Glowworm swarm optimisation: a new method for optimising multi-modal functions

K.N. Krishnanand

Department of Aerospace Engineering,
Indian Institute of Science,
Bangalore 560 012, India
and
Department of Computer Science,
University of Vermont, USA
E-mail: krishna.iisc@gmail.com

D. Ghose*

Department of Aerospace Engineering,
Indian Institute of Science,
Bangalore 560 012, India
E-mail: dghose@aero.iisc.ernet.in
*Corresponding author

Abstract: This paper presents an exposition of a new method of swarm intelligence based algorithm for optimising multi-modal functions. The main objective of using this method is to ensure capture of all local maxima of the function. The application of this method is in the area of multiple signal source location or identification of odour sources and hazardous spills. The method is based upon a dynamic decision domain for each agent in the swarm that decides its direction of movement by the strength of the signal picked up from its neighbours. This is somewhat similar to the luciferin induced glow of a glowworm which is used to attract mates or prey. The brighter the glow more is the attraction. The method is memory-less and gradient free and does not require the knowledge of any global information. Moreover, the method is amenable to robotic implementation. Several illustrative examples are given to show the effectiveness of the method in comparison to existing swarm intelligence algorithms.

Keywords: computational intelligence; glowworm swarm optimisation; GSO; multimodal function optimisation; multiple signal source localisation.

Reference to this paper should be made as follows: Krishnanand, K.N. and Ghose, D. (2009) 'Glowworm swarm optimisation: a new method for optimising multi-modal functions', *Int. J. Computational Intelligence Studies*, Vol. 1, No. 1, pp.93–119.

Biographical notes: K.N. Krishnanand received his BE (Hons.) in Electrical Engineering from the Birla Institute of Technology and Science, Pilani, India, in 1998. He received his MSc (Engg.) and PhD from the Indian Institute of Science, Bangalore, in 2004 and 2007, respectively. Currently, he is a Postdoctoral Associate in the Department of Computer Science at the University of Vermont, USA. His research interests are in swarm intelligence, collective robotics, computer vision and social robotics. He was a JRD Tata

Research Fellow from 2007 to 2008. He received the Best Paper Award in the Second International Conference on Artificial Intelligence, Pune, December 2005 and the NASAS Medal of Excellence for the Best Master's Thesis Award for the year 2004.

Debasish Ghose is a Professor in the Department of Aerospace Engineering at the Indian Institute of Science, Bangalore, India. He obtained his BSc (Engg.) from the National Institute of Technology, Rourkela, India in 1982 and his ME and PhD from the Indian Institute of Science, Bangalore in 1984 and 1990, respectively. His research interests include guidance and control, collective robotics, multiple agent decision making and distributed computing. He authored the book *Scheduling Divisible Loads in Parallel and Distributed Systems* (Wiley-IEEE Computer Society Press). He is in the Editorial Board of the *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans* and the *IEEE Transactions on Automation Science and Engineering*. He is a Fellow of the Indian National Academy of Engineering.

1 Introduction

Multimodal function optimisation generally focuses on algorithms to find either a local optimum or the global optimum while avoiding local optima. However, there is another class of optimisation problems which have the objective of finding multiple optima with either equal or unequal function values (Sikora and Shaw, 1994; Sedeno and Vemuri, 1997; Horn et al., 1994; Miller and Shaw, 1996; Singh and Deb, 2006; Goldberg and Richardson, 1987; Petrowski, 1996; Brits et al., 2002; Li, 2004; Parsopoulos and Vrahatis, 2004). The knowledge of multiple optima has three principal advantages:

1 The likelihood of finding the global optimum is increased (Li, 2004).

2 An alternative solution can be selected when dynamic nature of constraints in the search space makes a previous optimum solution infeasible to implement.

3 An insight can be gained into the nature of the function landscape.

One of the domains that requires identification of multiple optima includes the learning of useful financial decision knowledge from examples, where identifying relevant knowledge and useful intermediate concepts is important (Sikora and Shaw, 1994). In stock markets where parameters influencing the price of stocks are constantly changing, the pay-off function is a multimodal dynamic landscape. Investment solutions that were not optimal at one time could potentially become optimal at a later time. In such a landscape, maintaining a set of possible investment strategies is useful and necessary (Sedeno and Vemuri, 1997). Another example is to identify a set of diverse rules that together can be used as the basis for a classifier (Horn et al., 1994). Real world applications of multimodal function optimisation include identification of multiple signal sources like sound, heat, light and leaks in pressurised systems (Fronczek and Prasad, 2005), hazardous plumes/aerosols resulting from nuclear/chemical spills (Zarzhitsky et al., 2005), fire-origins in forest fires (Casbeer et al., 2005), deep-sea hydrothermal vent plumes (Jakuba, 2007), hazardous chemical discharge in water bodies (Farrell et al., 2003), oil spills (Clark and Fierro, 2005), etc. Signals such as sound, light and other electromagnetic radiations propagate in the form of a wave. Therefore, the nominal

source profile that spreads in the environment can be represented as a multimodal function and hence, the problem of localising their respective origins can be modelled as optimisation of multimodal functions. Multi-modality in search and optimisation problems gives rise to several attractors and thereby presents a challenge to any optimisation algorithm in terms of finding global optimum solutions (Singh and Deb, 2006). However, the problem is compounded when multiple (global and local) optima are sought.

In this paper, we introduce glowworm swarm optimisation (GSO), a new method of swarm intelligence based algorithm for optimising multi-modal functions. The main objective of using this method is to ensure capture of all local maxima of the function. The significant difference between GSO and earlier approaches to multimodal function optimisation problems is the dynamic decision domain used by agents in the swarm to effectively locate multiple peaks. Each agent in the swarm uses the decision domain to select its neighbours and decides its direction of movement by the strength of the signal picked up from them. This is somewhat similar to the luciferin induced glow of a glowworm which is used to attract mates or prey. The brighter the glow, the more is the attraction. Therefore, we use the glowworm metaphor to represent the underlying principles of our optimisation approach. Consequently, the agents in GSO are thought of as glowworms that encode the objective function values at their current locations into a luciferin value and broadcast the same within their neighbourhood. The glowworm depends on its dynamic decision domain, which is bounded above by a circular sensor range, to identify its neighbours and compute its movements. Each glowworm selects a neighbour that has a luciferin value more than its own, using a probabilistic mechanism and moves toward it. That is, they are attracted to neighbours that glow brighter. These movements that are based only on local information enable the swarm of glowworms to partition into disjoint subgroups that converge to multiple optima of a given multimodal function.

The paper is organised as follows. Related work on multimodal function optimisation is presented in Section 2. The GSO algorithm is briefly described in Section 3. Evolution of GSO from its initial to current version is described in Section 4. The working of GSO is illustrated through examples in Section 5. A summary of results from numerical experiments on several benchmark multimodal functions is presented in Section 6. A comparison between GSO and PSO is provided in Section 7. A summarised description of the performance of GSO in the presence of noise, theoretical foundations and real world applications of GSO is presented in Section 8. Conclusions are given in Section 9.

2 Multimodal function optimisation

Population-based approaches are particularly suited to solving multimodal function optimisation problems and could be broadly divided into two categories: evolutionary computation (EC) techniques that are based on evolutionary mechanisms encountered in natural selection and swarm intelligence (SI) methods. Seeking multiple optima by maintaining population diversity has received some attention in the domain of genetic algorithms (GAs) (Sikora and Shaw, 1994; Sedeno and Vemuri, 1997; Horn et al., 1994; Miller and Shaw, 1996; Singh and Deb, 2006; Goldberg and Richardson, 1987;

Petrowski, 1996; Harick, 1997) and particle swarm optimisation (PSO) algorithms (Clerc, 2007; Brits et al., 2002; Parsopoulos and Vrahatis, 2004).

Traditional GAs can successfully identify the best rule (optimum) in the domain, but are incapable of maintaining rules of secondary importance (Miller and Shaw, 1996). Several niche-preserving techniques have been proposed that allow a GA to identify multiple optima of a multimodal function (Goldberg and Richardson, 1987; Petrowski, 1996; Mahfoud, 1992, 1995; Mengsheol and Goldberg, 1999; Harick, 1997). The niche metaphor is inspired from nature where different subspaces (niches) within an environment support different types of species. The number of organisms contained within a niche is determined by the carrying capacity of the niche and the efficiency of each organism at exploiting niche fertility.

