 > 0\ \forall z \in r_{h_1}(\vec{q}_1) \cup \psi_{h_2}(\vec{q}_2)\ \forall\epsilon > 0,$$
$$P\left(n_{h_1,h_2}(\vec{q}_1, \vec{q}_2) \in r_\epsilon(z)\right) > 0$$

where P is the probability measure.

According to Notation 1 any point in $r_h(\vec{q})$ can be written as $\vec{q} + \vec{a}$ where $\|\vec{a}\| \leq h$.

According to Notation 2 any point in $\psi_h(\vec{q})$ can be written as $\vec{q} + \vec{a}$ where $\|\vec{a}\| \leq h$.

According to Notation 3 any point generated by $n_{h_1,h_2}(\vec{q}_1, \vec{q}_2)$ has nonzero probability to be in $r_{h_1}(\vec{q}_1)$ and also nonzero probability to be in $\psi_{h_2}(\vec{q}_2)$.

Fig. 9 shows examples of the areas defined in Notations 1, 2, and 3.

We define a local minimizer and an optimality region of the objective function F over the search space S as follows.

Definition 1: c_i is a local minimizer of an objective function F over the search space S if there exists an open set $I_i \subset S$ such that $F(c_i) \leq F(x)$ for all $x \in I_i$ and $c_i \in I_i$.

Definition 2: The optimality region of the objective function F is defined as $R_\epsilon = \bigcup_i R_{\epsilon,i}$ where $R_{\epsilon,i} = \{x \in I_i : F(x) < F(c_i) + \epsilon\}$ and ϵ is an arbitrarily small positive value.

The aim of a local search algorithm is to find a point in the search space that is within the optimality region [12], [22], [33].

We define a general form of stochastic algorithms (General Stochastic Algorithm, GSA) as follows.

Definition 3: GSA has following three steps.

1) Initialize $\vec{p}_0$ from the search space S and set $t = 1$.
2) Generate a random sample $\vec{x}_t$ from S.
3) Generate the candidate solution $\vec{p}_t = D(\vec{p}_{t-1}, \vec{x}_t)$, set $t = t + 1$, and go to 2 where $D(a, b)$ is defined by

$$D(a, b) = \begin{cases} b & F(b) < F(a) - \epsilon_0 \\ a & \text{otherwise} \end{cases}$$

and ϵ_0 is a positive value that is smaller than or equal to ϵ (ϵ in the definition of $R_{\epsilon,i}$).

The operator D updates $\vec{p}_t$ if and only if the new solution $\vec{x}_t$ is better than $\vec{p}_{t-1}$ by at least the constant ϵ_0. The constant ϵ_0 specifies the precision of the calculations and can be considered arbitrarily small. This value needs to be considered in the calculations as numerical methods are simulated on physically limited computers, hence, the lack of this consideration can affect the generality of the proofs and statements. One can set ϵ_0 to the smallest possible float/double value [20], [33] in computer simulations to achieve maximum precision.

Lemma 1: If a GSA guarantees

$$\begin{aligned} &\exists \epsilon > 0 \; \exists \eta > 0 \; \exists \delta \in (0, 1] \; \forall t \geq 0 \; \exists t' > 0, \\ &P\big(F\big(\vec{p}_{t+t'}\big) \leq F(\vec{p}_t) - \eta\big) \geq \delta \text{ or } \vec{p}_t \in R_\epsilon \end{aligned} \tag{14}$$

then that GSA is locally convergent.

Proof: This has been proven as Lemma 1 in [33]. ■

In fact, if the probability that $F(\vec{p}_{t+t'})$ is smaller than $F(\vec{p}_t)$ by at least η is larger than δ unless $\vec{p}_t$ is already in the optimality region then GSA is locally convergent.

Lemma 2: If a GSA guarantees

$$\begin{aligned} &\forall t > 0 \; \exists z, h' > 0 \; \forall u \in r_{h'}(\vec{p}_t) \; \forall \epsilon_0 > 0, \\ &P(\|\vec{x}_{t+z} - u\| < \epsilon_0) > 0 \end{aligned}$$

then the condition in Lemma 1 is satisfied, i.e., if there exists an iteration z that $\vec{x}_{t+z}$ has nonzero probability to be arbitrarily close to any point in $r_{h'}(\vec{p}_t)$ then the condition in Lemma 1 is satisfied.

Proof: There are two cases: 1) there exists a set $A \subset r_{h'}(\vec{p}_t)$ such that all points in A are better than $\vec{p}_t$ by at least ϵ_0 or, 2) there is no point inside $r_{h'}(\vec{p}_t)$ that is better than $\vec{p}_t$ by at least ϵ_0. In the first case, because $\vec{x}_{t+z}$ can be arbitrarily close to any point in $r_{h'}(\vec{p}_t)$ with nonzero probability, and $A \subset r_{h'}(\vec{p}_t)$, the point $\vec{x}_{t+z}$ has nonzero probability to be inside A, that satisfies the condition in Lemma 1 with setting $\epsilon_0 = \eta$. In the second case, $\vec{p}_t$ is already in the optimality region that also satisfies the condition in Lemma 1. Thus, in both cases the condition in Lemma 1 is satisfied, that completes the proof. ■

Lemma 3: Any recursion in the form of $\vec{x}_{t+1} = \omega(\vec{x}_t - \vec{x}_{t-1}) + n_{h_1,h_2}(\vec{p}_t, \vec{x}_t)$, where $\vec{p}_t = D(\vec{p}_{t-1}, \vec{x}_t)$, is locally convergent for any $\omega \in (0, 1)$, $\vec{x}_0 \in S$, and $\vec{p}_0 \in S$.

