Unified Particle Swarm Optimization in Dynamic Environments

K.E. Parsopoulos and M.N. Vrahatis

Computational Intelligence Laboratory (CI Lab), Department of Mathematics,
University of Patras, GR–26110 Patras, Greece
{kostasp, vrahatis}@math.upatras.gr
University of Patras Artificial Intelligence Research Center (UPAIRC),
University of Patras, GR–26110 Patras, Greece

Abstract. A first investigation of the recently proposed Unified Particle Swarm Optimization algorithm on dynamic environments is provided and discussed on widely used test problems. Results are very promising compared to the corresponding results of the standard Particle Swarm Optimization algorithm, indicating the superiority of the new scheme.

1 Introduction

Particle Swarm Optimization (PSO) is a stochastic optimization algorithm that belongs to the category of *swarm intelligence* methods [1,2]. PSO has attained increasing popularity due to its ability to solve efficiently and effectively a plethora of problems in diverse scientific fields [3, 4]. Most of these problems involve the minimization of a static objective function, i.e., the main goal is the computation of a global minimizer that does not change.

Dynamic optimization problems, where the global minimizer moves in the search space, arise very often in engineering applications. In contrast to the static optimization case, the main goal in dynamic problems is to track the orbit of the minimizer [5, 6, 7]. Many algorithms that address efficiently static problems, fail when applied to dynamic problems due to their inability to adapt and respond to changes in the environment. Therefore, studies on static environments are usually insufficient to reveal an algorithm's performance when the problem is dynamic. Carlisle and Dozier [8, 9, 10] conducted a thorough investigation of PSO on a large number of dynamic test problems. Modifications of PSO that can tackle dynamic problems efficiently have been recently proposed [3, 11, 12].

A Unified PSO (UPSO) scheme has been recently introduced [13]. This scheme harnesses the local and global variant of PSO, combining their exploration and exploitation abilities without imposing additional requirements in terms of function evaluations. Convergence in probability was proved for the new scheme, and experimental results on widely used static benchmark functions justified its superiority against the standard PSO [13].

In this paper, the performance of UPSO in dynamic environments is investigated and compared with both the local and the global variant of the stan-

F. Rothlauf et al. (Eds.): EvoWorkshops 2005, LNCS 3449, pp. 590–599, 2005.

dard PSO algorithm. A test suite of five widely used test problems is employed. The movement of the global minimizer is simulated by adding to its position a random vector. Numerous experiments are performed and analyzed to justify UPSO's superiority and provide empirical rules regarding the most promising parameter configuration. The paper is organized as follows: PSO and UPSO are described in Section 2. Experimental results are reported in Section 3 and the paper concludes in Section 4.

2 Unified Particle Swarm Optimization

PSO was introduced by Eberhart and Kennedy [1, 14]. Similarly to the evolutionary algorithms, PSO exploits a population of potential solutions to probe the search space simultaneously. However, its dynamic is based on laws that govern socially organized colonies rather than natural selection. PSO adheres to the five basic principles of swarm intelligence [15, 16], therefore it is categorized as a swarm intelligence algorithm.

In PSO's context, the population is called a *swarm* and its individuals (search points) are called *particles*. Each particle moves in the search space with an adaptable velocity. Moreover, each particle has a memory where it retains the best position it has ever visited in the search space, i.e., the position with the lowest function value. Also, the particles share information among them. More specifically, each particle has an index number, and, according to this index, it is assigned a neighborhood of particles with (usually) neighboring index numbers. In the *global* variant of PSO, the neighborhood of each particle is the whole swarm. In the *local* variant, the neighborhoods are strictly smaller and they usually consist of a few particles.

Assume an n–dimensional function, $f : S \subset \mathbb{R}^n \to \mathbb{R}$, and a swarm, $\mathbb{S} = \{X_1, X_2, \ldots, X_N\}$, of N particles. The i–th particle, $X_i \in S$, its velocity, V_i, as well as its best position, $P_i \in S$, are n–dimensional vectors. A neighborhood of radius m of X_i consists of the particles $X_{i-m}, \ldots, X_i, \ldots, X_{i+m}$. The particles are usually assumed to be organized in a cyclic topology with respect to their indices. Thus, X_N and X_2 are the immediate neighbors of the particle X_1.

