Swarm Intell (2009) 3: 87–124
DOI 10.1007/s11721-008-0021-5

Glowworm swarm optimization for simultaneous capture of multiple local optima of multimodal functions

K.N. Krishnanand · D. Ghose

Received: 12 June 2007 / Accepted: 4 November 2008 / Published online: 9 December 2008

Abstract This paper presents glowworm swarm optimization (GSO), a novel algorithm for the simultaneous computation of multiple optima of multimodal functions. The algorithm shares a few features with some better known swarm intelligence based optimization algorithms, such as ant colony optimization and particle swarm optimization, but with several significant differences. The agents in GSO are thought of as glowworms that carry a luminescence quantity called luciferin along with them. The glowworms encode the fitness of their current locations, evaluated using the objective function, into a luciferin value that they broadcast to their neighbors. The glowworm identifies its neighbors and computes its movements by exploiting an adaptive neighborhood, which is bounded above by its sensor range. Each glowworm selects, using a probabilistic mechanism, a neighbor that has a luciferin value higher than its own and moves toward it. These movements—based only on local information and selective neighbor interactions—enable the swarm of glowworms to partition into disjoint subgroups that converge on multiple optima of a given multimodal function. We provide some theoretical results related to the luciferin update mechanism in order to prove the bounded nature and convergence of luciferin levels of the glowworms. Experimental results demonstrate the efficacy of the proposed glowworm based algorithm in capturing multiple optima of a series of standard multimodal test functions and more complex ones, such as stair-case and multiple-plateau functions. We also report the results of tests in higher dimensional spaces with a large number of peaks. We address the parameter selection problem by conducting experiments to show that only two parameters need to be selected by the user. Finally, we provide some comparisons of GSO with PSO and an experimental comparison with Niche-PSO, a PSO variant that is designed for the simultaneous computation of multiple optima.

This work is partially supported by a project grant from the Ministry of Human Resources Development, India and by DRDO-IISc Mathematical Engineering Program.

K.N. Krishnanand (✉) · D. Ghose
Department of Aerospace Engineering, Indian Institute of Science, Bangalore 560 012, India
e-mail: krishna@aero.iisc.ernet.in

D. Ghose
e-mail: dghose@aero.iisc.ernet.in

Keywords Multimodal function optimization · Ant colony optimization · Particle swarm optimization · Glowworm swarm optimization · Multiple signal source localization

1 Introduction

Swarm intelligence (SI) (Bonabeau et al. 1999) as observed in natural swarms is the result of actions that individuals in the swarm perform exploiting local information. Usually, the swarm behavior serves to accomplish certain complex colony-level goals. Examples include group foraging by ants, division of labor between scouts and recruits in honeybee swarms, evading of predators by fish schools, flocking of birds, and group-hunting as observed in canids, herons, and several cetaceans.

The decentralized decision-making mechanisms found in the above examples, and others in the natural world, offer an insight on how to design distributed algorithms that solve complex problems related to diverse fields, such as optimization, multi-agent decision making, and collective robotics. Recent literature abounds with examples of such algorithms including bacterial chemotaxis based optimization (Muller et al. 2002), ant colony optimization techniques (Dorigo et al. 1996; Dorigo and Gambardella 1997; Dorigo and Stützle 2004), particle swarm optimization algorithms (Kennedy and Eberhart 1995; Clerc 2007; Poli et al. 2007), and several swarm based collective robotic algorithms (Dorigo et al. 2004; Fronczek and Prasad 2005; Zarzhitsky et al. 2005).

Multimodal function optimization has been addressed extensively in the recent literature. The traditional class of problems related to this topic focused on developing algorithms to find either the global optimum or all the global optima of the given multimodal function, while avoiding local optima. However, there is another class of optimization problems which is different from the problem of finding only the global optimum of a multimodal function. The objective of this class of multimodal optimization problems is to find multiple optima having either equal or unequal function values (Brits et al. 2002; Kennedy 2000; Parsopoulos and Vrahatis 2004). The knowledge of multiple local and global optima has several advantages, such as providing insight into the function landscape and allowing for the selection of an alternative solution when the dynamic nature of constraints in the search space makes a previous optimum solution infeasible to implement. Multi-modality in a search and optimization problem gives rise to several attractors and thereby presents a challenge to any optimization algorithm in terms of finding global optimum solutions (Singh and Deb 2006). The problem is compounded when multiple (global and local) optima are sought.

In this paper, we present a novel algorithm called glowworm swarm optimization (GSO) for the simultaneous computation of multiple optima of multimodal functions. The algorithm shares some common features with ant colony optimization (ACO) and with particle swarm optimization (PSO), but with several significant differences. The agents in GSO are thought of as glowworms[1] that carry a luminescent quantity called *luciferin*.[2] Each glowworm uses an artificial equivalent to luciferin, which we will call luciferin for short in the following, to broadcast the fitness of its current location, evaluated using the objective function, to its neighbors. The glowworms depend on a variable neighborhood, which is bounded

[1]The glowworm belongs to a family of beetles known as the *Lampyridae* or fireflies and produces natural light that is used as a signal to attract a mate (Tyler 1994).

[2]Luciferin is one of the several pigments used by glowworms to produce bioluminescent light.

above by a radial sensor range, to identify their neighbors and compute their movements. Each glowworm, using a probabilistic mechanism, selects a neighbor that has a luciferin value higher than its own and moves toward it. That is, glowworms are attracted to neighbors that glow brighter. These movements, that are based only on local information and selective neighbor interactions, enable the swarm to partition into disjoint subgroups that converge to multiple optima of a given multimodal function.

Preliminary results obtained with GSO were presented in (Krishnanand and Ghose 2005, 2006a). We have reported the related theoretical foundations in (Krishnanand and Ghose 2006b, 2008). In (Krishnanand et al. 2006a), we examine the behavior of GSO in the presence of noise: a comparison with a non-SI based (gradient-ascent) approach reveals the superiority of GSO in coping with uncertainty.

The paper is organized as follows. A complete description of GSO is given in Sect. 2. The multimodal functions and performance measures used to evaluate GSO are presented in Sect. 3. Some basic issues and related simulation results are presented in Sect. 4. Section 5 deals with the effect of algorithm's parameters on GSO's performance. Test results in higher dimensional spaces are given in Sect. 6. GSO is compared to PSO in Sect. 7. The paper concludes with some remarks in Sect. 8.

2 The glowworm swarm optimization (GSO) algorithm

In GSO, a swarm of agents are initially randomly distributed in the search space. Agents are modeled after glowworms and will be called glowworms in the following of this paper. Accordingly, they carry a luminescent quantity called *luciferin* along with them. The glowworms emit a light whose intensity is proportional to the associated luciferin and interact with other agents within a variable neighborhood. In particular, the neighborhood is defined as a local-decision domain that has a variable neighborhood range r_d^i bounded by a radial sensor range r_s $(0 < r_d^i \leq r_s)$. A glowworm i considers another glowworm j as its neighbor if j is within the neighborhood range of i and the luciferin level of j is higher than that of i. The decision domain enables selective neighbor interactions and aids in formation of disjoint sub-swarms. Each glowworm is attracted by the brighter glow of other glowworms in the neighborhood. Agents in GSO depend only on information available in their neighborhood to make decisions. For instance, in Fig. 1(a), agent i is in the sensor range of (and is equidistant to) both j and k. However, j and k have different neighborhood sizes, and only j uses the information of i. Figure 1(b) shows the directed graph based on the relative luciferin level of each agent and on the availability of only local information. Each glowworm selects, using a probabilistic mechanism, a neighbor that has a luciferin value higher than its own and moves toward it. These movements, that are based only on local information and selective neighbor interactions, enable the swarm of glowworms to partition into disjoint subgroups that steer toward, and meet at, multiple optima of a given multimodal function.

2.1 Algorithm description

The GSO algorithm is given in Fig. 2. It starts by placing a population of n glowworms randomly in the search space so that they are well dispersed. Initially, all the glowworms contain an equal quantity of luciferin ℓ_0. Each iteration consists of a luciferin-update phase followed by a movement phase based on a transition rule.

Luciferin-update phase: The luciferin update depends on the function value at the glowworm position. During the luciferin-update phase, each glowworm adds, to its previous luciferin level, a luciferin quantity proportional to the fitness of its current location in the

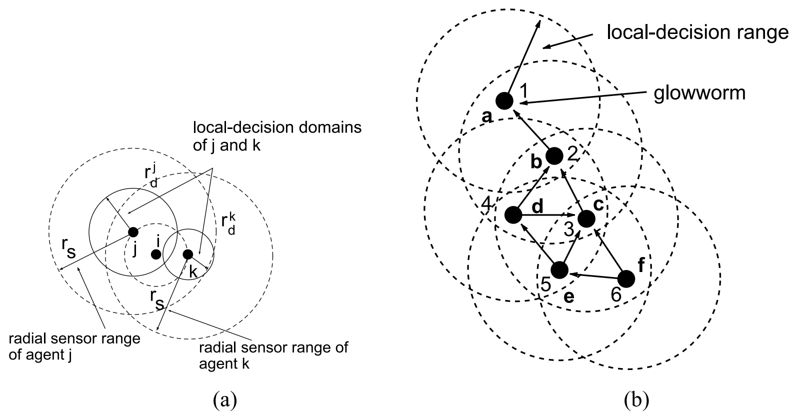

local-decision domains of j and k
r_d^j
r_d^k
r_s
j
i
k
r_s
radial sensor range of agent j
radial sensor range of agent k
(a)
local-decision range
glowworm
a
1
b
2
4
d
c
3
5
e
6
f
(b)

Fig. 1 (**a**) $r_d^k < d(i,k) = d(i,j) < r_d^j < r_s$. Agent i is in the sensor range of (and is equidistant to) both j and k. However, j and k have different neighborhood sizes, and only j uses the information of i. (**b**) Directed graph based on the relative luciferin level of each agent and availability of only local information. Agents are ranked according to the increasing order of their luciferin values. For instance, agent 'a', whose luciferin value is highest, is ranked '1' in the figure

objective function domain. Also, a fraction of the luciferin value is subtracted to simulate the decay in luciferin with time. The luciferin update rule is given by:

$$\ell_i(t+1) = (1-\rho)\ell_i(t) + \gamma J\big(x_i(t+1)\big), \tag{1}$$

where $\ell_i(t)$ represents the luciferin level associated with glowworm i at time t, ρ is the luciferin decay constant $(0 < \rho < 1)$, γ is the luciferin enhancement constant, and $J(x_i(t))$ represents the value of the objective function at agent i's location at time t.

Movement phase: During the movement phase, each glowworm decides, using a probabilistic mechanism, to move toward a neighbor that has a luciferin value higher than its own. That is, glowworms are attracted to neighbors that glow brighter. Figure 1(b) shows the directed graph among a set of six glowworms based on their relative luciferin levels and availability of only local information. For instance, there are four glowworms (a, b, c, and d) that have relatively more luciferin than glowworm e. Since e is located in the sensor-overlap region of c and d, it has only two possible directions of movement. For each glowworm i, the probability of moving toward a neighbor j is given by:

$$p_{ij}(t) = \frac{\ell_j(t) - \ell_i(t)}{\sum_{k \in N_i(t)} \ell_k(t) - \ell_i(t)}, \tag{2}$$

where $j \in N_i(t)$, $N_i(t) = \{j : d_{ij}(t) < r_d^i(t); \ell_i(t) < \ell_j(t)\}$ is the set of neighbors of glowworm i at time t, $d_{ij}(t)$ represents the Euclidean distance between glowworms i and j at time t, and $r_d^i(t)$ represents the variable neighborhood range associated with glowworm i at time t. Let glowworm i select a glowworm $j \in N_i(t)$ with $p_{ij}(t)$ given by (2). Then, the discrete-time model of the glowworm movements can be stated as:

$$x_i(t+1) = x_i(t) + s\left(\frac{x_j(t) - x_i(t)}{\|x_j(t) - x_i(t)\|}\right), \tag{3}$$

where $x_i(t) \in \mathbb{R}^m$ is the location of glowworm i, at time t, in the m-dimensional real space $\mathbb{R}^m$, $\|\cdot\|$ represents the Euclidean norm operator, and s (> 0) is the step size.

```
GLOWWORM SWARM OPTIMIZATION (GSO) ALGORITHM

Set number of dimensions $= m$
Set number of glowworms $= n$
Let $s$ be the step size
Let $x_i(t)$ be the location of glowworm $i$ at time $t$
deploy_agents_randomly;
for $i = 1$ to $n$ do $\ell_i(0) = \ell_0$
$r_d^i(0) = r_0$
set maximum iteration number $=$ iter_max;
set $t = 1$;
while ($t \leq$ iter_max) do:
{
    for each glowworm $i$ do: % Luciferin-update phase
        $\ell_i(t) = (1-\rho)\ell_i(t-1) + \gamma J(x_i(t))$; % See (1)

    for each glowworm $i$ do: % Movement-phase
    {
        $N_i(t) = \{j : d_{ij}(t) < r_d^i(t); \ell_i(t) < \ell_j(t)\}$;
        for each glowworm $j \in N_i(t)$ do:
            $p_{ij}(t) = \frac{\ell_j(t)-\ell_i(t)}{\sum_{k \in N_i(t)} \ell_k(t)-\ell_i(t)}$; % See (2)
        $j = select\_glowworm(\vec{p})$;
        $x_i(t+1) = x_i(t) + s(\frac{x_j(t)-x_i(t)}{\|x_j(t)-x_i(t)\|})$ % See (3)
      $r_d^i(t+1) = \min\{r_s, \max\{0, r_d^i(t) + \beta(n_t - |N_i(t)|)\}\}$; % See (4)
    }
    $t \leftarrow t+1$;
}
```

Fig. 2 The GSO algorithm

Neighborhood range update rule: We associate with each agent i a neighborhood whose radial range r_d^i is dynamic in nature $(0 < r_d^i \leq r_s)$. The fact that a fixed neighborhood range is not used needs some justification. When the glowworms depend only on local information to decide their movements, it is expected that the number of peaks captured would be a function of the radial sensor range. In fact, if the sensor range of each agent covers the entire search space, all the agents move to the global optimum and the local optima are ignored. Since we assume that a priori information about the objective function (e.g., number of peaks and inter-peak distances) is not available, it is difficult to fix the neighborhood range at a value that works well for different function landscapes. For instance, a chosen neighborhood range r_d would work relatively better on objective functions where the minimum inter-peak distance is more than r_d rather than on those where it is less than r_d. Therefore, GSO uses an adaptive neighborhood range in order to detect the presence of multiple peaks in a multimodal function landscape.

Let r_0 be the initial neighborhood range of each glowworm (that is, $r_d^i(0) = r_0\ \forall i$). To adaptively update the neighborhood range of each glowworm, the following rule is applied:

$$r_d^i(t+1) = \min\big\{r_s, \max\big\{0, r_d^i(t) + \beta\big(n_t - \big|N_i(t)\big|\big)\big\}\big\}, \qquad (4)$$

Table 1 Values of algorithm parameters that are kept fixed for all the experiments

ρ	γ	β	n_t	s	ℓ_0
0.4	0.6	0.08	5	0.03	5

where β is a constant parameter and n_t is a parameter used to control the number of neighbors.

The quantities $\rho, \gamma, s, \beta, n_t$, and ℓ_0 are algorithm parameters for which appropriate values have been determined based on extensive numerical experiments and are kept fixed in this paper (Table 1). The quantity r_0 is made equal to r_s in all the experiments. Thus, n and r_s are the only parameters that influence the algorithm behavior (in terms of the total number of peaks captured) and need to be selected.