In niching methods, each peak of a multimodal domain is thought of as a niche that can support a certain number of concepts. The number of individuals supported by a niche is in direct proportion to the niches carrying capacity, as measured by the niche peak's fitness relative to other niche peaks' fitness values present in the domain. As the number of individuals contained within a niche indirectly indicates the amount of computational effort the GA will spend to improve the niche, the niches are populated according to their fitness relative to the other peaks. In this manner, GAs can maintain the population diversity of its members in a multimodal domain. Some of the niching schemes that allow a GA to capture multiple optima of multimodal functions include sharing (Goldberg and Richardson, 1987), clearing (Petrowski, 1996), deterministic and probabilistic crowding (Mahfoud, 1992, 1995; Mengsheol and Goldberg, 1999) and restricted tournament selection (Harick, 1997).

Kennedy (2000) investigated modifying the PSO algorithm with stereotyping where clustering based on particles previous position, with cluster centres substituted for individual's or neighbour's previous bests, forces the clusters to focus on local regions.

Brits et al. (2002) proposed a niching particle swarm optimisation (NichePSO) algorithm that locates and tracks multiple solutions simultaneously. In NichePSO, the notion of using subswarms to improve swarm diversity and avoid premature convergence is adapted to maintain and optimise niches in the objective function space. The success of the NichePSO depends on the proper initial distribution of particles throughout the search space. Particles in the main swarm do not share knowledge about the best solution and use only their own knowledge. If a particle's fitness shows very little change over a small number of iterations, a niche is created with the particle and its closest topological neighbour. A niche radius is defined as the maximum distance between the global best particle (particle with the maximum fitness in the niche) and any other particle within the niche. A particle that is outside of the radius of a niche is merged into it when the particle moves into the niche, that is, when its distance to the global best particle becomes less than the niche radius. In this manner, population diversity is preserved in the form of different niches and each niche converges to a different optimum solution. The algorithm was reported to be successful at detecting global maxima and sometimes local maxima.

Li (2004) proposed a species-based PSO (SPSO) that incorporates the idea of species into PSO for solving multimodal optimisation problems. Initially, a population of particles is generated randomly. At each iteration step, different species seeds are identified for multiple species and then used as the lbest (particle that has the best fitness among the members of the same topological neighbourhood) for different species accordingly. For this purpose, all the particles are evaluated and sorted in descending order of their fitness values. The particle with the best fitness is set as the initial species

seed. All particles that are within a radius r_s of the species seed's position, along with the seed, form one species. The next best particle that falls outside the r_s range of the first seed is set as the next species seed. The above process is repeated until all the particles are checked against the species seeds. These multiple adaptively formed species are then used to optimise towards multiple optima in parallel, without interference across different species.

While the application of evolutionary approaches is largely limited to numerical (discrete combinatorial or continuous) optimisation problems that involve computer-based experiments, the particle-nature of individuals in swarm based optimisation algorithms enable their application to realistic collective robotics tasks, with some modifications where necessary.

3 Glowworm swarm optimisation

In GSO, physical entities/agents $\{i : i = 1,\ldots,n\}$ are considered that are initially randomly deployed $\{x_i(0) : x_i \in \mathbb{R}^m, i = 1,\ldots,n\}$ in the objective function space $\mathbb{R}^m$. Each agent in the swarm decides its direction of movement by the strength of the signal picked up from its neighbours. This is somewhat similar to the luciferin induced glow of a glowworm which is used to attract mates or prey. The brighter the glow, the more is the attraction. Therefore, we use the glowworm metaphor to represent the underlying principles of our optimisation approach. Hereafter, we refer to the agents in GSO as glowworms. However, the glowworms in GSO are endowed with other behavioural mechanisms (not found in their natural counterparts) that enable them to selectively interact with their neighbours and decide their movements at each iteration. In natural glowworms, the brightness of a glowworm's glow as perceived by its neighbour reduces with increase in the distance between the two glowworms. However, in GSO, we assume that luciferin value of a glowworm agent as perceived by its neighbour does not reduce due to distance.

Each glowworm i encodes the objective function value $J(x_i(t))$ at its current location $x_i(t)$ into a luciferin value ℓ_i and broadcasts the same within its neighbourhood. Each glowworm i regards only those incoming luciferin data as useful that are broadcast by its neighbours; the set of neighbours $N_i(t)$ of glowworm i consists of those glowworms that have a relatively higher luciferin value and that are located within a dynamic decision domain whose range r_d^i is bounded above by a circular sensor range $r_s (0 < r_d^i \le r_s)$. Each glowworm i selects a neighbour j with a probability $p_{ij}(t)$ and moves toward it. These movements that are based only on local information, enable the glowworms to partition into disjoint subgroups, exhibit a simultaneous taxis-behaviour toward and eventually co-locate at the multiple optima of the given objective function.

The GSO algorithm is given in the inset box [please refer to Krishnanand and Ghose (2006, 2008a) for a detailed description of the various algorithmic phases of GSO]. The algorithm is presented for maximisation problems. However, it can be easily modified and used to find multiple minima of multimodal functions.

The number of parameters and how much their 'optimal' values are dependent on the problem have a major impact on the practical applicability of optimisation algorithms. A good algorithm would consist of a small number of problem-specific parameters and a set of algorithmic parameters tuned to fixed values that yield optimal performance over a wide range of problems.

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 (Table 1). The neighbourhood threshold n_t indirectly controls the number of neighbours of each glowworm by influencing the neighbourhood 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 less in relation to the size of the search space. Fixed value of $s = 0.03$ resulted in similar algorithmic performance across a large variety of test functions (Krishnanand, 2007). The same value of s may not be efficient for domains and ranges of objective functions whose sizes significantly differ from those of the class of functions used to test the algorithm. However, it can be easily scaled to account for such variations in scaling of the domains and ranges of objective functions. 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 found to be a good choice. The parameter β affects the rate of change of the neighbourhood range. A relatively high value of β would lead to saturation resulting in the switching of the neighbourhood range between its upper and lower limits. Therefore, a small value of β (= 0.08) was chosen, which worked well for different test functions. A value ρ = 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 ρ = 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.

As mentioned above, the algorithmic parameters are kept fixed and are not specifically tuned for every problem. Thus, only n and r_s need to be selected. A full factorial analysis is carried out in Krishnanand and Ghose (2008b) to 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.

Table 1 The GSO algorithm

ρ	γ	β	n_t	s	ℓ_0
0.4	0.6	0.08	5	0.03	5

Figure 1 The GSO algorithm

Glowworm swarm optimisation (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

$\ell_i(t) = (1-\rho)\ell_i(t-1) + \gamma J(x_i(t))$;

for each glowworm i do: % Movement-phase

{

$N_i(t) = \{j : \|x_j(t) - x_i(t)\| < r_d^i(t); \ell_i(t) < \ell_j(t)\}$;

where $\|\vec{x}\|$ is the norm of $\vec{x}$

for each glowworm $j \in N_i(t)$ do:

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

j = *select_glowworm* $(\vec{p})$

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

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

}

$t \leftarrow t + 1$;

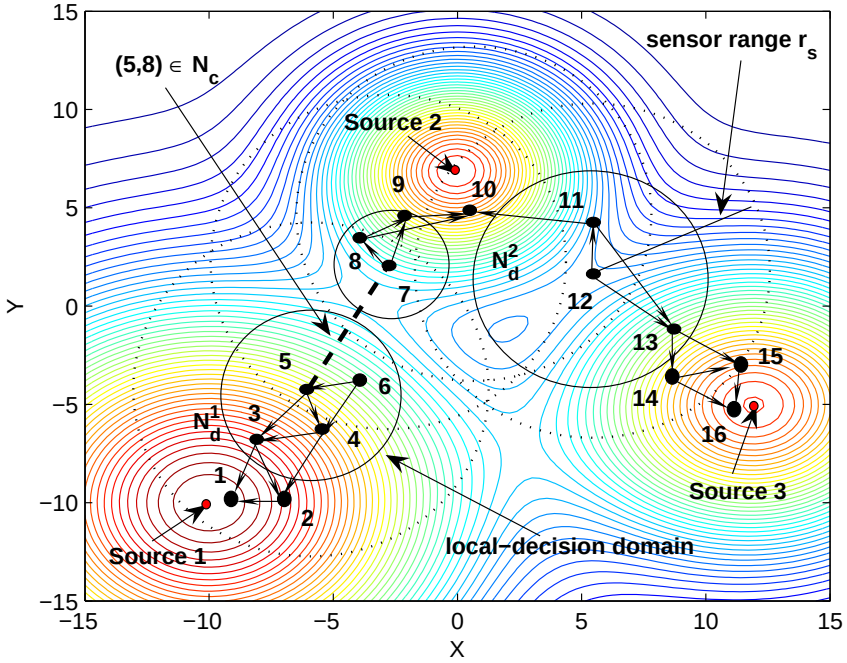
}

Implementation of GSO at the individual agent level gives rise to two major phases at the group level: Formation of dynamic networks that results in splitting of the swarm into subswarms and local convergence of glowworms in each subgroup to the peak locations. The two phases are briefly described below:

3.1 Dynamic networks: special suitability for multiple source localisations

Let $G(V,E)$ be a graph with vertex set $V=\{v_1,\ldots,v_n\}$ and edge set $E=\{(v_i,v_j):v_i,v_j\in V\}$. If E is a set of unordered pairs, then G is said to be an undirected graph. If E is a set of ordered pairs, then G is said to be a directed graph. The graph G is said to be connected if it has a path between each distinct pair of vertices v_i and v_j where by a path (of length m) we mean a sequence of distinct edges of G of the form $(v_i,k_1),(k_1,k_2),\ldots(k_m,v_j)$. The constraints of sensor range r_s and adaptive decision-range r_d^i that are imposed on each glowworm lead to the emergence of two different networks – an undirected communication-network N_c and a directed decision-network N_d – of the same set of glowworms. From the algorithm's description, it is clear that agents in the GSO algorithm do not maintain fixed-neighbours during their movements, which means that existing links between neighbours may break and new links may get established between glowworm pairs. Therefore, the networks N_c and N_d are dynamic in nature. We use an illustrative example (Figure 2) in order to describe the network architecture of agents in a glowworm swarm and its special suitability for localising multiple sources. A group of 16 glowworms is randomly deployed in a workspace that consists of three sources placed at different locations. Note from Figure 2 that the graph N_c is connected. If the glowworms use a constant local-decision domain whose range is equal to the maximum sensing range r_s, all the glowworms converge to the global peak. However, note that the graph N_d is partitioned into two disjoint weakly connected components N_d^1 and N_d^2, which can be explained in the following way. When the glowworms use an adaptive decision-domain, the glowworms adjust their decision-domain ranges until they acquire a pre-specified number of neighbours ($n_t=2$ in the example shown in Figure 2). This property enables each glowworm to select its neighbours in such a manner that its movements get biased toward the nearest peak. The above individual agent behaviour leads to a collective behaviour of agents that constitutes the automatic splitting of the whole group into disjoint subgroups where each subgroup of agents gets allocated to a nearby peak (that is a peak to which the agent-distance, averaged over the subgroup, is minimum among those distances to all the peaks in the environment). In Figure 2, glowworms 5, 6, 8, 9 and 10 are within the sensing-range of glowworm 7. However, glowworm 7 considers only lowworms 8 and 9 as neighbours. Therefore, its movements get biased toward Source 2, while it avoids moving toward glowworm 5, even though glowworm 5 has a higher luciferin value than that of itself. Accordingly, the subgroups of glowworms {1, 2, 3, 4, 5, 6}, {7, 8, 9, 10} and {13, 14, 15, 16} taxis toward Source 1, Source 2 and Source 3, respectively. Note that glowworms 11 and 12 may move toward either Source 2 or Source 3. The local-search and convergence of each subgroup to a nearby source location is achieved by the 'leapfrogging' effect inherent in GSO, which is explained below.

Figure 2 Networks resulting from the two different constraints imposed on the agents in the GSO algorithm: an undirected communication network N_c due to sensing range r_s and a directed decision network N_d due to the adaptive range r_d^i (see online version for colours)



Note: The graph N_d is partitioned into two disjoint weakly connected components N_d^1 and N_d^2.

3.2 Local search due to leapfrogging behaviour

In GSO, a glowworm with the maximum luciferin at a particular iteration remains stationary during that iteration. The above property leads to a dead-lock situation when all the glowworms are located such that the peak-location lies outside the convex-hull formed by the glowworm positions. 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 get co-located (meet at a single location) away from the peak-location. However, the discrete nature of the movement-update rule automatically takes care of this problem which could be described in the following way. During the movement phase, each glowworm moves a distance of finite step-size s toward a neighbour. Hence, when a glowworm i approaches closer to a neighbour j such that the inter-agent distance becomes 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 interchanging of roles between i and j occurs repetitively giving rise to a local-search behaviour of the glowworm pair along a

single ascent direction. A group of glowworms use the same principle to perform an improved local-search and eventually converge to the peak location. The leapfrogging behaviour of the agents in the GSO algorithm are supported by simulations results shown in Section 5.

4 Evolution of GSO

The GSO algorithm, in its present form, has evolved out of several significant modifications incorporated into the earlier versions of the algorithm (Krishnanand and Ghose, 2005a, 2006) that lead to improvement in algorithmic performance from one version to the next. We will discuss some of the important steps in this evolution briefly in order to convey the fact that many ideas were considered in the development process before we converged upon the current GSO version.

The algorithm was first introduced in Krishnanand and Ghose (2005a). The equations that modelled the luciferin-update, probability distribution used to select a neighbour, movement update and local-decision range update are given below:

$$\ell_i(t+1) = \max\left\{0, (1-\rho)\,\ell_i(t) + \gamma J\left(x_i(t+1)\right)\right\} \tag{1}$$

$$p_j(t) = \frac{\ell_j(t)}{\sum_{k \in N_i(t)} \ell_k(t)} \tag{2}$$

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

$$r_d^i(t+1) = \frac{r_s}{1 + \beta D_i(t)} \tag{4}$$

where,

$$D_i(t) = \frac{N_i(t)}{\pi r_s^2} \tag{5}$$

is the neighbour-density of agent i at iteration t and β is a constant parameter.

The local-decision range update rule (4) faces a problem at an instant when a glowworm i has no neighbour in its current local-decision domain range $r_d^i(t)$ but has some neighbours within its sensor range r_s. In particular, when the glowworm i isolated, $D_i(t) = 0$. From (4), $r_d^i(t+1) = r_s$. Suppose glowworm i acquires some neighbours at $t+1, D_i(t+1) \neq 0$. If we use a large value of β, (4) gives $r_d^i(t+2) \approx 0$. Therefore, the above kind of neighbour-distribution associated with i results in an oscillatory behaviour of the decision range, with r_d^i switching between r_s and a value closer to zero. Therefore, all glowworms in a neighbourhood situation as described above move only in alternative iterations slowing down the convergence of the algorithm approximately by a factor of two.

To solve the above problem, we modify the local-decision update rule in Krishnanand and Ghose (2005b) by forcing a nonzero lower bound on the decision range such that each glowworm improves its own chances of finding a neighbour at every instant. The modified update rule for $r_d^i(t)$ is given by:

$$r_d^i(t+1) = \alpha + \frac{r_s - \alpha}{1 + \beta N_i(t)} \tag{6}$$

where α represents the lower bound of the decision domain range.

We propose a new update rule in Krishnanand et al. (2006b), where an explicit threshold parameter n_t is used to control the number of neighbours at each iteration. We notice that there is a substantial enhancement in performance by using this rule:

$$r_d^i(t+1) = \begin{cases} r_d^i(t) + \beta_1 \left| N_i(t) \right|, \text{if } \left| N_i(t) \right| \le n_t \\ r_d^i(t) - \beta_2 \left| N_i(t) \right|, \text{otherwise} \end{cases} \tag{7}$$

where, β_1 and β_2 are constant parameters.

A further improvement in algorithmic performance is observed by using the present decision-range update rule, which was introduced in Krishnanand and Ghose (2006). This can be attributed to the fact that the second term (refer to the decision update rule given in the inset box of Figure 1), which either increments or decrements $r_d^i(t)$, is proportional to the difference between the desired number of neighbours n_t and the actual number of neighbours $\left| N_i(t) \right|$.

Since actual values of luciferin (instead of differences in luciferin values as used in the current version) are used in the probability distribution formula (2), the luciferin cannot take negative values. This is taken care of in the luciferin update formula (1), by forcing a zero lower bound on the luciferin value. However, the algorithm does not work in regions where the objective function has negative values unless the function is shifted appropriately. In order to address these problems, actual luciferin values in the probability distribution formula are replaced by relative luciferin values in Krishnanand et al. (2006a). As a consequence, the luciferin values are allowed to take negative values. Therefore, the luciferin-update formula was accordingly modified to its current form where the constraint of zero lower bound on the luciferin values is removed (Krishnanand et al., 2006a).

