Constricted Particle Swarm Optimization based Algorithm for Global Optimization

Gonzalo Nápoles, Isel Grau, and Rafael Bello

***Abstract*—Particle Swarm Optimization (PSO) is a bioinspired meta-heuristic for solving complex global optimization problems. In standard PSO, the particle swarm frequently gets attracted by suboptimal solutions, causing premature convergence of the algorithm and swarm stagnation. Once the particles have been attracted to a local optimum, they continue the search process within a minuscule region of the solution space, and escaping from this local optimum may be difficult. This paper presents a modified variant of constricted PSO that uses random samples in variable neighborhoods for dispersing the swarm whenever a premature convergence (or stagnation) state is detected, offering an escaping alternative from local optima. The performance of the proposed algorithm is discussed and experimental results show its ability to approximate to the global minimum in each of the nine well-known studied benchmark functions.**

***Index Terms*—Particle Swarm Optimization, Local optima, Global Optimization, Premature Convergence, Random Samples, Variable Neighborhoods.**

I. Introduction

Particle Swarm Optimization is a bioinspired search technique that simulates the social behavior observed in groups or swarms of biological individuals [1], [2]. It is based on the principle that intelligence does not lie in individuals but in the collective, allowing for the solution of complex optimization problems from a distributed point of view, without centralized control in a specific individual. Each organism (particle) adjusts its position by using a combination of an attraction to the best solution that they individually have found, and an attraction to the best solution that any particle has found [3], imitating those who have a better performance. Thus, the particle swarm overflies the search space detecting promising regions.

Although the PSO meta-heuristic has proved to be efficient for solving real-value optimization problems, the particle swarm is often attracted to stable points that are not necessarily global optima [4], [5]. This behavior causes premature convergence of the algorithm, in which the particles are grouped about suboptimal solutions with little chance of escaping from this situation. Once the particles have converged prematurely, they continue converging within extremely close proximity of one another so that the global best and all personal bests are within one minuscule region of the search space [6], limiting the algorithm exploration.

Several approaches have been proposed in the literature for addressing this undesirable situation. For example, in Attraction-Repulsion based PSO (ATREPSO) [7] the swarm switches between the attraction phase, repulsion phase, and in between phase which consist of a combination of attraction and repulsion to the optimal position. Another approach called Quadratic Interpolation based PSO (QIPSO) [8] uses a quadratic crossover operator and the swarm diversity as a measure to guide the population for finding a better solution in the search space. Using a certain threshold of diversity, Gaussian Mutation PSO (GMPSO) [9] activates a mutation operator with the hope to increase the diversity of the swarm. Lastly, in [10] the authors propose a new hybrid variant called HPSO-SA that combines PSO and Simulating Annealing (SA) to avoid premature convergence using the strong local-search ability of SA.

Although these approaches generally outperform classical PSO, there is a need to incorporate alternative procedures for enhancing the PSO search process. In this work we present a modified constricted PSO called Particle Swarm Optimization with Random Sampling in Variable Neighborhoods (PSO-RSVN) which detects and treats the premature convergence state, achieving promising results compared to several approaches reported in the literature.

The rest of the paper is organized as follows: in next Section II a theoretical background of standard PSO is described. In Section III we introduce the proposed PSO-RSVN algorithm. Section IV gives the experimental settings, the numerical benchmark problems used for comparison and the result discussion. Finally, conclusions and further research aspects are given in Section V.

II. Particle Swarm Optimization

The PSO technique involves a set of agents or particles known as swarm which "flies" through the solution space trying to locate promising regions. The particles are interpreted as possible solutions for the optimization problem and are represented as points in n-dimensional search space. In the case of standard PSO, each particle (X_i) has its own velocity (V_i) bounded by a maximum value (V_{max}), a memory

Manuscript received June 20, 2012. Manuscript accepted for publication July 24, 2012.

Gonzalo Nápoles and Rafael Bello are with the Laboratory of Artificial Intelligence, Universidad Central "Marta Abreu" de Las Villas (UCLV), Santa Clara, Cuba (e-mail: {gnapoles, rbellop}@ uclv.edu.cu).