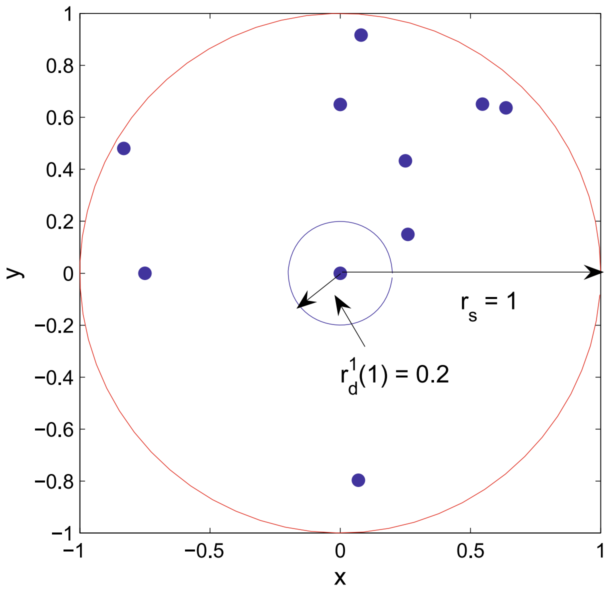
We use the following example to demonstrate the variable nature of the neighborhood. For the purpose of simplicity, we consider a static placement of glowworms and observe how the neighborhood range of the glowworm at $(0,0)$ varies according to (4). In Figs. 3 and 4, we notice that the neighborhood range adjusted until the glowworm acquires the number of neighbors that is specified by the parameter n_t (=3). Figure 3(a) shows an initial placement where the glowworm at $(0,0)$ is isolated. It increases its neighborhood range (Fig. 3(b)) until it either acquires a set of three neighbors or reaches the maximum range. In Fig. 4(a), the glowworm is crowded by a large number of neighbors ($|N_i(t)| > n_t$) that causes the neighborhood range to shrink until $|N_i(t)| = n_t$. Figure 4(b) shows the variation of the neighborhood range as a function of the iteration number.

When a glowworm is located far away from a peak, typically it has only a few neighbors or it may even be isolated (as in Fig. 3(a)). Rule (4) aids a glowworm in this situation to find neighbors by increasing its neighborhood range at each iteration. In contrast, when the glowworm is located in the vicinity of multiple peaks, it is more likely that the glowworm is surrounded by a large number of neighbors (as in Fig. 4(a)). Consequently, rule (4) restricts its visibility to a few neighbors so that its movement is biased toward a nearby peak.

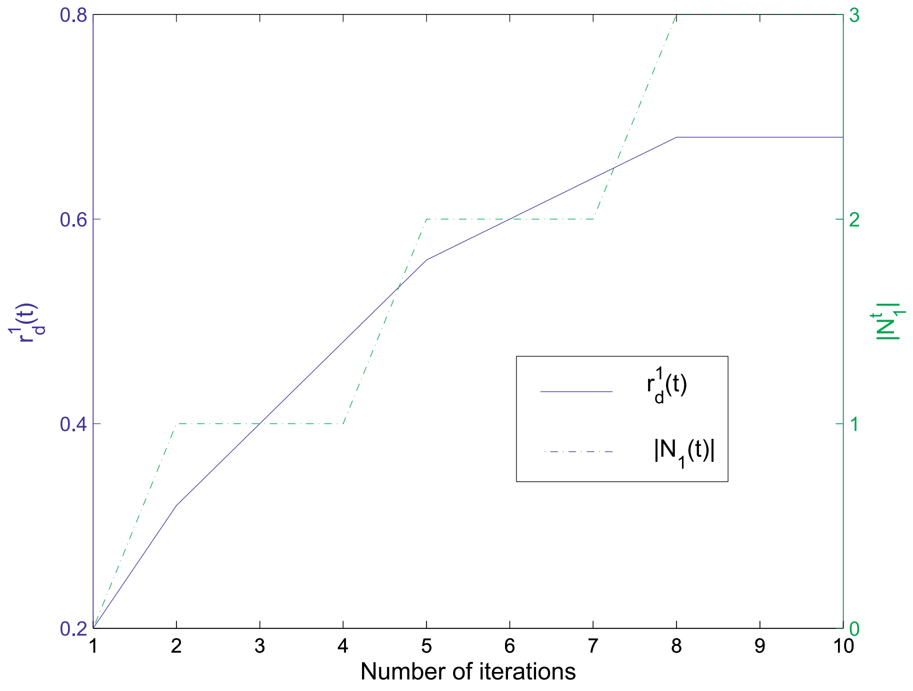
The leapfrogging effect: According to the glowworm algorithm presented in this paper, in any given iteration the glowworm with the maximum luciferin remains stationary. The above property may lead to a dead-lock situation where all the glowworms in the vicinity of a peak converge to the glowworm that is located closest to the peak. Since the agent movements are restricted to the interior region of the convex-hull, all the glowworms converge to a glowworm that attains maximum luciferin value during its movements within the convex-hull. As a result, all the glowworms are trapped away from the peak. However, the discrete nature of the movement update rule automatically takes care of this problem. In fact, during the movement phase, each glowworm moves a distance of finite step size s toward a neighbor. Hence, when the distance between a glowworm i approaching a neighbor j is less than s, i leapfrogs over the position of j and becomes a leader to j. In the next iteration, i remains stationary and j overtakes the position of i thus regaining its leadership. This process of role interchange between i and j repeats, giving rise to a local-search behavior of the glowworm pair along a single ascent direction. A group of glowworms uses the same principle to perform an improved local-search and eventually converge to the peak location. The leapfrogging behavior of the agents in GSO can be observed in simulations in Sect. 4 (Fig. 6(d)).

2.2 Convergence results

We provide some theoretical proofs for the luciferin update rule proposed in the previous section. First, we prove that, due to luciferin decay, the maximum luciferin level $\ell^{\max}$ is

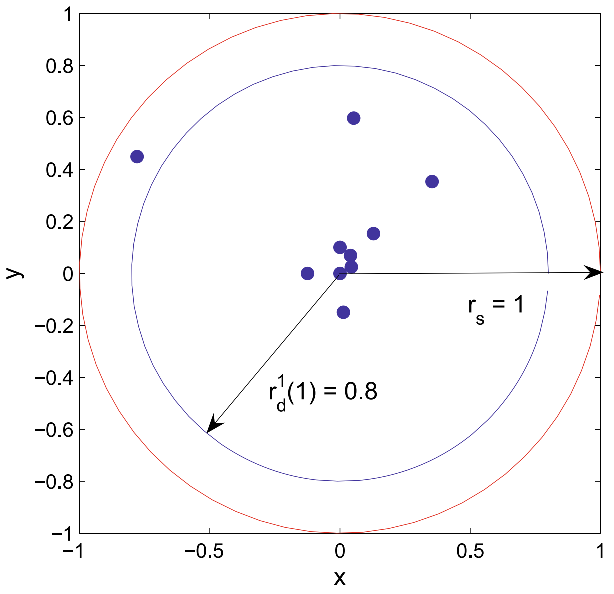


(a)

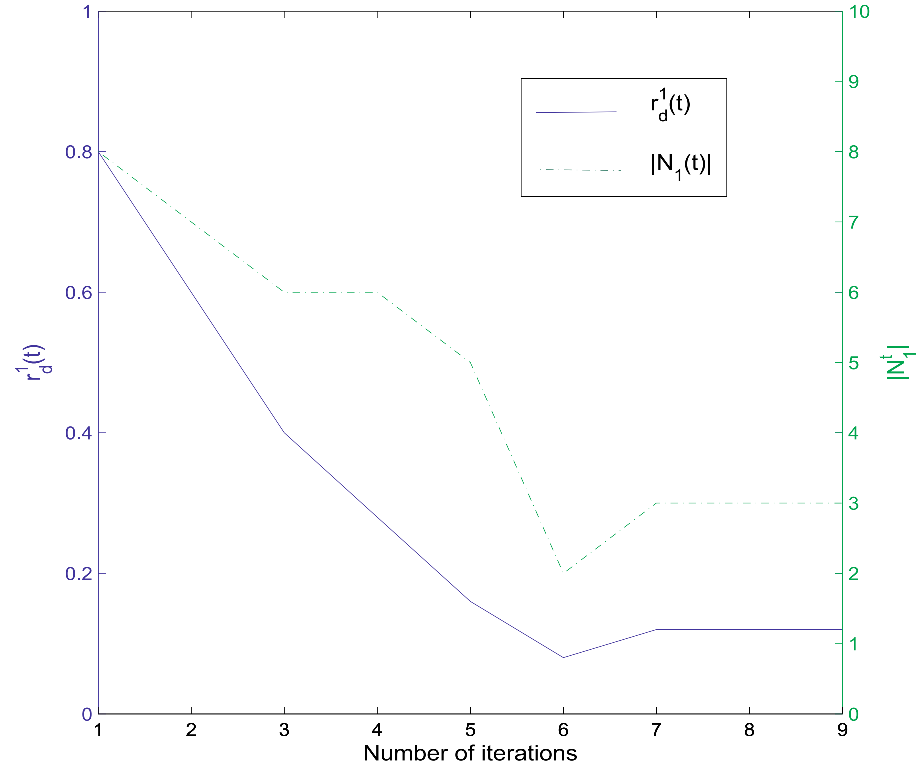


(b)

Fig. 3 (**a**) Initial placement where the glowworm at (0, 0) is isolated. (**b**) Plot of $r_d^1(t)$. Values of $r_s = 1$ and $n_t = 3$ are used



(a)



(b)

Fig. 4 (**a**) Initial placement where the glowworm at (0, 0) is crowded by a large number of glowworms. (**b**) Plot of $r_d^1(t)$. Values of $r_s = 1$ and $n_t = 3$ are used bounded asymptotically. Second, we show that the luciferin ℓ_j of each glowworm j located at a peak X_i converges to the same value ℓ_i^*.

Theorem 1 *Assuming that the luciferin update rule in (1) is used, the luciferin level $\ell_j(t)$ for any glowworm j is bounded above asymptotically as follows:*

$$\lim_{t\to\infty} \ell_j(t) \le \lim_{t\to\infty} \ell^{\max}(t) = \left(\frac{\gamma}{\rho}\right) J_{\max}, \tag{5}$$

where $J_{\max}(> 0)$ is the global maximum value of the objective function.

Proof Given that the maximum value of the objective function is $J_{\max}$ and the luciferin update rule in (1) is used, the maximum luciferin $\ell^{\max}(t)$ of any glowworm j at any iteration t is given by:

$$\ell^{\max}(t) = (1-\rho)^t \ell_0 + \sum_{i=0}^{t-1} (1-\rho)^i \gamma J_{\max} \Rightarrow \lim_{t\to\infty} \ell^{\max}(t) = \left(\frac{\gamma}{\rho}\right) J_{\max}.$$ □

Theorem 2 *For all glowworms j located at peak locations X_i^* associated with objective function values $J_i^* \le J_{\max}$ (>0) (where $i = 1, 2, \ldots, n_p$, with n_p the number of peaks), if the luciferin update rule (1) is used, then $\ell_j(t)$ increases or decreases monotonically and asymptotically converges to $\ell_i^* = (\frac{\gamma}{\rho}) J_i^*$.*

Proof If $\ell_j(t) < \ell_i^*$ for glowworm j located at peak location X_i^*, then it is easy to show that $\ell_j(t)$ increases monotonically. Similarly, if $\ell_j(t) > \ell_i^*$, we can show that $\ell_j(t)$ decreases monotonically.

Now, we prove the convergence of the sequence $\ell_j(t)$ by showing that the fixed point ℓ_i^* is asymptotically stable. In particular,

$$\left|\ell_j(t) - \ell_i^*\right| = (1-\rho)^t \left|\ell_j(0) - \ell_i^*\right|. \tag{6}$$

Therefore,

$$\lim_{t\to\infty} \left|\ell_j(t) - \ell_i^*\right| = \lim_{t\to\infty} (1-\rho)^t \left|\ell_j(0) - \ell_i^*\right| = 0, \quad \text{since } 0 < (1-\rho) < 1. \tag{7}$$

From (7), it is clear that the luciferin $\ell_j(t)$ of glowworm j, located at a peak location X_i^*, asymptotically converges to ℓ_i^*. □

In the same way, we can show that the luciferin value of each glowworm will converge to $\frac{\gamma}{\rho} J(x_i)$, where x_i is the position of the glowworm as $t \to \infty$. Hence, glowworms at the global maximum will reach a luciferin value that is not reachable for other agents located at other local peaks.

Note that we consider only cases with $J_{\max}$ strictly positive in both the theorems. However, the proofs can be easily extended to take into account the cases with negative $J_{\max}$.

3 Test functions and performance measures

We consider a representative set of multimodal test functions including unequal peaks, equal peaks, peaks of concentric circles, peak-regions involving step-discontinuities, and plateaus of equal heights. In the next section, we conduct some experiments to study the dependence of the algorithm's behavior on the critical parameters n and r_s. We will use the conclusions drawn from these experiments to tune the parameter r_s in Sect. 5, where we will test the performance of GSO in higher dimensions with a large number of peaks. Finally, we will carry out an experimental comparison of GSO with a PSO variant developed for capturing multiple peaks of equal function values (Sect. 7.2).

3.1 Multimodal test functions

GSO is tested on the following set of multimodal functions:

Peaks function:

$$J_1(x,y) = 3(1-x)^2 e^{-[x^2+(y+1)^2]} - 10\left(\frac{x}{5} - x^3 - y^5\right)e^{-(x^2+y^2)} - \left(\frac{1}{3}\right)e^{-[(x+1)^2+y^2]}.$$

The *Peaks* function is a function of two variables, obtained by translating and scaling Gaussian distributions (Reutskiy and Chen 2006). It has multiple peaks and valleys (Fig. 5(a)). Local maxima are located at $(0, 1.58)$, $(-0.46, -0.63)$, and $(1.28, 0)$ with different peak function values.

Rastrigin's function:

$$J_2(x,y) = 20 + \left(x^2 - 10\cos(2\pi x) + y^2 - 10\cos(2\pi y)\right).$$

The Rastrigin's function (Fig. 5(b)) was first proposed by Rastrigin (Törn and Zilinskas 1989) as a 2-dimensional function and has been generalized by Mühlenbein et al. in (Mühlenbein et al. 1991). This function presents a fairly difficult problem due to its large search space and its large number of local minima and maxima. For instance, while the Peaks function ($J_1(x,y)$) consists of only three peaks, the Rastrigin's ($J_2(x,y)$) function consists of as many as 100 peaks in the considered range. The Rastrigin's function has been used as a benchmark function to test several algorithms designed to solve global and multimodal function optimization problems (Fevrier and Patricia 2007; Muller et al. 2002).