Proof: The recursion $\vec{x}_{t+1} = \omega(\vec{x}_t - \vec{x}_{t-1}) + n_{h_1,h_2}(\vec{p}_t, \vec{x}_t)$, where $\vec{p}_t = D(\vec{p}_{t-1}, \vec{x}_t)$, is an instance of GSA. Thus, if we prove that

$$\begin{aligned} &\exists \epsilon > 0 \; \exists \eta > 0 \; \exists \delta \in (0, 1] \; \forall t \geq 0 \; \exists t' > 0, \\ &P\big(F\big(\vec{p}_{t+t'}\big) \leq F(\vec{p}_t) - \eta\big) \geq \delta \text{ or } \vec{p}_t \in R_\epsilon \end{aligned}$$

then, according to Lemma 1, this recursion is locally convergent.

For any $z > 0$, two cases might take place until iteration $t + z > 0$.

1) *Case 1:* $F(\vec{p}_{t+z}) < F(\vec{p}_t)$.
2) *Case 2:* $F(\vec{p}_{t+z}) = F(\vec{p}_t)$.

In Case 1, because the operator D updates $\vec{p}_t$ if the new found solution is better than $\vec{p}_t$ by at least ϵ_0, condition in Lemma 1 is satisfied for $\delta = 1$ if we set $\epsilon_0 = \eta$ because $\vec{p}_t$ has been already updated. We continue with Case 2. Note that, from here on we always write $\vec{p}_t$ as $\vec{p}_{t+z} = \vec{p}_t$ for all z. For Case 2, we prove that for all $t > 0$, there exists $z > 0$ and $h' > 0$ such that for any point u in $r_{h'}(\vec{p}_t)$, $P(\|\vec{x}_{t+z} - u\| < \epsilon_0) > 0$, that, according to Lemma 2, satisfies the condition in Lemma 1 and completes the proof for the local convergence.

One can write $\vec{x}_{t+2} = \omega(\vec{x}_{t+1} - \vec{x}_t) + n_{h_1,h_2}(\vec{p}_t, \vec{x}_{t+1})$ that indicates $\vec{x}_{t+2} = \omega(\omega(\vec{x}_t - \vec{x}_{t-1}) + n_{h_1,h_2}(\vec{p}_t, \vec{x}_t) - \vec{x}_t) n_{h_1,h_2}(\vec{p}_t, \vec{x}_{t+1})$. According to Notation 3, $n_{h_1,h_2}(\vec{p}_t, \vec{x}_t)$ can sample a point in $\psi_{h_2}(\vec{x}_t)$ with nonzero probability, thus can be written as $\vec{x}_t + \vec{a}$ where $\vec{a}$ is a random vector and $\|\vec{a}\| \leq h_2$. Hence, with nonzero probability, $\vec{x}_{t+2} = \omega(\omega(\vec{x}_t - \vec{x}_{t-1}) + \vec{x}_t + \vec{a}_1 - \vec{x}_t) + n_{h_1,h_2}(\vec{p}_t, \vec{x}_{t+1}) = \omega^2(\vec{x}_t - \vec{x}_{t-1}) + \omega\vec{a}_1 + n_{h_1,h_2}(\vec{p}_t, \vec{x}_{t+1})$ where $\vec{a}_1$ is a random vector and $\|\vec{a}_1\| \leq h_2$. Also, one can write $\vec{x}_{t+3} = \omega(\vec{x}_{t+2} - \vec{x}_{t+1}) + n_{h_1,h_2}(\vec{p}_t, \vec{x}_{t+2}) = \omega(\omega^2(\vec{x}_t - \vec{x}_{t-1}) + \omega\vec{a}_1 + n_{h_1,h_2}(\vec{p}_t, \vec{x}_{t+1}) - \vec{x}_{t+1}) + n_{h_1,h_2}(\vec{p}_t, \vec{x}_{t+2})$. Again, according to Notation 3, $\vec{x}_{t+3} = \omega^3(\vec{x}_t - \vec{x}_{t-1}) + \omega^2\vec{a}_1 + \omega\vec{a}_2 + n_{h_1,h_2}(\vec{p}_t, \vec{x}_{t+2})$ with nonzero probability, where $\vec{a}$ is a random vector and $\|\vec{a}_2\| \leq h_2$. If we continue the same strategy, we can write $\vec{x}_{t+z}$ as

$$\vec{x}_{t+z} = \omega^z(\vec{x}_t - \vec{x}_{t-1}) + \sum_{j=1}^{z-1} \omega^{z-j}\vec{a}_j + n_{h_1,h_2}(\vec{p}_t, \vec{x}_{t+z-1}) \tag{15}$$

where $\vec{a}_j$ are random vectors and $\|\vec{a}_j\| \leq h_2$ for all j.

If we prove that

$$\forall h_1 > 0 \; \exists z, h_2, h' > 0, r_{h'}(\vec{p}_t) \subseteq Y_{z,h_1,h_2}$$

where Y_{z,h_1,h_2} is the set of all possible points that can be sampled by $\omega^z(\vec{x}_t - \vec{x}_{t-1}) + \sum_{j=1}^{z-1} \omega^{z-j}\vec{a}_j + n_{h_1,h_2}(\vec{p}_t, \vec{x}_{t+z-1})$, then $\vec{x}_{t+z}$ can be arbitrarily close to any point in $r_{h'}(\vec{p}_t)$ with nonzero probability, that satisfies the condition in Lemma 2. Clearly, the maximum length of $\sum_{j=1}^{z-1} \omega^{z-j}\vec{a}_j$ is $(\omega - \omega^z)/(1 - \omega)h_2$. Let us assume $h' = h_1 - [\omega^z(\|\vec{x}_t - \vec{x}_{t-1}\|) + (\omega - \omega^z)/(1 - \omega)h_2]$. Thus, we need to guarantee that

$$\forall h_1 > 0 \; \exists z, h_2 > 0, h_1 - \left[\omega^z(\vec{x}_t - \vec{x}_{t-1}) + \frac{\omega - \omega^z}{1 - \omega}h_2\right] > 0.$$