Assume g_i to be the index of the particle that attained the best previous position among all the particles in the neighborhood of X_i, and t to be the iteration counter. Then, according to the *constriction factor* version of PSO, the swarm is updated using the equations [17],

$$V_i(t+1) = \chi\Big[V_i(t) + c_1 r_1 \big(P_i(t) - X_i(t)\big) + c_2 r_2 \big(P_{g_i}(t) - X_i(t)\big)\Big], \quad (1)$$

$$X_i(t+1) = X_i(t) + V_i(t+1), \quad (2)$$

where $i = 1, \ldots, N$; χ is the constriction factor; c_1 and c_2 are positive constants, referred to as *cognitive* and *social* parameters, respectively; and r_1, r_2 are random vectors with components uniformly distributed in $[0, 1]$. All vector operations in Eqs. (1) and (2) are performed componentwise.

The constriction factor was introduced as means of controlling the magnitude of the velocities, in order to avoid the "swarm explosion" effect that was detrimental for the convergence of early PSO versions, and it is determined through the formula [17,18],

$$\chi = \frac{2\kappa}{|2-\phi-\sqrt{\phi^2-4\phi}|}, \tag{3}$$

for $\phi > 4$, where $\phi = c_1 + c_2$, and $\kappa = 1$. This selection is based on the stability analysis provided in [17].

There are two main characteristics of a population–based algorithm that affect its performance, namely *exploration* and *exploitation*. The first is the ability to probe effectively the search space, while the latter is the ability to converge to the most promising solutions with the smallest possible computational cost. A proper trade–off between exploration and exploitation is necessary for the efficient and effective operation of the algorithm. The global variant of PSO promotes exploitation since all particles are attracted by the same best position, thereby, converging faster towards the same point. On the other hand, local variant has better exploration properties, since the information regarding the best position of each neighborhood is communicated slower to the rest of the swarm through neighboring particles. Therefore, the attraction to specific points is weaker, thus, preventing the swarm from getting trapped in local minima. Obviously, the proper selection of neighborhood size affects the trade–off between exploration and exploitation. However, there are no general rules regarding the selection of neighborhood size, and it is usually based on the experience of the user.

The *Unified Particle Swarm Optimization* (UPSO) scheme was recently proposed as an alternative that combines the exploration and exploitation properties of both the local and global PSO variants [13]. For completeness purposes, a brief description of UPSO is provided in the following paragraphs. The presented scheme is based on the constriction factor version of PSO, although it can be straightforwardly defined also for the inertia weight version. Let $\mathcal{G}_i(t+1)$ and $\mathcal{L}_i(t+1)$ denote the velocity update of the i–th particle, X_i, for the global and local PSO variant, respectively [13],

$$\mathcal{G}_i(t+1) = \chi\left[V_i(t) + c_1 r_1\big(P_i(t) - X_i(t)\big) + c_2 r_2\big(P_g(t) - X_i(t)\big)\right], \tag{4}$$

$$\mathcal{L}_i(t+1) = \chi\left[V_i(t) + c_1 r_1'\big(P_i(t) - X_i(t)\big) + c_2 r_2'\big(P_{g_i}(t) - X_i(t)\big)\right], \tag{5}$$

where t denotes the iteration number; g is the index of the best particle of the whole swarm (global variant); and g_i is the index of the best particle in the neighborhood of X_i (local variant). The search directions defined by Eqs. (4) and (5) are aggregated in a single equation, resulting in the main UPSO scheme [13],

$$\mathcal{U}_i(t+1) = (1-u)\,\mathcal{L}_i(t+1) + u\,\mathcal{G}_i(t+1), \quad u \in [0,1], \tag{6}$$

$$X_i(t+1) = X_i(t) + \mathcal{U}_i(t+1). \tag{7}$$

We named the parameter u, *unification factor*. This factor balances the influence of the global and local search directions in the final scheme. The standard global

PSO variant is obtained by setting $u = 1$ in Eq. (6), while $u = 0$ corresponds the standard local PSO variant. All values of $u \in (0, 1)$, correspond to composite variants of PSO that combine the exploration and exploitation characteristics of its global and local variant.

Besides the aforementioned scheme, a stochastic parameter that imitates the mutation of evolutionary algorithms can also be incorporated in Eq. (6) to further enhance the exploration capabilities of UPSO [13]. Thus, depending on which variant UPSO is mostly based, Eq. (6) can be written either as [13],

$$\mathcal{U}_i(t+1) = (1-u)\,\mathcal{L}_i(t+1) + r_3\,u\,\mathcal{G}_i(t+1), \tag{8}$$

which is mostly based on the local variant, or

$$\mathcal{U}_i(t+1) = r_3\,(1-u)\,\mathcal{L}_i(t+1) + u\,\mathcal{G}_i(t+1), \tag{9}$$

which is mostly based on the global variant, where $r_3 \sim \mathcal{N}(\mu, \sigma^2 I)$ is a normally distributed parameter, and I is the identity matrix. Although r_3 imitates mutation, its direction is consistent with the PSO dynamics. For these UPSO schemes, convergence in probability was proved in static environments [13]. Experimental results on widely used test problems justified the superiority of UPSO against the standard PSO, for various configurations of the PSO parameters proposed in the relative literature [13, 18].

