Applied Soft Computing xxx (2015) xxx–xxx

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Galactic Swarm Optimization: A new global optimization metaheuristic inspired by galactic motion

Venkataraman Muthiah-Nakarajan*, Mathew Mithra Noel

School of Electrical Engineering, VIT University, India

ARTICLE INFO

Article history:
Received 16 January 2015
Received in revised form 17 August 2015
Accepted 16 October 2015
Available online xxx

Keywords:
Global optimization
Galactic Swarm Optimization
Metaheuristics
Benchmark functions
Stochastic optimization

ABSTRACT

This paper proposes a new global optimization metaheuristic called Galactic Swarm Optimization (GSO) inspired by the motion of stars, galaxies and superclusters of galaxies under the influence of gravity. GSO employs multiple cycles of exploration and exploitation phases to strike an optimal trade-off between exploration of new solutions and exploitation of existing solutions. In the explorative phase different subpopulations independently explore the search space and in the exploitative phase the best solutions of different subpopulations are considered as a superswarm and moved towards the best solutions found by the superswarm. In this paper subpopulations as well as the superswarm are updated using the PSO algorithm. However, the GSO approach is quite general and any population based optimization algorithm can be used instead of the PSO algorithm. Statistical test results indicate that the GSO algorithm proposed in this paper significantly outperforms 4 state-of-the-art PSO algorithms and 4 multiswarm PSO algorithms on an overwhelming majority of 15 benchmark optimization problems over 50 independent trials and up to 50 dimensions. Extensive simulation results show that the GSO algorithm proposed in this paper converges faster to a significantly more accurate solution on a wide variety of high dimensional and multimodal benchmark optimization problems.

1. Introduction

For a myriad of applications such as training of neural nets [1–4], power system optimization [5–7], and signal analysis [8,9] the cost function to be minimized is multimodal (many local minima) and depends on numerous variables (high dimensional). Hence, a global optimization technique that can yield robust performance for high dimensional and multimodal functions is of long standing interest. Global optimization can be defined mathematically as:

$$\text{Minimize } f(\mathbf{x}) \tag{1}$$

where $f : \Re^D \rightarrow \Re$ is the function to be minimized and D is the dimensionality of $\mathbf{x}$.

It is well known that classical deterministic search methods get stuck in local minimum and do not perform well on high dimensional problems. Heuristic approaches like Genetic Algorithms (GA) [10], Tabu Search (TS) [11,12], Ant Colony Optimization (ACO) [13], Simulated Annealing (SA) [14], Particle Swarm Optimization (PSO) [15] use stochasticity to escape from local minima and are more effective in locating an approximation to the global minimum. The general global optimization problem is a NP-complete problem and hence accurate computation of the global minimum will require exorbitant computational resources. Thus heuristic based algorithms that compute a good local minimum are of interest.

This paper addresses the frequently arising problem of single objective optimization of multimodal functions by a new metaheuristic framework called Galactic Swarm Optimization (GSO). This framework increases the efficiency and efficacy of the embedded algorithm by providing multiple cycles of exploration and exploitation, thereby improving the odds of locating the global minimum accurately.

The GSO algorithm draws inspiration from the motion of stars inside galaxies and motions of clusters and superclusters of galaxies. Analogous to the motion of stars in galaxies under the attractive influence of large masses, the population of potential solutions is divided into subpopulations where each individual solution is attracted towards better solutions. However on a larger scale, entire galaxies appear as point masses and are attracted to other galaxies. In the GSO algorithm this is implemented by treating the best solution found by individual subpopulations as a superswarm. Movement of members of the population towards better solutions can be implemented with other population based strategies, like GAs.

Since PSO algorithm has been used as the demonstration vehicle the proposed strategy is termed as Galactic Swarm Optimization (GSO). However, the strategy by itself can be easily extended to other population based algorithms. The choice of PSO algorithm in this work relies on the fact that, it

1 can be realized using few lines of code,
2 has faster convergence rate and
3 free from complex constructs such as mutation/crossover or pheromone.

The GSO algorithm continues to possess the above mentioned desirable characteristics. Experimental results consistently support the enhanced ability of the proposed algorithm on multimodal cost functions when compared with the other state-of-the-art PSO versions. The proposed algorithm is also competitive in terms of running time and complexity, measured in terms of function evaluations.

The rest of the paper is organized as follows. Section 2 reviews the original PSO algorithm and associated limitations. In Section 3, the structure and intuition behind GSO algorithm is presented. In Section 4, experimental results are summarized and

* Corresponding author. Tel.: +91 9884520878; fax: +91 416 2243092.
E-mail address: mnvenkataraman@vit.ac.in (V. Muthiah-Nakarajan).

http://dx.doi.org/10.1016/j.asoc.2015.10.034
1568-4946/

discussed. Finally, Section 5 concludes the paper and gives possible future research directions.

2. Particle Swarm Optimization

The PSO algorithm mimics the behaviour of birds and some other species that collectively search for food [15,16]. The initial swarm or population is a set of points in the D-dimensional search space. The components of the vector $\mathbf{x}$ are real numbers. The PSO algorithm is initialized with population of particles (or swarm) randomly inside the search space. The swarm movement is guided by the equation stated below:

$$\mathbf{v}_{id}(k+1) \leftarrow \mathbf{v}_{id}(k) + c_1\phi_{1d}(\mathbf{p}_{id}(k) - \mathbf{x}_{id}(k)) + c_2\phi_{2d}(\mathbf{g}_d(k) - \mathbf{x}_{id}(k)) \quad (2)$$

$$\mathbf{x}_{id}(k+1) \leftarrow \mathbf{x}_{id}(k) + \mathbf{v}_{id}(k) \quad (3)$$

where $\mathbf{v}_i = [v_{i1}, v_{i2}, \cdots, v_{iD}]$ is termed as velocity of the particle i; $\mathbf{x}_i = [x_{i1}, x_{i2}, \cdots, x_{iD}]$ is the position of the particle i. ϕ_{1d} and ϕ_{2d} are uniformly distributed random numbers which differ for each dimension d. For the PSO version used in the paper, the constants ϕ do not change with dimension. Associated with each particle $\mathbf{x}^{(i)}$, there is a personal best $\mathbf{p}^{(i)}$, a particle that fetched the least function value during one of the past iterations. Associated with the entire swarm $\mathbf{x}$, there is a global best $\mathbf{g}$, a point that gave the least function value during prior movements. c_1 and c_2 are the acceleration constants that pull the flying direction towards the local best and global best solutions, respectively.

Eq. (2) gives the velocity of the next iteration for a given $\mathbf{x}^{(i)}$, $\mathbf{p}^{(i)}$ and $\mathbf{g}$. The velocity and position vectors are bounded by the same limits to avoid random relocation of the particle, subsequently losing its normal path and momentum [17]. Eq. (3) indicates that the velocity term is added to the position to update the new position. The personal best is updated with new $\mathbf{x}^{(i)}$ if $f(\mathbf{x}^{(i)}) < f(\mathbf{p}^{(i)})$. In each iteration, the global best is updated with the personal best, if the personal best assumes a smaller function value $f(\mathbf{p}^{(i)}) < f(\mathbf{g})$.

Currently there are numerous varieties of PSOs in existence and for a detailed historical summary the reader is referred to [18,19]. For the PSO version used in this paper inertial weight ω is used to narrow the search space with time. The form of inertial weight used in this paper is different from [20] as better experimental results were obtained with the modified inertial weights used in this paper. The Eqs. (2) and (3) are modified as follows:

$$\mathbf{v}^{(i)}(k+1) \leftarrow \omega\mathbf{v}^{(i)}(k) + c_1\phi_1(\mathbf{p}^{(i)}(k) - \mathbf{x}^{(i)}(k)) + c_2\phi_2(\mathbf{g}(k) - \mathbf{x}^{(i)}(k)) \quad (4)$$

$$\mathbf{x}^{(i)}(k+1) \leftarrow \mathbf{x}^{(i)}(k) + \mathbf{v}^{(i)}(k) \quad (5)$$

ω is reduced linearly from unity to nearly zero with each iteration.

As the inertial weight is gradually reduced the explorative ability of the PSO algorithm decreases and the PSO algorithm might prematurely converge to a poor local minimum. The different versions of PSO algorithm discussed in this paper and elsewhere try to address the problem of premature convergence by making particles learn from the historical best of other particles. The efforts in this direction have yielded limited success.

One of the common but important issues with heuristic algorithms, including PSO algorithms is the issue of striking a balance between the exploration and exploitation, each of which is obtained at the cost of the other. Most of the heuristic algorithms, including the PSO variants, work under the premise of highly explorative action at the beginning with a gradual transition to highly exploitative action towards the end. Exploration allows a search algorithm to come close to the global minimum or good local minimum and exploitation helps locate the global minimum accurately. Most optimization algorithms have a learning rate parameter that is reduced with time to explore many potential local minima in the beginning and accurately compute a good solution at the end. The scheme breaks down on multimodal functions if the algorithm converges to a poor local minimum during the exploration stage resulting in suboptimal solution.

Galactic Swarm Optimization (GSO) provides multiple cycles of exploration and exploitation by dividing the search in terms of epochs, providing the algorithm more opportunity to accurately locate the global minimum.

3. The Galactic Swarm Optimization (GSO) Algorithm

The original PSO algorithm by Eberhart and Kennedy was inspired by swarming like flocking behaviour in birds and schooling behaviour in fish. Both of these swarms do not perform any global optimization and only provide a mechanism for confusing predators and predator avoidance. Since the original PSO algorithm itself is motivated by swarming behaviour that is associated with swarms that do not perform global optimization it is not necessary for other PSO inspired heuristics as well as they perform well on benchmark test functions.

The GSO algorithm emulates the movement of stars, galaxies and superclusters of galaxies in the cosmos [21]. Stars are not distributed uniformly in the cosmos but clustered into galaxies which in turn are not uniformly distributed. On a large enough scale individual galaxies appear as point masses. The attraction of stars inside a galaxy to large masses and galaxies themselves to other large masses is emulated in the GSO algorithm as follows: First, individuals in each subpopulation are attracted to better solutions in the subpopulation according to the PSO algorithm. Secondly, each subpopulation is assumed to be represented by the best solution found by the subpopulation and treated as a superswarm. The individual in the superswarm comprising of the best solutions found by each subpopulation also move according to the PSO algorithm. This approach is general and movement of the swarm and superswarm can be accomplished using other population based optimization algorithms. This is analogous to the members of a particular swarm being attracted towards the global best.

On a larger scale, individual galaxies appear to be point masses and can be clustered with neighbouring galaxies to form superclusters of galaxies. In the GSO algorithm a galaxy of stars is analogous to a subswarm and a cluster of galaxies is analogous to the superswarm. The cluster of galaxies is identified with the Centre of Mass (CM) of the galaxies. Similarly each individual in the superswarm represents the global best solution of individual subswarms.

This analogy of stars clustered into galaxies and galaxies clustered into clusters and superclusters can be extended further and concepts such as super superclusters can be introduced. However in this paper the analogy is restricted to galaxies of stars and clusters of galaxies.

In the GSO algorithm the swarm is a set X of $D-$tuples containing elements ($\mathbf{x}_j^{(i)} \in \Re^D$) consists of M partitions, called subswarms X_i, each of size N. The elements of X are initialized randomly within the search space $[x_{min}, x_{max}]^D$. The complete swarm framework is defined by:

$$X_i \subset X : i = 1, 2, \cdots, M$$

$$\mathbf{x}_j^{(i)} \in X_i : j = 1, 2, \cdots, N$$

$$X_i \cap X_j = \phi : \text{ if } i \neq j$$

$$\bigcup_{i=1}^{M} X_i = X$$

X_i is a swarm of size N. The velocity and personal best associated with each particle $\mathbf{x}_j^{(i)}$ are represented by $\mathbf{v}_j^{(i)}$ and $\mathbf{p}_j^{(i)}$, respectively. The nomenclature used in this paper is summarized in Table 1 for convenience.

At the first level of clustering, PSO is performed independently for each subswarm. Since the swarm X is subdivided in to M groups,

Please cite this article in press as: V. Muthiah-Nakarajan, M.M. Noel, Galactic Swarm Optimization: A new global optimization metaheuristic inspired by galactic motion, Appl. Soft Comput. J. (2015), http://dx.doi.org/10.1016/j.asoc.2015.10.034

Table 1
Nomenclature.

f	Cost function
X	Set of all particles $\mathbf{x}_j^{(i)}$ in the swarm
D	Dimension of the search space
X_i	Subset of particles belonging to subswarm i
$\mathbf{x}_j^{(i)}$	Position (Members of subswarm) of particle $\mathbf{x}_j^{(i)}$
$\mathbf{v}_j^{(i)}$	Velocity of particle $\mathbf{x}_j^{(i)}$
$\mathbf{p}_j^{(i)}$	Personal best of particle $\mathbf{x}_j^{(i)}$
$\mathbf{g}^{(i)}$	Global best solution of subswarm i
$\mathbf{y}^{(i)}$	Member i of superswarm
$\mathbf{v}^{(i)}$	Velocity of member i superswarm
$\mathbf{g}$	Global best solution for the entire swarm X
EP_{max}	Maximum number of epochs
M	Number of partitions of X
N	Size of set X_i
c_i	Acceleration coefficient
r_i	Uniformly distributed random number between −1 and 1
ω_i	Inertial weight

the PSO algorithm is run for M times. Each subswarm X_i has an associated global best $\mathbf{g}^{(i)}$ and is updated if any of its personal bests $\mathbf{p}_j^{(i)}$ happen to take a smaller function value than $\mathbf{g}^{(i)}$, $f(\mathbf{p}_j^{(i)}) < f(\mathbf{g}^{(i)})$. Particles in each subswarm are attracted stochastically to the best solution found by that particular subswarm. Since the subswarm gbest usually lies in the vicinity of some local minimum the particles in each subswarm are stochastically attracted towards that local minimum. Also the subswarms do not share good solutions as in other approaches so each swarm is attracted towards its own subswarm best (local minimum) which in some cases might overlap. In the GSO algorithm strict subdivision of search region is not imposed.