If we set $h_2 < ((1 - \omega)(h_1 - \omega^z(\|\vec{x}_t - \vec{x}_{t-1}\|)))/(\omega - \omega^z)$ then $h_1 - [\omega^z(\vec{x}_t - \vec{x}_{t-1}) + (\omega - \omega^z)/(1 - \omega)h_2] > 0$ is guaranteed for large enough z. The reason is that for any h_1 the term $((1 - \omega)(h_1 - \omega^z(\|\vec{x}_t - \vec{x}_{t-1}\|)))/(\omega - \omega^z)$ is positive if z is large enough. In fact, a large value for z diminishes the

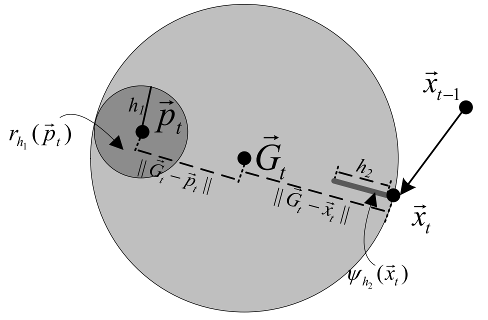


Fig. 10. Dark-gray area represents $n_{h_1,h_2}(\vec{p}^{\tau}_t, \vec{x}^{\tau}_t)$, see Lemma 4.

effect of $\omega^z\|\vec{x}_t - \vec{x}_{t-1}\|$ and, as $\omega - \omega^z$, h_1, and $1 - \omega$ are larger than 0, $((1-\omega)(h_1 - \omega^z(\|\vec{x}_t - \vec{x}_{t-1}\|)))/(\omega - \omega^z)$ is also larger than 0. Hence, for any h_1, there exists z and h_2 such that $h' > 0$ with nonzero probability. Therefore, $\vec{x}_{t+z}$ can be arbitrarily close to any point inside a hyper-ball with the center $\vec{p}_t$ and radius $h' > 0$ with nonzero probability. According to Lemma 2, in this case, the condition in Lemma 1 is guaranteed, which guarantees local convergence of the algorithm and completes the proof. ■

This lemma can be useful to prove local convergence of any PSO variant that follows the mentioned recursion form. In order to prove local convergence of the modified SPSO2011, we first need to find whether the modified SPSO2011 follows the recursion form in Lemma 3.

Lemma 4: In SPSO2011, if

$$\exists\delta > 0\ \forall\phi > 1\ \forall t > 0,\ \left\|\vec{G}^{\tau}_t - \vec{x}^{\tau}_t\right\| > \delta$$

where $\phi = \phi_1 = \phi_2$ then

$$\forall\vec{p}^{\tau}_t, \vec{x}^{\tau}_t \in S\ \exists h_1, h_2 > 0\ \forall z \in r_{h_1}\left(\vec{p}^{\tau}_t\right) \cup \psi_{h_2}\left(\vec{x}^{\tau}_t\right)$$
$$\forall\epsilon > 0, P\left(H\left(\vec{G}^{\tau}_t, \left\|\vec{G}^{\tau}_t - \vec{x}^{\tau}_t\right\|\right) \in r_\epsilon(z)\right) > 0.$$

In other words, for any $\vec{p}^{\tau}_t$ and $\vec{x}^{\tau}_t$ in the search space, there exist $h_1, h_2 > 0$ such that $H(\vec{G}^{\tau}_t, \|\vec{G}^{\tau}_t - \vec{x}^{\tau}_t\|)$ can be arbitrarily close to any point in $r_{h_1}(\vec{p}^{\tau}_t) \cup \psi_{h_2}(\vec{x}^{\tau}_t)$ with nonzero probability, where τ is the index of the best particle in the swarm.

Proof: $H(\vec{G}^{\tau}_t, \|\vec{G}^{\tau}_t - \vec{x}^{\tau}_t\|)$ can be arbitrarily close to any point on the line segment that connects $\vec{G}^{\tau}_t$ and $\vec{x}^{\tau}_t$. As $\|\vec{G}^{\tau}_t - \vec{x}^{\tau}_t\| > \delta$, thus this line segment always exists, and we can consider $0 < h_2 < \delta$. Hence, this line segment can be considered as the set $\psi_{h_2}(\vec{G}^{\tau}_t)$ in Notation 1 and $h_2 > 0$ always exists[8] (see Fig. 10). If we prove that $\|\vec{G}^{\tau}_t - \vec{p}^{\tau}_t\| < \|\vec{G}^{\tau}_t - \vec{x}^{\tau}_t\|$ with nonzero probability then there exists a $h_1 > 0$ such that $H(\vec{G}^{\tau}_t, \|\vec{G}^{\tau}_t - \vec{x}^{\tau}_t\|)$ can generate a point that can be arbitrarily close to any point in $r_{h_1}(\vec{p}^{\tau}_t)$ as well (see Fig. 10), which completes the proof. We calculate $\vec{G}^{\tau}_t = \vec{x}^{\tau}_t + (\phi/2)(\vec{p}^{\tau}_t - \vec{x}^{\tau}_t)$ [note that for the particle τ, $\vec{p}^{\tau}_t = \vec{l}^{\tau}_t = \vec{g}_t$, where g_t is the best solution found over the swarm, see (6)]. Thus, $\|\vec{G}^{\tau}_t - \vec{p}^{\tau}_t\| = \|\vec{x}^{\tau}_t + (\phi/2)(\vec{p}^{\tau}_t - \vec{x}^{\tau}_t) - \vec{p}^{\tau}_t\| = \|((\phi/2) - 1)(\vec{p}^{\tau}_t - \vec{x}^{\tau}_t)\|$. This value is obviously smaller than $\|\vec{G}^{\tau}_t - \vec{x}^{\tau}_t\| = \|\vec{x}^{\tau}_t + (\phi/2)(\vec{p}^{\tau}_t - \vec{x}^{\tau}_t) - \vec{x}^{\tau}_t\| = \|(\phi/2)(\vec{p}^{\tau}_t - \vec{x}^{\tau}_t)\|$ for any $\phi > 1$. Hence, there exists $h_1, h_2 > 0$ such that $H(\vec{G}^{\tau}_t, \|\vec{G}^{\tau}_t - \vec{x}^{\tau}_t\|)$ can be arbitrarily close to any point in $r_{h_1}(\vec{p}^{\tau}_t) \cup \psi_{h_2}(\vec{x}^{\tau}_t)$ with nonzero probability. ■

This shows that $H(\vec{G}^{\tau}_t, \|\vec{G}^{\tau}_t - \vec{x}^{\tau}_t\|)$ with $\|\vec{G}^{\tau}_t - \vec{x}^{\tau}_t\| \geq \delta$ is an instance of $n_{h_1,h_2}(\vec{p}_t, \vec{x}_t)$ as it is a probability distribution with the condition in Notation 3.