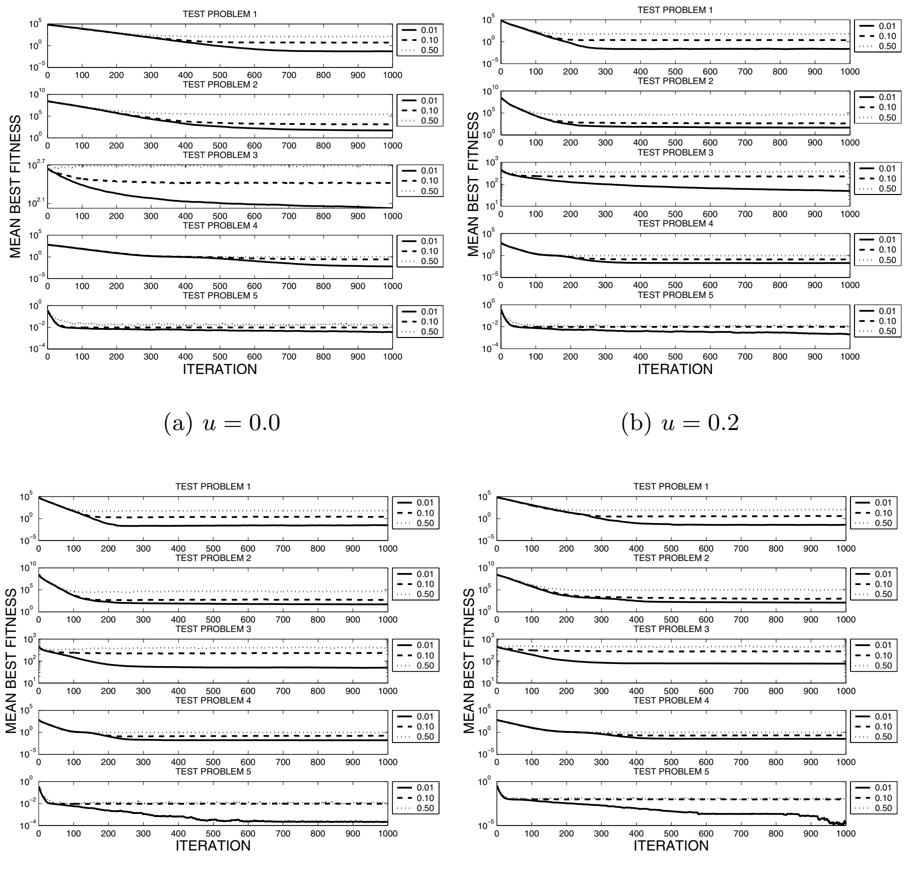
3 Results and Discussion

UPSO's performance was investigated on the most common DeJong test suite, which consists of the Sphere, Rosenbrock, Rastrigin, Griewank and Schaffer's F_6 function [9,17,18]. We will refer to these problems as Test Problem (TP) 1–5, respectively. Test Problems 1–4 were considered in 30 dimensions, while Test Problem 5 was considered 2–dimensional. The initialization ranges were $[-100, 100]^{30}$, $[-30, 30]^{30}$, $[-5.12, 5.12]^{30}$, $[-600, 600]^{30}$, and $[-100, 100]^2$, respectively.

The aforementioned static optimization problems were transformed to dynamic problems by moving their global minimizer. In order to make the simulation more realistic, we considered the global minimizer moving randomly and unbounded in the search space. This was performed by adding to the global minimizer a normally distributed random vector with mean value equal to zero and three different values of the standard deviation (denoted as MStD), 0.01, 0.10 and 0.50. Moreover, the movement was considered to be asynchronous, i.e., at each iteration, the global minimizer moved with a probability equal to 0.5.

Regarding PSO, we used the constriction factor version with the standard default parameter values, namely, $\chi = 0.729$, $c_1 = c_2 = 2.05$. Since the global minimizer was moving without constraints, no bounds were posed on velocities and particles. The best positions of the particles were re–evaluated after each movement of the global minimizer (a technique for the detection of changes in the environment is proposed in [9]). In order to fully exploit the exploration abilities of local PSO, a neighborhood of radius 1 was used for the computation of the search direction $\mathcal{L}$ of the local PSO variant.