The movement of a subswarm in X_i is independent and has no influence on another subswarm X_j for $i \neq j$, thereby allowing an unaffected and comprehensive search possible. To take full advantage of this new found exploration ability of multiple subswarms, a galactic best is defined $\mathbf{g}$ which is updated whenever any of the global bests $\mathbf{g}^{(i)}$ assumes a lower function value, $f(\mathbf{g}^{(i)}) < f(\mathbf{g})$. The GSO algorithm keeps record of its best solution by updating $\mathbf{g}$.

Rather than having a single swarm exploring towards a particular direction, when multiple swarms are present a synergistic effect can be observed resulting in a much improved exploration.

Each subswarm independently explores the search space freely on its own. The iteration begins by computing the velocity and position. The expressions for velocity and position updates are:

$$\mathbf{v}_j^{(i)} \leftarrow \omega_1 \mathbf{v}^{(i)} + c_1 r_1 \left(\mathbf{p}_j^{(i)} - \mathbf{x}_j^{(i)}\right) + c_2 r_2 \left(\mathbf{g}^{(i)} - \mathbf{x}_j^{(i)}\right) \tag{6}$$

$$\mathbf{x}_j^{(i)} \leftarrow \mathbf{x}_j^{(i)} + \mathbf{v}_j^{(i)} \tag{7}$$

where the inertial weight ω_1, and the random numbers r_1 and r_2 are given by

$$\omega_1 = 1 - \frac{k}{L_1 + 1} \tag{8}$$

$$r_i = U(-1, 1) \tag{9}$$

k is the current integer iteration number that varies from 0 to L_1. Eq. (9) indicates that r_i is a random number chosen in the range between −1 and 1.

Global bests participate in the next stage of clustering to form the superclusters. Analogously, a new superswarm Y is created by collecting the global bests from subswarms X_i.

$$\begin{aligned} &\mathbf{y}^{(i)} \in Y : i = 1, 2, \ldots, M \\ &\mathbf{y}^{(i)} = \mathbf{g}^{(i)} \end{aligned} \tag{10}$$

The velocity $\mathbf{v}^{(i)}$ and position vectors $\mathbf{y}^{(i)}$ are updated as per the equations given below:

$$\mathbf{v}^{(i)} \leftarrow \omega_2 \mathbf{v}^{(i)} + c_3 r_3 (\mathbf{p}^{(i)} - \mathbf{y}^{(i)}) + c_4 r_4 (\mathbf{g} - \mathbf{y}^{(i)}) \tag{11}$$

$$\mathbf{y}^{(i)} \leftarrow \mathbf{y}^{(i)} + \mathbf{v}^{(i)} \tag{12}$$

where $\mathbf{p}^{(i)}$ is the personal best associated with the vector $\mathbf{y}^{(i)}$. The defining relations of ω_2, r_3 and r_4 are similar to equations Eqs. (8) and (9). At this level $\mathbf{g}$ serves as the global best exemplar and is not updated unless the search locates a better point. Since the superswarm concentrates on the global bests from subswarm, it can enhance the exploitation.

The superswarm uses the best solution already computed by the subswarms and hence exploits information already computed. Although the individuals of the superswarm are more spread out compared to the subswarm individuals, the superswarm does not perform independent exploration. The following analogy provides some insight. Consider a team leader summarizing the findings of his team members, the leader does not come up with independent ideas or explores new solutions but simply exploits information given to him by his team members.

The global bests of the subswarms influence the superswarms, but there is no feedback effect or information flow from the superswarm to the subswarms to preserve diversity of solutions. Avoiding feedback helps the GSO strategy to retain highly diverse subswarms and constant global search ability every epoch.

When the next epoch begins, the search starts from where it left off and continues exploring the space again with the same rigor – the subswarms are not restarted. The first level of the GSO algorithm consisting of motion of the subswarms is primarily an exploratory phase and the second level involving the superswarm is primarily an exploitative phase. Thus the GSO algorithm alternates between exploratory and exploitative phases. Hence, the GSO algorithm gets opportunity to search many more local minima; this is important since any local minimum can potentially be the global minimum. This repeated exploration–exploitation cycle can be intuitively reasoned to be responsible for the superior performance of the GSO algorithm on highly multimodal test functions.

3.1. Analogy between galactic motion and optimization

The relation between galactic motion and optimization is as follows. Firstly, stellar masses in a galaxy experience a force which is given by the negative gradient of the gravitational potential energy. Thus, stellar masses experience a pull in the direction of greatest decrease in the gravitational potential energy, leading to an optimization like behaviour. In particular the well-known Hamilton's Principle of Least Action states that motion of a system of masses always takes place in such a way so as to minimize or maximize the time integral of the Lagrangian function.

Individual masses gravitate towards large masses like in our solar system where individual planets are primarily attracted towards the most massive neighbouring object namely the Sun. In the GSO algorithm this is implemented by moving particles stochastically towards better solutions. Also there are two types of motion associated with galaxies: first is the motion of individual masses inside galaxies and second in the motion of the centre of mass (CM) of the galaxies themselves. In the GSO algorithm this is implemented by updating all the subswarms in Level 1 and then forming and updating the superswarm in Level 2. Motion of the superswarm is analogous to the motion of the centre of masses of the galaxies. This is a good analogy since the CM is not a physical point but a mathematically defined average just like the members of the superswarm.

Please cite this article in press as: V. Muthiah-Nakarajan, M.M. Noel, Galactic Swarm Optimization: A new global optimization metaheuristic inspired by galactic motion, Appl. Soft Comput. J. (2015), http://dx.doi.org/10.1016/j.asoc.2015.10.034

Algorithm 1. GSO(f)

Level 1 Initialization: $\mathbf{x}_j^{(i)}, \mathbf{v}_j^{(i)}, \mathbf{p}_j^{(i)}, \mathbf{g}^{(i)}$ within $[x_{min}, x_{max}]^D$ randomly.
Level 2 Initialization: $\mathbf{v}^{(i)}, \mathbf{p}^{(i)}, \mathbf{g}$ within $[x_{min}, x_{max}]^D$ randomly.
for $EP \leftarrow 1$ **to** EP_{max}
do
- Begin PSO: Level 1
- **for** $i \leftarrow 1$ **to** M
  - **do**
    - **for** $k \leftarrow 0$ **to** L_1
      - **do**
        - **for** $j \leftarrow 1$ **to** N
          - **do**
            - $\mathbf{v}_j^{(i)} \leftarrow \omega_1 \mathbf{v}_j^{(i)} + c_1 r_1 (\mathbf{p}_j^{(i)} - \mathbf{x}_j^{(i)}) + c_2 r_2 (\mathbf{g}^{(i)} - \mathbf{x}_j^{(i)})$;
            - $\mathbf{x}_j^{(i)} \leftarrow \mathbf{x}_j^{(i)} + \mathbf{v}_j^{(i)}$;
            - **if** $f(\mathbf{x}_j^{(i)}) < f(\mathbf{p}_j^{(i)})$
              - **then**
                - $\mathbf{p}_j^{(i)} \leftarrow \mathbf{x}_j^{(i)}$;
                - **if** $f(\mathbf{p}_j^{(i)}) < f(\mathbf{g}^{(i)})$
                  - **then**
                    - $\mathbf{g}^{(i)} \leftarrow \mathbf{p}_j^{(i)}$;
                    - **if** $f(\mathbf{g}^{(i)}) < f(\mathbf{g})$
                      - **then** $\mathbf{g} \leftarrow \mathbf{g}^{(i)}$;
- Begin PSO: Level 2
- Initialize Swarm $\mathbf{y}^{(i)} = \mathbf{g}^{(i)} : i = 1, 2, \cdots, M$;
- **for** $k \leftarrow 0$ **to** L_2
  - **do**
    - **for** $i \leftarrow 1$ **to** M
      - **do**
        - $\mathbf{v}^{(i)} \leftarrow \omega_2 \mathbf{v}^{(i)} + c_3 r_3 (\mathbf{p}^{(i)} - \mathbf{y}^{(i)}) + c_4 r_4 (\mathbf{g} - \mathbf{y}^{(i)})$;
        - $\mathbf{y}^{(i)} \leftarrow \mathbf{y}^{(i)} + \mathbf{v}^{(i)}$;
        - **if** $f(\mathbf{y}^{(i)}) < f(\mathbf{p}^{(i)})$
          - **then**
            - $\mathbf{p}^{(i)} \leftarrow \mathbf{y}^{(i)}$;
            - **if** $f(\mathbf{p}^{(i)}) < f(\mathbf{g})$
              - **then** $\mathbf{g} \leftarrow \mathbf{p}^{(i)}$;

Return $\mathbf{g}, f(\mathbf{g})$

The position of the galactic best after the final epoch $\mathbf{g}$ and its fitness value $f(\mathbf{g})$ are returned as locations of minimum and cost of minimum respectively by the algorithm. The complete pseudocode is given in Algorithm 1.

3.2. Comparison of the GSO and multiswarm PSO algorithms

Just like the center of mass (CM) of a system of particles the superswarm has no individual or independent existence but only formed temporarily in Level 2 of the GSO algorithm. Since the superswarm exists only temporarily during Level 2 in the GSO algorithm, this is conceptually as well as computationally different from other multi-population or multi-swarm approaches which usually involve a masterswarm that runs continuously and independently while periodically sharing information with the slaveswarms.

Fundamental differences between GSO algorithm and a host of many other multi-swarm PSO variants described in [22–26] are that the flow of information between the subswarms and the superswarm is unidirectional to promote exploration and avoid premature convergence. This one way causality implies that the superswarm does not affect the exploration of the subswarms by injecting good solutions like the multiswarm approaches. Solutions computed by the superswarm are not reinserted or into the subswarms (or the subswarms are regrouped) to promote independent exploration of the subswarms.

This effect can also be observed in teams of human problem solvers; if different teams working on a problem share information all of them might converge to the first good solution identified by any one team. Since this independence of subpopulations leading to better performance appears to be a general phenomenon one might formally enunciate it as "**the principle of independence of subpopulations of problem solvers**".

The superswarm is created from the global bests of individual subswarms, evolves for a period of time, updates the global best solution and dies at the end of Level 2. Thus a new superswarm is created again after the completion of Level 1 each time. Level 1 is primarily an exploratory phase where potential good local minima are identified and Level 2 is an exploitative phase. Superswarm exists only temporarily during Level 2 of the GSO algorithm and has no independent or continuous existence unlike master and slave swarms shown in Niu et al. [27].

The superswarm was inspired by the concept of CM of a collection of particles (CM of individual galaxies). The location of the CM of a collection of particles does not necessarily coincide with the location of a physical particle but is a mathematically computed average location having no independent physical existence. Similarly the individuals comprising the superswarm have no individual or continuous existence but are created from the subswarm global best solutions and die after completion of Level 1. Thus, the motion of the superswarm represents the motion of galactic CM points.

4. Summary of results and discussion

Tables 2 and 3 list the benchmark functions that are used to compare different algorithms. All functions given in the table have zero as the global minimum value. The search range is kept wider than usual for Sphere, Rosenbrock, Rastrigin, Ackley and Weierstrass functions to demonstrate the search ability of GSO algorithm

Please cite this article in press as: V. Muthiah-Nakarajan, M.M. Noel, Galactic Swarm Optimization: A new global optimization metaheuristic inspired by galactic motion, Appl. Soft Comput. J. (2015), http://dx.doi.org/10.1016/j.asoc.2015.10.034

Table 2
Details of test functions used for comparison in this study.