Now we are ready to prove the following theorem.

Theorem 1: If $\|\vec{G}^{\tau}_t - \vec{x}^{\tau}_t\| \geq \delta > 0$, $0 < \omega < 1$, and $\phi > 1$ are guaranteed then SPSO2011 is locally convergent, where τ is the index of the particle whose personal best is the best solution found by the swarm at iteration t.

Proof: The position update rule for the best particle in the swarm in SPSO2011 is written as: $\vec{x}_{t+1} = \omega(\vec{x}_t - \vec{x}_{t-1}) + H(\vec{G}^{\tau}_t, \|\vec{G}^{\tau}_t - \vec{x}^{\tau}_t\|)$. According to Lemma 4, if $\|\vec{G}^{\tau}_t - \vec{x}^{\tau}_t\| \geq \delta > 0$ and $\phi > 1$ then $H(\vec{G}^{\tau}_t, \|\vec{G}^{\tau}_t - \vec{x}^{\tau}_t\|)$ is an instance of $n_{h_1,h_2}(\vec{p}^{\tau}_t, \vec{x}^{\tau}_t)$. Thus, the best particle in SPSO2011 follows the recursion introduced in Lemma 3. Also, the best particle in SPSO2011 is updated if the new found solution is better than the previous best found solution by at least an ϵ. Thus, SPSO2011 is locally convergent if $\|\vec{G}^{\tau}_t - \vec{x}^{\tau}_t\| \geq \delta > 0$, $0 < \omega < 1$, and $\phi > 1$. ■

This shows that the proposed modification guarantees local convergence of the algorithm under specified conditions.

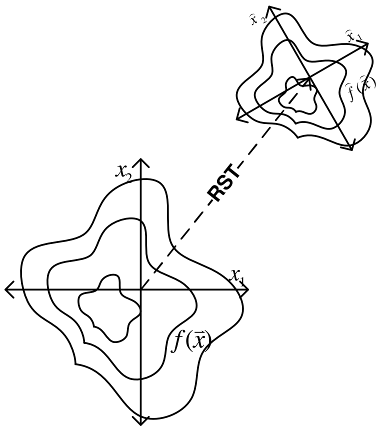


Fig. 11. RST of a search space.

V. Sensitivity to Transformations

Sensitivity to transformations, especially to RST, has been investigated for optimization algorithms [13], [33]. It is expected that the performance of an optimization algorithm does not change if the search space is rotated, scaled, or translated. The special form of transformation that only includes RST transformation in this paper (see Fig. 11).

Definition 4: Let $S' = \{\vec{y} : sQ\vec{x} + \vec{b}, \vec{x} \in S\}$ where $s \in R$, $Q \in R^d \times R^d$ is a rotation matrix, and $\vec{b} \in R^d$. Assume that $\hat{x}_t$ is an arbitrary point in S' and $\vec{x}_t \in S$ and $\hat{x}_t = sQ\vec{x}_t + \vec{b}$. An optimization algorithm is RST invariant if

$$\forall t > 0\ \forall s \in R\ \forall Q \in R^d \times R^d\ \forall\vec{b} \in R^d$$
$$\hat{x}_{t+1} = sQ\vec{x}_{t+1} + \vec{b} \qquad (16)$$

where $\hat{x}_{t+1}$ is the generated point by the algorithm at the iteration $t+1$ in $\hat{S}$ while $\vec{x}_{t+1}$ is the generated point by the algorithm in S.

[8] Another alternative for this set is the intersection between $r_{h_2}(\vec{x}^{\tau}_t)$ and $H(\vec{G}^{\tau}_t, \|\vec{G}^{\tau}_t - \vec{x}^{\tau}_t\|)$, where $h_2 < \delta$.

This definition means that the algorithm should preserve the relation between the points in the search space and the transformed space for any RST transformation. Also, based on this definition, a PSO algorithm is RST invariant if updating the position of particle i and then rotating, scaling, and translating the search space ($sQ\vec{x}_{t+1} + \vec{b}$) is the same as rotating, scaling, and translating the search space and then updating the position ($\hat{x}_{t+1}$) [13], [33], [38].

It is important that an algorithm is transformation invariant. In fact, if an algorithm is invariant of a transformation T then the performance of the algorithm on any problem P can be generalized to the complete class of problems C that are introduced by T and $P \in C$. This in fact enables researchers to make stronger statements about the performance of the algorithm. Although such statement might be about a bad or a good performance of the algorithm, it is valuable as one can study the reasons behind the performance and also generalize the claims. One well-known set of transformations is linear transformations (e.g., RST) that are frequently used in different areas. Hence, it is valuable to understand if an algorithm is invariant under any linear transformation.

In terms of real-world optimization, the variables (dimensions) of a problem usually represent physical concepts (e.g., temperature) that are related to each other through a function. If such function is nonseparable in its original form while becoming separable under a rotation α, then an algorithm that is sensitive to rotation might struggle to find an optimum solution of the original function even if it can find the optimum solution of the rotated function. However, as the rotation α is not known by the algorithm, another procedure is needed to find a rotation that enables the algorithm to find an optimum of the function successfully. This procedure might be very time consuming, especially for large-scale problems (note that the rotation should be performed along all possible axes). If the algorithm is rotation invariant, however, there is no need for such procedure that speeds up the optimization process. Of course making an algorithm rotationally invariant should not affect its performance.