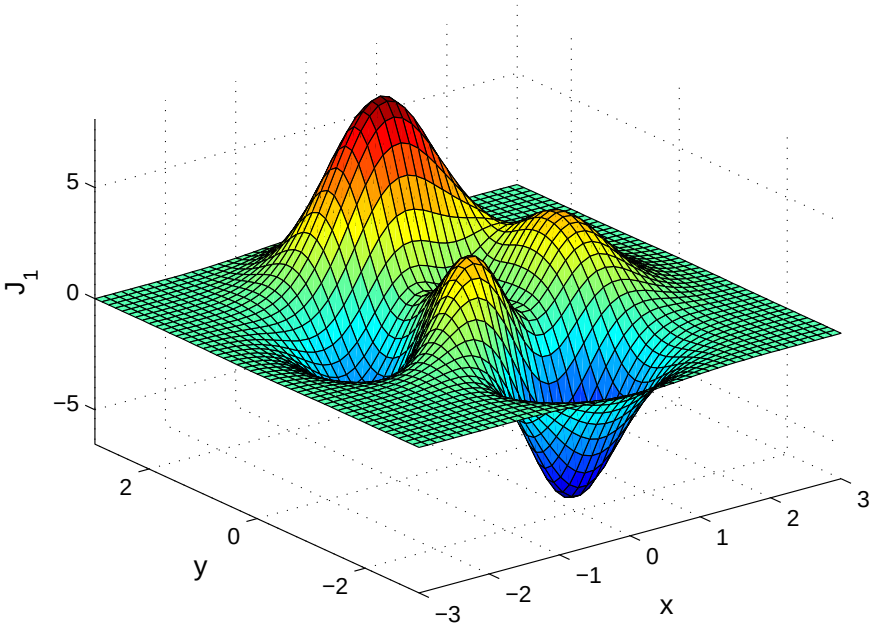
5 Simulation experiments to illustrate GSO

Simulation experiments[1] demonstrating the capability of GSO to capture multiple peaks of a number of benchmark multimodal functions are reported in Krishnanand (2007). Results from a set of 30 trials of each experiment show that the peak capturing behaviour of the algorithm is similar across different random initialisations. Here, we demonstrate the basic working of the algorithm using the Peaks function, which is obtained by translating and scaling Gaussian distributions (Reutskiy and Chen, 2006):

$$J_1(x,y) = 3(1-x)^2 e^{-[x^2+(y+1)^2]} - 10\left(\frac{x}{5} - x^3 - y^5\right)e^{-(x^2+y^2)} - \left(\frac{1}{3}\right)e^{-[(x+1)^2+y^2]} \quad (8)$$

The Peaks function $J_1(x,y)$ has multiple peaks and valleys [Figure 3(a)]. Local maxima are located at (0, 1.58), (–0.46, –0.63) and (1.28, 0) with different peak function values. A set of 50 glowworms are randomly deployed in a two-dimensional workspace of size 6 × 6 square units. The dynamic decision range is the key to GSO's ability of capturing multiple peaks. For instance, when the decision range is kept constant at $r_d^i = 2$, only one peak is captured; when r_d is decreased to 1.8, two peaks are captured. However, when a dynamic decision range is used, all the three peaks are captured (Krishnanand and Ghose, 2006). Figure 3(b) shows the emergence of the solution when a dynamic decision range is used. During this simulation, a value of $r_d^i(0) = 3$ is chosen for each glowworm i. Note that all the peaks are detected within 200 iterations. In particular, 23, 19 and 8 glowworms get co-located at the maxima of (0, 1.58), (–0.46, –0.63) and (1.28, 0), respectively.

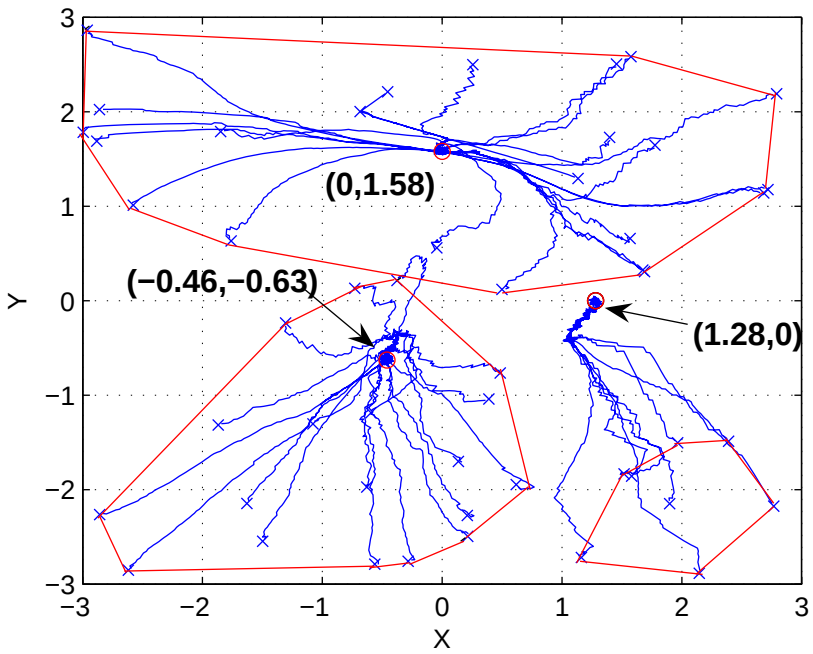
Figure 3 (a) Peaks function $J_1(x,y)$ used in simulations to demonstrate the basic working of GSO. The function has three local maxima located at (0, 1.58), (–0.46, –0.63) and (1.28, 0) (b) Trajectories (200 iterations) followed by the glowworms: they start from an initial random location (cross) and they end up at one of the peaks (all three peaks are captured) (see online version for colours)



(a)

Source: Reutskiy and Chen (2006)

Figure 3 (a) Peaks function $J_1(x,y)$ used in simulations to demonstrate the basic working of GSO. The function has three local maxima located at (0, 1.58), (–0.46, –0.63) and (1.28, 0). (b) Trajectories (200 iterations) followed by the glowworms: they start from an initial random location (cross) and they end up at one of the peaks (all three peaks are captured) (continued) (see online version for colours)



(b)

Source: Reutskiy and Chen (2006)

We present a summary of the results of another set of simulations that are conducted in Krishnanand and Ghose (2008a), in order to clearly characterise the splitting behaviour of the agent-swarm. We show how the same initial placement of agents gives rise to different splitting behaviours as conditioned by factors like the placements of various peaks, peak values and slope of the objective function in the vicinity of the peaks. We consider the $J_2(x,y)$ function (9) for this set of experiments (Figure 4).

$$J_2(x,y) = \sum_{i=1}^{Q} a_i \exp(-b_i((x-x_i)^2 + (y-y_i)^2)) \tag{9}$$

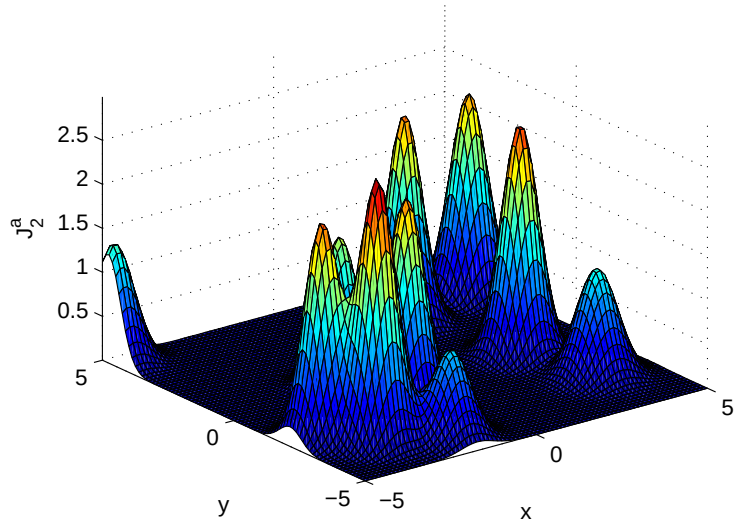
where, Q represents the number of peaks and (x_i, y_i) represents the location of each peak. The function $J_2(x,y)$ represents a linear sum of two dimensional exponential functions centred at the peak-locations. The constant b_i determines how the objective function changes slope in the vicinity of the peak i. We keep b_i equal for all the individual exponentials by choosing $b_i = 3, i = 1,\ldots Q$ [please refer to Krishnanand and Ghose (2008a) for a detailed description of the varied set of complexities presented by this function]. A workspace of [–5, 5] × [–5, 5] and a set of ten peaks $(Q = 10)$ are

considered for the purpose. The values of a_i, x_i and y_i are generated according to the following equations:

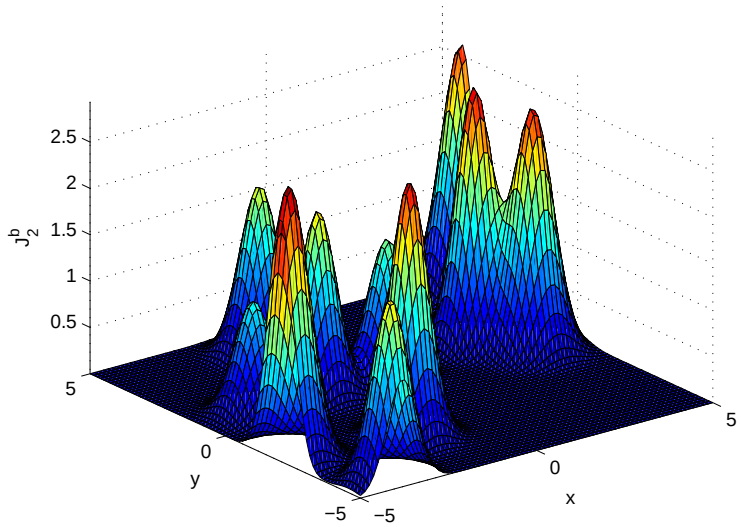
$$a_i = 1 + 2\vartheta$$
$$x_i = -5 + 10\vartheta$$
$$y_i = -5 + 10\vartheta$$

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

Figure 4 Multimodal function profiles J_2^a and J_2^b used to demonstrate the splitting behavior of agents in the GSO algorithm (see online version for colours)



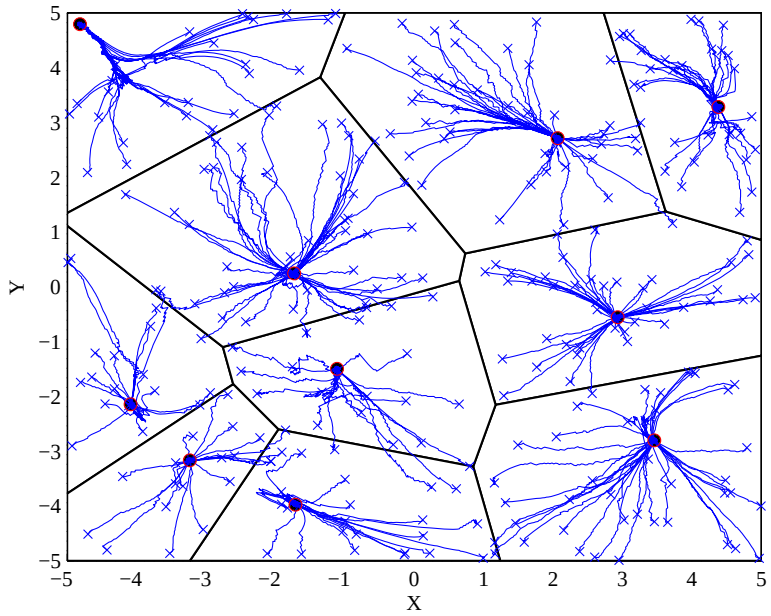
(a)



(b)

Figures 5(a) and 5(b) show the emergence of solution for each objective function when $n = 300$ and $r_s = 5$. The random initial agent placement is kept same for both the simulations. We observe that when the slopes of the peaks are equal, the swarm splits into subgroups according to the Voronoi-partition of the peak locations, that is, agents deployed in the Voronoi-partition of a peak location remain in the same partition, during their movements, and eventually converge to the corresponding Voronoi-centre that coincides with the peak location. However, a few agents located near the border region of a Voronoi-partition migrate into adjacent partitions and eventually get co-located at the respective peaks. Moreover, the same placement of agents, with a change in the location of the peaks, gives rise to a different partitioning of the agents. However, note that the splitting behaviour of the swarm respects the Voronoi-partition of the new set of peak locations [Figure 5(b)]. In the case where different slope profiles (b_i) are used for the individual exponentials in the $J_2(x, y)$ function, it was observed in Krishnanand and Ghose (2008a) that the subswarms partition according to the boundary line that passes through each equi-valued contour at a point where the gradient shifts direction from one peak to the other. It is easy to see that the above partitioning of the regions coincides with the Voronoi-partition of the peaks when their slopes become equal.

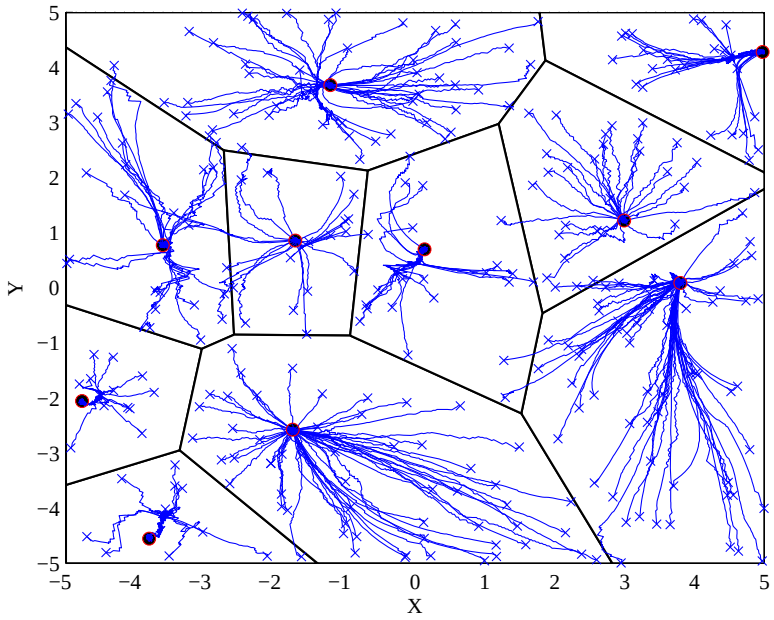
Figure 5 (a) Trajectories followed by the glowworms when $J_2^a(x, y)$ is used (b) trajectories followed by the glowworms when $J_2^b(x, y)$ is used (see online version for colours)



(a)

Note: The swarm splits according to the Voronoi-partition of the peak locations in both the cases.

Figure 5 (a) Trajectories followed by the glowworms when $J_2^a(x,y)$ is used (b) trajectories followed by the glowworms when $J_2^b(x,y)$ is used (continued) (see online version for colours)



(b)

Note: The swarm splits according to the Voronoi-partition of the peak locations in both the cases.

6 Performance of GSO on benchmark multimodal functions

The efficacy of GSO in capturing multiple peaks was shown through numerical simulations on a number of benchmark multimodal functions (Krishnanand, 2007). A brief summary of the related results are presented here. A representative set of multimodal test functions that pose different cases of complexity – unequal peaks, equal peaks, peaks of concentric circles, peak-regions involving step-discontinuities and plateaus of equal heights – are considered:

Rastrigin's function

$$J_3(x,y) = 20 + (x^2 - 10\cos(2\pi x) + y^2 - 10\cos(2\pi y)) \tag{10}$$

The Rastrigin's function was first proposed by Rastrigin (Törn and Zilinskas, 1989) as a two-dimensional function and has been generalised by 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 $(J_3(x,y))$ function consists of as many as

100 peaks in the considered range ([–5, 5] × [–5, 5]). The Rastrigin's function has been used as a benchmark function to test several algorithms designed to solve global and multimodal function optimisation problems (Muller et al., 2002; Fevrier and Patricia, 2007).

A set of 1,500 agents is deployed in a search space of [–5, 5] × [–5, 5] that contains a total number of 100 peaks. From the graph of the final co-location of groups of glowworms at various peaks (Krishnanand, 2007), it was observed that 92 peaks are captured in the dynamic decision domain case $(r_d^i(0) = 2)$ as against only eight peaks in the constant decision domain case $(r_d^i(t) = 2, \forall t)$. This result serves as a representative example to support the fact that the use of a dynamic decision domain, instead of a constant one, significantly improves the ability of the algorithm to capture multiple peaks.

Circles function

$$J_4(x,y) = (x^2 + y^2)^{0.25}((\sin^2(50(x^2 + y^2)^{0.1})) + 1.0) \tag{11}$$

The circles function (Muller et al., 2002) contains multiple concentric circles as the regions of local maxima. Unlike $J_1(x,y)$, $J_2(x,y)$ and $J_3(x,y)$ functions where each peak is a single point, the circular lines of local peaks present a infinite-peaks case.