Function (Type)	Definition	Search range	Global minimum location $\mathbf{x}^*$
Sphere (Unimodal Convex)	$f_1(\mathbf{x}) = \sum_{i=1}^{D} x_i^2$	$[-100, 100]^D$	$[0, 0, \ldots, 0]$
Rosenbrock (Multimodal)	$f_2(\mathbf{x}) = \sum_{i=1}^{D-1} \left[100(x_{i+1} - x_i^2)^2 + (1 - x_i)^2\right]$	$[-30, 30]^D$	$[1, 1, \ldots, 1]$
Rastrigin (Multimodal)	$f_3(\mathbf{x}) = 10D + \sum_{i=1}^{D} \left[x_i^2 - 10\cos(2\pi x_i)\right]$	$[-100, 100]^D$	$[0, 0, \ldots, 0]$
Rotated Rastrigin (Multimodal)	$f_4(\mathbf{y}) = 10D + \sum_{i=1}^{D} \left[y_i^2 - 10\cos(2\pi y_i)\right]$ $\mathbf{y}=\mathbf{M}^*\mathbf{x}$	$[-100, 100]^D$	$[0, 0, \ldots, 0]$
Griewangk (Multimodal)	$f_5(\mathbf{x}) = \sum_{i=1}^{D} \frac{x_i^2}{4000} - \prod_{i=1}^{D} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	$[-600, 600]^D$	$[0, 0, \ldots, 0]$
Rotated Griewangk (Multimodal)	$f_6(\mathbf{y}) = \sum_{i=1}^{D} \frac{y_i^2}{4000} - \prod_{i=1}^{D} \cos\left(\frac{y_i}{\sqrt{i}}\right) + 1$ $\mathbf{y}=\mathbf{M}^*\mathbf{x}$	$[-600, 600]^D$	$[0, 0, \ldots, 0]$
Ackley (Multimodal)	$f_7(\mathbf{x}) = 20 - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)\right) + e - 20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right)$	$[-40, 40]^D$	$[0, 0, \ldots, 0]$
Rotated Ackley (Multimodal)	$f_8(\mathbf{y}) = 20 - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi y_i)\right) + e - 20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} y_i^2}\right)$ $\mathbf{y}=\mathbf{M}^*\mathbf{x}$	$[-40, 40]^D$	$[0, 0, \ldots, 0]$

Table 3
Details of test functions used for comparison in this study (contd. . .).

Function (Type)	Definition	Search range	Global minimum location $\mathbf{x}^*$
Weierstrass (Multimodal)	$f_9(\mathbf{x}) = \sum_{i=1}^{D}\left[\sum_{j=0}^{K}\left[a^j\cos(2\pi b^j(x_i+0.5))\right]\right] - D\sum_{j=0}^{K}\left[a^j\cos(\pi b^j)\right]$; $a = 0.5, b = 3, K = 20$	$[-10, 10]^D$	$[0, 0, \ldots, 0]$
Non-Continuous Rastrigin (Multimodal)	$f_{10}(\mathbf{y}) = 10D + \sum_{i=1}^{D}\left[y_i^2 - 10\cos(2\pi y_i)\right]$ $y_i = \begin{cases} x_i & \lvert x_i\rvert \le 0.5 \\ \frac{\lceil 2x_i + 0.5\rceil}{2} & \lvert x_i\rvert > 0.5 \end{cases}$	$[-100, 100]^D$	$[0, 0, \ldots, 0]$
Noisy Sphere (Unimodal)	$f_{11}(\mathbf{y}) = \sum_{i=1}^{D} y_i^2$ $\mathbf{y} = \mathbf{x} + \mathbf{N}(0, 1)$	$[-100, 100]^D$	$\mathbf{N}(0, 1)$
Shifted-Rotated Rastrigin (Multimodal)	$f_{12}(\mathbf{y}) = 10D + \sum_{i=1}^{D}\left[y_i^2 - 10\cos(2\pi y_i)\right]$ $\mathbf{y} = \mathrm{M} * (\mathrm{x} - \mathrm{o})$	$[-5.12, 5.12]^D$	$[o_1, o_2, \ldots, o_D]$
Shifted-Rotated Ackley (Multimodal)	$f_{13}(\mathbf{y}) = 20 - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi y_i)\right) + e - 20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} y_i^2}\right)$ $\mathbf{y} = \mathrm{M} * (\mathrm{x} - \mathrm{o})$	$[-10, 10]^D$	$[o_1, o_2, \ldots, o_D]$
Zakharov (Unimodal)	$f_{14}(\mathbf{x}) = \sum_{i=1}^{D} x_i^2 + \left(\sum_{i=1}^{D}\frac{i}{2}x_i\right)^2 + \left(\sum_{i=1}^{D}\frac{i}{2}x_i\right)^4$	$[-10, 10]^D$	$[0, 0, \ldots, 0]^D$
Shifted-Rotated Weierstrass (Multimodal)	$f_{15}(\mathbf{x}) = \sum_{i=1}^{D}\left[\sum_{j=0}^{K}\left[a^j\cos(2\pi b^j(x_i+0.5))\right]\right] - D\sum_{j=0}^{K}\left[a^j\cos(\pi b^j)\right]$; $a = 0.5, b = 3, K = 20$ $\mathbf{y} = \mathrm{M} * (\mathrm{x} - \mathrm{o})$	$[-0.5, 0.5]^D$	$[o_1, o_2, \ldots, o_D]$

on more stringent conditions. A discussion about the nature of benchmark functions follow [28–31]:

The Sphere function is continuous, differentiable and scalable. The Sphere function is a unimodal convex function; hence its global minimum can be easily computed. Zakharov is also a unimodal function with a flat region near the minimum. The Rastrigin function is known to have large number of local minima near the global minimum. The Griewangk function is continuous, differentiable, non-separable, scalable and multimodal. The Ackley function is continuous, differentiable, non-separable, scalable, and multimodal. The Weierstrass function is multimodal and continuous everywhere. Rastrigin, Griewangk, Ackley and Weierstrass benchmark functions test the ability of the algorithm to accurately locate the global minimum in the presence of multiple local minima. The minima of the Noisy Sphere function, is offset by a Gaussian random number vector.

Rosenbrock's function has a global minimum at $[1, 1, \ldots, 1]$. The global minimum of Shifted-Rotated Rastrigin and Shifted-Rotated Ackley is randomly located in the hypercube $[-2, 2]^D$. For Shifted-Rotated Weierstrass global minimum is placed randomly in the hypercube $[-0.02, 0.02]^D$. Locating the global minimum for Rosenbrock's function and Shifted-Rotated functions is particularly difficult if the algorithm is biased to find minima closer to the origin.

Rotated version of the test function is achieved by multiplying each vector in the search space by an orthogonal matrix. An orthogonal matrix $\mathbf{M}$ multiplies the search vector $\mathbf{x}$ to produce the rotated vector $\mathbf{y}$. The orthogonal matrix is obtained from Salomon's method and is the same as the work presented in [18]. Locating the global minimum of rotated functions and shift-rotated functions is known to be more challenging than the unrotated versions.

4.1. Choice of parameters

The choice of parameters is based on the values used in existing literature [36] for the competing algorithms. The acceleration constants are taken to be 2.05, a value recommended by prior work [32]. In the recent paper by Md Nasir et al. [36], the population size was set to be 50, 100 and 100 for the 10D, 30D and 50D, respectively. The maximum number of iterations was set to 2000, 3000 and 5000 respectively. The number of function evaluations is given by the product of population size and the maximum number of iterations and is presented in Table 5.

Please cite this article in press as: V. Muthiah-Nakarajan, M.M. Noel, Galactic Swarm Optimization: A new global optimization metaheuristic inspired by galactic motion, Appl. Soft Comput. J. (2015), http://dx.doi.org/10.1016/j.asoc.2015.10.034

Table 4
Parameters used in GSO.

Dimension	M	N	L_1	L_2	EP_{max}	$c_1 = c_2 = c_3 = c_4$
$D = 10$	10	5	198	1000	5	
30	20	5	280	1500	5	2.05
50	20	5	250	1500	9	

Table 5
Function evaluations.

Dimension	$D = 10$	30	50
GSO	99,800	290,600	496,080
Other PSOs	100,000	300,000	500,000

To make the GSO experimentation identical with [36] the swarm size X is also fixed to be 50, 100 and 100 for the respective dimensions. Choosing $M > N$ assists in maintaining high exploration demanded by the multimodal functions. For example in the 30D case, the population of the subswarm is $N = 5$ and there are $M = 20$ such subswarms. To make the function evaluations for all algorithms comparable the maximum number of epochs EP_{max} is fixed as indicated in Table 4 which results in Table 5.

The values of L_1 and L_2 must be chosen carefully as it provides the trade-off between exploration and exploitation. In the GSO algorithm exploration and exploitation are given equal importance so that the algorithm is applicable to a wide class of problems. Each member in the population X_i undergoes L_1 iterations. One iteration involves one function count and hence the total number of iterations in the first level is $M \times N \times L_1$.

Likewise for the second level the M global bests of the first level are collected and the second level of PSO is run for the superswarm. Using the same reasoning the number of function evaluations turns out to be $M \times L_2$. In order to obtain exploitation at level comparable to exploration, the first level and second level should perform equal function evaluations.

$$M \times L_2 \approx M \times N \times L_1 \tag{13}$$

$$\Rightarrow L_2 \approx L_1 \times N \tag{14}$$

Eq. (14) indicates that setting $L_2 = N \times L_1$ should be the ideal choice of obtaining equal exploration and exploitation. The thumb rule of $L_2 = N \times L_1$ is confirmed to give the best experimental results for all dimensions. The Table 4 shows in detail the parameters that are deployed for the GSO algorithm.

The performance of the following PSO variants, are compared with the GSO algorithm in this work.

1 Fully informed particle swarm optimization (FIPS),
2 Unified particle swarm optimizer (UPSO),
3 Comprehensive learning particle swarm optimizer (CLPSO) and
4 Dynamic neighbourhood learning particle swarm optimizer (DNLPSO).

All the listed algorithms given above attempt to preserve swarm divergence to avoid premature convergence, which is a serious and known issue with the canonical PSO algorithm. The general approach is to keep all members of the population aware of the performance of few of other members thereby keeping the global search ability active at all times.

In FIPS (2004), each member is influenced by neighbours which could be from a U-ring topological structure [33]. This algorithm is different from the canonical PSO in which the pbest information of a particle in consideration is shared among predetermined neighbours. The U-ring topology offered the best neighbourhood for information sharing in terms of the results obtained.

UPSO (2006) strategy is divided into constrained and unconstrained depending on the position and velocity parameter [34,35] being bounded or unbounded. This scheme also uses neighbourhoods to enhance sharing of information.

In CLPSO (2006), the inclusion of global best as an exemplar is abandoned and particles move in the search space influenced by the best pbest positions located by other particles or its own depending upon a random number [18]. This is a major step to increase exploration. However this strategy takes large number of iterations to converge for multimodal functions as there is no gbest to guide the search.

In DNLPSO (2012) the velocity depends on the global best along with the particle best of a neighbourhood whose position is determined by a ring topology varying dynamically [36]. DNLPSO is a slight modification of CLPSO in which the pbest information is shared within a small neighbourhood and hence can balance global and local search abilities.

All the algorithms discussed above share information either over the entire population or a small neighbourhood making the search vulnerable by getting stuck in a local minimum. The same argument holds for multiswarm based algorithms sharing best solutions. In GSO algorithm, exploration and exploitation are independent from each other making it suitable to find minima for a given multimodal and high dimensional search space. The galactic best $\mathbf{g}$ is influenced by both exploration and exploitation phases but $\mathbf{g}$ does not influence the exploration by any feedback mechanism. Information sharing is fully avoided in GSO algorithm, making the search uninfluenced towards some local minimum.

Each algorithm is tested with the benchmark functions for 50 independent trials and the results are given for the 10 dimensions in Table 6. In this paper each algorithm is tested for 50 independent trials to achieve higher confidence in the results.

To capture the statistics better, in each entry the mean (μ), standard deviation (σ), median (m_d) and the comparison results of Mann–Whitney U test (h) with respect to GSO are provided. The value of h is binary, if $h = 0$ then the null hypothesis that "the medians are equal" cannot be rejected at 5% significance level [37–39]. For, $h = 1$ the null hypothesis can be rejected at 5% level. Mean and standard deviation are sensitive to outliers; hence median is also provided to enable a better comparison. The best solutions are highlighted in bold.

From Table 6 it can be seen that for the multimodal functions like Rastrigin, Rotated Rastrigin, Griewangk, Rotated Griewangk, Ackley, Rotated Ackley, Weierstrass, Noisy-Sphere and Zakharov, the GSO algorithm returns zero within machine precision. Since PSO variants are already shown to perform better than the normal PSO in an earlier work [18] comparison with the classical PSO algorithm is not repeated in this work.

It is evident from the Table 6 that the GSO algorithm outperforms other algorithms for most of the functions for 10-dimensional problems. DNLPSO, CLPSO and UPSO algorithms have performance similar to GSO algorithm for Rosenbrock function. DNLPSO does better for the Weierstrass test function. Griewangk is known to be difficult [18] for lower dimensions, however GSO algorithm locates global minimum without any impediment. This is in contrast to UPSO which performs well for high dimensional Griewangk function but has difficulties with lower dimensions. For the Shifted-Rotated Rastrigin GSO algorithm shows better results. For the Shifted-Rotated Ackley many competing algorithms show statistically equivalent performance. For Shifted-Rotated Weierstrass

Please cite this article in press as: V. Muthiah-Nakarajan, M.M. Noel, Galactic Swarm Optimization: A new global optimization metaheuristic inspired by galactic motion, Appl. Soft Comput. J. (2015), http://dx.doi.org/10.1016/j.asoc.2015.10.034

Table 6
Results for 10-dimensional problem.