The values $u = 0.2$ and $u = 0.5$ of the unification factor have been proved to be very efficient in static optimization problems [13]. Preliminary experiments on dynamic problems were in accordance with the results for static problems. Thus, initially, we considered the UPSO approach defined by Eqs. (6) and (7) for $u = 0.0$ (standard local PSO), $u = 0.2$, $u = 0.5$, and $u = 1.0$ (standard global PSO). For each test problem, 100 experiments were conducted and the algorithm was allowed to perform 1000 iterations per experiment. Since the main goal in dynamic environments is to track the orbit of the global minimizer rather than to acquire it [7], the quality assessments of static problems, such as the position with the smallest function value, are not valid in our case [6]. Instead, at each iteration (out of 1000) of an experiment, the mean function value of the particles' best positions was recorded. This value provides a more robust measure of the true quality of the particles [6]. Thus, 1000 such values



(a) $u = 0.0$ (b) $u = 0.2$

(c) $u = 0.5$ (d) $u = 1.0$

Fig. 1. Behavior of UPSO for different values of the unification factor

Table 1. Results for Test Problems 1–3

TP	MStD	u	Mean	StD	Min	Max
1	0.01	0.0	3.1655×10^3	4.8187×10^2	1.9664×10^3	4.3169×10^3
		0.2	9.1959×10^2	1.2136×10^2	6.3073×10^2	1.3489×10^3
		0.5	7.2230×10^2	9.7704×10^1	5.0892×10^2	1.0408×10^3
		1.0	1.6995×10^3	3.0792×10^2	1.0917×10^3	2.4738×10^3
	0.10	0.0	3.1021×10^3	5.3588×10^2	1.8875×10^3	4.4504×10^3
		0.2	9.1283×10^2	1.2416×10^2	5.8145×10^2	1.2892×10^3
		0.5	7.2875×10^2	1.0179×10^2	5.1276×10^2	9.3233×10^2
		1.0	1.6461×10^3	3.1279×10^2	9.9135×10^2	2.7464×10^3
	0.50	0.0	3.2536×10^3	4.7804×10^2	2.2689×10^3	4.4672×10^3
		0.2	9.5902×10^2	1.2177×10^2	6.6160×10^2	1.3458×10^3
		0.5	7.8051×10^2	9.9330×10^1	5.4604×10^2	1.0057×10^3
		1.0	1.8109×10^3	3.2161×10^2	1.1427×10^3	2.7648×10^3
2	0.01	0.0	8.6011×10^6	2.2275×10^6	3.0879×10^6	1.3499×10^7
		0.2	1.9842×10^6	3.8800×10^5	1.1650×10^6	2.9605×10^6
		0.5	1.4565×10^6	2.9872×10^5	8.9335×10^5	2.4064×10^6
		1.0	4.7003×10^6	1.5037×10^6	1.4236×10^6	9.8277×10^6
	0.10	0.0	8.5898×10^6	2.1055×10^6	3.8289×10^6	1.3550×10^7
		0.2	1.9888×10^6	3.1969×10^5	1.4178×10^6	3.0690×10^6
		0.5	1.4439×10^6	3.1333×10^5	7.1354×10^5	2.5139×10^6
		1.0	4.6714×10^6	1.4462×10^6	2.0045×10^6	8.4536×10^6
	0.50	0.0	8.7243×10^6	2.1533×10^6	3.6296×10^6	1.4842×10^7
		0.2	2.0783×10^6	4.5228×10^5	1.3253×10^6	4.2118×10^6
		0.5	1.4959×10^6	3.2928×10^5	7.0034×10^5	2.5733×10^6
		1.0	5.1983×10^6	2.0042×10^6	2.3351×10^6	1.1613×10^7
3	0.01	0.0	1.5523×10^2	1.9187×10^1	1.0409×10^2	2.1289×10^2
		0.2	9.7620×10^1	1.4826×10^1	6.9749×10^1	1.2776×10^2
		0.5	7.5825×10^1	1.2203×10^1	4.9258×10^1	1.0963×10^2
		1.0	1.0740×10^2	1.7714×10^1	6.4876×10^1	1.5081×10^2
	0.10	0.0	2.7790×10^2	6.1293×10^0	2.6168×10^2	2.9438×10^2
		0.2	2.3185×10^2	4.1795×10^0	2.2192×10^2	2.4213×10^2
		0.5	2.3210×10^2	6.5381×10^0	2.1548×10^2	2.4953×10^2
		1.0	2.8625×10^2	9.3753×10^0	2.6475×10^2	3.0655×10^2
	0.50	0.0	4.9098×10^2	1.8086×10^1	4.4882×10^2	5.4354×10^2
		0.2	3.6957×10^2	8.9636×10^0	3.4994×10^2	3.9674×10^2
		0.5	3.9097×10^2	1.5980×10^1	3.5973×10^2	4.2600×10^2
		1.0	4.5141×10^2	1.8168×10^1	4.1512×10^2	5.1713×10^2

were obtained per experiment. The behavior of UPSO for each test problem and unification factor, are illustrated in Fig. 1, for the three levels of MStD. Each line style corresponds to a different value of MStD and it stands for the mean value of the particles' best position per iteration, averaged over 100 experiments.