The initial placement consists of random deployment of 1,000 glowworms deployed in the search space [–10, 10] × [–10, 10]. The pattern of the emerging solution resembles the concentric-circle contours of local maxima. However, rather than spreading uniformly on the circles, the glowworms converge to distinct points on the circles. This mainly occurs as a result of the glowworms seeking to move towards relatively brighter neighbours even as the associated differences in luciferin levels may be very small. Some amount of spreading can be achieved by disabling (nullifying) the attraction between two glowworms whenever the difference in their luciferin values is less than a small threshold value.

Staircase function

$$J_5(x,y) = 25 - \lfloor x \rfloor - \lfloor y \rfloor \tag{12}$$

The staircase function (Dréo and Siarry 2004) contains a series of stairs. In (12), $\lfloor x \rfloor$ is the floor function and is defined as the nearest integer lesser 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, towards the tallest stair (global peak region). The $J_5(x,y)$ function contains six regions at seven different heights within a search space of [–2, 2] × [–2, 2]. All the glowworms initially located in region [–2, –1] × [–2, –1] remain stationary as they are located in the flat region of global maxima. Note that glowworms in the interior locations (except the ones that are very close to and on the edges) of all other regions move towards and settle on the edges of the next higher peak-regions.

Plateaus function

$$J_6(x,y) = \text{sign}(\cos(\text{x}) + \cos(\text{y})) \tag{13}$$

The plateaus function contains multiple plateaus 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 $J_6(x,y)$ function have equal objective function values, thereby providing no uphill direction.

A set of 1,000 glowworms are deployed in a search space of $[-2\pi, 2\pi] \times [-2\pi, 2\pi]$. All the glowworms, except 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 plateaus function provide an interesting intuition regarding the working of the algorithm. For instance, we notice that glowworms located in the interior region of a plateau are always stationary. This could be attributed to their equal luciferin-levels. A large percent of the glowworms that climb the gradient-steps settle on the plateau-edges. However, a few glowworms, after climbing the plateaus, continue movement 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.

In summary, the Rastrigin's function and the circles function show the ability of GSO to simultaneously capture a very large number of peaks. By choosing complex functions such as the stair-case and multiple-plateau functions, it is shown that the algorithm can also handle discontinuities in the objective function. In Krishnanand and Ghose (2008b), GSO is tested on higher dimensional spaces with up to five dimensions. The test function used has a total number of 3^m peaks of equal function value (where m is the number of dimensions), in a search space whose range is $\prod_{i=1}^{m}[-\pi, \pi]$. The results show that the number of glowworms needed to compute a given fraction of the total number of peaks does not increase exponentially. However, by increasing the dimension m by 1, the number of glowworms n_m 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 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

PSO is a population-based stochastic search algorithm (Clerc, 2007). In PSO, a population of solutions $\left\{X_i : X_i \in \mathbb{R}^m, i = 1,\ldots,N\right\}$ is evolved, which is modelled 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:

$$V_i(t+1) = V_i(t) + c_1 r_1 (P_i(t) - X_i(t)) + c_2 r_2 (P_g(t) - X_i(t)) \tag{14}$$

$$X_i(t+1) = X_i(t) + V_i(t+1) \tag{15}$$

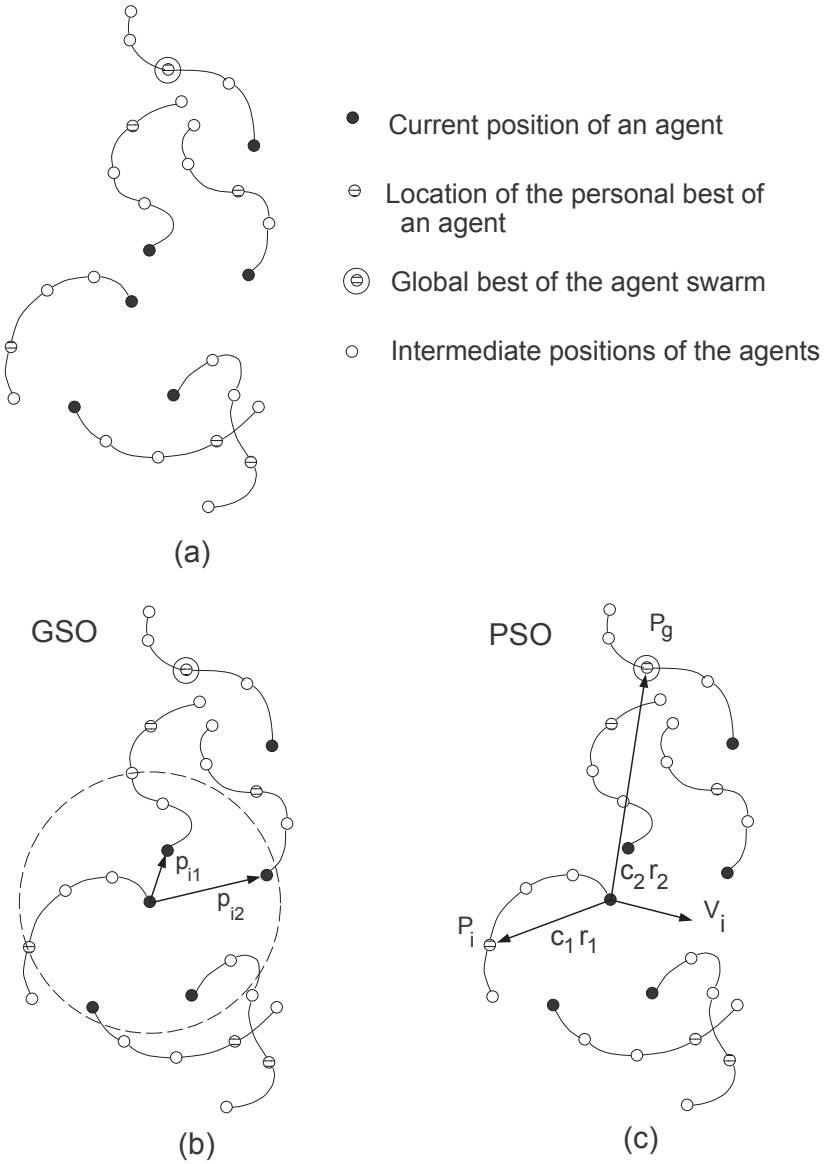
where c_1 and c_2 are positive constants, referred to as the 'cognitive' and 'social' parameters, respectively, r_1 and r_2 are random numbers that are uniformly distributed within the interval $[0,1], t = 1, 2, \ldots,$ indicates the iteration number. PSO uses a memory element in the velocity update mechanism of the particles. Differently, 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 shares 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 direction based on respective position update formulae. While the directions of a particle movements in original PSO are adjusted according to its own and global best previous positions, movement directions are aligned along the line-of-sight between neighbours in GSO. In PSO, The net improvement in the objective function at the iteration t is stored in $P_g(t)$. However, in the GSO algorithm, a glowworm with the highest luciferin value in a populated neighbourhood indicates its potential proximity to an optimum location.

Figures 6(a)–6(c) illustrate the basic concepts behind PSO and GSO and bring out the differences between them. Figure 6(a) shows the trajectories of six agents, their current positions and the positions at which they encountered their personal best and the global best. Figure 6(b) shows the GSO decision for an agent which has four neighbours in its dynamic range of which two have higher luciferin value and thus only two probabilities are calculated. Figure 6(c) shows the PSO decision, which is a random combination between the current velocity of the agent, the vector to the personal best location and the vector to the global best location.

In PSO, the next velocity direction and magnitude is dependent on combination of the agent's own current velocity and randomly weighted global best vector and personal best vector. While this is implementable in a purely computational platform, implementation of this algorithm in robotics platform or a platform containing realistic agents would demand large speed fluctuations, presence of memory of the personal best position of each agent and knowledge of the global best position which requires global communication with all other agents. In the local variant of PSO, P_g is replaced by the best previous position encountered by particles in a local neighbourhood. In one of the local variants of PSO, the dynamic neighbourhood is achieved by evaluating the first k nearest neighbours. However, such a neighbourhood topology is also limited to computational models only and is not applicable in a realistic scenario where the neighbourhood size is defined by the limited sensor range of the mobile agents.

Figure 6 (a) Trajectories of six agents, their current positions, and the positions at which they encountered their personal best and the global best (b) GSO decision for an agent which has four neighbors in its dynamic range of which two have higher luciferin value and thus only two probabilities are calculated (c) PSO decision which is a random combination between the current velocity of the agent, the vector to the personal best location and the vector to the global best location



In GSO, the next direction of movement of an agent is determined by the position of the higher luciferin carrying neighbours within a dynamic decision range and the weights are determined by the actual values of the luciferin level. Thus, in GSO the implementation is much simpler as the algorithm demands communication only with a limited number of neighbours and therefore, does not require retaining personal best position information, nor does it require collecting data from all agents to determine the global best position.