Algorithm	Benchmark Functions					
	f_1(Sphere)		f_2(Rosenbrock)		f_3(Rastrigin)	
GSO	μ =**0**	σ =**0**	**3.346343E+00**	**3.346343E+00**	**0**	**0**
	m_d =**0**		**4.549212E−03**		**0**	
DNLPSO	μ =7.454117E−02	σ =5.258471E−01	**4.579636E+00**	**9.187347E+00**	6.633415E+00	7.472971E+00
	m_d =4.218421E−115	h =1	**2.714104E+00**	**0**	3.473367E+00	1
CLPSO	μ =9.180086E−26	σ =9.700885E−26	**5.413461E−01**	**4.864544E−01**	2.486900E−16	8.798009E−16
	m_d =6.459944E−26	h =1	**3.719262E−01**	**0**	0	1
UPSO	μ =2.180218E−89	σ =3.849323E−89	**4.341650E−01**	**5.958843E−01**	7.926291E+00	2.919806E+00
	m_d =3.507327E−90	h =1	**2.578580E−01**	**0**	7.959667E+00	1
FIPS	μ =8.015314E−20	σ =7.222630E−20	4.047522E+00	1.145596E+00	1.909959E+00	1.208438E+00
	m_d =6.246882E−20	h =1	4.404810E+00	1	1.645338E+00	1

Algorithm	Benchmark Functions					
	f_4(Rotated Rastrigin)		f_5(Griewangk)		f_6(Rotated Griewangk)	
GSO	**0**	**0**	**0**	**0**	**0**	**0**
	0		**0**		**0**	
DNLPSO	8.868549E+00	6.565880E+00	8.823940E−02	6.289002E−02	1.063180E−01	6.145364E−02
	6.964711E+00	1	7.239125E−02	1	8.977682E−02	1
CLPSO	4.550047E+00	2.230538E+00	1.555388E−04	1.045524E−03	2.151283E−02	1.274882E−02
	3.986174E+00	1	9.065542E−07	1	2.081254E−02	1
UPSO	1.183094E+01	3.662598E+00	3.046884E−02	1.827492E−02	2.464885E−02	1.457674E−02
	1.193949E+01	1	2.710324E−02	1	2.460154E−02	1
FIPS	8.112330E+00	3.173953E+00	3.547640E−02	2.039213E−02	7.146974E−02	4.869865E−02
	8.180680E+00	1	3.474983E−02	1	6.340074E−02	1

Algorithm	Benchmark Functions					
	f_7(Ackley)		f_8(Rotated Ackley)		f_9(Weierstrass)	
GSO	**0**	**0**	**0**	**0**	**0**	**0**
	0		**0**		**0**	
DNLPSO	9.273778E+00	9.351956E+00	8.638006E+00	9.916434E+00	**0**	**0**
	4.568173E+00	1	9.537844E−02	1	**0**	**0**
CLPSO	1.593747E−13	1.041292E−13	6.864130E−09	3.107415E−08	4.272828E−01	2.389322E−01
	1.350031E−13	1	4.774137E−11	1	3.952151E−01	1
UPSO	3.268497E−15	9.736124E−16	3.197442E−15	1.076635E−15	6.912884E−01	4.657697E−01
	3.552714E−15	1	3.552714E−15	1	5.853086E−01	1
FIPS	1.548194E−10	5.874201E−11	1.769742E−10	6.736748E−11	7.769260E+00	7.315156E−01
	1.524203E−10	1	1.773621E−10	1	7.797454E+00	1

Algorithm	Benchmark Functions					
	f_{10}(Non-Continuous Rastrigin)		f_{11}(Noisy Sphere)		f_{12}(Shifted-Rotated Rastrigin)	
GSO	**0**	**0**	**0**	**0**	**8.463998E−03**	**4.838831E−02**
	0		**0**		**3.437369E−04**	
DNLPSO	7.827265E+00	8.178500E+00	2.918126E−21	1.604597E−20	6.053397E+00	6.442646E+00
	5.004860E+00	1	3.908552E−145	1	3.979836E+00	1
CLPSO	1.492140E−15	2.670209E−15	1.455050E−13	1.270846E−13	3.949970E+00	1.874530E+00
	0	1	9.786138E−14	1	4.051978E+00	1
UPSO	5.099152E+00	1.159187E+00	2.910285E−82	1.141397E−81	8.861903E+00	3.048016E+00
	5.000000E+00	1	2.491372E−83	1	7.959672E+00	1
FIPS	5.184917E+00	1.006310E+00	2.503216E−19	2.892327E−19	6.315532E+00	3.570602E+00
	5.291141E+00	1	1.840689E−19	1	5.395516E+00	1

Algorithm	Benchmark Functions					
	f_{13}(Shifted-Rotated Ackley)		f_{14}(Zakharov)		f_{15}(Shifted-Rotated Weierstrass)	
GSO	**5.692032E+00**	**3.359089E+00**	**4.480297E−07**	**2.649336E−06**	2.149116E−01	5.704419E−01
	4.081029E+00		**0**		1.059948E−01	
DNLPSO	**1.137964E+01**	**1.002040E+01**	4.986472E−04	3.228070E−03	5.137679E−01	1.010746E+00
	1.999960E+01	**0**	6.489030E−33	1	1.784634E−02	1
CLPSO	1.839891E+01	1.473934E+00	8.954142E−05	5.639834E−05	8.984718E−02	7.160551E−02
	1.873671E+01	1	7.830674E−05	1	6.881252E−02	1
UPSO	**1.211909E+01**	**9.732503E+00**	5.475182E−40	1.156220E−39	9.896272E−01	9.360765E−01
	1.991524E+01	**0**	4.955043E−41	1	7.166146E−01	1
FIPS	**1.215265E+01**	**9.923105E+00**	2.291714E−11	2.051249E−11	**4.020933E−03**	**1.130769E−02**
	2.025075E+01	**0**	1.483331E−11	1	**1.069669E−04**	**1**

Please cite this article in press as: V. Muthiah-Nakarajan, M.M. Noel, Galactic Swarm Optimization: A new global optimization metaheuristic inspired by galactic motion, Appl. Soft Comput. J. (2015), http://dx.doi.org/10.1016/j.asoc.2015.10.034

Table 7
Results for 30-dimensional problem.

Algorithm	Benchmark Functions					
	f_1(Sphere)		f_2(Rosenbrock)		f_3(Rastrigin)	
GSO	μ =**0**	σ =**0**	**1.014230E+01**	**1.344685E+01**	**0**	**0**
	m_d =**0**		**1.697883E−02**		**0**	
DNLPSO	μ =2.067459E−01	σ =1.070726E+00	4.638780E+03	2.453529E+04	1.029995E+02	2.016368E+02
	m_d =1.773520E−43	h =1	1.581263E+01	1	5.173780E+01	1
CLPSO	μ =7.458703E−11	σ =2.531589E−11	1.523131E+01	7.457680E+00	1.306592E−03	5.717197E−04
	m_d =6.776015E−11	h =1	1.364200E+01	1	1.206642E−03	1
UPSO	μ =4.461058E−51	σ =5.327222E−51	1.313221E+01	7.076222E+00	8.013614E+01	1.514800E+01
	m_d =2.630685E−51	h =1	1.562824E+01	1	8.108888E+01	1
FIPS	μ =4.257349E−06	σ =1.416584E−06	2.589907E+01	9.059257E−01	1.097708E+02	1.334847E+01
	m_d =4.094418E−06	h =1	2.607102E+01	1	1.121884E+02	1

Algorithm	Benchmark Functions					
	f_4(Rotated Rastrigin)		f_5(Griewangk)		f_6(Rotated Griewangk)	
GSO	**0**	**0**	**0**	**0**	**0**	**0**
	0		**0**		**0**	
DNLPSO	1.629827E+02	6.258233E+02	1.004828E−01	3.024862E−01	1.055049E+00	7.048781E+00
	5.571764E+01	1	9.857285E−03	1	9.864672E−03	1
CLPSO	4.568478E+01	8.323703E+00	6.758184E−08	4.586442E−08	1.223805E−04	1.326182E−04
	4.576024E+01	1	5.222917E−08	1	8.022365E−05	1
UPSO	8.465418E+01	1.540084E+01	**1.030780E−06**	**7.288718E−06**	1.479208E−04	1.045958E−03
	8.419507E+01	1	**0**	**0**	0	1
FIPS	1.584953E+02	1.554764E+01	8.163290E−05	6.490223E−04	3.264383E−04	7.776936E−04
	1.613222E+02	1	1.036757E−04	1	1.355055E−04	1

Algorithm	Benchmark Functions					
	f_7(Ackley)		f_8(Rotated Ackley)		f_9(Weierstrass)	
GSO	**0**	**0**	**0**	**0**	**0**	**0**
	0		**0**		**0**	
DNLPSO	1.999329E+01	7.177566E−15	1.422131E+01	9.299878E+00	**0**	**0**
	1.999329E+01	1	2.000305E+01	1	**0**	**0**
CLPSO	7.264776E−06	1.334031E−06	5.215065E−03	3.145045E−03	1.784184E+00	3.385508E−01
	7.301766E−06	1	4.293644E−03	1	1.722107E+00	1
UPSO	3.552714E−15	0	2.680843E−02	1.895642E−01	1.064807E+01	2.401397E+00
	3.552714E−15	1	3.552714E−15	1	1.103862E+01	1
FIPS	5.902425E−04	9.105905E−05	7.287851E−04	1.343584E−04	3.754392E+01	1.011229E+00
	5.922426E−04	1	7.475233E−04	1	3.759877E+01	1

Algorithm	Benchmark Functions					
	f_{10}(Non-Continuous Rastrigin)		f_{11}(Noisy Sphere)		f_{12}(Shifted-Rotated Rastrigin)	
GSO	**0**	**0**	**5.748128E−05**	**4.064540E−04**	**6.375119E−01**	**4.245655E+00**
	0		**0**		**1.731989E−03**	
DNLPSO	1.388704E+02	2.252864E+02	6.901643E+01	4.862303E+02	4.132378E+01	2.751363E+01
	5.272042E+01	1	2.827110E−54	1	3.581848E+01	1
CLPSO	3.529142E−03	1.555435E−03	1.631486E+00	6.780138E−01	4.338192E+01	6.738683E+00
	3.391873E−03	1	1.490353E+00	1	4.423278E+01	1
UPSO	8.028696E+01	1.636549E+01	4.294932E−30	6.700421E−30	5.733123E+01	1.061382E+01
	8.170863E+01	1	1.786204E−30	1	5.870250E+01	1
FIPS	9.979245E+01	1.446637E+01	4.779729E−05	1.741262E−05	1.359108E+02	1.359680E+01
	1.020049E+02	1	4.696219E−05	1	1.390697E+02	1

Algorithm	Benchmark Functions					
	f_{13}(Shifted-Rotated Ackley)		f_{14}(Zakharov)		f_{15}(Shifted-Rotated Weierstrass)	
GSO	**1.149663E+01**	**3.052314E+00**	**6.375815E−04**	**4.270195E−03**	5.920445E−01	6.334273E−01
	1.132023E+01		**0**		4.968520E−01	
DNLPSO	1.903383E+01	5.682626E+00	5.262054E+00	2.267857e+01	5.735469E+00	6.021071E+00
	2.092214E+01	1	5.155775E−08	1	2.664586E+00	1
CLPSO	1.930537E+01	8.361213E−01	2.621608E+01	5.222562E+00	2.871897E+00	1.289431E+00
	1.961541E+01	1	2.578291E+01	1	2.536836E+00	1
UPSO	2.000011E+01	1.691343E−02	9.658180E−04	5.409316E−04	1.539890E+01	3.251883E+00
	1.999881E+01	1	2.309064E−03	1	1.506032E+01	1
FIPS	2.057435E+01	2.360332E+00	1.885101E+01	4.999178E+00	**4.722291E−01**	**3.510055E−01**
	2.091578E+01	1	1.868058E+01	1	**4.016529E−01**	**1**

Please cite this article in press as: V. Muthiah-Nakarajan, M.M. Noel, Galactic Swarm Optimization: A new global optimization metaheuristic inspired by galactic motion, Appl. Soft Comput. J. (2015), http://dx.doi.org/10.1016/j.asoc.2015.10.034

Table 8
Results For 50-Dimensional Problem.