Table 2. Results for Test Problems 4 and 5

TP	MStD	u	Mean	StD	Min	Max
4	0.01	0.0	2.8443×10^{1}	4.7740×10^{0}	1.6007×10^{1}	3.9408×10^{1}
		0.2	8.5797×10^{0}	1.0504×10^{0}	6.2604×10^{0}	1.1255×10^{1}
		0.5	6.6704×10^{0}	9.2881×10^{-1}	5.0150×10^{0}	9.5947×10^{0}
		1.0	1.5233×10^{1}	3.1764×10^{0}	6.4421×10^{0}	2.3072×10^{1}
	0.10	0.0	2.8802×10^{1}	4.3512×10^{0}	1.8535×10^{1}	3.9056×10^{1}
		0.2	8.6834×10^{0}	1.1195×10^{0}	6.5475×10^{0}	1.2439×10^{1}
		0.5	6.8352×10^{0}	8.0967×10^{-1}	4.9374×10^{0}	8.6670×10^{0}
		1.0	1.5398×10^{1}	2.7152×10^{0}	8.5176×10^{0}	2.2396×10^{1}
	0.50	0.0	2.9844×10^{1}	4.6330×10^{0}	1.6961×10^{1}	3.9251×10^{1}
		0.2	9.2982×10^{0}	1.0643×10^{0}	7.4027×10^{0}	1.1693×10^{1}
		0.5	7.6360×10^{0}	1.0219×10^{0}	5.4295×10^{0}	1.0125×10^{1}
		1.0	1.6169×10^{1}	3.2054×10^{0}	9.3358×10^{0}	2.6588×10^{1}
5	0.01	0.0	8.0300×10^{-3}	3.3491×10^{-3}	1.6961×10^{-3}	1.3762×10^{-2}
		0.2	6.2539×10^{-3}	3.2677×10^{-3}	6.4010×10^{-4}	1.2861×10^{-2}
		0.5	3.3174×10^{-3}	1.6358×10^{-3}	9.0027×10^{-4}	1.2290×10^{-2}
		1.0	3.4830×10^{-3}	1.6805×10^{-3}	1.0990×10^{-3}	1.0637×10^{-2}
	0.10	0.0	1.2160×10^{-2}	1.0920×10^{-3}	1.0468×10^{-2}	1.6584×10^{-2}
		0.2	1.1681×10^{-2}	7.3334×10^{-4}	1.0157×10^{-2}	1.3808×10^{-2}
		0.5	1.1334×10^{-2}	6.7662×10^{-4}	9.9373×10^{-3}	1.3156×10^{-2}
		1.0	1.1156×10^{-2}	5.1947×10^{-4}	1.0125×10^{-2}	1.2706×10^{-2}
	0.50	0.0	2.1850×10^{-2}	2.0692×10^{-3}	1.5954×10^{-2}	2.8011×10^{-2}
		0.2	1.4647×10^{-2}	9.3145×10^{-4}	1.2654×10^{-2}	1.6941×10^{-2}
		0.5	1.3380×10^{-2}	9.3176×10^{-4}	1.1682×10^{-2}	1.8421×10^{-2}
		1.0	1.3399×10^{-2}	7.7779×10^{-4}	1.1639×10^{-2}	1.6081×10^{-2}

For statistical comparison purposes, the mean function values obtained per experiment, were averaged over the 1000 iterations. Thus, a single averaged mean function value was obtained for each experiment. Therefore, for each test problem, we obtained a total of 100 such averaged means. The mean, standard deviation (StD), minimum (Min) and maximum (Max) of the sample of these 100 averaged means are reported for all test problems and unification factor values in Tables 1 and 2. We observe that UPSO always outperformed both the local and global variant of PSO, which correspond to the values $u = 0.0$ and $u = 1.0$, respectively. The unification factor with the most promising behavior, with respect to the reported mean, is $u = 0.5$, which exhibits the smallest means in most cases, and the best overall behavior for both small and large values of MStD, which is an indication of its robustness.