Conceptually, the fact that GSO does not use the global best position or the personal best position and the fact that it uses information only from a dynamic neighbour set helps it to detect local maxima, whereas PSO gets easily attracted to the global maxima.

The above figures and explanation imply that GSO is a completely different algorithm than PSO although they have some minor commonalities. The main differences between GSO and PSO are summarised in Table 2.

Table 2 PSO versus GSO

	PSO	*GSO*
1	Net improvement in the objective function at iteration t stored in the global best position Pg(t)	A glowworm with the highest luciferin value in the swarm indicates its potential proximity to an optimum
2	Direction of movement based on previous best positions	Agent movement along line-of-sight with a neighbour
3	Neighbourhood range covers the entire search space	Maximum range hard limited by finite sensor range
4	Dynamic neighbourhood based on k nearest (in a local variant of PSO)	Adaptive neighbourhood based on varying range
5	Limited to numerical optimisation models	Can be applied to multiple source localisation in addition to numerical optimisation

7.2 Experimental comparison between PSO and GSO

In Krishnanand and Ghose (2008b), we have 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 the Niche-PSO and GSO on the modified Himmelblau's (J_7), equal-peaks-B (J_8) and the Rastrigin's (J_2) functions. A set of 30 trials was conducted for each test case. For comparison purpose, we used the following performance metrics:

1 number of runs R_i, for which at least $i(i = 1, 2, \ldots, Q)$ peaks are captured, for various values of n

2 average number of peaks captured as a function of n

3 mean distance travelled by an agent

4 computation time

5 total numbers of iterations for convergence.

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). For Niche-PSO, the performance improves as the number of agents increases

until $n = 40$ (when the number of runs for which all the 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 increase in the value of $n(R_4 = 24,$ which occurs at $n = 100)$. The above observation is also supported by the gradual increase in the average number of peaks captured with increase in n. A maximum average peak capture of 3.8 ± 0:5 was observed in the case of GSO as against 2.8 ± 1 in the case of Niche-PSO.

The Equal-peaks-B function (Krishnanand and Ghose, 2008b) has 12 peaks, of equal function value, in the search space of [–5, 5] × [–5, 5]. We get similar comparison results as in the experiments with Himmelblau's function. In the case of Niche-PSO, the performance degrades drastically above $n = 100$ where only one peak is captured most number of times with the exception of two peaks captures in a few instances. However, the performance steadily improves as n increases in the case of GSO ($R_{12} = 25$ for GSO, which occurs at $n = 500$ as against $R_{12} = 1$ for PSO, which occurs at $n = 100$). The maximum average number of peaks captured was approximately 12 in the case of GSO as against only 2 in the case of Niche-PSO.

The Rastrigin's function consists of 16 peaks, of unequal function values, in a search space of [–2, 2] × [–2, 2]. $R_{16} = 28$ for GSO, which occurs at $n = 500$ as against $R_{16} = 1$ for Niche-PSO, which occurs at $n = 300$. The maximum average number of peaks captured was 16 ± 0.1 in the case of GSO as against only 2 ± 2 in the case of Niche-PSO.

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 that leads to 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 increase in values of n, in terms of the average number of peaks captured. It was observed that the average distance travelled by an agent is relatively very low in the case of GSO when compared to that of Niche-PSO, in all the cases. Even though GSO takes more time than Niche-PSO at higher values of n, GSO performs better than Niche-PSO in terms of the number of peaks captured.

8 Noise issues, theoretical foundations and other applications

8.1 Influence by noisy sensor data

The GSO algorithm was tested in the presence of sensor noise in Krishnanand and Ghose (2008c). A Gaussian distribution was used to model the sensor noise. The GSO algorithm was compared with a gradient-ascent algorithm. The GSO algorithm showed good performance even with fairly high noise levels. There was a degradation of performance only with significant increase in levels of measurement noise. Whereas, gradient based algorithm degraded fast in the presence of noise.

8.2 Theoretical foundations

Some theoretical analysis of GSO has been presented in Krishnanand and Ghose (2008a), where we provided local convergence results under certain restricted set of assumptions. In particular, we considered local convergence of agents to leaders that may represent agents closer to the peaks than other agents [refer to Krishnanand and Ghose (2008a) for a more precise definition of a leader]. Note that multiple peaks may give rise to multiple leaders. A leader does not move until another agent crosses the equi-valued contour that passes through the leader's position or until another agent leapfrogs over the leader. We obtain the simplified model by making the following modifications to the actual GSO algorithm: Each agent i contains a constant luciferin value ℓ_i; the local-decision domain range is kept constant and is made equal to the maximum sensing range r_s; the step-size s is modified such that an agent reaches the leader's location in one step, whenever it is situated within a distance s to the leader.

A discussion is provided in Krishnanand and Ghose (2008a) justifying the fact that these modifications lead to a simplified model of the algorithm, making it amenable to analysis, while still reflecting most of the features of the original algorithm. We find a finite upper bound on the time taken by the agents to converge to an 'isolated-leader' [refer to Theorem 3 in Krishnanand and Ghose (2008a)] and on the time taken by the agents to converge to one of the leaders with 'non-isolated' and 'non-overlapping' neighbourhoods [refer to Theorem 4 in Krishnanand and Ghose (2008a)]. Finally, we show that agents under the influence of multiple leaders with overlapping neighbourhoods asymptotically converge to one of the leaders [refer to Theorem 5 in Krishnanand and Ghose (2008a)]. In all the above analysis, we assumed that the agents have constant luciferin values. However, the luciferin values of glowworms in actual GSO vary with time and their position in the search space. Therefore, we relax the constant luciferin assumption to some extent by varying the luciferin values as a function of the agent's position from a leader and show similar results [refer to Theorem 7 in Krishnanand and Ghose (2008a)].

8.3 Signal source localisation

Embodied Simulations and real-robot-experiments were used in Krishnanand (2007) and Krishnanand and Ghose (2008c) to demonstrate the potential of GSO for signal source localisation applications. The problem involves the deployment of a group of mobile robots that use their sensory perception of signal-signatures at distances that are potentially far from a source and interaction with their neighbours as cues to guide their movements toward and eventually co-locate at, the signal-emitting source. The GSO, originally designed for solving optimisation problems, offers itself as one of the swarm robotics approaches to the above problem. Certain algorithmic aspects need modifications while implementing in a robotic network mainly because of the point-agent model of the basic GSO algorithm and the physical dimensions and dynamics of a real robot. Embodied robot simulations were carried out in order to assess GSO's suitability for multiple source localisation tasks. Next, we extended this work to collective robotics experiments. For this purpose, we used a set of four wheeled robots, called Kinbots (Krishnanand and Ghose, 2005c), that were originally built for experiments related to robot formations. By making necessary modifications to the Kinbot hardware, the robots

are endowed with the capabilities required to implement the various behavioural primitives of GSO. We presented the results from an experiment where two Kinbots use GSO to localise a light source.

8.4 Application to ubiquitous environments

The basic GSO algorithm, which was applied to multimodal optimisation and signal source localisation problems, considers a homogeneous swarm of mobile agents. However, in Krishnanand and Ghose (in press), we developed a GSO variant for a heterogeneous swarm of mobile and stationary agents with applications to ubiquitous environments. In particular, we proposed a GSO based ubiquitous environment to address the problem of sensing hazards that manifest themselves in various forms such as nuclear leaks, healthrisk causing gas or inflammable gas leakages in factories, fires caused by electrical shortcircuits/sparks, etc. The ubiquitous environment constitutes deployment of a heterogeneous swarm of nano-agents into the region of interest. The heterogeneous swarm consists of two types of agents:

1 'stationary nano-agents' that are embedded into the environment and act only as fixed sensors at locations of their deployment

2 'mobile nano-agents' that use the information cues received from stationary and/or mobile neighbours in order to find their way to the source locations of the hazard.

The agents deployed in the environment implement a modification to the original GSO algorithm. This modified algorithm is unlike the previous GSO implementations where homogeneous mobile agent swarms are considered. Extensive simulations in various experimental scenarios and comparison of results with that of homogeneous swarms demonstrate the significance of supplementing a mobile agent group with a large number of inexpensive stationary agents for hazard sensing applications. In a graph of minimum number of mobile agents required for 100% source-capture as a function of the number of stationary agents, we showed that deployment of the stationary agents in a grid-configuration leads to multiple phase-transitions in the heterogeneous swarm behaviour.