Circles function:

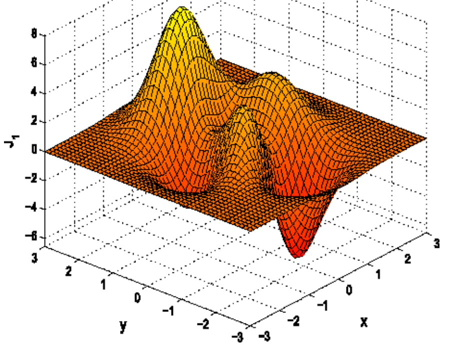
$$J_3(x,y) = \left(x^2+y^2\right)^{0.25}\left(\left(\sin^2\left(50\left(x^2+y^2\right)^{0.1}\right)\right) + 1.0\right).$$

The Circles function (Muller et al. 2002) contains multiple concentric circles as the regions of local maxima (Fig. 5(c)). Unlike $J_1(x,y)$ and $J_2(x,y)$ functions where each peak is a single point, the circular lines of local peaks present an infinite-peaks case.

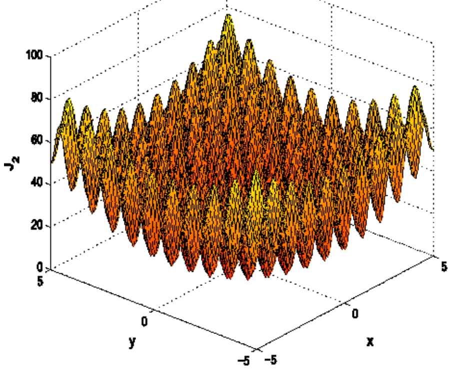
Staircase function:

$$J_4(x,y) = 25 - \lfloor x \rfloor - \lfloor y \rfloor.$$

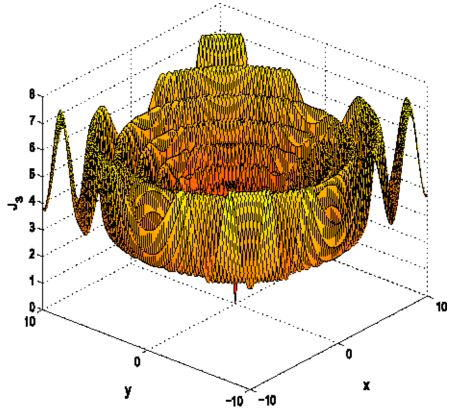
The Staircase function (Dréo and Siarry 2004) contains a series of stairs as shown in Fig. 5(d). In J_4, $\lfloor x \rfloor$ is the floor function and is defined as the nearest integer less than x. Peaks are flat regions that are surrounded by step-discontinuities. The Staircase function presents a case where the function value increases in discrete steps as we move, from one stair to the next, toward the tallest stair (global peak region).



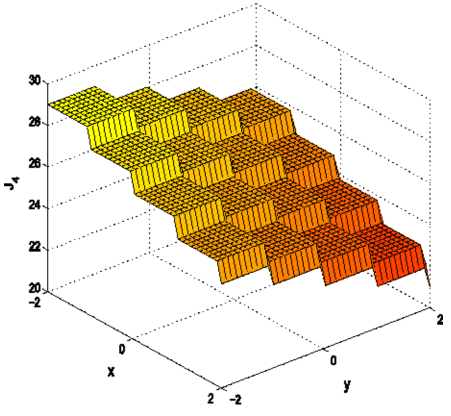
(a) $J_1(x, y)$



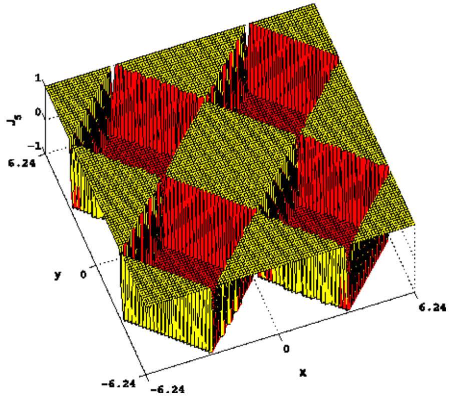
(b) $J_2(x, y)$



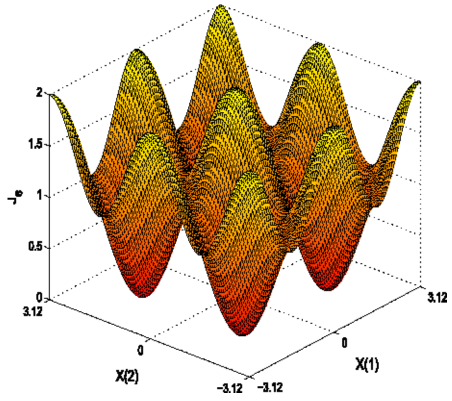
(c) $J_3(x, y)$



(d) $J_4(x, y)$

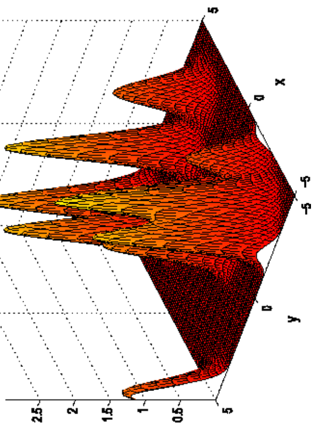


(e) $J_5(x, y)$

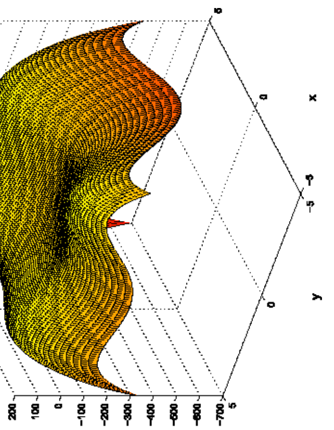


(f) $J_6(x, y), m = 2$

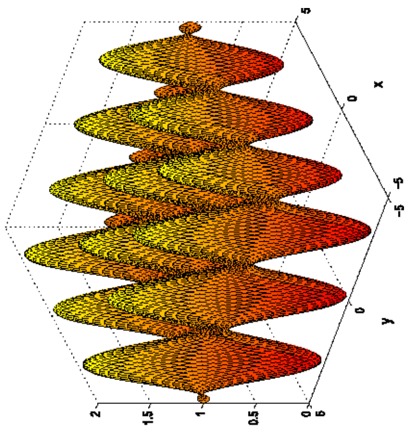
Fig. 5 The multimodal function profiles chosen to test GSO: (**a**) Peaks (J_1), (**b**) Rastrigin's (J_2), (**c**) Circles (J_3), (**d**) Staircase (J_4), (**e**) Plateaus (J_5), (**f**) Equal-peaks-A (J_6), (**g**) Random-peaks (J_7), (**h**) Himmelblau's (J_8), (**i**) Equal-peaks-B (J_9)



(g) $J_7(x, y)$



(h) $J_8(x, y)$



(i) $J_9(x, y)$

Fig. 5 (*Continued*)

Plateaus function:

$$J_5(x, y) = \operatorname{sign}\big(\cos(x) + \cos(y)\big).$$

The Plateaus function contains multiple plateaus (Fig. 5(e)) and is similar to Ackley's Plateau function described in (Singh et al. 2004). Even though the Plateaus function is similar to the Staircase function in terms of the nature of peaks (both have flat regions as peaks), the plateaus in the $J_5(x, y)$ function have equal objective function values, thereby providing no uphill direction.

Equal-peaks-A function:

$$J_6(X) = \sum_{i=1}^{m} \cos^2\big(X(i)\big).$$

In $J_6(X)$, $X \in \Re^m$ is a position vector in the m-dimensional search space with $m \geq 2$. All local maxima of the Equal-peaks-A function have equal function values (Fig. 5(f)). We use the function $J_6(X)$ for tests on higher dimensional search spaces (results for $m = 2, 3, 4$, and 5 are reported).

Random-peaks function:

$$J_7(x, y) = \sum_{i=1}^{Q} a_i \exp\big(-b_i\big((x - x_i)^2 + (y - y_i)^2\big)\big),$$

where Q represents the number of peaks and (x_i, y_i) represents the location of each peak. The values of a_i, b_i, x_i, and y_i are generated according to the following equations:

$$a_i = 1 + 2\vartheta; \qquad b_i = 2 + \vartheta; \qquad x_i = y_i = -5 + 10\vartheta,$$

where ϑ is uniformly distributed within the interval [0, 1].

Himmelblau's function:

$$J_8(x, y) = 200 - \big(x^2 + y^2 - 11\big)^2 - \big(x + y^2 - 7\big)^2.$$

The modified Himmelblau's function (Brits et al. 2002) has four maxima of equal objective function value (200) at $(3, 2)$, $(-3.779, -3.283)$, $(-2.805, 3.131)$, and $(3.584, -1.848)$.

Equal-peaks-B function:

$$J_9(x, y) = \cos^2(x) + \sin^2(y).$$

All local maxima of the Equal-peaks-B function (Parsopoulos and Vrahatis 2004) have equal function values. The only difference between $J_9(x, y)$ and $J_6(x, y)$ is a phase shift of $\frac{\pi}{2}$ in the landscape from each other.

3.2 Performance measures

To evaluate the performance of GSO, we use the fraction of peaks captured and the average minimum distance to the peak locations. We consider a glowworm to be located at a peak when its position is within a small ϵ-distance from the peak and we assume that a peak is captured when at least three glowworms are located at the peak. The value $\epsilon = 0.05$ is chosen for all experiments. The average minimum distance to the peak locations $dmin_{\text{av}}$ is given by:

$$dmin_{\text{av}} = \frac{1}{n}\sum_{i=1}^{n} \min_{\{1,\dots,Q\}}\{\delta_{i1},\dots,\delta_{iQ}\}, \tag{8}$$

where $\delta_{ij} = \|X_i - S_j\|$, $i = 1,\dots,n$, $j = 1,\dots,Q$, X_i and S_j are the locations of glowworm i and peak j, respectively, and Q is the number of peak locations.

4 Some basic issues and simulation results

This section addresses some basic issues such as variable vs. fixed neighborhood range, effect of step size, and the performance of GSO on functions with special characteristics, through detailed simulations of selected test functions. The values of the algorithm parameters ($\rho, \gamma, \beta, n_t, s$, and ℓ_0) are kept fixed for all experiments (Table 1). These values have been obtained from a large number of numerical experiments. Note that these are not parameters to be selected as they are fixed irrespective of the problem chosen. Further studies are carried out in Sect. 5 in order to show that the values chosen for these algorithm parameters work well for a wide range of simulation scenarios and that only n and r_s are parameters that influence the algorithm's behavior (in terms of the total number of peaks captured). The search space size considered for each function, the values of parameters n and r_s, the maximum number of iterations, and the corresponding computation times are given in Table 2. Note that the computation times depend on several factors such as the search space size, the nature of peaks, the number of peaks, and the number of glowworms used.

4.1 Variable vs. constant neighborhood range

The Peaks function (J_1) is used in the following set of experiments. A set of 50 glowworms is randomly deployed in a two-dimensional search space of size 6×6 square units.

Constant neighborhood range: As a first step, the radial range r_d^i of each glowworm is kept constant in order to characterize the sensitivity of the number of peaks detected to the size of the neighborhood. As noted earlier, the neighborhood range greatly influences the

Table 2 The values of n and r_s used for various test functions. The number of iterations and the corresponding computation times are given in the last two columns, respectively

Function	Search space size	n	r_s	No. of iterations	Computation time (s)[†]
J_1	$(-3,3)\times(-3,3)$	50	3, 2, 1.8, 1.5	200	2.67
J_2	$(-5,5)\times(-5,5)$	1500	2	500	1534
J_3	$(-10,10)\times(-10,10)$	1000	4	500	584
J_4	$(-2,2)\times(-2,2)$	1000	0.75	500	557
J_5	$(-2\pi,2\pi)\times(-2\pi,2\pi)$	1000	3	500	883
J_6	$\prod_{i=1}^{m}(-\pi,\pi)$	200–6000	1.25, 1.5, 4, 5	150	9.87
			(for $m = 2, 3, 4, 5$)	(for $m = 2$)	(for $m = 2$)

[†]The algorithm is coded using Matlab 7.0 and all simulations are implemented on a 2 GB RAM, 3 GHz Intel P4 machine with a Windows XP operating system

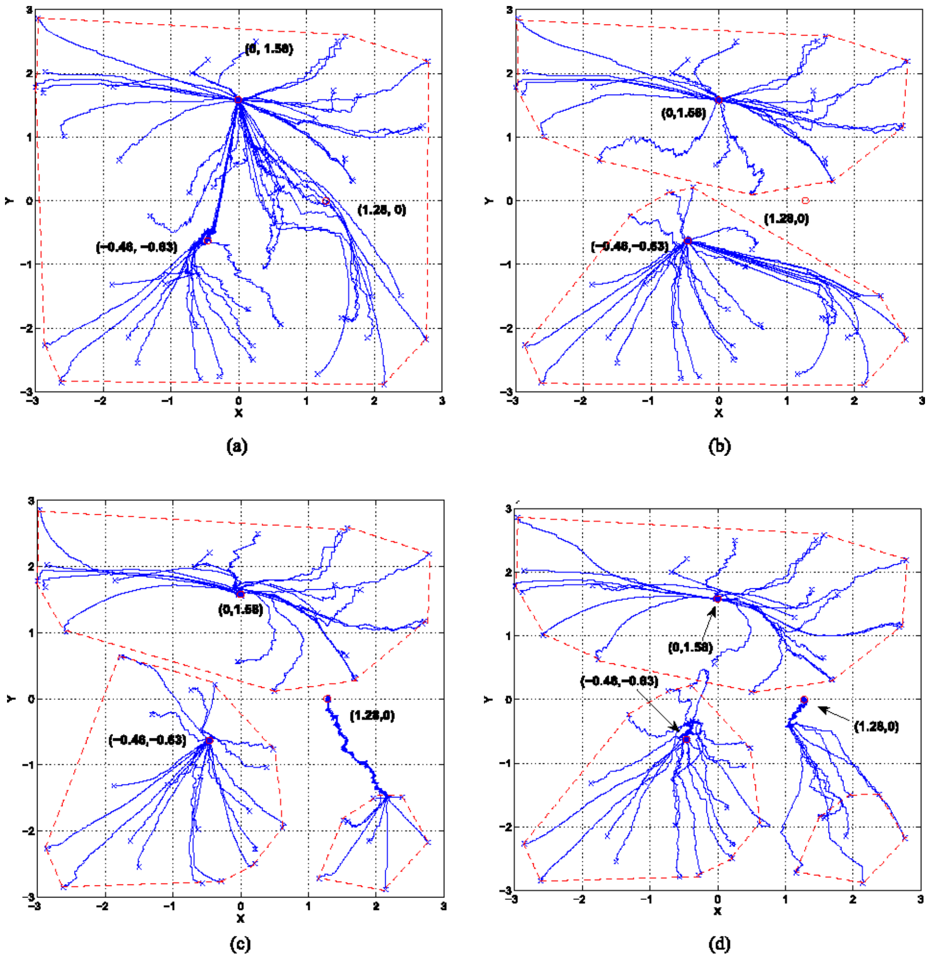


Fig. 6 Peaks function (J_1): trajectories (200 iterations) followed by the glowworms: they start from an initial random location (*cross*) and they end up at one of the peaks. (**a**) The neighborhood range is kept constant with $r_d^i = 2$: only one peak is captured. (**b**) The neighborhood range is kept constant with $r_d^i = 1.8$: only two peaks are captured. (**c**) The neighborhood range is kept constant with $r_d^i = 1.5$: all three peaks are captured. (**d**) Neighborhood range is made adaptive according to (4), with $r_d^i(0) = 3$: all three peaks are captured. The leapfrogging effect can be observed in the movements of glowworms that reach the peak location $(1.28, 0)$ in (**c**) and (**d**)

determination of various peaks. When the range is more than 2, all the glowworms move to the global maximum. Figures 6(a), (b), and (c) show the trajectories followed by the glowworms within 200 iterations (they start from an initial random location and end up at one of the peaks), when r_d^i of each glowworm j is kept constant at 2 (only one peak is captured), 1.8 (two peaks are captured), and 1.5 (all three peaks are captured), respectively.