Algorithm	Benchmark Functions					
	f_1(Sphere)		f_2(Rosenbrock)		f_3(Rastrigin)	
GSO	μ =**0**	σ =**0**	**1.754761E+01**	**2.345558E+01**	**0**	**0**
	m_d =**0**		**1.361893E−03**		**0**	
DNLPSO	μ =2.507253E+01	σ =1.490451E+02	1.511956E+03	9.407639E+03	2.909075E+02	4.918555E+02
	m_d =2.128158E−23	h =1	7.801566E+01	1	1.318858E+02	1
CLPSO	μ =5.969617E−11	σ =1.370316E−11	5.152155E+01	2.030110E+01	3.835930E−03	1.106694E−03
	m_d =5.725084E−11	h =1	4.811863E+01	1	3.695767E−03	1
UPSO	μ =2.315557E−55	σ =1.605565E−55	4.249652E+01	2.288394E+01	1.960009E+02	3.237489E+01
	m_d =2.012665E−55	h =1	3.844941E+01	1	1.955102E+02	1
FIPS	μ =4.309850E−04	σ =1.131916E−04	4.901800E+01	3.723694E+00	2.816954E+02	1.849331E+01
	m_d =4.193802E−04	h =1	4.809043E+01	1	2.842823E+02	1

Algorithm	Benchmark Functions					
	f_4(Rotated Rastrigin)		f_5(Griewangk)		f_6(Rotated Griewangk)	
GSO	**0**	**0**	**0**	**0**	**0**	**0**
	0		**0**		**0**	
DNLPSO	7.555488E+02	2.076073E+03	1.044678E+01	6.244802E+01	4.943715E+00	2.646668E+01
	1.545108E+02	1	9.860978E−03	1	1.722146E−02	1
CLPSO	1.016781E+02	1.340658E+01	4.726969E−09	2.853674E−09	5.138687E−05	5.491315E−05
	1.012697E+02	1	4.014630E−09	1	2.868487E−05	1
UPSO	2.184566E+02	2.977827E+01	**0**	**0**	**0**	**0**
	2.159625E+02	1	**0**	**0**	**0**	**0**
FIPS	3.553786E+02	1.766012E+01	1.140669E−03	9.179803E−04	3.939815E−03	1.720350E−03
	3.584236E+02	1	8.799764E−04	1	3.416944E−03	1

Algorithm	Benchmark Functions					
	f_7(Ackley)		f_8(Rotated Ackley)		f_9(Weierstrass)	
GSO	**8.953224E−03**	**6.287450E−02**	**0**	**0**	**0**	**0**
	0		**0**		**0**	
DNLPSO	1.999329E+01	7.177566E−15	1.665212E+01	8.228409E+00	**0**	**0**
	1.999329E+01	1	2.107242E+01	1	**0**	**0**
CLPSO	4.943769E−06	8.188170E−07	8.031692E−03	3.996753E−03	1.607327e+00	2.780002E−01
	4.757539E−06	1	7.111928E−03	1	1.586666E+00	1
UPSO	6.465939E−15	1.378765E−15	4.596364E−01	6.851495E−01	2.416791E+01	3.762256E+00
	7.105427E−15	1	7.105427E−15	1	2.428753E+01	1
FIPS	4.615299E−03	5.937149E−04	6.291656E−03	8.435986E−04	7.066436E+01	1.396962E+00
	4.597196E−03	1	6.236044E−03	1	7.086165E+01	1

Algorithm	Benchmark Functions					
	f_{10}(Non-Continuous Rastrigin)		f_{11}(Noisy Sphere)		f_{12}(Shifted-Rotated Rastrigin)	
GSO	**0**	**0**	**1.961313E−07**	**1.386858E−06**	**8.848112E−01**	**4.036588E+00**
	0		**0**		**4.353496E−04**	
DNLPSO	2.346039E+02	2.863932E+02	6.252992E+02	2.099279E+03	1.029401E+02	5.732838E+01
	1.300310E+02	1	2.791151E−22	1	9.253103E+01	1
CLPSO	1.043312E−02	4.030864E−03	4.830669E+02	1.336575E+02	9.305714E+01	1.363626E+01
	9.697179E−03	1	4.899347E+02	1	9.263139E+01	1
UPSO	2.366899E+02	4.277788E+01	2.043920E−14	9.402070E−15	1.254077E+02	1.722605E+01
	2.346250E+02	1	8.470176E−15	1	1.243723E+02	1
FIPS	2.714630E+02	2.164392E+01	3.233628E−02	9.586050E−03	3.095742E+02	1.199357E+01
	2.775908E+02	1	3.182972E−02	1	3.101737E+02	1

Algorithm	Benchmark Functions					
	f_{13}(Shifted-Rotated Ackley)		f_{14}(Zakharov)		f_{15}(Shifted-Rotated Weierstrass)	
GSO	**1.696910E+01**	**2.656514E+00**	**6.268439E−03**	**4.432139E−02**	**1.851850E+00**	**2.645764E+00**
	1.749924E+01		**0**		**9.361264E−01**	
DNLPSO	2.043704E+01	2.982264E+00	9.173593E+01	3.329294E+02	1.468015E+01	1.177599E+01
	2.109250E+01	1	2.192383E−01	1	1.067914E+01	1
CLPSO	1.947488E+01	5.500538E−01	1.428127E+02	1.455452E+01	4.724226E+00	1.351783E+00
	1.954028E+01	1	1.413192E+02	1	4.872217E+00	1
UPSO	2.001171E+01	2.610259E−02	2.760691E+00	1.054970E+00	3.751234E+01	3.849334E+00
	2.000263E+01	1	2.696308E+00	1	3.779544E+01	1
FIPS	2.110762E+01	3.085342E−02	2.213914E+02	4.039525E+01	2.772891E+00	1.340077E+00
	2.110594E+01	1	2.216122E+02	1	2.403706E+00	1

Please cite this article in press as: V. Muthiah-Nakarajan, M.M. Noel, Galactic Swarm Optimization: A new global optimization metaheuristic inspired by galactic motion, Appl. Soft Comput. J. (2015), http://dx.doi.org/10.1016/j.asoc.2015.10.034

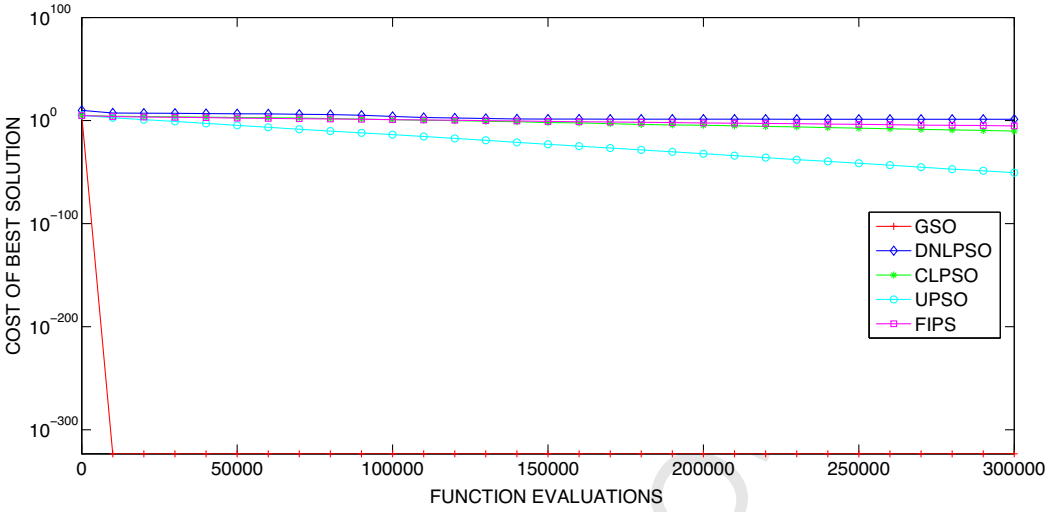

Fig. 1. Convergence characteristics for Sphere function.

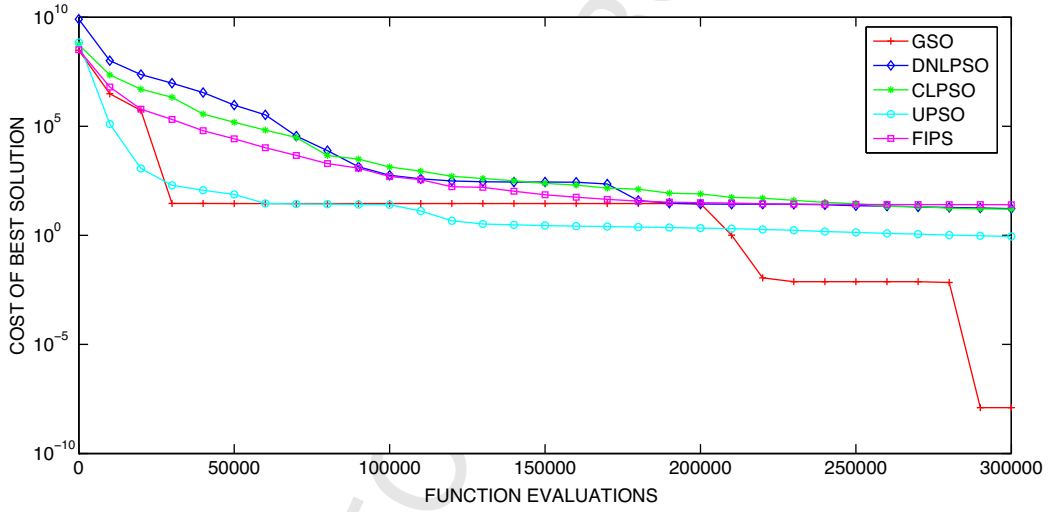

Fig. 2. Convergence characteristics for Rosenbrock function.

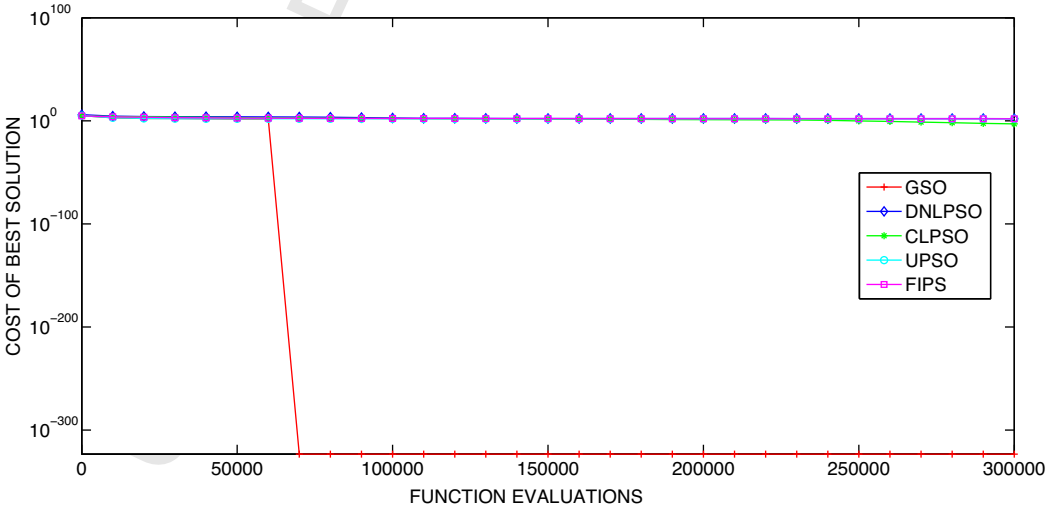

Fig. 3. Convergence characteristics for Rastrigin function.

Please cite this article in press as: V. Muthiah-Nakarajan, M.M. Noel, Galactic Swarm Optimization: A new global optimization metaheuristic inspired by galactic motion, Appl. Soft Comput. J. (2015), http://dx.doi.org/10.1016/j.asoc.2015.10.034

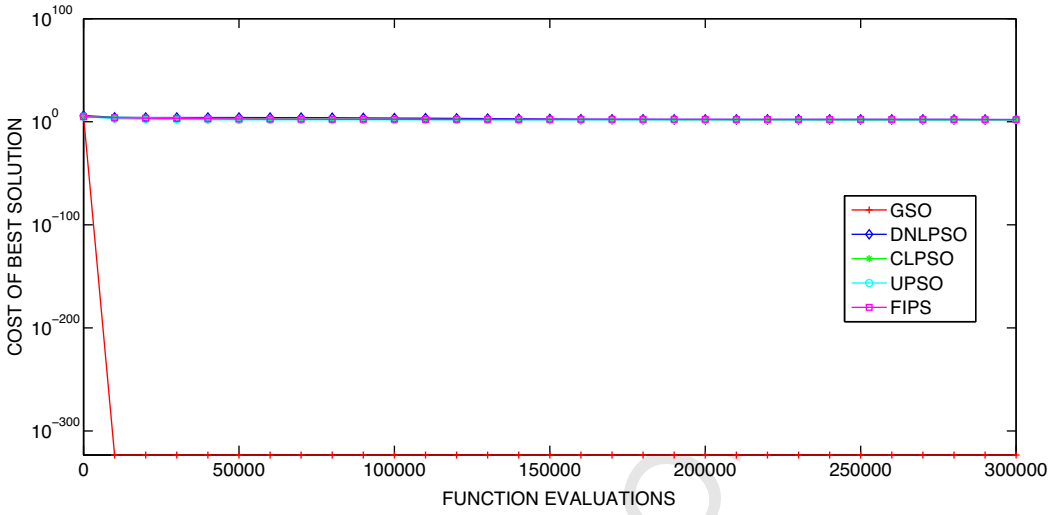

Fig. 4. Convergence characteristics for Rotated-Rastrigin function.