A. Transformation Sensitivity for IPSO

It has been proven [13], [33], [39] that IPSO is scaling and translation invariant while it is rotation variant. It was proven [13] that the linear PSO (a PSO variant for which all values on the diagonal of R_{1t} and R_{2t} are equal) is RST invariant (this variant is called LPSO). However, LPSO suffers from another limitation investigated in [40] and [33]. If $(\vec{p}_t^i - \vec{x}_t^i) || (\vec{g}_t - \vec{x}_t^i)$ and $\vec{V}_t^i || (\vec{p}_t^i - \vec{x}_t^i)$, particle i oscillates between its personal best and the global best and it cannot sample other points in the search space.

A PSO variant was proposed in [13] that was proven to be rotationally invariant. In that variant, the random diagonal matrices were replaced by random rotation matrices to rotate the velocity vector. As generating random rotation matrices is computationally expensive, an approximation idea (an exponential map) was used that generated a rotation matrix M as

$$M = I + \sum_{i=1}^{\max_i} \frac{1}{i!}\left(\frac{\alpha\pi}{180}\left(E - E^T\right)\right)^i \tag{17}$$

where E is a $d \times d$ matrix with elements generated randomly in the interval [-0.5, 0.5], α is a real value representing the angle, and I is the identity matrix. The generated matrix M is an approximation of a random rotation matrix with the angle α. The value of $\max_i$ determines the accuracy of the estimation of a rotation matrix with the angle α [see (17)]. The value of $\max_i$ was set to 1 in [13] that limits the approximation to one term only. Thus, the approximation error of the random rotation matrix method grows with the rotation angle (α). The time complexity for generating the approximated rotation matrix with $\max_i = 1$ is in $O(d^2)$. Also, multiplying this matrix into a vector (SI or PI vectors) is also in $O(d^2)$.

It was shown [41] that IPSO performs poorly in comparison with covariance matrix adaptation evolutionary strategy [42] and differential evolution [43] when it is applied to nonseparable optimization problems. As rotating a search space usually makes the problem nonseparable [44], IPSO performs also poorly when the search space is rotated (it is rotation variant). This comparison was also conducted in [45] where it was found that IPSO performs poorly when the problem is nonseparable. A given explanation for this weak performance in comparison to other methods is that potential correlation between the variables is ignored in IPSO as all calculations in the algorithm are done for each dimension separately [45], which makes the algorithm rotationally variant.

The impact of rotation matrices on the performance of several PSO variants was studied by [46], where random Euclidean rotation matrices were used rather than the random diagonal matrices in several PSO variants (this variant is called RotPSO). A dynamic programming technique was used to generate Euclidean rotation matrices in low computational time, i.e., $O(d^2)$. A normal distribution was used to generate the directions of rotations with the mean of current direction of velocity and the variance that was set experimentally for different variants. This makes the particles mutate their direction of movement to find better solutions. Experiments showed that random rotation matrices can improve the performance of several PSO variants in most cases on the tested benchmarks.

The rotation variance issue was also investigated by [33] from theoretical perspective [see (12)]. It was proven [33] that if $sQf(\vec{y}) + \vec{b} = f(sQ\vec{y} + \vec{b})$ is true for all scalar s, orthogonal matrices Q ($Q \in \mathbb{R}^d \times \mathbb{R}^d$), and vectors $\vec{b}$ and $\vec{y}$ ($\vec{b}, \vec{y} \in \mathbb{R}^d$), then the PSO variant that uses velocity update rule in (12) is RST invariant.

B. Transformation Sensitivity for SPSO2011

In this section, we investigate RST invariance of SPSO2011.

Theorem 2: SPSO2011 is invariant under any scaling $s \in R$, rotation Q, and translation $\vec{b}$ of the search space.

Proof: The position update rule for SPSO2011 for a particle i is written as $\vec{x}_{t+1}^i = \omega(\vec{x}_t^i - \vec{x}_{t-1}^i) + H(\vec{G}_t^i, \|\vec{G}_t^i - \vec{x}_t^i\|)$.

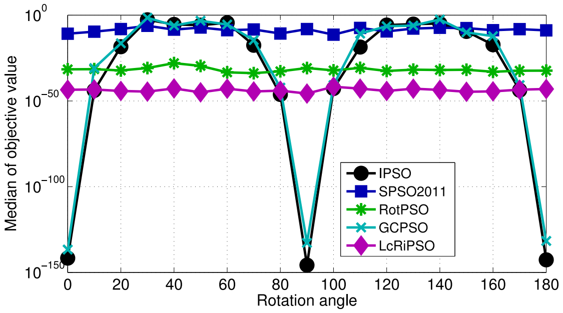


Fig. 12. Application of five PSO variants to an Ellipse function.

This can be rewritten as

$$\vec{x}^i_{t+1} = \omega\left(\vec{x}^i_t - \vec{x}^i_{t-1}\right) + \vec{T}^i_t + \vec{x}^i_t + H\left(0, \left\|\vec{T}^i_t\right\|\right) \tag{18}$$

where $\vec{T}^i_t = (\phi_1(\vec{p}^i_t - \vec{x}^i_t) + \phi_1(\vec{g}_t - \vec{x}^i_t))/3$. Let us calculate $\hat{x}^i_{t+1}$

$$\begin{aligned}\hat{x}^i_{t+1} &= \omega\left(\hat{x}^i_t - \hat{x}^i_{t-1}\right) + \hat{x}^i_t + \hat{T}^i_t + H\left(0, \left\|\hat{T}^i_t\right\|\right)\\ &= \omega\left(sQ\vec{x}^i_t - sQ\vec{x}^i_{t-1}\right) + sQ\vec{x}^i_t + \vec{b} + \hat{T}^i_t + H\left(0, \left\|\hat{T}^i_t\right\|\right)\end{aligned} \tag{19}$$

where $\hat{T}^i_t$ is calculated by

$$\begin{aligned}\hat{T}^i_t &= \frac{\phi_1\left(\hat{p}^i_t - \hat{x}^i_t\right) + \phi_1\left(\hat{g}_t - \hat{x}^i_t\right)}{3}\\ &= sQ\frac{\phi_1\left(\vec{p}^i_t - \vec{x}^i_t\right) + \phi_1\left(\vec{g}_t - \vec{x}^i_t\right)}{3} = sQ\vec{T}^i_t.\end{aligned}$$