Isel Grau is with the Laboratory of Bioinformatics, Universidad Central "Marta Abreu" de Las Villas (UCLV), Santa Clara, Cuba (e-mail: igrau@ uclv.edu.cu).

of the best position it has obtained (P_i) and knowledge of the best solution found in its neighborhood (G). In the search process the particles adjust their positions according to the following equations (1) and (2):

$$V_i^{(k+1)} = V_i^{(k)} + c_1 r_1 \left(P_i - X_i^{(k)}\right) + c_2 r_2 \left(G - X_i^{(k)}\right) \quad (1)$$

$$X_i^{(k+1)} = X_i^{(k)} + V_i^{(k+1)} \quad (2)$$

where k indexes the current generation, c_1 and c_2 are positive constants, r_1 and r_2 are random numbers with uniform distribution on the interval $[0, 1]$.

A commonly used parameter that changes the original PSO is the *constriction coefficient* (χ), which was introduced by Clerc *et al.* [11] to guarantee the algorithm convergence, avoiding the explosion of the particle swarm (i.e. the state where the particles velocities and positional coordinates careen toward infinity). It can be expressed in terms of c_1 and c_2 as shown in (3):

$$\chi = \frac{2}{\left|2 - \varphi - \sqrt{\varphi^2 - 4\varphi}\right|} \text{ and } \varphi = c_1 + c_2\,,\ \varphi > 4 \quad (3)$$

Another parameter that modifies the standard PSO is the inertia weight (ϖ) added by Shi *et al.* [12]. The incorporation of this parameter guarantees the balance between the capacities of local and global search; a higher weight value ($\varpi > 1$) will facilitate the exploration, while a low weight ($\varpi < 1$) facilitates the exploitation. The wrong choice of this parameter value will affect the algorithm convergence speed, so it is recommended to adjust it dynamically as shown in the following equation (4):

$$\varpi_k = \varpi_{max} - \frac{\varpi_{max} - \varpi_{min}}{F_{max}} F_k \quad (4)$$

where, ϖ_{min} and ϖ_{max}match the end points of the interval on which the *k*-th inertia weight is defined, F_k denotes the number of evaluations at the *k*-cycle, whereas F_{max} corresponds to the maximal number of evaluations allowed. So, both factors are applied to the equation (1) as follow:

$$V_i^{(k+1)} = \chi\left(\varpi_{k+1} V_i^{(k)} + \delta_1\left(P_i - X_i^{(k)}\right) + \delta_2\left(G - X_i^{(k)}\right)\right) \quad (5)$$

Extensive experiments carried out in [3] showed that constricted PSO returns improved performance over the original PSO; however, it provides no mechanism to detect and treat premature convergence (or stagnation state) of the particle swarm, which could adversely affect the algorithm effectiveness.

III. PSO-RSVN Algorithm

In this section we introduce a modification for constricted PSO algorithm defined by equations (2) and (5). First, several mechanisms to detect the premature convergence state along the progress of the algorithm are discussed. Next, a new procedure of swarm reorganization based on randomly selected particles from the neighborhoods of the global best particle is presented.

A. Detection of the premature convergence state

The first step to enhance the performance of constricted PSO algorithm is to detect the premature convergence state. When PSO falls into a local optimum all individuals are grouped around this solution, which is why diversity is lost among the swarm particles, making more difficult to find better solutions in the algorithm progress. In [13] several ways to detect this state are discussed:

i. **Cluster analysis:** a percentage of the particles are at a certain Euclidean distance of the best global particle.
ii. **Objective function without progress:** the objective function does not suffer significantly improvement in several iterations of the generational cycle. This criterion also may be used for detecting the stagnation state.
iii. **Maximum radius of the swarm:** the particle with more Euclidean distance respect to the global best particle found, have a distance less than a pre-set threshold. This criterion it is formally defined as:

$$\rho(k) = \frac{\max_{1 \le i \le |\Omega|} \left\| X_i^{(k)} - G \right\|}{|\sigma_{max} - \sigma_{min}|} \quad (6)$$

where $\|.\|$ denotes the Euclidean norm on $\mathbb{R}^n$, while σ_{min} and σ_{max} are the end points on which each dimension of the particle X_i is defined (assuming same domain for all dimensions), whereas Ω represents the particle swarm. In this way the threshold is normalized for each generation *k*. Particularly, we use ii) to detect the stagnation state and iii) for identify premature convergence.

B. Treatment of the Premature Convergence State

Once premature convergence signals are detected it is necessary to take some action to allow the algorithm to escape from this state, for example:

i. Moving the position of the globally best particle found.
ii. Reorganizing the particle swarm (applying genetic operators in order to diversify the population, re-initializing the swarm, etc.).

Although both strategies have reported good results in solving global optimization problems, they have some drawbacks. In the first case is not trivial to find a better particle, and even when once found, the population is poorly diversified.