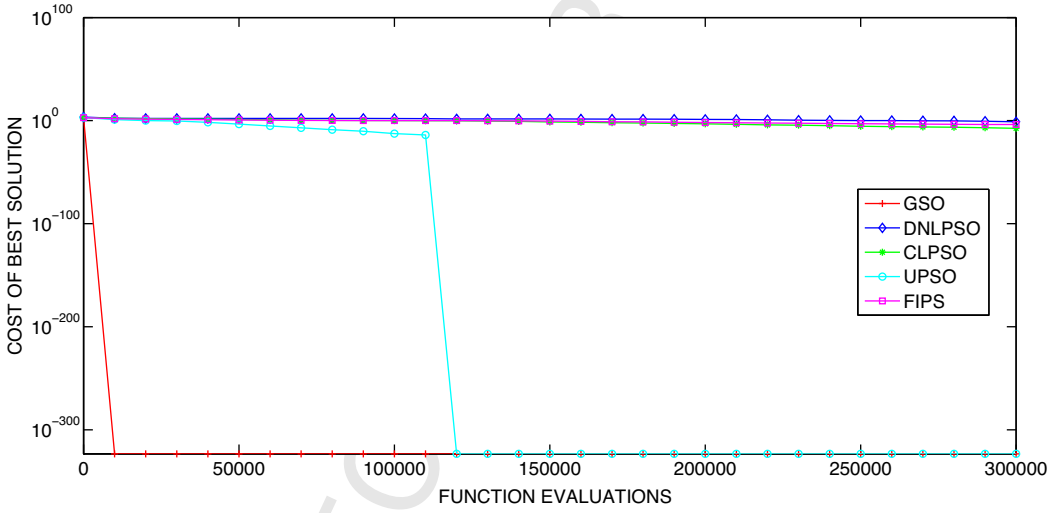

Fig. 5. Convergence characteristics for Griewangk function.

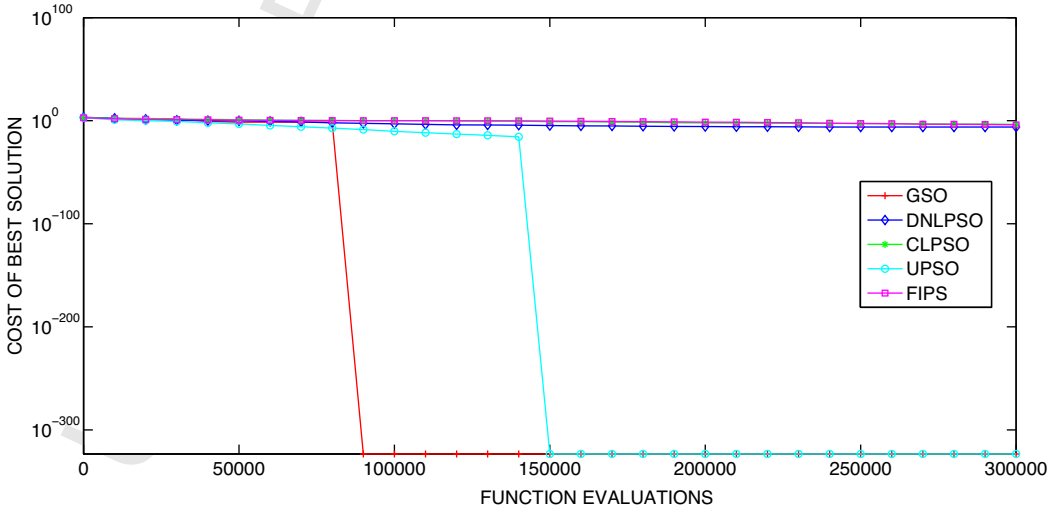

Fig. 6. Convergence characteristics for Rotated-Griewangk function.

Please cite this article in press as: V. Muthiah-Nakarajan, M.M. Noel, Galactic Swarm Optimization: A new global optimization metaheuristic inspired by galactic motion, Appl. Soft Comput. J. (2015), http://dx.doi.org/10.1016/j.asoc.2015.10.034

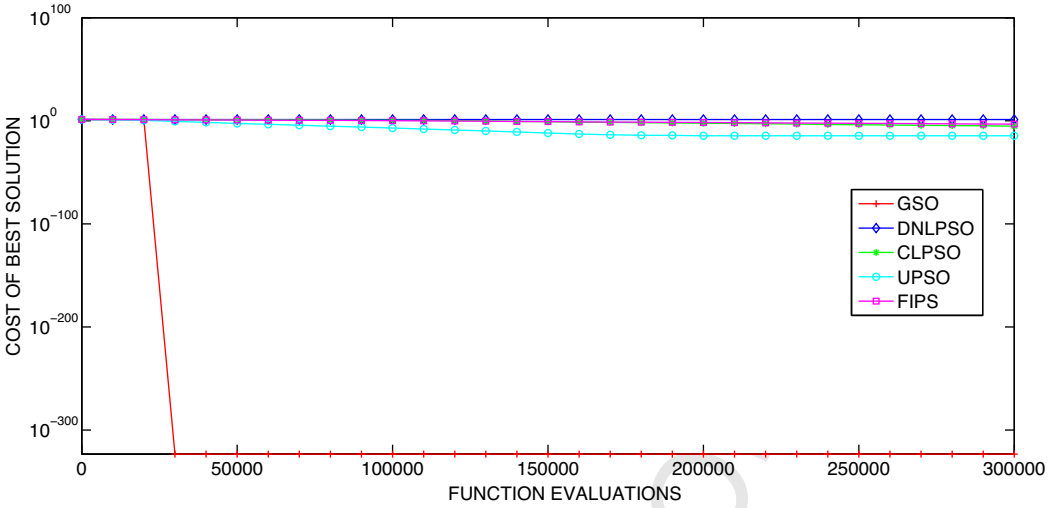

Fig. 7. Convergence characteristics for Ackley function.

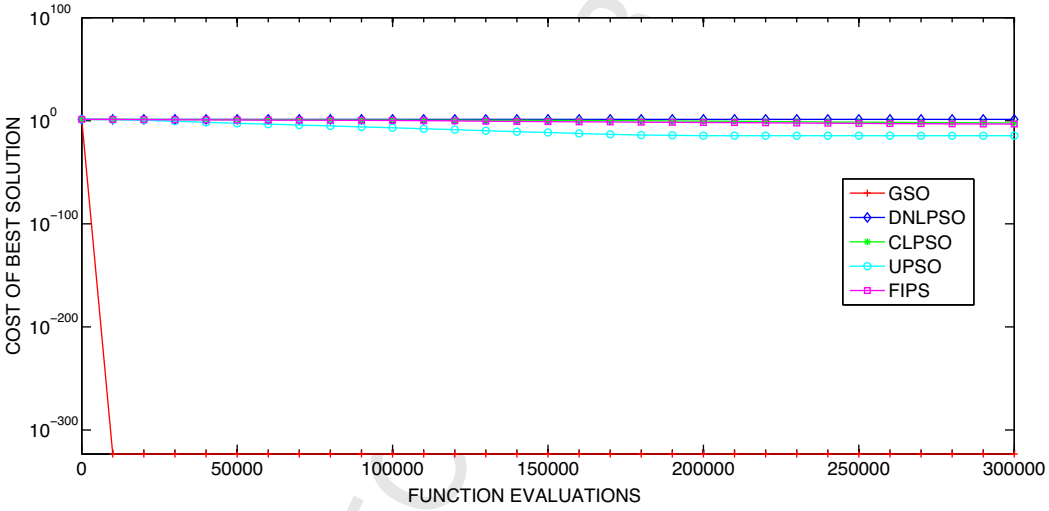

Fig. 8. Convergence characteristics for Rotated-Ackley function.

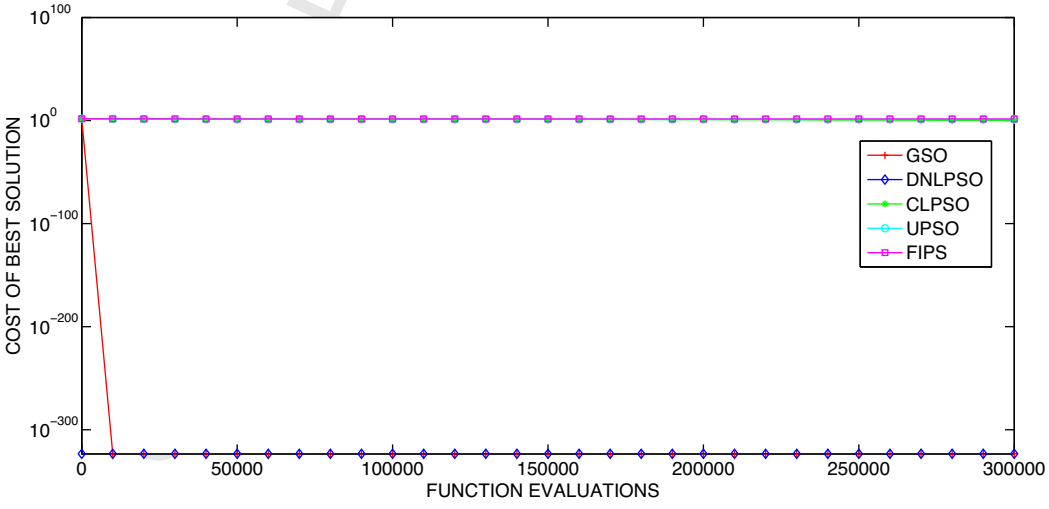

Fig. 9. Convergence characteristics for Weierstrass function.

Please cite this article in press as: V. Muthiah-Nakarajan, M.M. Noel, Galactic Swarm Optimization: A new global optimization metaheuristic inspired by galactic motion, Appl. Soft Comput. J. (2015), http://dx.doi.org/10.1016/j.asoc.2015.10.034

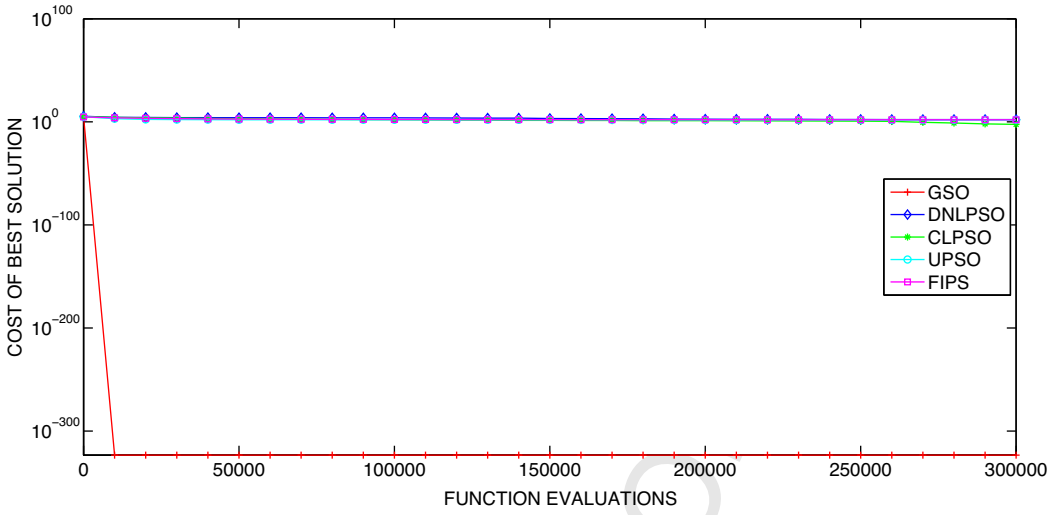

Fig. 10. Convergence characteristics for Non-Continuous Rastrigin function.

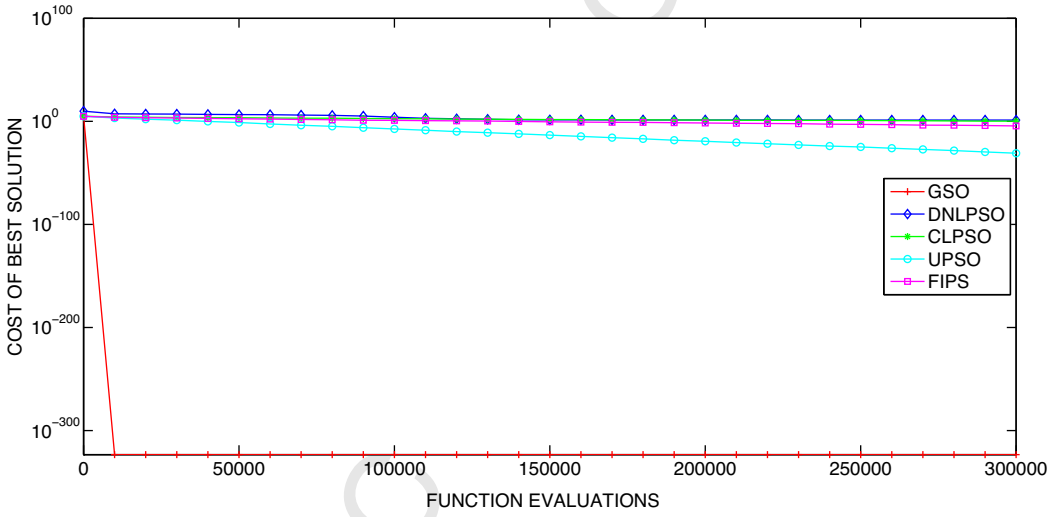

Fig. 11. Convergence characteristics for Noisy Sphere function.

function the performance of GSO algorithm is poor compared to all other algorithms for 10-dimensions.