Hence, (19) is rewritten as

$$\begin{aligned}\hat{x}^i_{t+1} &= \omega\left(sQ\vec{x}^i_t - sQ\vec{x}^i_{t-1}\right)\\ &\quad + sQ\vec{x}^i_t + \vec{b} + sQ\vec{T}^i_t + H\left(0, \left\|sQ\vec{T}^i_t\right\|\right).\end{aligned} \tag{20}$$

Also, we calculate $sQ\vec{x}^i_{t+1} + \vec{b}$ as

$$\begin{aligned}sQ\vec{x}^i_{t+1} + \vec{b} &= sQ\omega\left(\vec{x}^i_t - \vec{x}^i_{t-1}\right)\\ &\quad + sQ\vec{x}^i_t + sQ\vec{T}^i_t + sQH\left(0, \left\|\vec{T}^i_t\right\|\right) + \vec{b}.\end{aligned} \tag{21}$$

By comparing (20) and (21), it is clear that the algorithm is RST invariant if

$$sQH\left(0, \left\|\vec{T}^i_t\right\|\right) = H\left(0, \left\|sQ\vec{T}^i_t\right\|\right). \tag{22}$$

Rotating the space (matrix Q) preserves the Euclidean distances. Also, scaling the space and then generating a point by $H(.,.)$ is the same as generating a point and then scaling the space. Hence, (22) is true for a spherical distribution, that completes the proof. ■

In order to see the impact of transformation on different algorithms, we applied SPSO2011, IPSO, RotPSO, LcRiPSO, and GCPSO to a 2-D Ellipse function proposed in [47]. The function is rotated in the space from 0 degree to 180 degree and different PSO variants are applied to the rotated Ellipse to find an optimum point in the search space (see Fig. 12).

According to Fig. 12 the performances of SPSO2011, LcRiPSO, and RotPSO are not significantly (we used the Wilcoxon rank test to compare the results of the algorithm for each rotation angle with the next) changed by rotating the search space with different angles. Slight changes in the performance of these methods are because of the involvement of random components in their calculations. It is clear, however, that the performance of IPSO and GCPSO is changed by rotating the search space.

VI. Conclusion

Stability of particles, local convergence, and rotation sensitivity are important characteristics of optimization algorithms including PSO. In this paper, we investigated these properties for a variant of PSO called SPSO2011. We analyzed the stability of particles through an estimation method (called EVCB) that uses an exhaustive search with defined step size to find convergence boundaries. EVCB was used to estimate the convergence boundaries for a particular PSO variant (called IPSO) for which the convergence boundaries are known. It was observed that the estimated boundary is in good agreement with what was found theoretically. EVCB was also used to estimate the convergence boundaries for SPSO2011. Results showed that the convergence boundaries for SPSO2011 are different from that of IPSO. It was observed that the convergence boundaries for SPSO2011 is affected by the number of dimensions. Further experiments showed, however, that the convergence boundaries for SPSO2011 for d-dimensional problems with $d > 10$ remain almost the same. This enables practitioners to conduct parameter settings for a large enough number of dimensions (e.g., $d = 10$) to make sure their results are scalable. Also, our experiments showed that the convergence boundaries under the stagnation assumption are similar to that of without the stagnation assumption. The behavior of particles before convergence was also analyzed through Fourier analysis of the movement of particles versus iteration numbers. This analysis also showed differences between the behavior of particles in IPSO with those in SPSO2011 with same values of coefficients. These results assist practitioners to select appropriate set of coefficients according to the specifications of the problem at hand. In addition, it was proven that SPSO2011 does not guarantee to locate a local optimum in the search space. We proved that any algorithm (PSO variant) that follows the form of $\vec{x}_{t+1} = \omega(\vec{x}_t - \vec{x}_{t-1}) + n_{h_1,h_2}(\vec{p}_t, \vec{x}_t)$, where $\vec{p}_t = D(\vec{p}_{t-1}, \vec{x}_t)$, is locally convergent for all $\omega \in (0, 1), \vec{x}_0, \vec{p}_0 \in S$ (see Lemma 3 and Notation 3). This lemma is applicable to study the local convergence property of a large class of PSO variants. We modified SPSO2011 in a way that this condition is satisfied so that the modified SPSO2011 is locally convergent. Finally, we provided a proof for the transformation invariance property of SPSO2011. Indeed, it was proven that SPSO2011 is RST invariant. As potential future work, one can consider a theoretical analysis of the stability of particles in SPSO2011, keeping in mind the traditional analyses are not readily applicable to this algorithm. Also, formulating the regions where particles' behave differently might be of high value. Another important area for further research is to investigate the earliest first hitting time of the algorithm [48]. As the local convergence is a prerequisite for the first hitting time analysis, it would be interesting to conduct such analysis for the modified SPSO2011. In addition, investigations of transformation sensitivity and convergence of other PSO variants using the methodologies proposed in this paper represent other potential future directions.

References

[1] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. IEEE Int. Conf. Neural Netw.*, vol. 4. Perth, WA, Australia, 1995, pp. 1942–1948.

[2] A. Engelbrecht, *Fundamentals of Computational Swarm Intelligence*. Hoboken, NJ, USA: Wiley, 2005.