In the second case, diversification increases the chances of escaping the local optimum but if the swarm is not properly reorganized, the particles may converge to the same solution or indiscriminately move away from the promising areas that were found. In order to mitigate these problems in this paper we assume a hybrid approach consisting in diversifying the population while trying to move the position of the best global particle found (G) in the search process.

The Variable Neighborhood Search (VNS) [14] is a simple and effective meta-heuristic for combinatorial problems and

global optimization which is based on the systematic change of the neighborhood in the search process. Inspired by this idea, we present a procedure called Random Sampling in Variable Neighborhoods (RSVN) which aims to disperse the swarm when the premature convergence or stagnation state is detected. The main idea of this procedure is to restructure the particle swarm from the selection of random samples uniformly distributed in several neighborhoods generated around the n-dimensional point G. Equations (7), (8) and (9) formalize the way to generate the set of samples in each neighborhood:

$$\lambda_{jd}^{-}=\begin{cases}\tau_{jd}, & \tau_{jd}\geq\sigma_{min}\\ \sigma_{min}, & \tau_{jd}<\sigma_{min}\end{cases} \text{ and } \tau_{jd}=\mathrm{G}_d-\xi_j|\sigma_{max}-\sigma_{min}| \quad (7)$$

$$\lambda_{jd}^{+}=\begin{cases}\varsigma_{jd}, & \varsigma_{jd}\leq\sigma_{max}\\ \sigma_{max}, & \varsigma_{jd}>\sigma_{max}\end{cases} \text{ and } \varsigma_{jd}=\mathrm{G}_d+\xi_j|\sigma_{max}-\sigma_{min}| \quad (8)$$

$$\mathrm{X}_t\in\Psi_j\ |\mathrm{X}_{td}\sim\mathrm{U}\left(\lambda_{jd}^{-},\lambda_{jd}^{+}\right),\ t=1,\ldots,\left|\Psi_j\right|;\ j=1,\ldots,\mathrm{M} \quad (9)$$

where d indexes the particle dimension, M is a user-specified integer parameter that denotes the number of neighborhoods, whereas $\xi_j\in(0,1]$is a fractional value called neighborhood factor that denotes the j-th neighborhood proportion to the size of the search space; and it is calculated as: $\xi_j=j/\mathrm{M}$. Lastly, $\Psi_j$represents the *j*-th set of uniformly distributed samples in the domain that defines the interval $\left[\lambda_{jd}^{-},\lambda_{jd}^{+}\right]$.

After collecting the samples, a selection process of the particles takes place. These agents will form the new swarm as shown below:

$$\Omega^k=\Phi_1\cup\Phi_2\cup\ldots\cup\Phi_m=\bigcup_{j=1}^{m}\Phi_j\ |\ \Phi_j\subseteq\Psi_j,\forall j \quad (10)$$

where $\Phi_j$is a subset of good enough particles compared to all samples Ψ_j using an elitist criterion. In this procedure each particle X_i is a candidate to replace the best global particle, which complements the swarm dispersion process. Next, a pseudocode summarizes the main ideas and constriction parameters of the PSO-RSVN algorithm.

An important aspect to be discussed is the selection of each subset Φ_j; due to the high computational cost that generally involve the evaluation of the objective function the elitist criterion may be replaced by a heuristic criterion, for example: "select the particles with greater Euclidean distance respect to the global best particle". However, for high-dimensional problems the extra-computational cost needed to compute the Euclidean distance could become significant. For solving this inconvenient, a simple but effective alternative might be: for each set of samples make $\left|\Phi_j\right|=\left|\Psi_j\right|=|\Omega|/\mathrm{M}$, i.e., the selection process is omitted and consequently reduced the extra-computational cost required for computing the swarm reorganization process. In fact, this last criterion is used in all experiments carried out in next Section IV.