Table 7 presents the results for 30 dimensions. The situation is similar to the 10*D* case for Sphere, Rastrigin, Rotated-Rastrigin, Griewangk, Rotated-Griewangk, Ackley, Rotated Ackley, Weierstrass, Non-Continuous Rastrigin, Zakharov. As dimensions increase the performance of all algorithms deteriorate. However, the performance worsens faster for other algorithms when compared with GSO algorithm. This graceful degradation in performance of the GSO algorithm with increasing dimension is evidenced by the significant difference in results observed for Rosenbrock, Noisy-Sphere, Shift-Rotated Rastrigin and Shift-Rotated Ackley. Though GSO's performance has improved relatively, FIPS algorithm still leads in 30-dimensions for Shifted-Rotated Weierstrass.

Table 8 illustrates the results for the 50-dimensions. The increased dimension seems to have hardly affected the results for the functions Sphere, Rastrigin, Rotated Rastrigin, Griewangk, Rotated Griewangk, Non-continuous Rastrigin, Noisy Sphere, Ackley, Rotated Ackley, Weierstrass, Zakharov as GSO algorithm continuous to be robust. Even in this high dimensional space the GSO algorithm is found to be robust and less susceptible to the "curse of dimensionality". In case of the Shifted-Rotated Weierstrass function the GSO algorithm is the worst performing algorithm for 10 dimensions. However it is worth noticing that the GSO algorithm performance improves significantly with increasing dimension and the GSO outperforms other algorithms for 50 dimensions. For the functions Rosenbrock, Shifted-Rotated Rastrigin and Shift-Rotated Ackley, GSO algorithm clearly leads all the other algorithms when the dimension is increased to 50.

4.2. Discussion on experimental results

Convergence of the GSO algorithm and other algorithms cannot be shown directly as zero is returned during some runs and hence cannot be plotted on a logarithmic scale. In this paper ϵ_0 (machine epsilon at '0') which is equal to 4.9407×10^{-324} is uniformly added to all plots to shift all the convergence characteristics upward by ϵ_0. This is only done because a value of zero cannot be plotted on a log scale. ϵ_0 is not added anywhere during the simulations but only in the plots to display the final results (since log(0) is undefined).

This has the effect of shifting the entire set of characteristics upward by a fixed amount of ϵ_0 and hence does not affect the relative performance of different algorithms. Also ϵ_0 being

Please cite this article in press as: V. Muthiah-Nakarajan, M.M. Noel, Galactic Swarm Optimization: A new global optimization metaheuristic inspired by galactic motion, Appl. Soft Comput. J. (2015), http://dx.doi.org/10.1016/j.asoc.2015.10.034

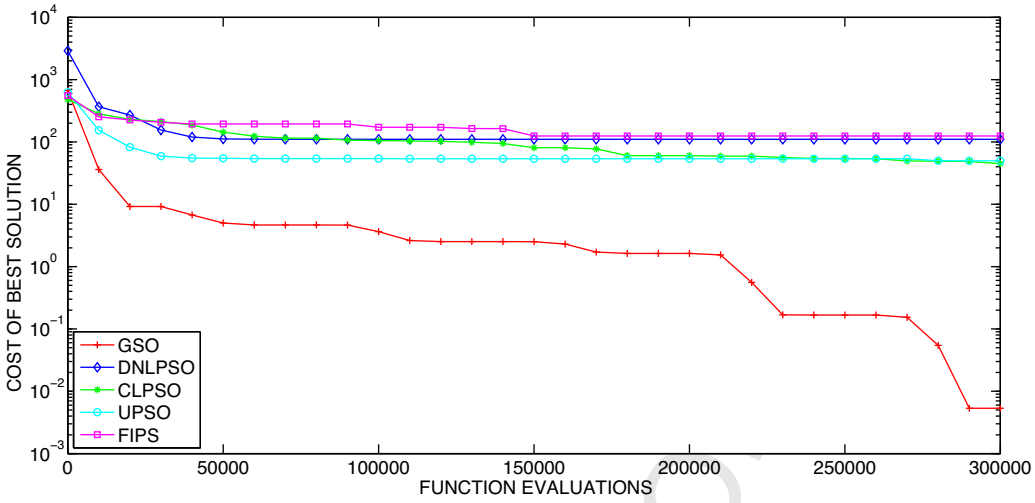

Fig. 12. Convergence characteristics for Shift-Rotated Rastrigin function.

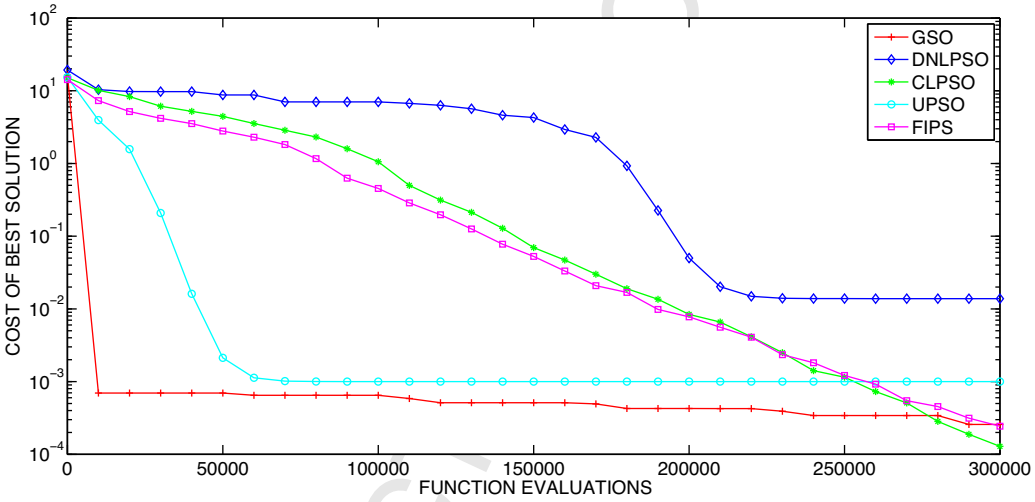

Fig. 13. Convergence characteristics for Shift-Rotated Ackley function.

4.9407×10^{-324} changes the all the plots only infinitesimally (by the same amount upward). When values are reported in the table the exact number returned by the algorithm is stated without any change (no epsilons added anywhere).

The plots are provided for visualization and conclusions can be based purely on statistical analysis of exact numerical values given in the tables. The convergence plots for single run are shown for the 30*D* case as it is representative of the convergence for 10*D* and 50*D* cases.

From the convergence graphs for Sphere, Rastrigin, Rotated Rastrigin, Griewangk, Ackley, Rotated Ackley, Weierstrass, Non-Continuous Rastrigin, Noisy Sphere and Zakharov functions shown in Fig. 1, Fig. 3, Fig. 4, Fig. 5, Fig. 7, Fig. 8 and Fig. 9, Fig. 10, Fig. 11 and Fig. 14 respectively it can be seen that the rate of convergence of GSO is rapid for these functions. It is observed that a best cost of zero is obtained by the GSO algorithm within 10,000 function evaluations. Fig. 9 shows that DNLPSO converges quite rapidly for Weierstrass function compared to other algorithms.

UPSO algorithm is competitive for Rosenbrock (Fig. 2), Griewangk (Fig. 5) and Rotated Griewangk (Fig. 6) but the GSO algorithm is significantly better. The GSO algorithm converges faster and shows continuous convergence for Shifted-Rotated Rastrigin as can be seen in Fig. 12. DNLPSO converges faster than the GSO algorithm for the Weierstrass test function. For Shifted-Rotated Ackley the GSO algorithm converges faster in the earlier 10,000 function evaluations and cost of the best solution shows slow improvement compared to CLPSO and FIPS over the observed range. Fig. 15 shows that for Shifted-Rotated Weierstrass function, FIPS and CLPSO algorithms show better convergence properties than the GSO algorithm but CLPSO has not overtaken the GSO algorithm in first 300000 function evaluations. FIPS converges to a lower function value than the GSO algorithm for Shifted-Rotated Weierstrass function as expected (Fig. 13).

Table 9 compares the GSO algorithm and 4 state-of-the-art multiswarm PSO algorithms from recent literature. Results in Table 9 present the best solution for the 30*D* case averaged over 30 independent trials. To facilitate comparison with already published work on multiswarm algorithms; the same test functions, dimension, number of trials and stopping criteria used by the authors of the multiswarm algorithms has been adopted in this paper. The maximum number of function evaluations is restricted to 200,000 for GSO, DMSPSO-HS and DMSPSO [25]. For the COM-MPSO and COL-MPSO the function evaluations is set to 320,000 [27]. Table 9 indicates that the GSO algorithm significantly outperforms all the multiswarm algorithms taken for comparison. The superior performance of the GSO approach demonstrates that sharing of good solutions between the subswarms and the superswarm must be unidirectional for good performance contrary to popular

Please cite this article in press as: V. Muthiah-Nakarajan, M.M. Noel, Galactic Swarm Optimization: A new global optimization metaheuristic inspired by galactic motion, Appl. Soft Comput. J. (2015), http://dx.doi.org/10.1016/j.asoc.2015.10.034

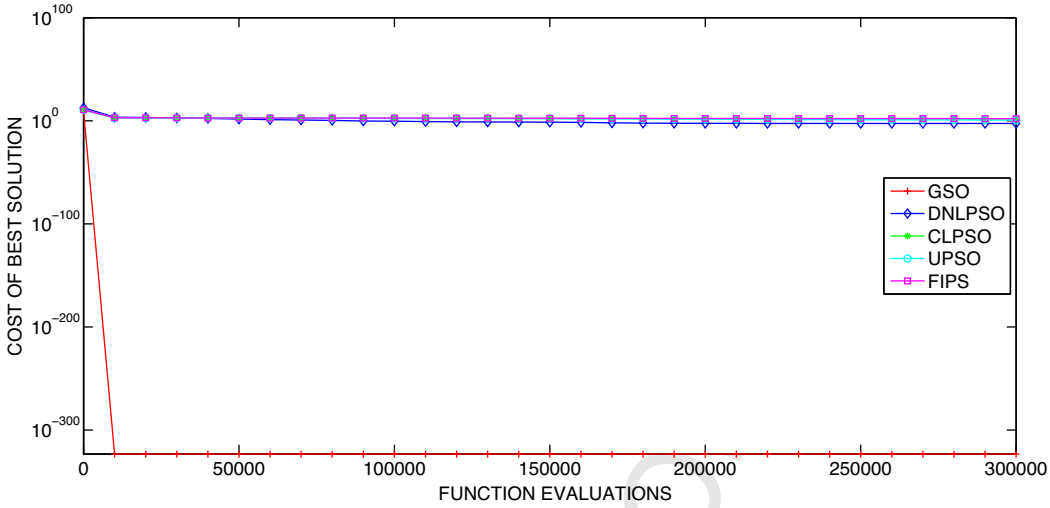

Fig. 14. Convergence characteristics for Zakharov function.

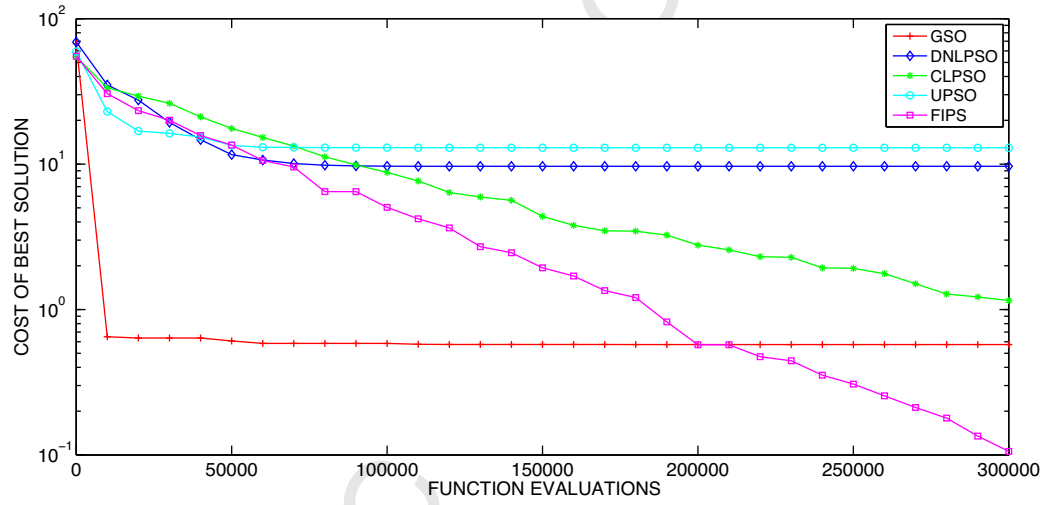

Fig. 15. Convergence characteristics for Shift-Rotated Weierstrass function.