Variable neighborhood range: We observe that the number of peaks captured is a function of the neighborhood range r_d^i. Since we assume that a priori information about the objective function (e.g., the number of maxima and minima) is not available, the neighborhood range was made a varying parameter. Figure 6(d) shows the trajectories followed by

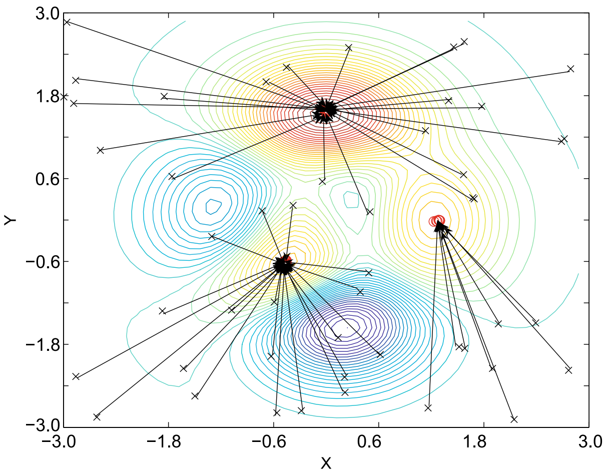


Fig. 7 The equi-contour plot with the initial placement and final location of the glowworms at the peak locations

the glowworms when the neighborhood range varies according to (4) at each iteration t. During this simulation, a value of $r_d^i(0) = r_s = 3$ is chosen for each glowworm i. Note that all the peaks are detected within 200 iterations. In particular, 23, 19, and 8 glowworms get located at the maxima of $(0, 1.58)$, $(-0.46, -0.63)$, and $(1.28, 0)$, respectively. All glowworms located at the same location have the same luciferin value.[3] Figure 7 shows the initial placement and location of all the glowworms on the equi-contour plot of the objective function. Figures 8(a) and (b) show the number of neighbors and the neighborhood range of glowworm 12, respectively. Initially, the value of $r_d^{12}(t)$ decreases as the glowworm 12 has more than five neighbors ($n_t = 5$). Note that between $t = 74$ and $t = 94$, the neighborhood range rises sharply up to the maximum sensing range r_s. This is because the number of glowworms that are within the neighborhood range of glowworm 12 is much higher than n_t, but among them, those having a brighter luciferin value are fewer than n_t (Fig. 8(b)). However, at $t = 94$, the neighborhood range starts shrinking again.

Experiments with the Rastrigin's function (J_2) give similar results. Here, a set of 1500 agents is deployed in a 10×10 units region centered in $(0, 0)$ that contains a total number of 100 peaks (Fig. 5(b)). Figures 9(a) and (b) show the trajectories (500 iterations) followed by the glowworms from initial random locations to one of the peaks, for the constant and adaptive neighborhood cases, respectively. Note that 92% of the peaks are captured in the variable neighborhood case ($r_d^i(0) = r_s = 2$), in comparison to a mere 8% in the constant neighborhood case ($r_d^i(t) = 2, \forall\, t$).

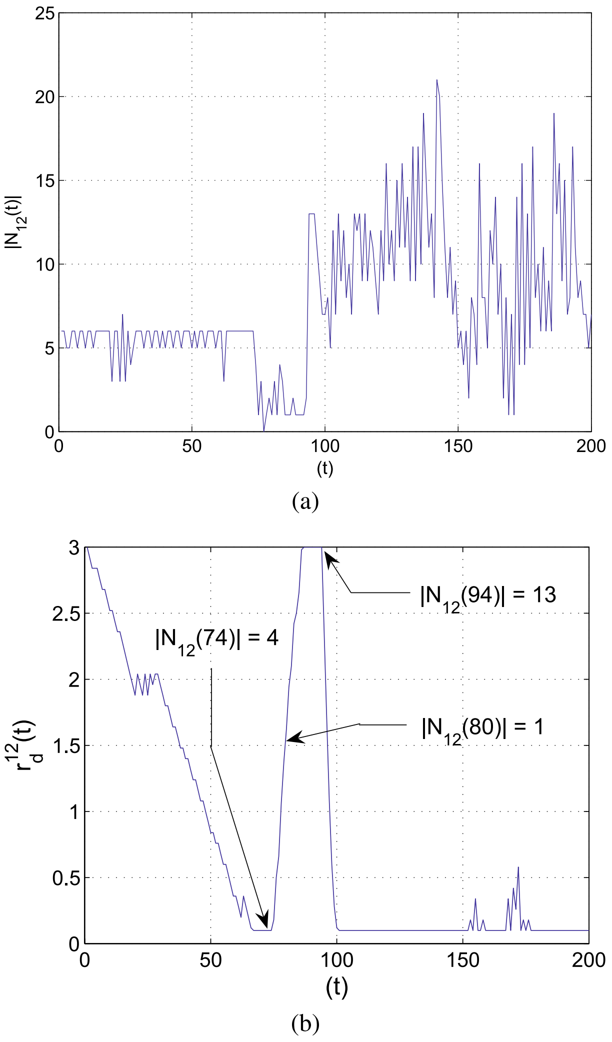
These experiments suggest that the use of a variable neighborhood, instead of a constant one, significantly improves the ability of the algorithm to compute multiple peaks.

4.2 Effect of step size on convergence

To obtain solutions within the desired ϵ-distance from the peaks, the step size s should be smaller than the value ϵ. However, this may lead to a slow convergence of the algorithm.

[3] According to Theorem 2, the value ℓ_j, for all glowworms j located at a peak location X_i^*, is given by $(\frac{\gamma}{\rho})J_i^*$. The luciferin values at the three peak locations $(0, 1.58)$, $(-0.46, -0.63)$, and $(1.28, 0)$ are 12.15 $(= (0.6/0.4) \times 8.1)$, 5.67 $(= (0.6/0.4) \times 3.78)$, and 5.4 $(= (0.6/0.4) \times 3.6)$, respectively. These values, obtained from simulations, exactly agree with the analytical values given by Theorem 2.

Fig. 8 (**a**) Plot of $|N_{12}(t)|$ as a function of iteration number t. (**b**) Plot of $r_d^{12}(t)$ as a function of iteration number t



This issue can be resolved by starting the algorithm with a coarse (large) step size and progressively reducing the step size to a value smaller than ϵ. For instance, in an experiment with the peaks function (J_1), when the step size $s(t)$ was updated as $s(t) = q^t s(0)$, with $q = 0.96$ and $s(0) = 0.2$, convergence was achieved within 50 iterations ($s(50) = 0.026$) as opposed to 200 iterations in the constant step size case. This result is only an illustrative example and various models for the adaptive step size have to be explored in order to make it work on a wide variety of problems. However, note that the number of peaks captured is not affected by the step size.

The step size is also a function of the size of the search space. For relatively large search spaces, we need to start with a larger step size. However, there is a chance of missing closely spaced peaks depending on the number of peaks in a given area. In these cases, an additional search procedure in each local region of a peak captured with a smaller step size could reveal the presence of more peaks.

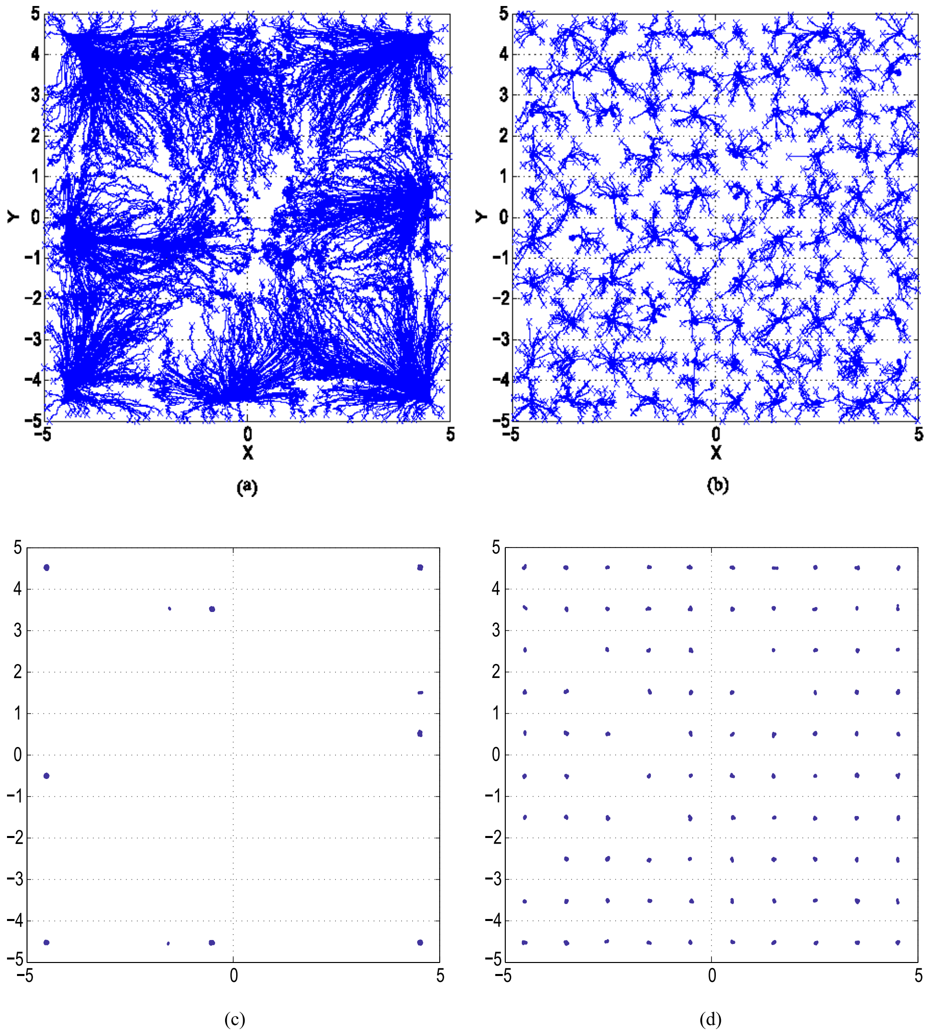


Fig. 9 Rastrigin's function (J_2): trajectories (500 iterations) followed by the glowworms: they start from an initial random location (*cross*) and end up at one of the peaks. (**a**) The neighborhood range r_d^i is kept constant. (**b**) The neighborhood range r_d^i is made adaptive. (**c**) Locations of groups of glowworms at various peaks in the constant neighborhood case. (**d**) Locations of groups of glowworms at various peaks in the variable neighborhood case

4.3 Performance of GSO on special functions

In this section we consider a few functions for which optimization algorithms have difficulties in capturing multiple maxima.

Circles function (J_3): The Circles function has connected maxima in the form of concentric circles (Fig. 5(c)). The initial placement consists of the random deployment of $n = 1000$ glowworms in the search space $(-10, 10) \times (-10, 10)$. The trajectories resemble the concentric-circle contours of local maxima, as shown in Fig. 10. However, rather than spreading uniformly on the circles, the glowworms converge to distinct points on the

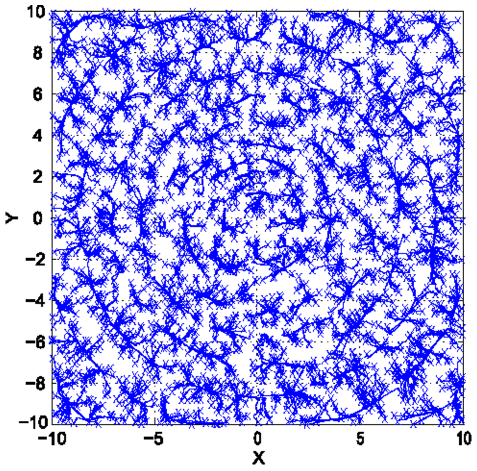


Fig. 10 Trajectories (500 iterations) followed by the glowworms for the Circles (J_3) function

circles. This mainly occurs as a result of the glowworms seeking to move toward relatively brighter neighbors even as the associated differences in luciferin levels may be very small. Some amount of spreading can be achieved by disabling the attraction between two glowworms whenever the difference in their luciferin values is less than a small threshold value.

Staircase function (J_4): The Staircase function (Fig. 5(d)) has connected local maxima at different levels with discontinuity in the function. It contains 16 regions at 7 different heights in the square $(-2, 2) \times (-2, 2)$. The initial locations of the glowworms, the trajectories followed by the glowworms, and the final solution are shown in Figs. 11(a), (b), and (c), respectively. Different square regions are marked according to the increasing order of their function values. Note that regions marked with the same number have equal function values. All the glowworms initially located in region 7 remain stationary as they are located in the flat region of global maxima. Note that glowworms in the interior locations (except those that are very close to or on the edges) of all other regions move toward and settle on the edges of the next higher peak-regions. Refer to Fig. 11(c) to observe this pattern.

Even though the Staircase function has step discontinuities, it provides an uphill direction toward the highest level in discrete steps whose size is equal to the width of each stair. Therefore, if the values of the sensor range r_s and of the number of glowworms n are such that there is sufficient connectivity among the swarm members, all the glowworms will reach the highest level. However, the values of r_s and n can be tuned (as is the case in the above simulation result) so that the glowworms detect the presence of other stairs. From a global optimization point of view (i.e., when the goal is to compute only the global peak), it appears that GSO does not perform well for optimization with J_4. However, in the context of this paper, since multiple local peaks are being sought, the GSO's performance on the J_4 function is indeed good.