Pseudocode of PSO with Random Sampling in Variable Neighborhoods

```
Generate the swarm vector and the velocity vector randomly
Select the best global particle of the swarm (G)
Initialize ω_max = 1.4, ω_min = 0.4, c_1 = 2.05, c_2 = 2.05
While(the maximal number of evaluations is not met)
  Calculate ϖ_k dynamically according to expression (4)
  ForEach X_i^(k) εΩ
    Calculate V_i^(k+1) according to expression(5)
    Adjust the position of X_i^(k+1) according to expression (2)
    Evaluate the new particle X_i^(k+1)
    IF(X_i^(k+1) is the best record for the i-th particle)
      Update the best record for the i-th particle with X_i^(k+1)
      IF(X_i^(k+1) is a better particle than G)
        Update G with X_i^(k+1) as the best global particle
      endIF
    endIF
  endForEach
  IF (premature convergence or stagnation state is detected)
    Disperse the swarm Ω according to the expressions (7-10)
    Reset velocity vector using a random sequence
    Update the vector P and the best particle G
    Reset the inertia weight
  endIF
endWhile
```

IV. Performance Study

In this section two implementations of the PSO-RSVN algorithm are evaluated; the first implementation called PSO-RSVN-α detects the premature convergence state, comparing the maximum radius of the swarm (according to equation (6)) with a pre-set threshold α. The second variant called PSO-RSVN-p is useful for identify a premature convergence state as well as a possible stagnation state, and involve a parameter (p) that represents the allowed maximal number of evaluations without progress. It must be mentioned that PSO-RSVN-p induce less extra-computational cost, since the estimation of the maximum radius of the swarm is not required.

Table I describes nine well-known benchmark functions taken from [15], which are used to compare the performance of the proposal with several approaches reported in the literature. These functions are minimization problems characterized by multiple local optima, especially when the complexity of the function increases, that is, when the dimensionality of the search space increases. The first seven problems are scalable and includes unimodal, multimodal and noisy functions, whereas the last two problems are highly multimodal in nature. Therefore, the use of these functions helps in deciding the credibility of an optimization algorithm.

The experimental results discussed in this section are addressed in two sub-sections which study the performance of the PSO-RSVN algorithm for 20-dimensional and 30-dimensional solution search spaces.

A. PSO-RSVN behavior in 20-dimensional search spaces

This section compares both PSO-RSVN variants against five approaches evaluated and discussed in [16], [10]: PSO,

TABLE I
STANDARD BENCHMARK FUNCTIONS USED IN THIS WORK. LAST COLUMN (F_{MIN}) REFERS TO GLOBAL MINIMUM VALUE EXISTING IN THE DOMAIN DEFINED BY BOUNDARIES.

Mathematical Formulation	Search Range	F_{min}
$f_1(\vec{x}) = \sum_{i=1}^{n} x_i^2$	[-100.0,100.0]	0.0000000
$f_2(\vec{x}) = \sum_{i=1}^{n} (x_i^2 - 10\cos(2\pi x_i) + 10)$	[-5.120,5.120]	0.0000000
$f_3(\vec{x}) = \left(\frac{1}{4000}\right)\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	[-600.0,600.0]	0.0000000
$f_4(\vec{x}) = \sum_{i=1}^{n-1} (100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2)$	[-2.048,2.048]	0.0000000
$f_5(\vec{x}) = \sum_{i=1}^{n} i x_i^4 + rand[0,1]$	[-1.280,1.280]	0.0000000
$f_6(\vec{x}) = -\sum_{i=1}^{n} (x_i \sin(\sqrt{\lvert x_i \rvert}))$	[-500.0,500.0]	-420.968*n*
$f_7(\vec{x}) = 20 + e - 20e^{-0.2\left(\left(\frac{1}{n}\right)\sum_{i=1}^{n} x_i^2\right)^{\frac{1}{2}}} - e^{(1/n)\sum_{i=1}^{n}\cos(2\pi x_i)}$	[-32.0,32.0]	0.0000000
$f_8(\vec{x}) = \sum_{i=1}^{5} i\cos((i+1)x_1 + i) \sum_{i=1}^{5} i\cos((i+1)x_2 + i)$	[-10.0,10.0]	-186.7309
$f_9(\vec{x}) = (x_2 + x_1^2 - 11)^2 + (x_1 + x_2^2 - 7)^2 + x_1$	[-5.00,5.00]	-3.783961

[a] F_8 and F_9 are defined in $\mathbb{R}^2$ space.

TABLE II
AVERAGE ERROR OBTAINED IN THE OPTIMIZATION PROCESS. THE BEST PERFORMING ALGORITHM FOR EACH FUNCTION IS EMPHASIZED IN BOLDFACE.