Table 9
Comparison of GSO with recent multiswarm algorithms for 30D.

f	GSO	DMSPSO-HS [25]	DMSPSO [25]	COM-MPSO [27]	COL-MPSO [27]
f_1	0	3.6514E−78	1.2658E−66	3.8877E−32	1.9008E−31
f_2	1.7543E+01	1.7023E+01	1.9828E+01	2.8349E+00	2.5875E+00
f_3	0	0	1.4014E+01	6.9850E−01	1.5561E+00
f_4	0	1.9269E+01	2.8101E+01	–	–
f_5	0	0	0	4.9100E−02	5.6500E−02
f_6	0	4.3147E−04	4.8307E−04	–	–
f_7	0	3.2145E−15	6.5178E−14	–	–

belief. Comparison with other multiswarm algorithms indicates that inserting good solutions from the masterswarm into the slaveswarm or exchanging information between swarms results in poor quality of the final solution.

A major difference between the GSO and MCPSO [27] algorithms is that in the GSO the superswarm comes into existence only temporarily during Level 2 (exploitative phase). Whereas in MCPSO strategies, the master swarm has a continuous and independent existence. The masterswarm retains a memory of good solutions computed in the previous epoch resulting in reduced exploration. In the GSO algorithm since the superswarm is created anew from the best solutions found by the subswarms everytime at the start of Level 2, the GSO algorithm has more opportunity to explore new solutions. This independence of search between two consecutive exploitative phases of the GSO results in better exploration and ultimately better quality of the final solution compared to MCPSO algorithms (both COM-MPSO and COL-MPSO versions).

4.3. Discussion on running time complexity

To make a comparison of the running-time complexity of different algorithms, the number of function evaluations is considered to be the most time consuming operation in relation to other operations such as variable updates. Table 5 shows the number of

Please cite this article in press as: V. Muthiah-Nakarajan, M.M. Noel, Galactic Swarm Optimization: A new global optimization metaheuristic inspired by galactic motion, Appl. Soft Comput. J. (2015), http://dx.doi.org/10.1016/j.asoc.2015.10.034

Table 10
Mean running time over 40 trials for the 10D case in seconds.

	f_1	f_2	f_3	f_4	f_5	f_6	f_7
GSO	**3.7862**	**3.8774**	**4.6338**	**5.4220**	**3.3010**	**4.3213**	**3.2778**
DNLPSO	5.1405	5.9077	6.5069	8.3406	5.4317	7.0744	5.7199
CLPSO	8.5077	9.3382	8.8199	9.2418	9.3534	10.3393	9.9792
UPSO	9.3104	10.0841	9.7654	10.0121	10.0630	11.0788	10.8894
FIPS	7.4584	8.4448	7.7237	8.0370	8.2562	9.2056	8.8707

function evaluations for different algorithms from which it is evident that the GSO algorithm converges faster than existing PSO variants.

Since the GSO algorithm as well as its competitors analysed in this paper run for a fixed number of iterations rather than a termination condition, the running time complexity is strongly influenced by the maximum number of iterations. The running time complexity is roughly of the order of $\mathcal{O}(D \times f.eval)$, where D is the dimensionality and $f.eval$ is the number of function evaluations. Apart from function evaluations, frequent variable updates are also performed in all the algorithms. However, as can be seen from pseudocode given in Algorithm 1, the GSO algorithm employs simple variable updates compared with other competing algorithms. DNLPSO and CLPSO as well as UPSO and FIPS use refreshing and regrouping while updating variables which require complex vector operations. Table 10 shows the mean running time over 40 runs for different objective functions. The best results are highlighted in bold. The results shown in Table 10 indicate that GSO has the smallest mean running time due to simpler variable updates. Thus the performance of the GSO algorithm is superior in terms of speed of convergence and accuracy of final solution compared to existing state-of-the-art PSO algorithm variants.

5. Conclusion

In this paper a new optimization algorithm inspired by the motion of stars, galaxies and superclusters of galaxies under the gravitational influence was proposed. The new GSO algorithm converged faster to a significantly better solution compared to state-of-the-art PSO algorithms on a wide variety of high dimensional and multi-modal benchmark optimization problems. Each epoch of the GSO algorithm employs an explorative phase in which different subpopulations independently explore and an exploitative phase where the best solutions of each subpopulation are updated. The use of an explorative and exploitative phase in each epoch prevents premature convergence and allows multimodal surfaces to be efficiently explored. The results obtained using the GSO algorithm are significantly superior to those presented in [33,34,18,36] as well as many representative multiswarm algorithms. Further work on the GSO algorithm may explore the benefits of using other population based optimization algorithms like GAs instead of the PSO algorithm in the basic framework of the GSO algorithm.

Acknowledgements

We would like to thank Prof.P. N. Suganthan, Nanyang Technological University, Singapore, for generously providing the PSO codes that were used for comparison in this work.

References

[1] F. van den Bergh, A.P. Engelbrecht, A cooperative approach to particle swarm optimization, IEEE Trans. Evolut. Comput. 8 (June (3)) (2004) 225–239.
[2] L. Messerschmidt, A.P. Engelbrecht, Learning to play games using a PSO-based competitive learning approach, IEEE Trans. Evolut. Comput. 8 (June (3)) (2004) 280–288.
[3] N. Franken, A.P. Engelbrecht, Particle swarm optimization approaches to coevolve strategies for the iterated prisoner's dilemma, IEEE Trans. Evolut. Comput. December (9) (6) (2005) 562–579.
[4] L. Jianli, J. Feng, F. Jiancheng, C. Junchao, Temperature Error Modeling of RLG based on neural network optimized by PSO and regularization, IEEE Sens. J. 14 (March (3)) (2014) 912–919.
[5] H. Yoshida, K. Kawata, Y. Fukuyama, S. Takayama, Y. Nakanishi, A particle swarm optimization for reactive power and voltage control considering voltage security assessment, IEEE Trans. Power Syst. 15 (November (4)) (2000) 1232–1239.
[6] M.A. Abido, Optimal design of power-system stabilizers using particle swarm optimization, IEEE Trans. Energy Convers. 17 (September (3)) (2002) 406–413.
[7] M. Mao, P. Jin, L. Channg, H. Xu, Economic analysis and optimal design on microgrids with SS-PVs for industries, IEEE Trans. Sustain. Energy 5 (September (4)) (2014) 1328–1336.
[8] M. Donelli, R. Azaro, F.G.B. De Natale, A. Massa, An innovative computational approach based on a particle swarm strategy for adaptive phased-arrays control, IEEE Trans. Antennas Propag. 54 (March (3)) (2006) 888–898.
[9] S.S. Mohamed, M.M.A. Salama, Prostate cancer spectral multifeature analysis using TRUS images, IEEE Trans. Med. Imaging 27 (April (4)) (2008) 548–556.
[10] D. Goldberg, Genetic Algorithms in Search, Optimization and Machine Learning, Addison-Wesley Professional, Reading, MA, 1989.
[11] F. Glover, Tabu search – part 1, ORSA J. Comput. 1 (2) (1989) 190–206.
[12] F. Glover, Tabu search – part 2, ORSA J. Comput. 2 (1) (1990) 4–32.
[13] M. Dorigo, L.M. Gambardella, Ant colony system: a cooperative learning approach to the traveling salesman problem, IEEE Trans. Evolut. Comput. 1 (1) (1997) 53–66.
[14] S. Kirkpatrick, C.D. Gelatt, M.P. Vecchi, Optimization by simulated annealing, Science 20 (4598) (1983) 671–680.
[15] R.C. Eberhart, J. Kennedy, A new optimizer using particle swarm theory, in: Proceedings of the Sixth International on Micro Machines and Human Science, 1995, October, pp. 39–43.
[16] J. Kennedy, R.C. Eberhart, Swarm Intelligence, Morgan Kaufmann Academic Press, 2001.
[17] R. Poli, J. Kennedy, T. Blackwell, Particle swarm optimization – an overview, in: Swarm Intelligence, 2007.
[18] J.J. Liang, A.K. Qin, P.N. Suganthan, S. Baskar, Comprehensive learning particle swarm optimizer for global optimization of multimodal functions, IEEE Trans. Evolut. Comput. 10 (June (3)) (2006) 281–293.
[19] D. Sedighizadeh, E. Masehian, Particle swarm optimization methods, taxonomy and applications, Int. J. Comput. Theory Eng. 1 (December (5)) (2009) 486–502.
[20] Y. Shi, R.C. Eberhart, A modified particle swarm optimizer, in: Proceedings of IEEE Congress in Evolutionary Computation, 1998, pp. 69–73.
[21] Encyclopdia Britannica, "Cosmos", Encyclopaedia Britannica Ultimate Reference Suite, Encyclopdia Britannica, Chicago, 2010.
[22] T. Blackwell, J. Branke, Multi-swarm optimization in dynamic environments, Lect. Notes Comput. Sci. 3005 (2004) 489–500.
[23] J.J. Liang, P.N. Suganthan, Dynamic multi-swarm particle swarm optimizer, Proc. Swarm Intell. Symp. 2005 (2005) 124–129.
[24] Y. Marinakis, M. Marinaki, A hybrid multi-swarm particle swarm optimization algorithm for the traveling salesman problem, Comput. Oper. Res. 37 (2010) 432–442.
[25] S.Z. Zhao, P.N. Suganthan, Q.K. Pan, M.F. Tasgetiren, Dynamic multi-swarm particle swarm optimizer with harmony search, Expert Syst. Appl. 38 (2011) 3735–3742.
[26] J. Zhang, X. Ding, A multi-swarm self-adaptive and co-operative particle swarm optimization, Eng. Appl. Artif. Intell. 24 (2011) 958–967.
[27] B. Niu, Y. Zhu, X. He, H. Wu, MCPSO: a multi-swarm cooperative particle swarm optimizer, Appl. Math. Comput. 185 (2007) 1050–1062.
[28] M. Jamil, X.S. Yang, A literature survey of benchmark functions for global optimization problems, Int. J. Math. Modell. Numer. Optim. 4 (2) (2013) 150–194.
[29] R.W. Garden, A.P. Engelbrecht, Analysis and classification of optimisation benchmark functions and benchmark suites, in: IEEE Congress on Evolutionary Computation, 2014, pp. 1641–1649.
[30] S.P. Lim, H. Haron, Performance comparison of Genetic Algorithm, Differential Evolution and Particle Swarm Optimization towards benchmark functions, in: IEEE Conference on Open Systems, 2013, December, pp. 41–46.
[31] J.J. Liang, B.Y. Qu, P.N. Suganthan, Problem Definitions and Evaluation Criteria for the CEC 2014 Special Session and Competition on Single Objective Real-Parameter Numerical Optimization, Technical Report 201311, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China And Technical Report, Nanyang Technological University, Singapore, 2013, December.
[32] R.C. Eberhart, Y. Shi, Comparing inertia weights and constriction factors in particle swarm optimization, in: Proceedings of the 2000 Congress on Evolutionary Computation, 2000, July, pp. 84–88.

Please cite this article in press as: V. Muthiah-Nakarajan, M.M. Noel, Galactic Swarm Optimization: A new global optimization metaheuristic inspired by galactic motion, Appl. Soft Comput. J. (2015), http://dx.doi.org/10.1016/j.asoc.2015.10.034

[33] R. Mendes, J. Kennedy, J. Neves, The fully informed particle swarm: simpler, maybe better, IEEE Trans. Evolut. Comput. 8 (3) (2004) 204–210.
[34] K.E. Parsopoulos, M.N. Vrahatis, UPSO – a unified particle swarm optimization scheme, Lect. Ser. Comput. Sci. (2004) 868–873.
[35] P.W. Moore, G.K. Venayagamoorthy, Empirical study of an unconstrained modified particle swarm optimization, in: IEEE Congress of Evolutionary Computation, 2006, pp. 1477–1482.
[36] M. Nasir, S. Das, D. Maity, S. Sengupta, U. Halder, P.N. Suganthan, A dynamic neighbourhood learning based particle swarm optimizer for global numerical optimization, Inf. Sci. (2012, May) 16–36.
[37] W.J. Conover, Practical Nonparametric Statistics, 2nd ed., John Wiley and Sons, 1980, pp. 225–226.
[38] E.L. Lehamnn, Elements of Large Sample Theory, Springer, 1999.
[39] Matlab Help, 2015 http://in.mathworks.com/help/stats/ranksum.html.

Please cite this article in press as: V. Muthiah-Nakarajan, M.M. Noel, Galactic Swarm Optimization: A new global optimization metaheuristic inspired by galactic motion, Appl. Soft Comput. J. (2015), http://dx.doi.org/10.1016/j.asoc.2015.10.034