Plateaus function (J_5): We use the function $J_5(x, y)$ to test the behavior of the algorithm on a profile that has multiple plateaus with equal function values (Fig. 5(e)). The plateaus function has connected local maxima at the same level with discontinuity in the function. One thousand glowworms are deployed in the square $(-2\pi, 2\pi) \times (-2\pi, 2\pi)$ (Fig. 12(a)). Figures 12(b) and (c) show the trajectories followed by the glowworms and the final location of the glowworms, respectively. Note that all the glowworms, with the exception of a very few that are located in the minima regions of the function, settle on the plateau-regions, after the solution is reached. The test results on the J_5 function provide an interesting intuition regarding the working of the algorithm. For instance, we notice from Fig. 12(b) that

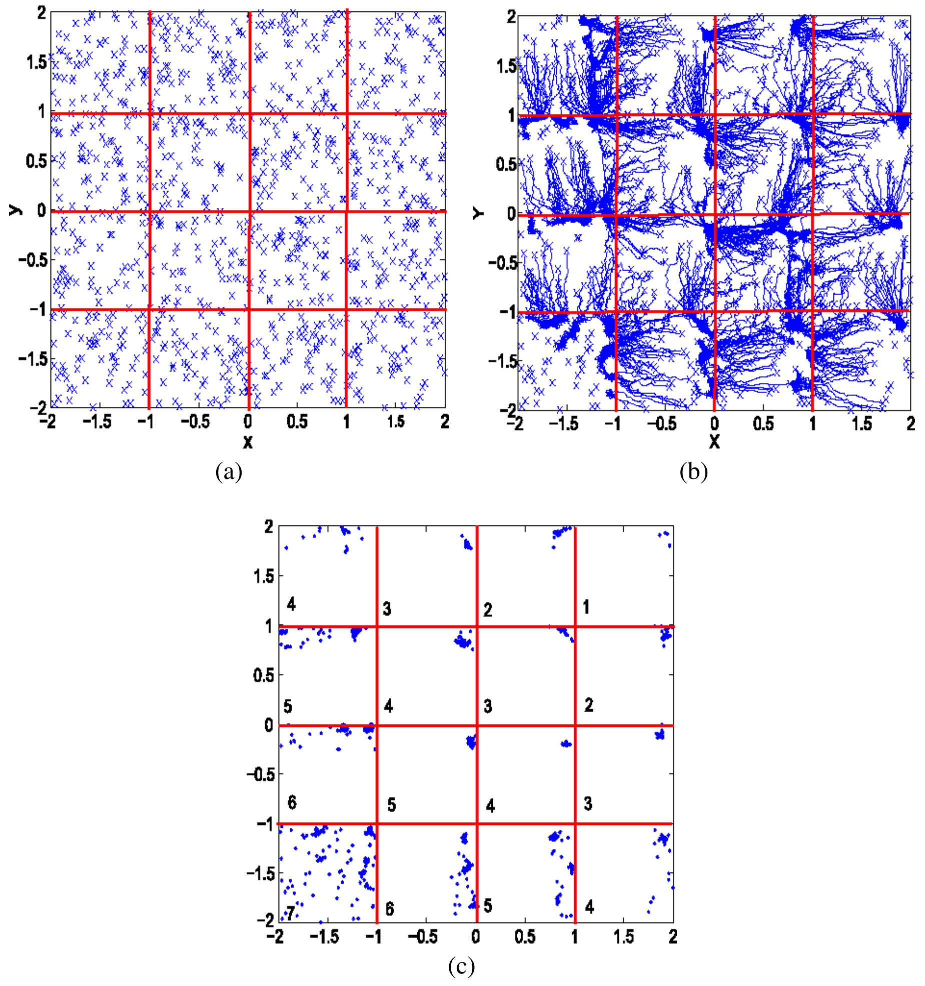
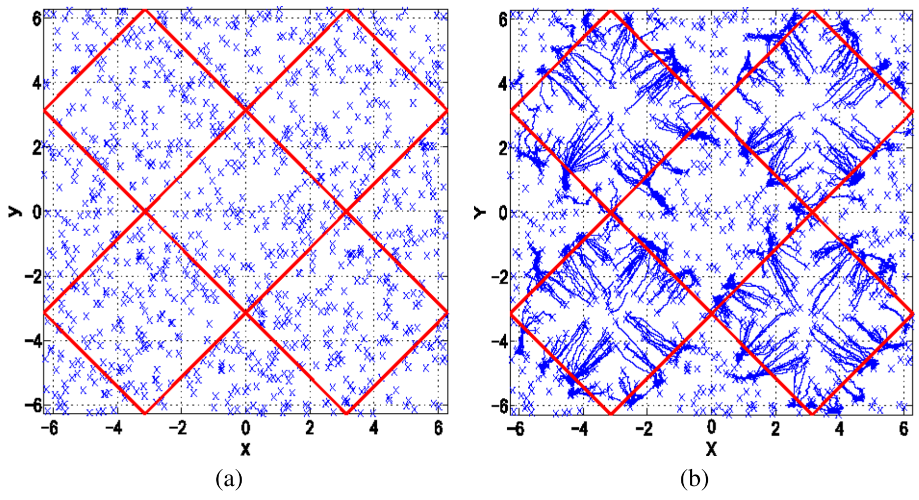


Fig. 11 Test results on the Staircase (J_4) function. (**a**) Initial placement of glowworms. (**b**) Trajectories (500 iterations) followed by the glowworms. (**c**) Locations of groups of glowworms after 500 iterations

glowworms located in the interior region of a plateau are always stationary. This could be attributed to their equal luciferin-levels. Note from Fig. 12(b) that most glowworms that climb the gradient steps settle on the plateau edges. However, a few glowworms, after climbing the plateaus, continue moving into the interior regions of the plateaus until their luciferin values reach a steady state and become equal to that of the glowworms located in the interior region.

5 Parameter selection and performance evaluation

The number of parameters and how much their "optimal" values are dependent on the problem have a major impact on the practical applicability of optimization algorithms. A good algorithm would consist of a small number of problem-specific parameters and a set of al-



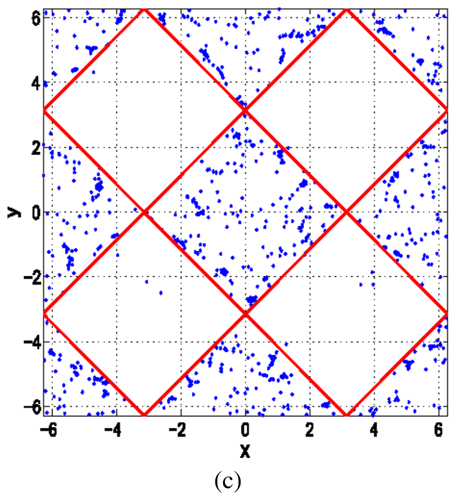


Fig. 12 Test results on the Plateaus (J_5) function. (**a**) Initial placement of glowworms. (**b**) Trajectories (500 iterations) followed by the glowworms. (**c**) Locations of groups of glowworms after 500 iterations

gorithm parameters tuned to fixed values that yield optimal performance over a wide range of problems.

In the set of experiments carried out in Sect. 4, fixed values of $\rho, \gamma, \beta, n_t, s$, and ℓ_0 worked well for all the five function-profile cases. Therefore, as mentioned earlier, these quantities are algorithm parameters that will remain the same and do not need to be specifically tuned for every problem. Thus, only n and r_s need to be selected. Fortunately, their selection is rather straightforward. We show that the choice of these parameters has some influence on the performance of the algorithm, in terms of the total number of peaks captured.

5.1 Effect of n and r_s on algorithm performance

In the following set of experiments, we study the influence of n and r_s on algorithm performance by carrying out a full factorial analysis on the following functions: Peaks (J_1), Rastrigin's (J_2), Equal-peaks-A (J_6), and Random-peaks (J_7).[4]

In particular, for each combination of n and r_s, the total number of peaks captured is monitored over a set of 30 trials. The average number of peaks captured and the standard deviation values for each function are shown in Tables 3–6. The Peaks function has a total number of 3 peaks. At $n = 10$, the number of peaks captured increases as r_s increases. When $n \geq 20$, good performance is observed for $r_s \geq 2.5$ (Table 3). The Rastrigin's function has a total number of 16 peaks in the chosen search space ($[-2, 2] \times [-2, 2]$). For fixed values of n, the best performance was observed for values of r_s between 0.5 and 0.75 (Table 4). The Equal-peaks-A function has a total number of 9 peaks in the chosen search space ($[-4, 4] \times [-4, 4]$). For fixed values of n, the best performance was observed for values of r_s between 2 and 2.5 (Table 5). The Random-peaks function has a total number of 10 peaks in the chosen search space ($[-5, 5] \times [-5, 5]$). For fixed values of n, the best performance was observed for $r_s = 2$ (Table 6). For all functions, at any fixed value of r_s, we observe that the peak capture level increases as n increases.

We can deduce four major conclusions from the above observations and the results shown in Tables 3–6.

(1) When a small value of r_s relative to the size of the search space is used, a significant fraction of glowworms may get isolated and become stationary at every iteration. Therefore, only a large number of glowworms can ensure connectivity among the swarm members, leading to the computation of all the multiple peaks.
(2) For large values of r_s, the number of peaks that influence the movements of a glowworm increases when r_s increases. This may result in a few peak locations being overpopulated by glowworms, while other peaks remain undetected at the final iteration, unless a large number of glowworms is used to ensure sufficient allocation of glowworms to all the peak locations.
(3) For a fixed value of n, the number of peaks captured increases with the increase in r_s, reaches a maximum at some r_s, and decreases with further increase in r_s.
(4) There exists a sensor range r_s^* that minimizes the number of necessary agents without affecting final performance in terms of reached peaks and that is invariant across different peak-capture levels. However, in order to increase the peak-capture level it is necessary to increase the number of glowworms.

From conclusion 4, note that the value of r_s^* is approximately the same for different values of n. Therefore, the value of r_s^* is initially found by keeping n at a lower value that takes relatively less computational time. Then, we set $r_s = r_s^*$ and keep increasing the value of n in order to obtain higher peak-capture levels.

5.2 Fixed algorithm parameters

The quantities $n_t, s, \ell_0, \beta, \rho$, and γ are algorithm parameters for which appropriate values have been determined based on extensive numerical experiments and are kept fixed in this

[4]The Circular, Stair-case, and Pleateaus functions have either lines or regions rather than points as peaks, making it difficult to use the number of peaks captured as a performance metric. This is why we restrict ourselves to J_1, J_2, J_6, and J_7 functions for these experiments.

Table 3 Results on the Peaks function: Average number of peaks captured, with standard deviation values, for various combinations of n and r_s (30 trials)

$n \| r_s$	0.5	1.0	1.5	2.0	2.5	3.0	3.5	4.0	4.5	5.0
10	0.0 ± 0.2	0.1 ± 0.3	0.3 ± 0.5	0.4 ± 0.5	0.5 ± 0.5	0.5 ± 0.5	0.6 ± 0.5	0.6 ± 0.5	0.6 ± 0.5	0.7 ± 0.5
20	0.1 ± 0.3	0.4 ± 0.5	0.7 ± 0.7	0.7 ± 0.5	0.9 ± 0.5	0.8 ± 0.5	0.8 ± 0.5	0.8 ± 0.5	0.8 ± 0.5	0.8 ± 0.5
30	0.4 ± 0.6	0.8 ± 0.8	1.3 ± 0.8	1.3 ± 0.6	1.3 ± 0.5	1.2 ± 0.6	1.1 ± 0.5	1.1 ± 0.4	1.1 ± 0.5	1.2 ± 0.5
40	0.6 ± 0.6	1.0 ± 0.7	1.3 ± 0.7	1.5 ± 0.6	1.4 ± 0.5	1.6 ± 0.6	1.4 ± 0.7	1.3 ± 0.5	1.5 ± 0.6	1.4 ± 0.5
50	0.8 ± 0.7	1.4 ± 0.8	1.7 ± 0.7	2.0 ± 0.6	2.1 ± 0.6	2.2 ± 0.7	2.1 ± 0.8	2.2 ± 0.6	2.1 ± 0.6	2.1 ± 0.7
60	0.6 ± 0.7	1.6 ± 0.8	1.8 ± 0.7	2.0 ± 0.5	2.3 ± 0.6	2.1 ± 0.8	2.4 ± 0.6	2.2 ± 0.7	2.3 ± 0.6	2.3 ± 0.6
70	1.0 ± 0.8	1.8 ± 0.7	2.0 ± 0.7	2.2 ± 0.7	2.6 ± 0.6	2.4 ± 0.5	2.4 ± 0.6	2.5 ± 0.5	2.4 ± 0.7	2.5 ± 0.5
80	1.2 ± 0.8	2.0 ± 0.7	2.2 ± 0.5	2.3 ± 0.5	2.6 ± 0.5	2.6 ± 0.5	2.7 ± 0.4	2.5 ± 0.6	2.6 ± 0.6	2.7 ± 0.5
90	1.2 ± 0.8	1.8 ± 0.9	2.3 ± 0.6	2.6 ± 0.6	2.8 ± 0.5	2.7 ± 0.6	2.7 ± 0.4	2.7 ± 0.5	2.8 ± 0.4	2.6 ± 0.5
100	1.5 ± 0.9	2.3 ± 0.6	2.4 ± 0.6	2.5 ± 0.6	2.8 ± 0.4	2.8 ± 0.4	2.8 ± 0.5	2.7 ± 0.5	2.8 ± 0.5	2.8 ± 0.4

Table 4 Results on the Rastrigin's function: Average number of peaks captured, with standard deviation values, for various combinations of n and r_s (30 trials)

$n \| r_s$	0.25	0.5	0.75	1.0	1.25	1.5	1.75	2.0
50	0.3 ± 0.5	3.0 ± 1.4	4.6 ± 1.2	4.2 ± 1.1	4.0 ± 0.7	3.5 ± 0.7	3.4 ± 0.7	3.0 ± 0.8
100	1.4 ± 1.0	7.2 ± 1.5	8.7 ± 1.4	6.6 ± 1.2	5.2 ± 1.1	4.8 ± 0.7	4.7 ± 0.9	4.2 ± 1.2
150	3.8 ± 1.4	11.2 ± 1.9	11.2 ± 1.5	7.6 ± 1.3	6.3 ± 1.1	6.0 ± 0.9	6.1 ± 1.1	6.2 ± 1.1
200	5.6 ± 1.9	13.3 ± 1.3	13.0 ± 1.3	8.2 ± 1.2	8.5 ± 1.9	8.8 ± 1.6	9.4 ± 1.7	8.6 ± 1.7
250	6.8 ± 1.8	14.8 ± 1.1	14.1 ± 1.2	10.1 ± 1.1	12.7 ± 1.3	12.5 ± 1.4	12.2 ± 1.4	11.6 ± 1.7
300	8.4 ± 2.1	15.3 ± 0.8	14.7 ± 1.1	11.6 ± 1.3	14.1 ± 1.1	14.1 ± 1.4	14.4 ± 1.0	14.4 ± 1.2
350	10.4 ± 1.7	15.6 ± 0.6	15.2 ± 0.8	13.2 ± 1.2	15.2 ± 1.1	15.4 ± 0.7	15.6 ± 0.6	15.4 ± 0.7

Table 5 Results on the Equal-peaks-A function: Average number of peaks captured, with standard deviation values, for various combinations of n and r_s (30 trials)

$n\|r_s$	0.5	1.0	1.5	2.0	2.5	3.0	3.5	4.0
50	0.7 ± 0.9	1.6 ± 1.3	2.6 ± 1.7	2.2 ± 1.7	2.9 ± 1.2	2.4 ± 1.2	1.2 ± 0.9	2.0 ± 0.9
100	2.2 ± 2.8	4.6 ± 2.4	4.3 ± 2.6	5.4 ± 1.8	5.8 ± 1.4	4.8 ± 1.1	3.2 ± 0.9	3.0 ± 1.0
150	3.5 ± 3.8	4.8 ± 2.3	6.8 ± 1.8	7.7 ± 1.4	7.4 ± 1.0	6.0 ± 0.7	4.7 ± 1.2	4.9 ± 1.2
200	2.6 ± 1.1	6.9 ± 1.2	8.1 ± 0.7	8.3 ± 0.6	8.1 ± 0.8	7.3 ± 1.0	6.4 ± 1.0	6.4 ± 1.5
250	3.8 ± 2.3	8.2 ± 0.7	8.6 ± 0.6	8.7 ± 0.5	8.6 ± 0.5	8.1 ± 0.7	7.8 ± 1.0	7.7 ± 0.9
300	5.2 ± 2.4	8.6 ± 0.5	8.8 ± 0.4	8.9 ± 0.3	8.9 ± 0.3	8.7 ± 0.6	8.4 ± 0.6	8.4 ± 0.7
350	4.3 ± 1.3	8.9 ± 0.3	8.9 ± 0.2	8.9 ± 0.2	8.9 ± 0.3	8.6 ± 0.5	8.6 ± 0.5	8.7 ± 0.5