ID	PSO	QIPSO	ATREPSO	GMPSO	HPSO-SA	RSVN-α	RSVN-p
F_1	1.167E-45	**0.0000000**	4.000E-17	7.263E-17	5.365E-32	**0.0000000**	**0.0000000**
F_2	22.339158	11.946888	19.425979	20.079185	**0.0000000**	**0.0000000**	**0.0000000**
F_3	0.0316460	0.0115800	0.0251580	0.0244620	3.322E-20	**0.0000000**	**0.0000000**
F_4	22.191725	8.9390110	19.490820	14.159547	0.2270481	2.635E-16	**7.312E-25**
F_5	8.6816020	0.4511090	8.0466170	7.1606750	0.0020199	**4.372E-06**	1.386E-05
F_6	2240.8010	2063.7740	2235.6830	2371.6900	**39.700000**	43.651000	1280.4350
F_7	3.483E-18	**2.461E-24**	0.0184930	1.474E-18	7.435E-16	4.440E-16	4.440E-16
F_8	1.420E-05	**0.0000000**	1.420E-05	1.530E-05	8.670E-14	**0.0000000**	**0.0000000**
F_9	0.4524730	**0.0000000**	0.0325030	0.3237280	1.000E-06	1.000E-06	**0.0000000**

QIPSO, ATREPSO, GMPSO, and HPSO-SA. In each simulation we used 30 particles and 300.000 objective function evaluations in a 20-dimensional search space. In addition, five variable neighborhoods (M=5) are used. For the PSO-RSVN-α implementation, a threshold for the maximum radius of the swarm α= 1.0E-5 is adopted, whereas for PSO-RSVN-p the allowed number of evaluations without progress is set to 200. Table II summarizes the average error obtained respect to the global optimum for each algorithm from 30 independent trials. In all tables, PSO-RSVN-αis abbreviated as RSVN-α and PSO-RSVN-p as RSVN-p.

Analyzing the results shown in Table II we observed that for the functions F_1, F_2, F_3 and F_8 both PSO-RSVN variants always finds the global optimum satisfactorily. For Rosenbrock (F_4) function PSO-RSVN-p outperforms all examined algorithms; in this function the stagnation state is frequently presented due to the search space properties. For the noisy function F_5PSO-RSVN-α computes the best performance, whereas for Shwefel (F_6), HPSO-SA locates better solutions. PSO-RSVN-α is slightly the best approach for minimizing the Himmelblau (F_9) function. Finally, QIPSO has the best results reported for Ackley (F_7), followed by both PSO-RSVN algorithms.

In a deeper statistical study of the algorithms performance we used several test for exploring significant differences among them. Depending on the concrete type of data employed, statistical procedures are grouped in two classes: parametric and nonparametric [18]. Parametric tests have been often used in the analysis of experiments in computational intelligence. Unfortunately, they are based on assumptions (independence, normally, homoscedasticity) which are most probably violated when analyzing the performance of stochastic algorithms based on computational intelligence [19], [20]. To overcome this problem, the researchers frequently use nonparametric statistical procedures when these previous assumptions cannot be satisfied.

First, we compute the Friedman test (Friedman two-way analysis of variances by ranks) [21], [22]. This test is a multiple comparisons procedure for detecting significant differences between the behaviors of two or more algorithms; i.e. it can be used for detecting whether at least two of the samples represent populations with different median values or not, in a set of *n* samples ($n \geq 2$). Table III shows the mean rank and the p-value associated with this test. Using a significance level of 0.05, corresponding to the 95% confidence interval, the Friedman test suggest rejecting the null hypothesis (p-value < 0.05), thus, there exist highly significant differences between at least two methods across benchmark. Also can be observed that PSO-RSVN-p and PSO-RSVN-α are the best ranked; however this information cannot be used to conclude that our proposals are involved on this differences.

TABLE III
MEAN RANK ACHIEVED BY THE FRIEDMAN TEST

Evaluated Algorithms	Mean Rank[a]
RSVN-p	2.39
RSVN-α	2.39
HPSO-SA	2.89
QIPSO	3.44
GMPSO	5.39
ATREPSO	5.50
PSO	6.00

[a] Monte Carlo signification (p-value) = 0.00

The main drawback of the Friedman's tests is that they only can detect significant differences over the whole multiple comparisons, being unable to establish proper comparisons between some of the algorithms considered [23]. For this reason we also compute the Wilcoxon signed ranks test [24]; it is used for answering a simple question: do two samples represents two different populations? Thus, Wilcoxon is a pairwise procedure that aims to detect significant differences between two sample means, that is, the behavior of two algorithms.