The good performance of $u = 0.5$ triggered our interest on its behavior using the UPSO schemes of Eqs. (8) and (9). These schemes enhanced significantly UPSO's performance in static optimization problems [13]. For each test problem, 100 experiments were conducted using the UPSO schemes that incorporate $r_3 \sim \mathcal{N}(\mu, \sigma^2 I)$ either on the term of $\mathcal{G}$ (cf. Eq. (8)) or on the term of $\mathcal{L}$ (cf. Eq. (9)), in the update of UPSO's search direction, $\mathcal{U}$. Two different vectors, $\mu = (0, \ldots, 0)^\top$ and $\mu = (1, \ldots, 1)^\top$, were investigated (for simplicity purposes we use the notion $\mu = 0$ and $\mu = 1$, respectively), while a small standard deviation, $\sigma = 0.01$, was selected to alleviate deterioration of UPSO's dynamics [13]. The results are reported in Tables 3 and 4. The addition of r_3 improved further the performance

Table 3. Results for $u = 0.5$ using $r_3 \sim \mathcal{N}(\mu, \sigma^2 I)$ for Test Problems 1–3

TP	MStD	μ	Position	Mean	StD	Min	Max
1	0.01	0.0	on $\mathcal{G}$	3.1805×10^2	1.2188×10^2	1.8903×10^2	7.1865×10^2
			on $\mathcal{L}$	5.0574×10^3	1.2547×10^3	2.3535×10^3	8.7807×10^3
		1.0	on $\mathcal{G}$	7.2196×10^2	9.3121×10^1	4.1754×10^2	9.3062×10^2
			on $\mathcal{L}$	7.3134×10^2	9.8959×10^1	4.9692×10^2	9.5854×10^2
	0.10	0.0	on $\mathcal{G}$	3.3886×10^2	1.1305×10^2	2.2138×10^2	9.6920×10^2
			on $\mathcal{L}$	5.1672×10^3	1.2852×10^3	2.5525×10^3	7.8607×10^3
		1.0	on $\mathcal{G}$	7.3028×10^2	1.0014×10^2	5.2279×10^2	9.9433×10^2
			on $\mathcal{L}$	7.2599×10^2	9.0081×10^1	5.3669×10^2	1.0628×10^3
	0.50	0.0	on $\mathcal{G}$	1.4828×10^3	2.7719×10^2	9.0263×10^2	2.2291×10^3
			on $\mathcal{L}$	6.8694×10^3	1.5433×10^3	3.7164×10^3	1.0982×10^4
		1.0	on $\mathcal{G}$	7.7892×10^2	1.0623×10^2	5.5643×10^2	1.0218×10^3
			on $\mathcal{L}$	7.8379×10^2	9.5146×10^1	5.7097×10^2	1.1437×10^3
2	0.01	0.0	on $\mathcal{G}$	7.2742×10^5	2.0365×10^5	4.2395×10^5	1.3934×10^6
			on $\mathcal{L}$	3.0387×10^6	1.2173×10^6	7.5792×10^5	6.5612×10^6
		1.0	on $\mathcal{G}$	1.4257×10^6	3.3186×10^5	8.3978×10^5	2.5889×10^6
			on $\mathcal{L}$	1.4241×10^6	2.9498×10^5	8.4386×10^5	2.5711×10^6
	0.10	0.0	on $\mathcal{G}$	7.2379×10^5	1.7563×10^5	3.6936×10^5	1.2948×10^6
			on $\mathcal{L}$	3.3831×10^6	1.3056×10^6	1.0576×10^6	7.5686×10^6
		1.0	on $\mathcal{G}$	1.4670×10^6	3.0998×10^5	8.2882×10^5	2.5353×10^6
			on $\mathcal{L}$	1.4631×10^6	2.9592×10^5	8.4093×10^5	2.2752×10^6
	0.50	0.0	on $\mathcal{G}$	2.0289×10^7	1.1652×10^7	6.5949×10^6	6.8726×10^7
			on $\mathcal{L}$	5.7990×10^7	2.7477×10^7	1.6168×10^7	1.4867×10^8
		1.0	on $\mathcal{G}$	1.5569×10^6	3.4020×10^5	9.4104×10^5	2.5788×10^6
			on $\mathcal{L}$	1.4809×10^6	3.0238×10^5	8.8182×10^5	2.6824×10^6
3	0.01	0.0	on $\mathcal{G}$	8.7090×10^1	2.0353×10^1	5.2522×10^1	1.6865×10^2
			on $\mathcal{L}$	1.7782×10^2	2.5795×10^1	1.3637×10^2	2.6563×10^2
		1.0	on $\mathcal{G}$	7.4911×10^1	1.3349×10^1	4.8584×10^1	1.1738×10^2
			on $\mathcal{L}$	7.3238×10^1	1.2202×10^1	5.0768×10^1	1.0567×10^2
	0.10	0.0	on $\mathcal{G}$	3.2810×10^2	1.7360×10^1	2.9064×10^2	3.7078×10^2
			on $\mathcal{L}$	4.2494×10^2	2.8040×10^1	3.6882×10^2	5.0882×10^2
		1.0	on $\mathcal{G}$	2.3175×10^2	7.3102×10^0	2.1569×10^2	2.5508×10^2
			on $\mathcal{L}$	2.3111×10^2	7.1345×10^0	2.1828×10^2	2.5152×10^2
	0.50	0.0	on $\mathcal{G}$	1.9161×10^3	3.8433×10^2	1.3110×10^3	3.0697×10^3
			on $\mathcal{L}$	2.1162×10^3	3.9866×10^2	1.3790×10^3	3.5007×10^3
		1.0	on $\mathcal{G}$	3.9092×10^2	2.4602×10^1	3.4996×10^2	5.1264×10^2
			on $\mathcal{L}$	3.8603×10^2	1.8018×10^1	3.4986×10^2	4.2635×10^2