Table 6 Results on the Random-peaks function: Average number of peaks captured for various combinations of n and r_s (30 trials)

$n\|r_s$	0.5	1.0	1.5	2.0	2.5	3.0	3.5	4.0	4.5	5.0
50	0.1 ± 0.2	0.4 ± 0.6	1.3 ± 1.0	2.2 ± 1.1	2.3 ± 1.1	2.4 ± 1.0	2.2 ± 1.0	1.9 ± 1.1	1.9 ± 0.6	2.0 ± 0.7
100	0.0 ± 0.2	1.6 ± 1.1	3.6 ± 1.2	3.8 ± 0.9	3.6 ± 0.8	3.5 ± 1.0	3.2 ± 1.0	3.5 ± 1.2	3.4 ± 1.0	3.3 ± 0.9
150	0.2 ± 0.5	3.1 ± 1.3	5.1 ± 1.0	5.4 ± 1.2	4.8 ± 1.0	4.8 ± 0.9	4.6 ± 1.0	4.6 ± 0.8	4.5 ± 1.0	4.6 ± 0.9
200	0.8 ± 0.8	4.6 ± 1.1	6.1 ± 1.1	6.3 ± 0.8	6.0 ± 0.8	5.7 ± 0.9	5.8 ± 0.8	5.4 ± 1.0	5.6 ± 0.8	5.4 ± 1.0
250	1.4 ± 1.0	6.3 ± 1.5	7.1 ± 1.3	7.2 ± 1.3	7.1 ± 1.1	6.6 ± 0.9	6.7 ± 1.1	6.6 ± 1.0	6.4 ± 1.1	6.6 ± 1.3
300	1.7 ± 1.2	6.9 ± 1.3	7.6 ± 0.8	7.5 ± 1.0	7.2 ± 1.0	7.0 ± 1.0	6.9 ± 0.8	6.7 ± 1.2	6.8 ± 1.1	6.6 ± 1.1
350	2.1 ± 1.1	7.6 ± 1.1	8.3 ± 1.0	8.4 ± 1.0	7.9 ± 1.0	7.8 ± 1.1	7.8 ± 1.0	7.6 ± 1.1	7.6 ± 1.0	7.6 ± 0.9

paper (Table 1). The neighborhood threshold n_t indirectly controls the number of neighbors of each glowworm by influencing the neighborhood range at each iteration. Whereas a very low value of n_t would not allow enough connectivity for interactions between glowworms, a high value of n_t would result in their strong grouping leading to reduced diversity in the swarm. It was observed that a value of $n_t = 5$ was sufficient to ensure that glowworms are not isolated, yet diversity is maintained between subswarms. The value of step-size s influences the number of iterations in which the peaks are reached by the glowworms and the precision of the solutions. The value of s was selected such that it is very small in relation to the size of the search space. The fixed value of $s = 0.03$ resulted in similar algorithm performance across all the test functions used in this paper. Even though all the glowworms start with the same luciferin value ℓ_0, their luciferin values get updated based on the objective fitness values at their initial positions before they start moving. Therefore, the value of ℓ_0 can be arbitrarily selected. A value of $\ell_0 = 5$ was used in all the simulations. The parameter β affects the rate of change of the neighborhood range. A relatively high value of β would lead to saturation, resulting in the switching of the neighborhood range between its upper and lower limits. Therefore, a small value of β (=0.08) was chosen, which worked well for all the test functions used in this paper. A value $\rho = 0$ renders the algorithm memoryless, where the luciferin value of each glowworm depends only on the fitness value of its current position. However, $\rho \in (0, 1]$ leads to the reflection of the cumulative goodness of the path followed by the glowworms in their current luciferin values. A value of $\rho = 0.4$ showed good performance across different test functions. The parameter γ only scales the function fitness values and the chosen value of γ (=0.6) showed good performance for all test functions used in this paper.

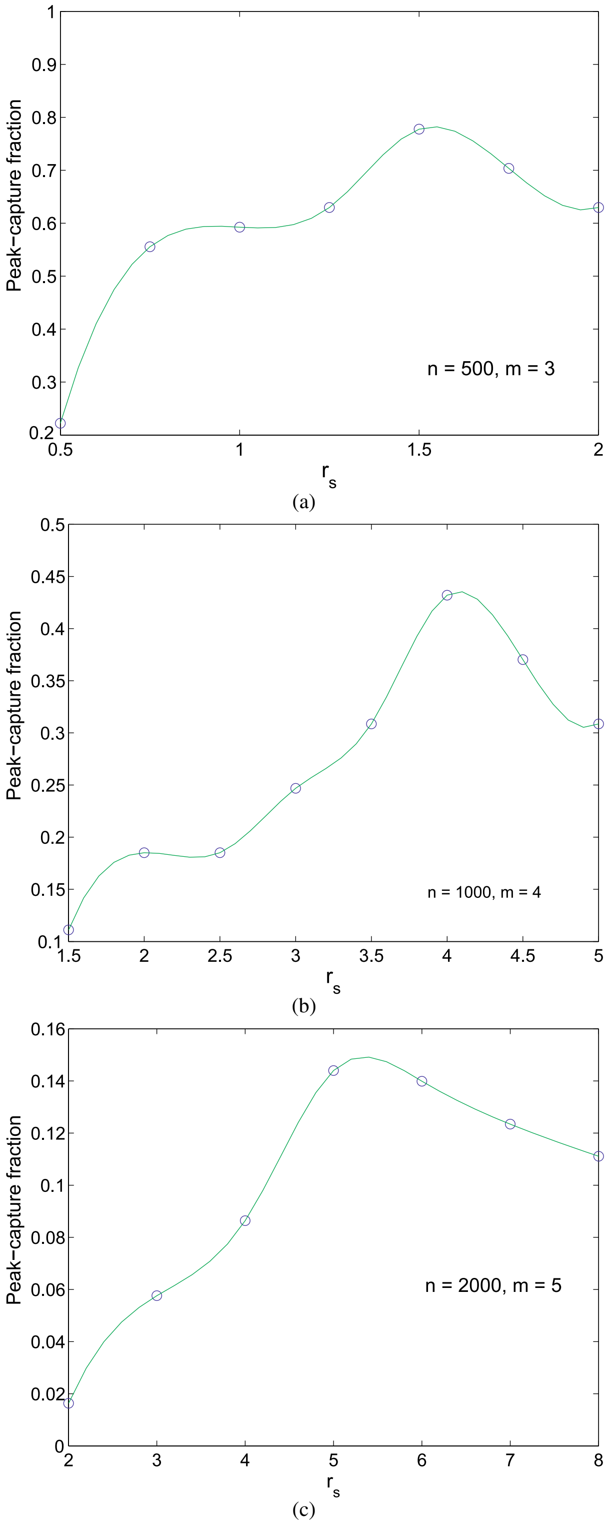
In the above analysis we have studied the influence of the parameters n and r_s on algorithm performance, and provided evidence that algorithm parameters are indeed constant values. Thus we conclude that n and r_s are the only parameters that need to be selected.

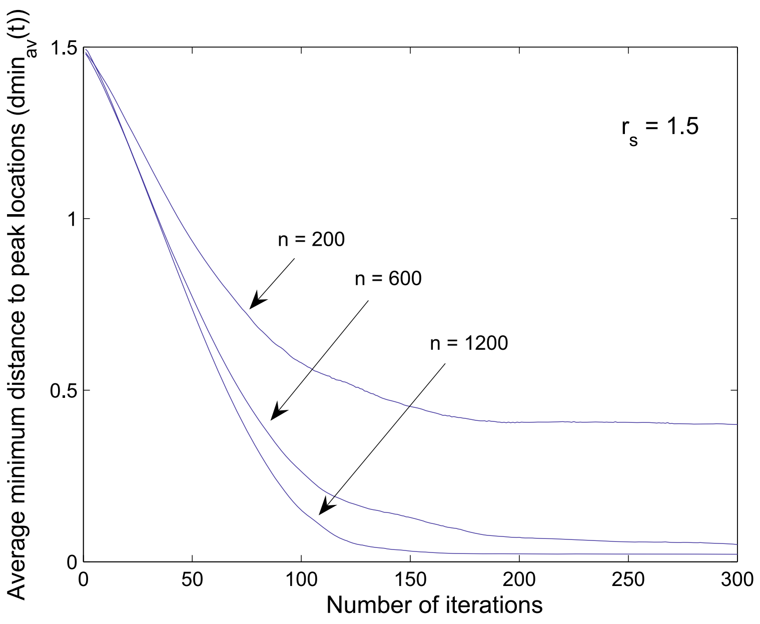
6 Tests in higher dimensional spaces

In the previous section, we conducted experiments for two dimensional cases. Here we show the efficacy of GSO in higher dimensional spaces as well. Again, we keep the algorithm parameters at the same values as given in Table 1. We use the $J_6(X)$ function to test GSO on higher dimensional spaces with up to five dimensions. $J_6(X)$ has a total number of $(2k + 1)^m$ peaks of equal function value, in a search space whose range is $\prod_{i=1}^{m}[-k\pi, k\pi]$, $k = 1, 2, \dots$. In our experiments, we set $k = 1$.

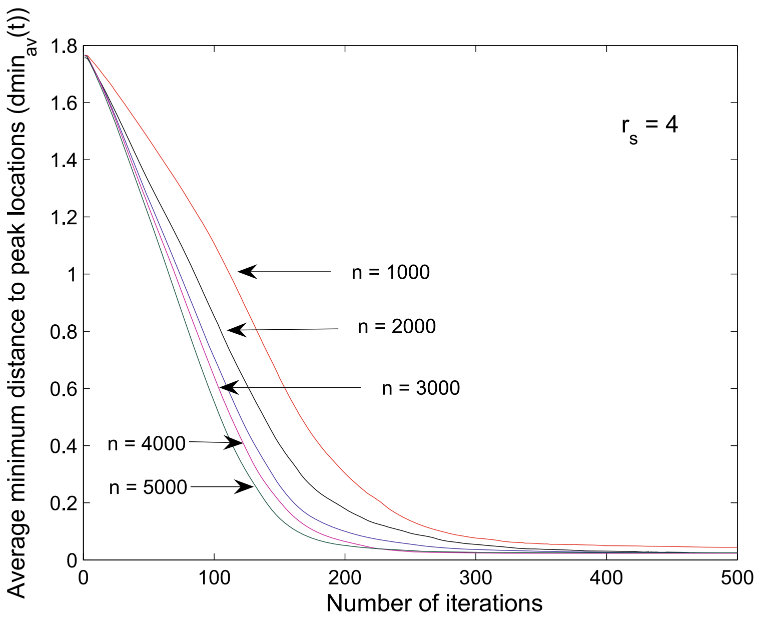
From the results of tests on $J_6(X)$ with $m = 2$, we observe that performance is best at some value of r_s. Therefore, preliminary simulation runs were carried out for each dimension case ($m = 3, 4, 5$) where the number of glowworms was fixed at some value and the percentage of peak-captures was logged for various values of r_s. A set of 10 trials was conducted for each value of r_s. The value of r_s that gave the best performance in each case was used in the main simulations to test the performance of the algorithm as a function of the number of agents. The results of these experiments are shown in Figs. 13(a), (b), and (c), for the three cases ($m = 3, 4, 5$), respectively. Maximum performance was observed at $r_s = 1.5, 4$, and 5 for $m = 3, 4$, and 5, respectively. In the three-dimensional case, peak-captures of 26%, 85%, and 100% are obtained, when $n = 200, 600$, and 1200, respectively. In accordance to observations in the previous section, the number of peak-captures increases when the value of n increases. In the four-dimensional case, peak-captures of 36%, 85%, and 90% are obtained when $n = 1000, 3000$, and 5000, respectively. The plots of mean minimum distance to peak locations $dmin_{\text{av}}(t)$ (defined in (8)) for $m = 3$ and $m = 4$ are shown

Fig. 13 Fraction of peaks captured for the $J_6(X)$ function for various values of r_s when (**a**) $m = 3$ and $n = 500$; (**b**) $m = 4$ and $n = 1000$; (**c**) $m = 5$ and $n = 2000$





(a)



(b)

Fig. 14 Plots of average minimum distance to peak locations $dmin_{\mathrm{av}}(t)$ for the $J_6(X)$ function: (**a**) $m = 3$; (**b**) $m = 4$

Table 7 Number of glowworms required for 46% peak-capture of J_6 function for $m = 1, \ldots, 5$ and $k = 1$

m	Number of peaks	n_m	n_m/n_{m-1}
1	3	6	
2	9	41	6.83
3	27	300	7.31
4	81	1500	5
5	243	6000	4

in Figs. 14(a) and (b), respectively. Note that increasing the number of glowworms also increases the slope of the $dmin_{\text{av}}(t)$ curve. In the five-dimensional case, a peak-capture of 46% was observed when $n = 6000$. To characterize the dependence of the required number of glowworms on the dimensionality of the problem, the number of glowworms n_m required for 46% peak capture for $m = 1, \ldots, 5$ are reported in Table 7. The values n_m/n_{m-1} show that the number of glowworms needed to compute a given fraction of the total number of peaks does not increase exponentially. However, note that by increasing the dimension m by 1, the number of glowworms has to be multiplied by a value between 4 ($m = 5$) and 7 ($m = 2$), even though the number of peaks increases only by a factor of 3, implying that more glowworms are needed in higher dimensions, which is seemingly counter-intuitive. But, the number of glowworms required for a particular dimensionality depends on factors such as the sensor range affecting agent connectivity and the nature of the peaks (e.g., a peak located in the interior is relatively easier to compute than those that are located on the edges and corners of the search space). These effects are more prominent at lower dimensions and become less important as the dimensionality increases (refer to Table 7) and the ratio n_m/n_{m-1} tends to values closer to 3 at higher dimensions.

7 Comparison of GSO with PSO

7.1 Comparison based on basic characteristics

Particle swarm optimization (PSO) is a population-based stochastic search algorithm (Clerc 2007). In PSO, a population of solutions $\{X_i : X_i \in \mathbb{R}^m, i = 1, \ldots, N\}$ is evolved, which is modeled by a swarm of particles that start from random positions in the objective function space and move through it searching for optima; the velocity V_i of each particle i is dynamically adjusted according to the *best so far* positions visited by itself (P_i) and the whole swarm (P_g). The position and velocity updates of the ith particle are given by:

$$V_i(t+1) = V_i(t) + c_1 r_1 \big(P_i(t) - X_i(t)\big) + c_2 r_2 \big(P_g(t) - X_i(t)\big), \tag{9}$$

$$X_i(t+1) = X_i(t) + V_i(t+1), \tag{10}$$

where c_1 and c_2 are positive constants, referred to as the *cognitive* and *social* parameters, respectively, r_1 and r_2 are random numbers that are uniformly distributed within the interval [0, 1], and $t = 1, 2, \ldots,$ indicates the iteration number. PSO uses a memory element in the velocity update mechanism of the particles. In contrast, the notion of memory in GSO is incorporated into the incremental update of the luciferin values that reflect the cumulative goodness of the path followed by the glowworms.