Table IV shows the p-values associated with each pairwise comparison. Then some important conclusions came out:

i. Using a significance level of 0.05, corresponding to the 95% confidence interval, the Wilcoxon test suggest to reject the null hypothesis (p-value < 0.05) for the following pairwise comparisons: RSVN-p vs. GMPSO, RSVN-p vs. ATREPSO, RSVN-p vs. PSO, RSVN-α vs. PSO, RSVN-α vs. GMPSO and RSVN-α vs. ATREPSO; thus we can conclude that there exist highly significant differences between them.
ii. Using a significance level of 0.1, corresponding to the 90% confidence interval, the Wilcoxon test suggest to reject the null hypothesis (p-value < 0.1) for the following pairwise comparisons: RSVN-p vs. QIPSO and RSVN-α vs. QIPSO; i.e. there exist fairly significant differences between them.
iii. For the pairwise comparisons that involve the following methods: RSVN-α, RSVN-p and HPSO-SA, there not exist perceptible differences among them. These results confirm the improvement of the proposed procedures.

TABLE IV
WILCOXON SIGNED RANKS TEST RESULTS

Pairwise Comparison	p-value[a]
RSVN-p vs. RSVN-α	0.621
RSVN-p vs. HPSO-SA	0.195
RSVN-p vs. GMPSO	0.013
RSVN-p vs. ATREPSO	0.004
RSVN-p vs. QIPSO	0.066
RSVN-p vs. PSO	0.013
RSVN-α vs. HPSO-SA	0.292
RSVN-α vs. GMPSO	0.013
RSVN-α vs. ATREPSO	0.004
RSVN-α vs. QIPSO	0.081
RSVN-α vs. PSO	0.013

[a] Monte Carlo signification

B. PSO-RSVN behavior in 30-dimensional search spaces

One of the most important variations to PSO is the introduction of the local model or local topology (*lbest*). In this model, each particle can only communicate with a subset of particles, limiting the overall exchange of information. In contrast to the global model or the global topology (*gbest*), the local model converges more slowly but is less prone to being trapped in suboptimal solutions. In fact, several authors suggest using the local topology to optimize complex multimodal functions, and the global topology to optimize unimodal functions [3].

Four different approaches have been evaluated in [6] that include considerations about the topology of the particle swarm in 30-dimensional spaces: *lbest* PSO with a ring topology, *gbest* PSO, Regrouping PSO (RegPSO) [6] and Opposition based PSO (OPSO) [17]. These simulations allow studying the stability of the PSO-RSVN algorithm when increasing dimensionality of the solution search space. Table V summarizes the average error obtained for each algorithm from 50 trials, 20 particles as the swarm size and 800.000 objective function evaluations. Moreover, five variable neighborhoods (M=5) are used. For the PSO-RSVN-α implementation, a tolerance for the maximum radius of the swarm α = 1.0E-5 is adopted, whereas for PSO-RSVN-p the allowed number of evaluations without progress is set to 500.

From the numerical results shown in Table V a conclusion came out: both variants of PSO-RSVN outperform other approaches in all cases. PSO-RSVN-α and PSO-RSVN-p always find the global optimum satisfactorily for Sphere (F_1),

TABLE V
AVERAGE ERROR OBTAINED IN THE OPTIMIZATION PROCESS. THE BEST PERFORMING ALGORITHM FOR EACH FUNCTION IS EMPHASIZED IN BOLDFACE.

ID	*gbest* PSO	*lbest* PSO	OPSO	RegPSO	RSVN-α	RSVN-p
F_1	2.470E-323	5.513E-160	9.881E-324	9.2696E-15	**0.00000000**	**0.00000000**
F_2	71.6368600	54.2849000	66.1646300	2.6824E-11	**0.00000000**	**0.00000000**
F_3	0.05500800	0.00939970	0.02574900	0.01386100	**0.00000000**	**0.00000000**
F_4	2.06915000	3.25523000	1.86410000	0.00393510	7.9800E-04	**2.4299E-18**
F_5	0.00394380	0.01325000	0.00101660	0.00064366	**2.1731E-06**	1.4155E-05
F_7	3.91150000	0.07546900	2.67240000	4.6915E-07	**4.4408E-16**	**4.4408E-16**

Rastringin (F_2), Griewank (F_3) and Ackley (F_7) functions. For Rosenbrock (F_4) PSO-RSVN-p has the best performance, whereas for Quartic (F_5) the other proposal achieves the best approximations. These results reveal that both PSO-RSVN are quite consistent across benchmarks when the dimensionality of the search space increases.