Table 4. Results for $u = 0.5$ using $r_3 \sim \mathcal{N}(\mu, \sigma^2 I)$ for Test Problems 4 and 5

TP	MStD	μ	Position	Mean	StD	Min	Max
4	0.01	0.0	on $\mathcal{G}$	2.8650×10^{0}	1.1803×10^{0}	1.7891×10^{0}	9.3589×10^{0}
			on $\mathcal{L}$	4.5350×10^{1}	1.2000×10^{1}	1.8176×10^{1}	8.4527×10^{1}
		1.0	on $\mathcal{G}$	6.7278×10^{0}	1.0156×10^{0}	4.7195×10^{0}	1.0510×10^{1}
			on $\mathcal{L}$	6.6302×10^{0}	8.5156×10^{-1}	4.8633×10^{0}	9.1128×10^{0}
	0.10	0.0	on $\mathcal{G}$	3.5698×10^{0}	1.0864×10^{0}	2.4488×10^{0}	7.1177×10^{0}
			on $\mathcal{L}$	4.5699×10^{1}	1.1305×10^{1}	2.3487×10^{1}	7.9473×10^{1}
		1.0	on $\mathcal{G}$	6.9523×10^{0}	8.7108×10^{-1}	5.5159×10^{0}	9.5926×10^{0}
			on $\mathcal{L}$	6.7420×10^{0}	8.7237×10^{-1}	4.5845×10^{0}	9.8724×10^{0}
	0.50	0.0	on $\mathcal{G}$	4.0269×10^{0}	9.8038×10^{-1}	3.0680×10^{0}	7.9478×10^{0}
			on $\mathcal{L}$	4.7370×10^{1}	1.0754×10^{1}	2.7856×10^{1}	7.0848×10^{1}
		1.0	on $\mathcal{G}$	7.5279×10^{0}	8.9793×10^{-1}	5.4762×10^{0}	1.0165×10^{1}
			on $\mathcal{L}$	7.6153×10^{0}	9.1264×10^{-1}	5.7358×10^{0}	9.8793×10^{0}
5	0.01	0.0	on $\mathcal{G}$	8.6723×10^{-3}	3.1477×10^{-3}	3.1126×10^{-3}	1.9211×10^{-2}
			on $\mathcal{L}$	7.6079×10^{-3}	5.4494×10^{-3}	1.4328×10^{-3}	3.8073×10^{-2}
		1.0	on $\mathcal{G}$	3.2053×10^{-3}	1.1405×10^{-3}	8.4245×10^{-4}	6.2444×10^{-3}
			on $\mathcal{L}$	2.9865×10^{-3}	1.3656×10^{-3}	9.8465×10^{-4}	1.0661×10^{-2}
	0.10	0.0	on $\mathcal{G}$	1.7457×10^{-2}	8.4832×10^{-3}	1.1007×10^{-2}	5.7459×10^{-2}
			on $\mathcal{L}$	2.0828×10^{-1}	1.1152×10^{-1}	5.0655×10^{-2}	4.8215×10^{-1}
		1.0	on $\mathcal{G}$	1.1240×10^{-2}	6.5926×10^{-4}	1.0022×10^{-2}	1.3759×10^{-2}
			on $\mathcal{L}$	1.1246×10^{-2}	6.7370×10^{-4}	9.4405×10^{-3}	1.3298×10^{-2}
	0.50	0.0	on $\mathcal{G}$	1.9668×10^{-1}	6.3972×10^{-2}	6.6878×10^{-2}	3.3205×10^{-1}
			on $\mathcal{L}$	4.6819×10^{-1}	2.8537×10^{-2}	4.0353×10^{-1}	5.2920×10^{-1}
		1.0	on $\mathcal{G}$	1.3523×10^{-2}	1.1829×10^{-3}	1.1475×10^{-2}	2.1030×10^{-2}
			on $\mathcal{L}$	1.3627×10^{-2}	1.3526×10^{-3}	1.1788×10^{-2}	2.1753×10^{-2}

of UPSO. For a given level of MStD, the best mean and minimum value both correspond to the same value of μ in all cases, with $\mu = 1$ being marginally better than $\mu = 0$, overall. The best behavior was obtained when r_3 was incorporated in the term of $\mathcal{G}$. Finally, we must notice that for large values of MStD (i.e., 0.5) $\mu = 1$ proved to be the best choice in all test problems with the exception of Test Problem 4.