GSO does share a feature with PSO: in both algorithms a group of particles/agents are initially deployed in the objective function space and each agent selects, and moves in, a

Table 8 PSO versus GSO

	PSO	GSO
1	Direction of movement based on previous best positions	Agent movement along line-of-sight with a neighbor
2	Dynamic neighborhood based on k nearest neighbors	Adaptive neighborhood based on varying range
3	Neighborhood range covers the entire search space	Maximum range hard limited by finite sensor range
4	Limited to numerical optimization models	Can be applied to multiple source localization in addition to numerical optimization

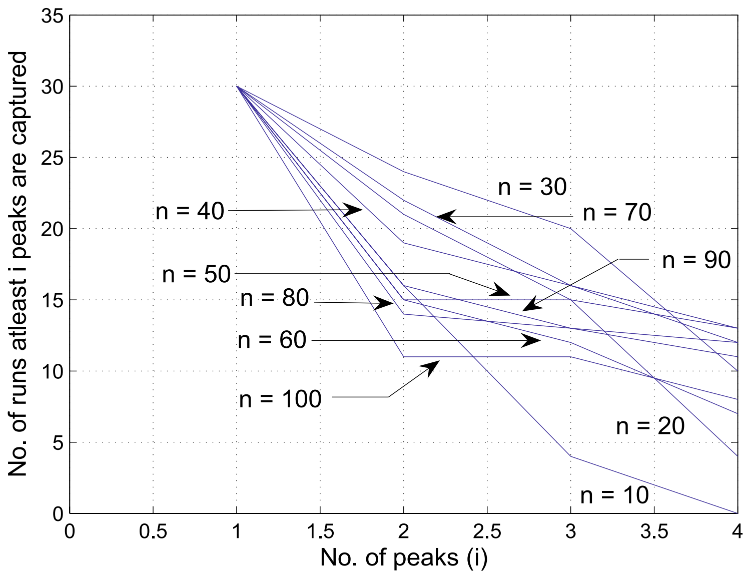
direction based on respective position update formulae. While the directions of a particle's movements in the original PSO are adjusted according to its own and global best previous positions, movement directions are aligned along the line-of-sight between neighbors in the GSO algorithm. The main differences between GSO and PSO are summarized in Table 8.

7.2 Experimental comparison between PSO and GSO

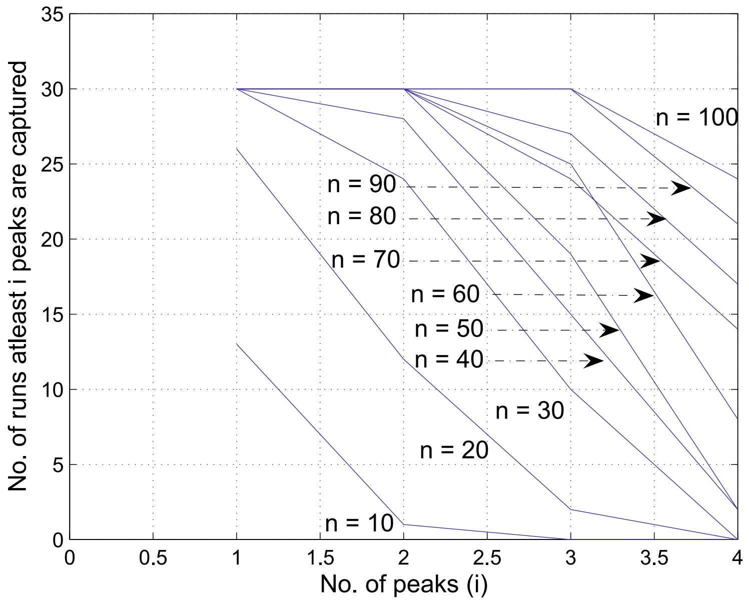
We compared GSO with Niche-PSO (Brits et al. 2002), a PSO variant that can be used to obtain multiple optima of multimodal functions. We tested Niche-PSO and GSO on Himmelblau's (J_8), the Equal-peaks-B (J_9), and Rastrigin's (J_2) functions. A set of 30 trials is conducted for each test case. For the purpose of comparison, we used the following two performance metrics: (i) the number of runs, R_i, for which at least i ($i = 1, 2, \dots, Q$) peaks are captured, for various values of n; (ii) the average number of peaks captured as a function of n; (iii) the mean distance traveled by an agent; (iv) the computation time; and (v) the total number of iterations for convergence.

Tests on Himmelblau's function: Himmelblau's function has four equal peaks of equal function value in the search space of $[-5, 5] \times [-5, 5]$. The plots of R_i for various values of n when Niche-PSO and GSO are used are shown in Figs. 15(a) and (b), respectively. For Niche-PSO, the performance improves as the number of agents increases until $n = 40$ (when the number of runs for which all four peaks are captured is maximum ($R_4 = 13$)), after which its performance deteriorates. However, in the case of GSO there is a steady improvement in performance with increasing values of n. The above observation is also supported by the gradual increase in the average number of peaks captured with the increase of n (Fig. 16) when GSO is used. The maximum average number of peaks captured is higher when GSO is used than when Niche-PSO is used. Moreover, the standard deviation values are relatively less in the case of GSO than those of PSO.

Tests on the Equal-peaks-B function: The Equal-peaks-B function has 12 peaks of equal function value in the search space of $[-5, 5] \times [-5, 5]$. The plots of R_i for various values of n when the Niche-PSO and GSO are used are shown in Figs. 17(a) and (b), respectively. We get similar comparison results as in the experiments with Himmelblau's function. In the case of Niche-PSO, note that performance degrades drastically above $n = 100$ (Fig. 17(a)), where only one peak is captured (with the exception of two peak captures in a few instances). The maximum average number of peaks captured is only 2 in the case of Niche-PSO and is approximately 12 in the case of GSO (Fig. 18).



(a)



(b)

Fig. 15 Number of runs where at least *i* peaks are captured for Himmelblau's function: (**a**) Using Niche-PSO. (**b**) Using GSO

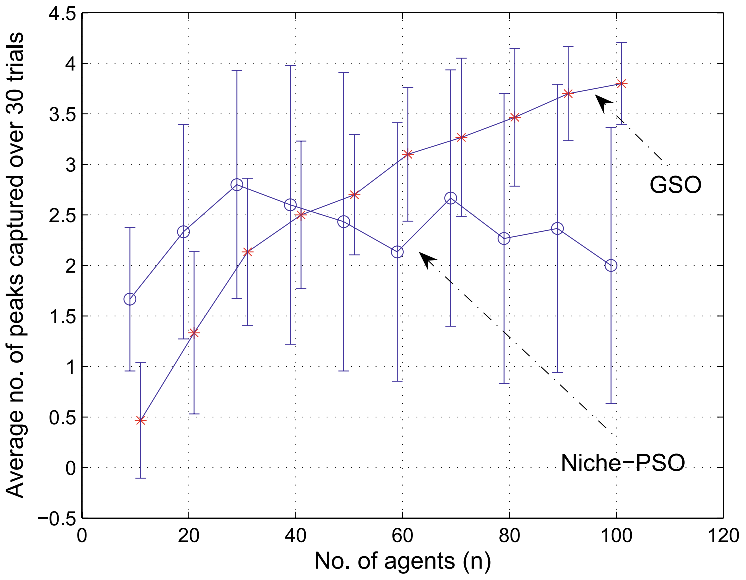


Fig. 16 Average number of peaks captured as a function of the number of agents for Himmelblau's function

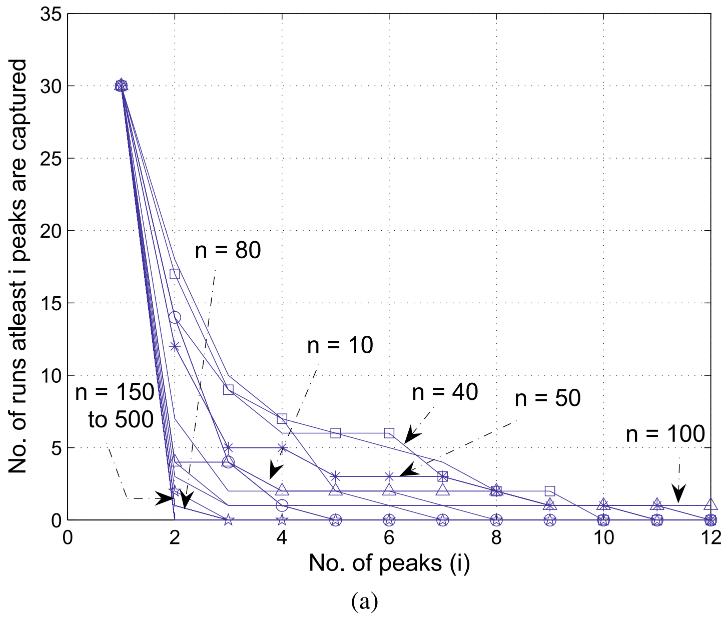
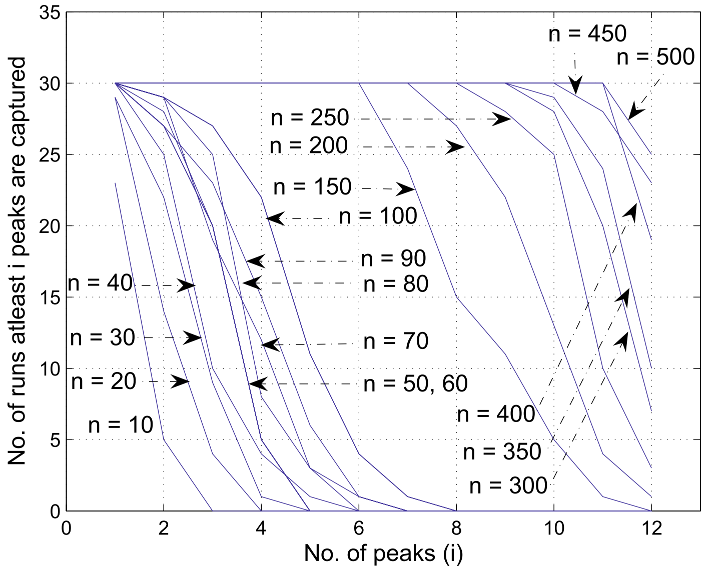


Fig. 17 Number of runs where at least i peaks are captured for the Equal-peaks-B function: (**a**) Using Niche-PSO. (**b**) Using GSO

Tests on Rastrgin's function: Rastrigin's function consists of 16 peaks of unequal function values in a search space of $[-2, 2] \times [-2, 2]$. The plots of R_i for various values of n when the Niche-PSO and GSO are used are shown in Figs. 19(a) and (b), respectively. The average



(b)

Fig. 17 (*Continued*)

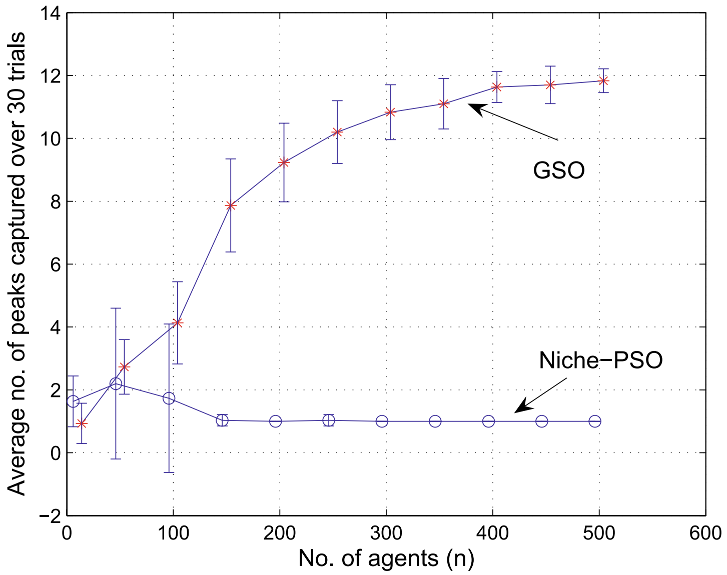
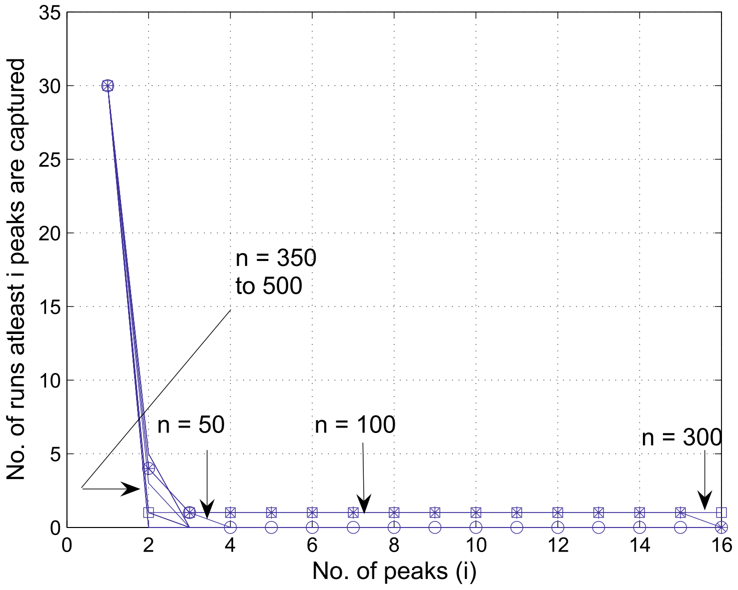
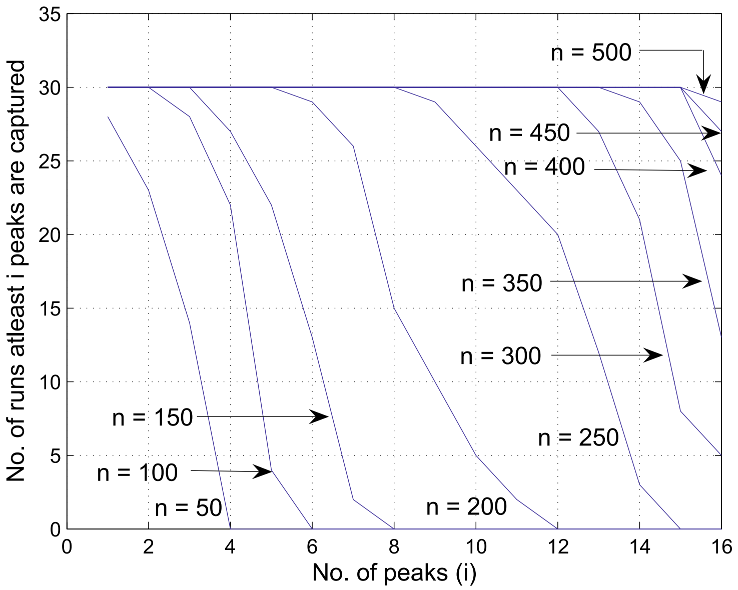


Fig. 18 Average number of peaks captured as a function of the number of agents for the Equal-peaks-B function

number of peaks captured as a function of n is shown in Fig. 20. Using the same analysis as before, it is evident that GSO performs better when compared to Niche-PSO on Rastrigin's function.