Figure 1 illustrates, as an example, the behavior of the swarm diversity in the optimization process of the Rastringin function, for proposed PSO-RSVN-αalgorithm and constricted PSO. In this simulation we use the maximum radius of the swarm for measuring the swarm diversity. So, in the generation number 4900 both methods prematurely converge to a local optimum, i.e. the whole population is grouped in a minuscule region of the search space. This situation degrades the PSO search capabilities. However, the swarm diversity introduced by PSO-RSVN-α ensures the exploration of new areas of the solution space, increasing the possibility of escape from suboptimal solutions.

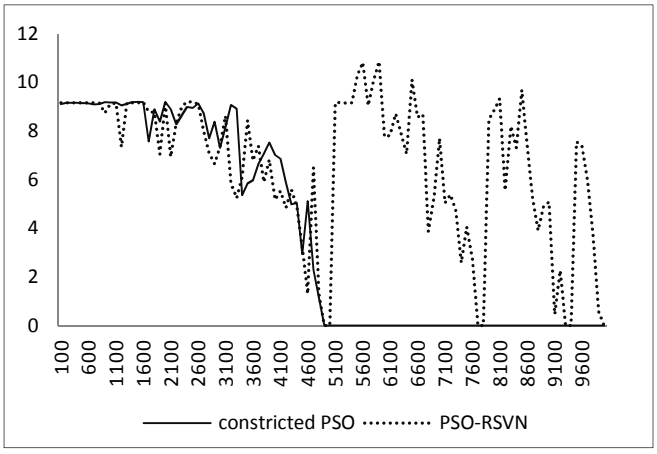

Fig. 1. Behavior of the swarm diversity for PSO-RSVN and constricted PSO, during the optimization process. The horizontal axis denotes the number of objective functions evaluated and the vertical axis refers to the maximum radius of the swarm.

In general, PSO-RSVN introduces a new parameter to be estimated by the user: the number of neighborhoods M; it is an integer value used to organize the sampling process by dividing the search space in several partitions. Recommended values for this parameter could be M = 5 or M =10, although other values are allowed. For the first implementation a normalized threshold α for the maximum radius of the swarm should be specified. The wrong selection of this parameter may be relevant to the algorithm performance: a higher value of α will affect the PSO-RSVN-α exploitation capability, due to a false premature convergence state could be induced, whereas a lower value could never detect an existing premature convergence state. Empirical experiments show that values from 1.0E-2 to 1.0E-8 are a good choice. Finally, for the PSO-RSVN-p implementation, a parameter for controlling the allowed number of evaluations without progress is required. This parameter is easy to set and will depend on the maximal number of the objective function evaluations.

Although PSO-RSVN generally provides superior results regarding examined approaches for well-known benchmark functions described in Table I, future work will study the algorithm performance across other well-known benchmark functions, for example, shifted or badly scaled functions.

V. CONCLUSIONS

In this paper was proposed a modified variant of the constricted PSO called PSO-RSVN, for enhancing the search capability of this algorithm, when solving complex optimization problems. Two PSO-RSVN implementations are presented: the first one is capable to detect the premature convergence state, and the second one is able to detect the premature convergence as well as the stagnation state. Both variants treat these undesirable states by reorganizing the particle swarm, which is, conducting a random sampling in several neighborhoods, emphasizing the neighborhood of the best particle found so far. The swarm diversity introduced by the proposed dispersion mechanism ensures the exploration of new areas of the solution space, increasing the possibility of escape from suboptimal solutions.

It was evaluated the algorithm performance in comparison with other variants of PSO reported in the literature, by using nine well-known benchmark functions for 20-dimensional search spaces. In addition it was verified the stability of the algorithm upon increasing the dimensionality of the search space, by using 30-dimensional spaces. In both experiments the new algorithm (PSO-RSVN) provides superior results in most cases. In fact, due to its simplicity, elitist properties and low computational cost, the RSVN procedure could be adapted and successfully integrated into other evolutionary paradigms. As mentioned, future work will be focused on extending the study of the algorithm performance across other well-known benchmark functions.

ACKNOWLEDGMENT

We would like to thank Prof. Dr. Ricardo Grau Abalo, from the Centre of Studies on Informatics, UCLV, for fruitful discussions and statistical advice.

REFERENCES

[1] J. Kennedy and R. Eberhart, "Particle Swarm Optimization," in *Proc. of the 1995 IEEE International Conference on Neural Networks*, Australia, 1995, pp. 1942–1948.

[2] R. Eberhart and J. Kennedy, "A New Optimizer using Particle Swarm Theory," in *Proc. of the Sixth International Symposium on Micromachine and Human Science*, Japan, 1995, pp. 39–43.