[3] R. Poli, "Analysis of the publications on the applications of particle swarm optimisation," *J. Artif. Evol. Appl.*, vol. 4, pp. 1–10, Jan. 2008, Art. ID 685175.

[4] PCS. (2006). *Standard Particle Swarm Optimisation*. [Online]. Available: http://www.particleswarm.info/index.html

[5] D. Bratton and J. Kennedy, "Defining a standard for particle swarm optimization," in *Proc. IEEE Swarm Intell. Symp.*, Honolulu, HI, USA, 2007, pp. 120–127.

[6] M. Clerc. (2011). *Standard Particle Swarm Optimisation From 2006 to 2011*. [Online]. Available: http://hal.archives-ouvertes.fr/hal-00764996

[7] M. Zambrano-Bigiarini, M. Clerc, and R. Rojas, "Standard particle swarm optimisation 2011 at CEC-2013: A baseline for future PSO improvements," in *Proc. IEEE Congr. Evol. Comput.*, Cancun, Mexico, 2013, pp. 2337–2344.

[8] M. R. Bonyadi and Z. Michalewicz, "SPSO 2011: Analysis of stability; local convergence; and rotation sensitivity," in *Proc. Conf. Genet. Evol. Comput.*, Vancouver, BC, Canada, 2014, pp. 9–16.

[9] M. Clerc and J. Kennedy, "The particle swarm—Explosion, stability, and convergence in a multidimensional complex space," *IEEE Trans. Evol. Comput.*, vol. 6, no. 1, pp. 58–73, Feb. 2002.

[10] I. C. Trelea, "The particle swarm optimization algorithm: Convergence analysis and parameter selection," *Inf. Process. Lett.*, vol. 85, no. 6, pp. 317–325, Mar. 2003.

[11] P. K. Lehre and C. Witt, *Finite First Hitting Time Versus Stochastic Convergence in Particle Swarm Optimisation*. New York, NY, USA: Springer, 2013, pp. 1–20.

[12] F. Van den Bergh and A. Engelbrecht, "A convergence proof for the particle swarm optimiser," *Fundamenta Informaticae*, vol. 105, no. 4, pp. 341–374, Dec. 2010.

[13] D. Wilke, S. Kok, and A. Groenwold, "Comparison of linear and classical velocity update rules in particle swarm optimization: Notes on scale and frame invariance," *Int. J. Numer. Methods Eng.*, vol. 70, no. 8, pp. 985–1008, May 2007.

[14] R. Eberhart and J. Kennedy, "A new optimizer using particle swarm theory," in *Proc. Int. Symp. Micro Mach. Human Sci.*, Nagoya, Japan, 1995, pp. 39–43.

[15] M. Clerc, *Particle Swarm Optimization*. London, U.K.: Wiley-ISTE, 2006.

[16] M. A. M. de Oca, T. Stützle, M. Birattari, and M. Dorigo, "Frankenstein's PSO: A composite particle swarm optimization algorithm," *IEEE Trans. Evol. Comput.*, vol. 13, no. 5, pp. 1120–1132, Oct. 2009.

[17] Y. Shi and R. Eberhart, "A modified particle swarm optimizer," in *Proc. IEEE World Congr. Comput. Intell.*, Anchorage, AK, USA, 1998, pp. 69–73.

[18] R. Poli, J. Kennedy, and T. Blackwell, "Particle swarm optimization: An overview," *Swarm Intell.*, vol. 1, no. 1, pp. 33–57, 2007.

[19] G. Marsaglia, "Choosing a point from the surface of a sphere," *Ann. Math. Stat.*, vol. 43, no. 2, pp. 645–646, Apr. 1972.

[20] J. Matyas, "Random optimization," *Autom. Remote Control*, vol. 26, no. 2, pp. 246–253, 1965.

[21] N. Baba, "Convergence of a random optimization method for constrained optimization problems," *J. Optim. Theory Appl.*, vol. 33, no. 4, pp. 451–461, Apr. 1981.

[22] F. J. Solis and R. J.-B. Wets, "Minimization by random search techniques," *Math. Oper. Res.*, vol. 6, no. 1, pp. 19–30, 1981.

[23] F. Van den Bergh and A. P. Engelbrecht, "A study of particle swarm optimization particle trajectories," *Inf. Sci.*, vol. 176, no. 8, pp. 937–971, Apr. 2006.

[24] M. Jiang, Y. P. Luo, and S. Y. Yang, "Stochastic convergence analysis and parameter selection of the standard particle swarm optimization algorithm," *Inf. Process. Lett.*, vol. 102, no. 1, pp. 8–16, Apr. 2007.

[25] R. Poli, "Mean and variance of the sampling distribution of particle swarm optimizers during stagnation," *IEEE Trans. Evol. Comput.*, vol. 13, no. 4, pp. 712–721, Aug. 2009.

[26] M. Clerc, "The swarm and the queen: Towards a deterministic and adaptive particle swarm optimization," in *Proc. IEEE Congr. Evol. Comput.*, vol. 3. Washington, DC, USA, 1999, pp. 1951–1957.

[27] C. W. Cleghorn and A. P. Engelbrecht, "A generalized theoretical deterministic particle swarm model," *Swarm Intell.*, vol. 8, no. 1, pp. 35–59, Mar. 2014.

[28] Q. Liu, "Order-2 stability analysis of particle swarm optimization," *Evol. Comput.*, vol. 23, no. 2, pp. 187–216, Apr. 2014.

[29] C. W. Cleghorn and A. P. Engelbrecht, "Particle swarm convergence: An empirical investigation," in *Proc. IEEE Congr. Evol. Comput.*, Beijing, China, 2014, pp. 2524–2530.

[30] S. Helwig, J. Branke, and S. Mostaghim, "Experimental analysis of bound handling techniques in particle swarm optimization," *IEEE Trans. Evol. Comput.*, vol. 17, no. 2, pp. 259–271, Apr. 2013.