Summarizing the results, the UPSO scheme of Eq. (8) can be considered a good default choice in unknown dynamic environments when no additional information is available.

4 Conclusions

We investigated the performance of the new, Unified Particle Swarm Optimization (UPSO) scheme in dynamic environments. Experiments on widely used benchmark problems were conducted with very promising results. UPSO outperformed both the local and global PSO variant. Guidelines regarding the most promising UPSO scheme are derived by analyzing the results, and support the claim that, besides static optimization problems, UPSO is a promising scheme also in dynamic environments.

References

1. Eberhart, R.C., Kennedy, J.: A new optimizer using particle swarm theory. In: Proc. 6th Symp. Micro Mach. Hum. Sci., IEEE Service Center (1995) 39–43
2. Kennedy, J., Eberhart, R.C.: Swarm Intelligence. Morgan Kaufmann Publ. (2001)
3. Parsopoulos, K.E., Vrahatis, M.N.: Recent approaches to global optimization problems through particle swarm optimization. Natural Computing **1** (2002) 235–306
4. Parsopoulos, K.E., Vrahatis, M.N.: On the computation of all global minimizers through particle swarm optimization. IEEE Trans. Evol. Comp. **8** (2004) 211–224
5. Branke, J.: Evolutionary Optimization in Dynamic Environments. Kluwer Academic Publishers (2001)
6. Bäck, T., U., H.: Evolution strategies applied to perturbed objective functions. In: Proc. IEEE Congr. Evol. Comput. (1994) 40–45
7. Angeline, P.J.: Tracking extrema in dynamic environments. In: Proc. Evolutionary Programming VI. (1997) 335–345
8. Carlisle, A., Dozier, G.: Adapting particle swarm optimization to dynamic environments. In: Proc. Int. Conf. Artif. Intell., Las Vegas (NV), USA (2000) 429–434
9. Carlisle, A.: Applying the Particle Swarm Optimizer to Non–Stationary Environments. PhD thesis, Auburn University, Auburn, Alabama, USA (2002)
10. Carlisle, A., Dozier, G.: Tracking changing extrema with adaptive particle swarm optimizer. In: Proc. 2002 World Autom. Congr., Orlando (FL), USA (2002)
11. Blackwell, T., Branke, J.: Multi–swarm optimization in dynamic environments. In: Lecture Notes in Computer Science. Volume 3005. Springer-Verlag (2004) 489–500
12. Janson, S., Middendorf, M.: A hierarchical particle swarm optimizer for dynamic optimization problems. In: Lecture Notes in Computer Science. Volume 3005. Springer-Verlag (2004) 513–524
13. Parsopoulos, K.E., Vrahatis, M.N.: UPSO: A unified particle swarm optimization scheme. In: Lecture Series on Computer and Computational Sciences, Vol. 1, Proc. Int. Conf. Comput. Meth. Sci. Eng. (ICCMSE 2004), VSP International Science Publishers, Zeist, The Netherlands (2004) 868–873
14. Kennedy, J., Eberhart, R.C.: Particle swarm optimization. In: Proc. IEEE Int. Conf. Neural Networks. Volume IV., IEEE Service Center (1995) 1942–1948
15. Millonas, M.M.: Swarms, phase transitions, and collective intelligence. In Palaniswami, M., Attikiouzel, Y., Marks, R., Fogel, D., Fukuda, T., eds.: Computational Intelligence: A Dynamic System Perspective. IEEE Press (1994) 137–151
16. Eberhart, R.C., Simpson, P., Dobbins, R.: Computational Intelligence PC Tools. Academic Press (1996)
17. Clerc, M., Kennedy, J.: The particle swarm–explosion, stability, and convergence in a multidimensional complex space. IEEE Trans. Evol. Comput. **6** (2002) 58–73
18. Trelea, I.C.: The particle swarm optimization algorithm: Convergence analysis and parameter selection. Information Processing Letters **85** (2003) 317–325
19. Matyas, J.: Random optimization. Automatization and Remote Control **26** (1965) 244–251