8.5 Pursuit of multiple mobile signal sources

In Krishnanand and Ghose (2007), we investigated the behaviour of agents that implement the GSO algorithm when mobile sources are considered. In particular, we use GSO to develop a coordination scheme that enables a swarm of mobile pursuers, with hard-limited sensing ranges, to split into subgroups, exhibit simultaneous taxis toward and eventually pursue a group of mobile signal sources. Examples of such sources include hostile mobile targets in a battlefield and moving fire-fronts that are created in forest fires. We assume that the mobile source radiates a signal whose intensity peaks at the source location and decreases monotonically with distance from the source. For the case where the positions of the pursuers and the moving source are collinear, we presented a theoretical result that provides an upper bound on the relative speed of the mobile source below which the pursuers succeed in chasing the source. We used simulations to demonstrate the efficacy of the algorithm in addressing these pursuit problems. In particular, we presented simulation results for single and two source cases,

respectively, where each source moves in a circular trajectory and at constant angular speed. In the case where the positions of the pursuers and the moving source are non-collinear, we used numerical experiments to determine an upper bound on the relative speed of the mobile source below which the pursuers succeed in chasing the source.

9 Conclusions

GSO is a new method that was recently developed for finding multiple optima of multimodal functions. We have here discussed all the major results available on GSO till date and presented a step-by-step development of the algorithm from its genesis to its current formulation. We have also provided the basic algorithmic differences between GSO and PSO and compared their performances in terms of capturing multiple peaks of several benchmark multimodal functions.

Acknowledgements

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

References

Brits, R., Engelbrecht, A.P. and 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.

Casbeer, D.W., Beard, R.W., McLain, T.W., Li, S-M. and Mehra, R.K. (2005) 'Forest fire monitoring with multiple small uavs', in *Proceedings of American Control Conference*, pp.3530–3535.

Clark, J. and Fierro, R. (2005) 'Cooperative hybrid control of robotic sensors for perimeter detection and tracking', in *Proceedings of American Control Conference*, pp.3500–3505.

Clerc (2007) *Particle Swarm Optimization*, ISTE Ltd, London, UK.

Dréo, J. and Siarry, P. (2004) 'Continuous interacting ant colony algorithm based on dense heterarchy', *Future Generations Computer Systems*, Vol. 20, pp.841–856.

Farrell, J., Li, W., Pang, S. and Arrieta, R. (2003) 'Chemical plume tracing experimental results with a remus auv', in *Oceans 2003 Marine Technology and Ocean Science Conference*, pp.962–968.

Fevrier, V. and 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.

Fronczek, J.W. and 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.

Goldberg, D. and Richardson, J. (1987) 'Genetic algorithms with sharing for multi-modal function optimization', in *Genetic Algorithms and Their Applications: Proceedings of the Second International Conference on Genetic Algorithms*, pp.44–49.

Harick, G. (1997) 'Finding multi-modal solutions using restricted tournament selection', in *Proceedings of the Sixth International Conference on Genetic Algorithms*, pp.24–31.

Horn, J., Goldberg, D.E. and Deb, K. (1994) 'Implicit niching in a learning classifier system: nature's way', *Evolutionary Computation*, Vol. 2, No. 1, pp.37–66.

Jakuba, M.V. (2007) 'Stochastic mapping for chemical plume source localization with application to autonomous hydrothermal vent discovery', PhD thesis.

Kennedy, J. (2000) 'Stereotyping: improving particle swarm performance with cluster analysis', in *Proceedings of the Congress on Evolutionary Computation*, pp.1507–1512.

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. and Ghose, D. (2005a) 'Detection of multiple source locations using a glowworm metaphor with applications to collective robotics', in *Proceedings of IEEE Swarm Intelligence Symposium*, pp.84–91.

Krishnanand, K.N. and Ghose, D. (2005b) 'Multimodal function optimization using a glowworm metaphor with applications to collective robotics', in *Proceedings of the Second Indian International Conference on Artificial Intelligence*, pp.328–346.

Krishnanand, K.N. and Ghose, D. (2005c) 'Formations of minimalist mobile robots using local-templates and spatially distributed interactions', *Robotics and Autonomous Systems*, Vol. 53, pp.194–213.

Krishnanand, K.N. and Ghose, D. (2006) 'Glowworm swarm based optimization algorithm for multimodal functions with collective robotics applications', *Multiagent and Grid Systems*, Vol. 2, No. 3, pp.209–222.

Krishnanand, K.N. and Ghose, D. (2007) 'Chasing multiple mobile signal sources: a glowworm swarm optimization approach', in *Third Indian International Conference on Artificial Intelligence (IICAI 07)*.

Krishnanand, K.N. and Ghose, D. (2008a) 'Theoretical foundations for rendezvous of glowworm-inspired agent swarms at multiple locations', *Robotics and Autonomous Systems*, Vol. 56, No. 7, pp.549–569.

Krishnanand, K.N. and Ghose, D. (2008b) 'Glowworm swarm optimization for simultaneous capture of multiple local optima of multimodal functions', *Swarm Intelligence*, to appear.

Krishnanand, K.N. and Ghose, D. (2008c) 'A glowworm swarm optimization based multi-robot system for signal source localization', *Design and Control of Intelligent Robotic Systems*, pp.53–74.

Krishnanand, K.N. and Ghose, D. (in press) 'Glowworm swarm optimization algorithm for hazard sensing in ubiquitous environments using heterogeneous agent swarms', in *Soft Computing Applications in Industry*, Springer-Verlag.

Krishnanand, K.N., Amruth, P., Guruprasad, M.H., Bidargaddi, S.V. and 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.): *Ant Colony Optimization and Swarm Intelligence*, Lecture Notes in Computer Science, pp.259–269, LNCS 4150, Springer, Berlin, Germany.

Krishnanand, K.N., Amruth, P., Guruprasad, M.H., Bidargaddi, S.V. and 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.

Li, X. (2004) 'Adaptively choosing neighbourhood bests using species in a particle swarm optimizer for multimodal function optimization', in *Proceedings of the Genetic and Evolutionary Computation Conference*, pp.105–116, Springer Verlag.

Mahfoud, S. (1992) 'Crowding and preselection revisited', *Parallel Problem Solving, Nature 2*, pp.27–37.

Mahfoud, S. (1995) 'Niching method for genetic algorithms', PhD thesis.

Mengsheol, O. and Goldberg, D. (1999) 'Probabilistic crowding: deterministic crowding with probabilistic replacement', in *Proceedings of the Genetic and Evolutionary Computation Conference*, pp.409–416.

Miller, B.L. and Shaw, M.J. (1996) 'Genetic algorithms with dynamic niche sharing for multimodal function optimization', in *Proceedings of the IEEE International Conference on Evolutionary Computation*, pp.786–791.

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

Muller, S.D., Marchetto, J. and Koumoutsakos, S.A.P. (2002) 'Optimization based on bacterial chemotaxis', *IEEE Transactions on Evolutionary Computation*, Vol. 6, No. 6, pp.16–29.

Parsopoulos, K. and Vrahatis, M.N. (2004) 'On the computation of all global minimizers through particle swarm optimization', *IEEE Transactions on Evolutionary Computation*, Vol. 8, No. 3. pp.211–224.

Petrowski, A. (1996) 'A clearing procedure as a niching method for genetic algorithms', in *Proceedings of Third IEEE International Conference on Evolutionary Computation*, pp.798–803.

Reutskiy, S.Y. and 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*, Vol. 67, No. 13, pp.1811–1829.

Sedeno, W. and Vemuri, V.R. (1997) 'On the use of niching for dynamic landscapes', in *Proceedings of the IEEE International Conference on Evolutionary Computation*, pp.361–366.

Sikora, R. and Shaw, M.J. (1994) 'A double-layered learning approach to acquiring rules for classification: integrating genetic algorithms with similarity-based learning', *ORSA Journal on Computing*, Vol. 6, No. 2, pp.174–187.

Singh, G. and 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.

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

Törn, A. and Zilinskas, A. (1989) *Global Optimization*, Springer Verlag, New York, NY.

Zarzhitsky, D., Spears, D.F. and Spears, W.M. (2005) 'Swarms for chemical plume tracing', in *Proceedings of IEEE Swarm Intelligence Symposium*, pp.249–256.

Notes

1 Movies of the simulations presented in this paper can be viewed at our lab website: http://guidance.aero.iisc.ernet.in/gso.html.