[31] C. W. Cleghorn and A. P. Engelbrecht, "Particle swarm convergence: Standardized analysis and topological influence," in *Swarm Intelligence*. Cham, Switzerland: Springer, 2014, pp. 134–145.

[32] E. Campana, G. Fasano, and A. Pinto, "Dynamic analysis for the selection of parameters and initial population, in particle swarm optimization," *J. Glob. Optim.*, vol. 48, no. 3, pp. 347–397, Nov. 2010.

[33] M. R. Bonyadi and Z. Michalewicz, "A locally convergent rotationally invariant particle swarm optimization algorithm," *Swarm Intell.*, vol. 8, no. 3, pp. 159–198, Sep. 2014.

[34] M. A. M. de Oca, T. Stützle, K. Van den Enden, and M. Dorigo, "Incremental social learning in particle swarms," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 41, no. 2, pp. 368–384, Apr. 2011.

[35] J. García-Nieto and E. Alba, "Restart particle swarm optimization with velocity modulation: A scalability test," *Soft Comput.*, vol. 15, no. 11, pp. 2221–2232, Nov. 2011.

[36] W.-N. Chen *et al.*, "Particle swarm optimization with an aging leader and challengers," *IEEE Trans. Evol. Comput.*, vol. 17, no. 2, pp. 241–258, Apr. 2013.

[37] M. Schmitt and R. Wanka, "Particle swarm optimization almost surely finds local optima," in *Proc. Genet. Evol. Comput. Conf.*, Amsterdam, The Netherlands, 2013, pp. 1629–1636.

[38] M. R. Spiegel, *Theory and Problems of Vector Analysis (Schaum's Outline Series)*. New York, NY, USA: McGraw Hill, 1959.

[39] D. Wilke, "Analysis of the particle swarm optimization algorithm," M.S. thesis, Mech. Aeronaut. Eng., Univ. Pretoria, Pretoria, South Africa, 2005.

[40] M. R. Bonyadi, X. Li, and Z. Michalewicz, "A hybrid particle swarm with velocity mutation for constraint optimization problems," in *Proc. Genet. Evol. Comput. Conf.*, Amsterdam, The Netherlands, 2013, pp. 1–8.

[41] N. Hansen, R. Ros, N. Mauny, M. Schoenauer, and A. Auger, "Impacts of invariance in search: When CMA-ES and PSO face ill-conditioned and non-separable problems," *Appl. Soft Comput.*, vol. 11, no. 8, pp. 5755–5769, 2011.

[42] N. Hansen and A. Ostermeier, "Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation," in *Proc. IEEE Congr. Evol. Comput.*, Nagoya, Japan, 1996, pp. 312–317.

[43] R. Storn and K. Price, "Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces," *J. Glob. Optim.*, vol. 11, no. 4, pp. 341–359, Dec. 1997.

[44] R. Salomon, "Reevaluating genetic algorithm performance under coordinate rotation of benchmark functions—A survey of some theoretical and practical aspects of genetic algorithms," *Biosystems*, vol. 39, no. 3, pp. 263–278, 1996.

[45] A. Auger, N. Hansen, J. P. Zerpa, R. Ros, and M. Schoenauer, "Empirical comparisons of several derivative free optimization algorithms," in *Proc. 9th Nat. Symp. Calculat. Struct.*, France, vol. 1. 2009, pp. 481–486.

[46] M. R. Bonyadi, Z. Michalewicz, and X. Li, "An analysis of the velocity updating rule of the particle swarm optimization algorithm," *J. Heuristics*, vol. 20, no. 4, pp. 417–452, Apr. 2014.

[47] W. Spears, D. Green, and D. Spears, "Biases in particle swarm optimization," *Int. J. Swarm Intell. Res.*, vol. 1, no. 2, pp. 34–57, Apr. 2010.

[48] C. Witt, "Why standard particle swarm optimisers elude a theoretical runtime analysis," in *Proc. Found. Genet. Algorithms*, Orlando, FL, USA, 2009, pp. 13–20.

Mohammad Reza Bonyadi received the Ph.D. degree in mathematics and computer science from University of Adelaide, Adelaide, Australia, in 2015.

He is an Honorary Adjunct Lecturer with University of Adelaide as well as a Senior Scientist with Complexica Pty Ltd., Adelaide. His research interests include theoretical analysis of optimization algorithms in continuous space, multicomponent optimization problems, and applied optimization. He has published over 35 articles in international journals, conferences, and chapter of books.

Dr. Bonyadi also serves the evolutionary computation community as a reviewer in over 15 international journals and conferences.

Zbigniew Michalewicz received the Ph.D. degree from the Institute of Computer Science, Polish Academy of Sciences, Warsaw, Poland, in 1981 and the Doctor of Science (Habilitation) degree in computer science from Polish Academy of Science, in 1997.

He was an Emeritus Professor with the School of Computer Science, University of Adelaide, Adelaide, Australia, and the Chief Scientist with Complexica Pty Ltd., Adelaide. In 2002, he received the title of Professor from the President of Poland, Mr. A. Kwasniewski. He is also a Professor with the Institute of Computer Science, Polish Academy of Sciences, the Polish–Japanese Institute of Information Technology, Warsaw, Poland, and an Honorary Professor with State Key Laboratory of Software Engineering, Wuhan University, Wuhan, China. In 2006, he was appointed a Business Ambassador for the state of South Australia. He has published several books, including a monograph entitled *Genetic Algorithms + Data Structures = Evolution Programs* (three editions and several translations), and over 250 technical papers in journals and conference proceedings.

Prof. Michalewicz was a recipient of the Order of the Rebirth of Polish Polonia Restituta (the second highest Polish state decoration for a civilian, after the Order of the White Eagle) in 2013 from the President of Poland, Mr. B. Komorowski, awarded for outstanding achievements in the field of education, science, sports, culture, arts, economy, national defense, social activities, civil service, and the development of good relations with other countries. He is a fellow of the Australian Computer Society.