(a)



(b)

Fig. 19 Number of runs where at least *i* peaks are captured for Rastrigin's function: (**a**) Using Niche-PSO. (**b**) Using GSO

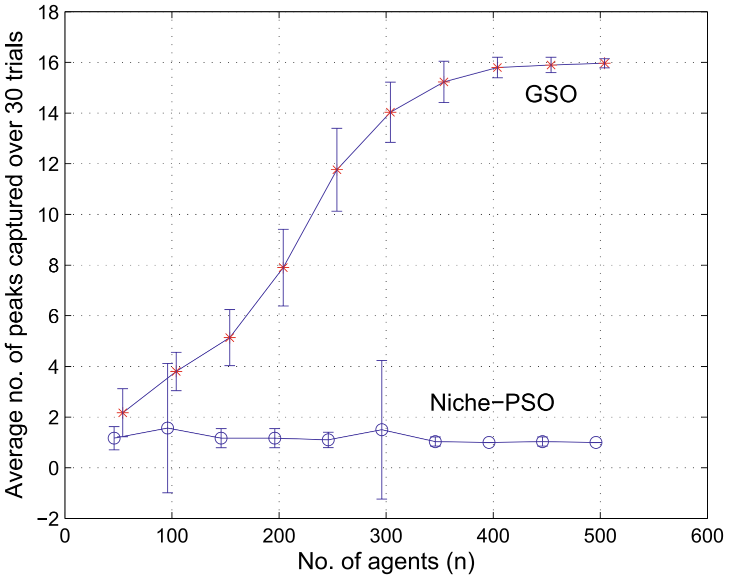


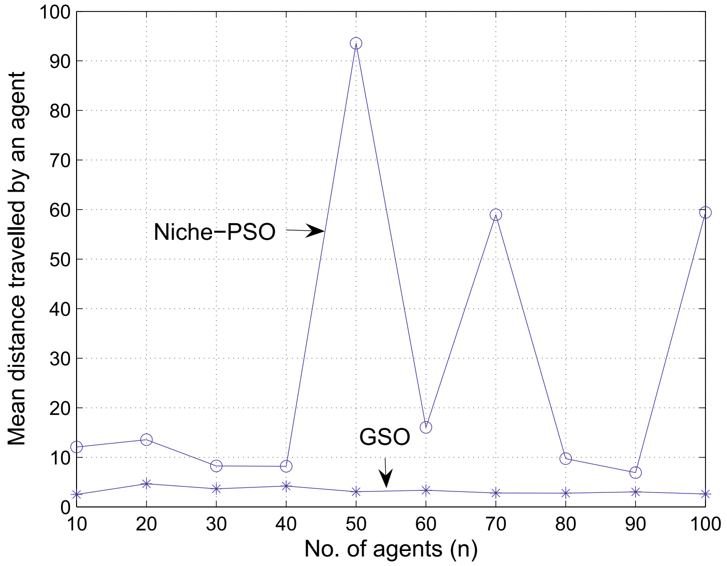
Fig. 20 Average number of peaks captured as a function of the number of agents for Rastrigin's function

A common observation in all the above experiments is the fact that Niche-PSO maintains disjoint sub-swarms at low values of n during its search for multiple peaks, but the performance in terms of number of peaks captured is not consistent over the number of trials. Moreover, as the number of agents increases, the algorithm loses the niching ability, leading to the computation of a single peak at relatively large values of n. However, GSO shows stable performance over different trials and shows a steady improvement in performance with increasing values of n in terms of the average number of peaks captured. The mean distances traveled by an agent for different cases are plotted in Figs. 21(a), (b), and (c). Note that the average distance traveled by an agent is relatively very low for GSO in comparison to Niche-PSO, in all the cases. Even though GSO takes more time than Niche-PSO at higher values of n (Fig. 22), GSO performs better than Niche-PSO in terms of the number of peaks captured.

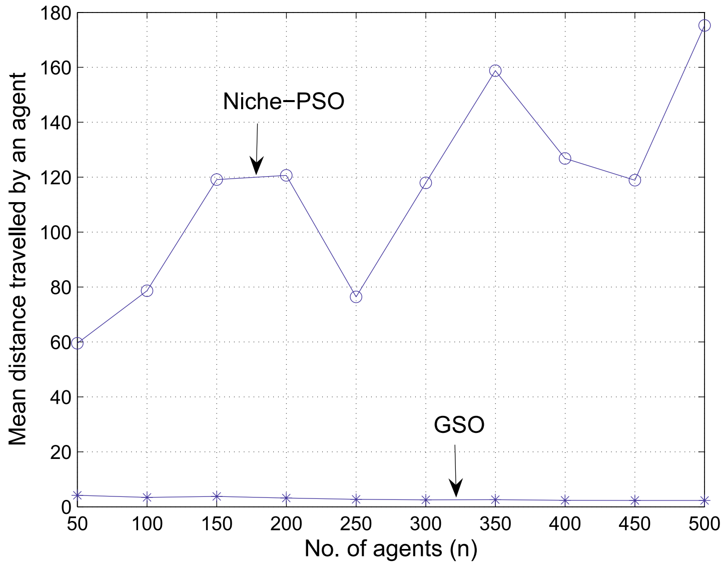
8 Concluding remarks

We have presented a novel algorithm called glowworm swarm optimization (GSO) that simultaneously finds multiple optima of multimodal functions. We described the underlying ideas behind the GSO technique, the notion of an adaptive neighborhood, and the steps involved in the implementation of the basic GSO algorithm. GSO is evaluated on several standard multimodal test functions borrowed from the literature. By choosing complex functions, such as the stair-case and multiple-plateau functions, we have shown that the algorithm can also handle discontinuities in the objective function. We reported test results in higher dimensions with large numbers of peaks. We also presented some comparisons of GSO with PSO and conducted an experimental comparison with a PSO variant used for capturing multiple peaks.

GSO was originally developed for numerical multimodal function optimization problems. However, it could be applied to a class of realistic collective robotics tasks related to



(a)



(b)

Fig. 21 Mean distance traveled by an agent in the case of different test functions: (**a**) Himmelblau's. (**b**) Equal-peaks. (**c**) Rastrigin's

simultaneous localization of multiple signal sources. Real robot implementation of GSO in the context of localization of sound and light sources are reported in (Krishnanand et al. 2006b) and (Krishnanand 2007), respectively.

The glowworm metaphor approach raises several intriguing questions about the relation of parameter values and the tuning of the algorithm for different applications. We answer

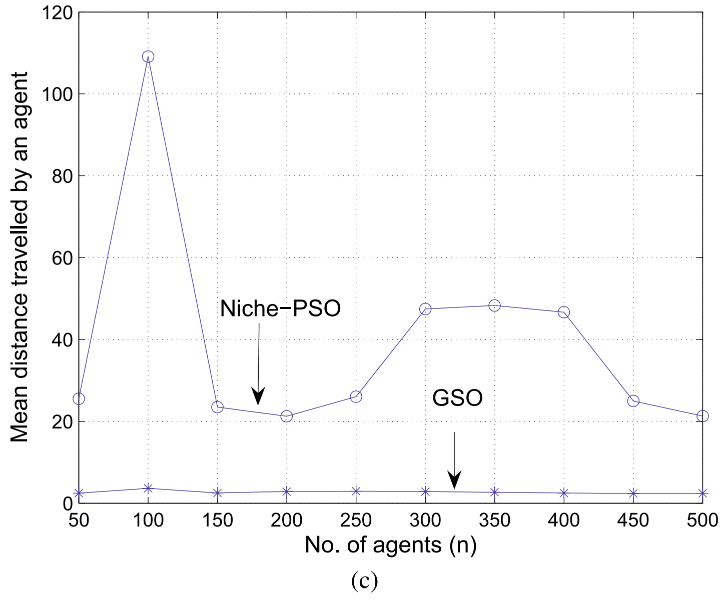


Fig. 21 (*Continued*)

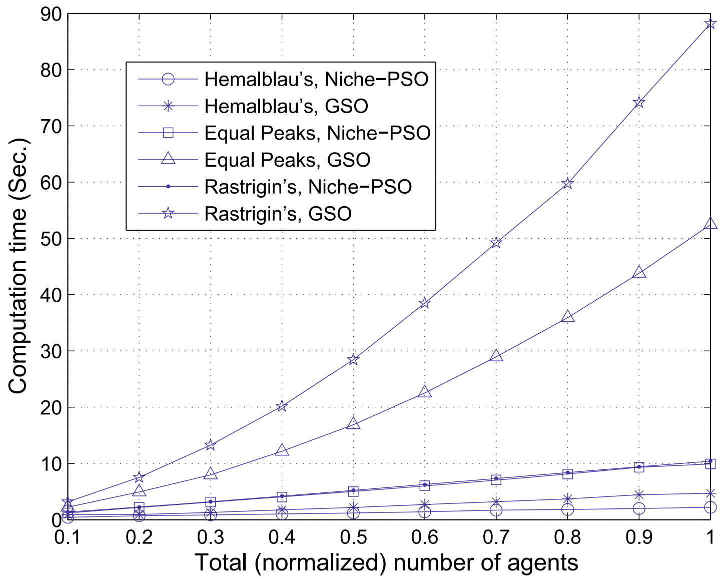


Fig. 22 Total computation time for 300 iterations

some of these questions by providing values for the algorithm parameters based on experimental observations and by showing that the values chosen work well for a wide range of experimental scenarios. Moreover, we conducted experiments to show that only two parameters, the number of agents n and the sensor range r_s, influence algorithm performance and need to be selected by the user. However, the experimental study in this regard gives a profound insight into selecting these two parameters for a large class of problems. Future work in this direction would involve a thorough analytical study of the effect of various parameters on algorithm performance, aimed primarily toward providing an analytical justification to the conclusions reached by experimentation. Some preliminary studies on the theoretical foundations of the glowworm algorithm using a simplified framework have been reported elsewhere (Krishnanand and Ghose 2008). Further work in this direction is needed to extend this theoretical treatment to a more complex model.

Acknowledgements We would like to thank the editors and reviewers and, in particular, Professor Marco Dorigo for their suggestions and comments.

References

Bonabeau, E., Dorigo, M., & Theraulaz, G. (1999). *Swarm intelligence: from natural to artificial systems*. New York: Oxford University Press.

Brits, R., Engelbrecht, A. P., & van den Bergh, F. (2002). A niching particle swarm optimizer. In *Proceedings of the 4th Asia-Pacific conference on simulated evolution and learning* (pp. 692–696).

Clerc (2007). *Particle swarm optimization*. London: ISTE Ltd.

Dorigo, M., & Gambardella, L. M. (1997). Ant colony system: a cooperative learning approach to the travelling salesman problem. *IEEE Transactions on Evolutionary Computation*, *1*(1), 53–66.

Dorigo, M., & Stützle (2004). *Ant colony optimization*. Cambridge: MIT Press.

Dorigo, M., Maniezzo, V., & Colorni, A. (1996). The ant system: optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, *26*(1), 29–41.

Dorigo, M., Trianni, V., Sahin, E., Gross, R., Labella, T. H., Baldassarre, G., Nolfi, S., Deneubourg, J.-L., Mondada, F., Floreano, D., & Gambardella, L. M. (2004). Evolving self-organizing behaviors for a swarm-bot. *Autonomous Robots*, *17*(2–3), 223–245.

Dréo, J., & Siarry, P. (2004). Continuous interacting ant colony algorithm based on dense hierarchy. *Future Generations Computer Systems*, *20*, 841–856.

Fevrier, V., & Patricia, M. (2007). Parallel evolutionary computing using a cluster for mathematical function optimization. In *Proceedings of the annual meeting of the North American fuzzy information processing society* (pp. 598–603). Piscataway: IEEE Press.

Fronczek, J. W., & Prasad, N. R. (2005). Bio-inspired sensor swarms to detect leaks in pressurized systems. In *Proceedings of IEEE international conference on systems, man and cybernetics* (pp. 1967–1972). Piscataway: IEEE Press.

Kennedy, J. (2000). Stereotyping: improving particle swarm performance with cluster analysis. In *Proceedings of the congress on evolutionary computation* (pp. 1507–1512). Piscataway: IEEE Press.

Kennedy, J., & Eberhart, R. C. (1995). Particle swarm optimization. In *Proceedings of the IEEE international conference on neural networks* (pp. 1942–1948). Piscataway: IEEE Press.

Krishnanand, K. N. (2007). *Glowworm swarm optimization: a multimodal function optimization paradigm with applications to multiple signal source localization tasks*. PhD thesis, Department of Aerospace Engineering, Indian Institute of Science.

Krishnanand, K. N., & Ghose, D. (2005). Detection of multiple source locations using a glowworm metaphor with applications to collective robotics. In *Proceedings of IEEE swarm intelligence symposium* (pp. 84–91). Piscataway: IEEE Press.

Krishnanand, K. N., & Ghose, D. (2006a). Glowworm swarm based optimization algorithm for multimodal functions with collective robotics applications. *Multiagent and Grid Systems*, *2*(3), 209–222.

Krishnanand, K. N., & Ghose, D. (2006b). Theoretical foundations for multiple rendezvous of glowworm-inspired mobile agents with variable local-decision domains. In *Proceedings of American control conference* (pp. 3588–3593). Piscataway: IEEE Press.

Krishnanand, K. N., & Ghose, D. (2008). Theoretical foundations for rendezvous of glowworm-inspired agent swarms at multiple locations. *Robotics and Autonomous Systems*, *56*(7), 549–569.

Krishnanand, K. N., Amruth, P., Guruprasad, M. H., Bidargaddi, S. V., & Ghose, D. (2006a). Rendezvous of glowworm-inspired robot swarms at multiple source locations: a sound source based real-robot implementation. In M. Dorigo et al. (Eds.), *Lecture notes in computer science: Vol. 4150. Ant colony optimization and swarm intelligence* (pp. 259–269). Berlin: Springer.

Krishnanand, K. N., Amruth, P., Guruprasad, M. H., Bidargaddi, S. V., & Ghose, D. (2006b). Glowworm-inspired robot swarm for simultaneous taxis toward multiple radiation sources. In *Proceedings of IEEE international conference on robotics and automation* (pp. 958–963). Piscataway: IEEE Press.

Mühlenbein, H., Schomisch, D., & Born, J. (1991). The parallel genetic algorithm as function optimizer. *Parallel Computing*, *17*(6–7), 619–632.

Muller, S. D., Marchetto, J., & Koumoutsakos, S. A. P. (2002). Optimization based on bacterial chemotaxis. *IEEE Transactions on Evolutionary Computation*, *6*(6), 16–29.

Parsopoulos, K., & Vrahatis, M. N. (2004). On the computation of all global minimizers through particle swarm optimization. *IEEE Transactions on Evolutionary Computation*, *8*(3), 211–224.

Poli, R., Kennedy, J., & Blackwell, T. (2007). Particle swarm optimization: an overview. *Swarm Intelligence*, *1*(1), 33–57.

Reutskiy, S. Y., & Chen, C. S. (2006). Approximation of multivariate functions and evaluation of particular solutions using Chebyshev polynomial and trigonometric basis functions. *International Journal for Numerical Methods in Engineering*, *67*(13), 1811–1829.

Singh, G., & Deb, K. (2006). Comparison of multi-modal optimization algorithms based on evolutionary algorithms. In *Proceedings of the genetic and evolutionary computation conference* (pp. 1305–1312). New York: ACM Press.

Singh, K. A., Mukherjee, A., & Tiwari, M. K. (2004). Incorporating kin selection in simulated annealing algorithm and its performance evaluation. *European Journal of Operational Research*, *158*, 34–45.

Törn, A., & Zilinskas, A. (1989). *Global optimization*. New York: Springer.

Tyler, J. (1994). *Glow-worms*. Sevenoaks: Tyler-Scagell.

Zarzhitsky, D., Spears, D. F., & Spears, W. M. (2005). Swarms for chemical plume tracing. In *Proceedings of IEEE Swarm Intelligence Symposium* (pp. 249–256). Piscataway: IEEE Press.