[3] D. Bratton and J. Kennedy, "Defining a Standard for Particle Swarm Optimization," in *Proc. of the 2007 IEEE Swarm Intelligence Symposium*, Honolulu, 2007, pp. 120–127.

[4] V.J. Huang, et al, "Comprehensive learning particle swarm optimizer for solving multiobjective optimization problems," *Int. J. of Intelligent Systems*, vol. 21, pp. 209–226, 2006.

[5] J. Kennedy and C.E. Russell, *Swarm Intelligence*, Morgan Kaufmann Publishers, 2001.

[6] G. Evers and M.B. Ghalia, "Regrouping Particle Swarm Optimization: A new Global Optimization Algorithm with Improved Performance Consistency Across Benchmarks," in *IEEE International Conference on Systems, Man and Cybernetics*, San Antonio, 2009, pp. 3901–3908.

[7] J. Riget and J.S. Vesterstrom, "A diversity-guide particle swarm optimizer – the arPSO", Tech. Rep., 2002.

[8] M. Pant and T. Radha, "A new Particle Swarm Optimization with quadratic crossover," in *International Conference on Advanced Computing and Communications*, India, 2007, pp. 81–86.

[9] M. Pant, T. Radha and V.P. Singh, "A new diversity based Particle Swarm Optimization using Gaussian Mutation," *Int. J. of Mathematical Modeling, Simulation and Applications*, vol. 1, pp. 47–60, 2007.

[10] L. Idoumghar, M. Melkemi, R. Schott and M.I. Aouad, "Hybrid PSO-SA Type Algorithms for Multimodal Function Optimization and Reducing Energy Consumption in Embedded Systems," *Applied Computational Intelligence and Soft Computing*, 12 pages, 2011.

[11] M. Clerc and J. Kennedy, "The particle swarm - explosion, stability, and convergence in a multidimensional complex space," *IEEE Trans. Evolutionary Computation*, vol. 6, pp. 58–73, 2002.

[12] Y. Shi and R. Eberhart, "A Modified Particle Swarm Optimizer," in *Proc. of the IEEE Congress on Evolutionary Computation*, 1998, pp. 69–73.

[13] F. Van den Bergh, "An Analysis of Particle Swarm Optimizers," PhD thesis, Univ. of Pretoria, South Africa, 2002.

[14] P. Hansen and N. Mladenović, "Variable neighborhood search: Principles and applications," *European Journal of Operations Research*, pp. 449–467, 2001.

[15] P.N. Suganthan, et al., "Problem definition and evaluation criteria for the CEC 2005 special session on real-parameter optimization," Tech. Rep. 2005005, 2005.

[16] M. Pant, R. Thangaraj and A. Abraham, "Particle swarm based metaheuristic for function optimization and engineering applications," in *Proc. of the 7th Computer Information Systems and Industrial Management Applications*, Washington, 2008, pp. 84–90.

[17] H. Wang, et al., "Opposition-based Particle Swarm Algorithm with Cauchy mutation," in *Proc. of the IEEE Congress on Evolutionary Computation*, 2007, pp. 4750–4756.

[18] J. Higgins, *Introduction to Modern Nonparametric Statistics*, Duxbury Press, 2003.

[19] S. García, A. Fernández, J. Luengo and F. Herrera, "A study of statistical techniques and performance measures for genetics-based machine learning: Accuracy and interpretability," *Soft Computing*, vol. 13, no. 10, pp. 959–977, 2009.

[20] S. García, D. Molina, M. Lozano and F. Herrera, "A study on the use of nonparametric tests for analyzing the evolutionary algorithms' behavior: A case study on the CEC'2005 special session on real parameter optimization," *J. Heuristics*, vol. 15, pp. 617–644, 2009.

[21] M. Friedman, "The use of ranks to avoid the assumption of normality implicit in the analysis of variance," *Journal of the American Statistical Association*, vol. 32, pp. 674–701, 1937.

[22] M. Friedman, "A comparison of alternative tests of significance for the problem of m rankings," *Annals of Mathematical Statistics*, vol. 11, pp. 86–92, 1940.

[23] J. Derrac, S. Garcia, D. Molina and F. Herrera, "A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms," *Swarm and Evolutionary Computation*, vol. 1, pp. 3–18, 2011.

[24] J.D. Gibbons and S. Chakraborti, "Nonparametric Statistical Inference," 5th ed., Chapman & Hall, 2010.