Knowledge-Based Systems xxx (xxxx) xxx

Contents lists available at ScienceDirect

Knowledge-Based Systems

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

Parameter tuning for meta-heuristics☆

Susheel Kumar Joshi, Jagdish Chand Bansal *

Department of Applied Mathematics, South Asian University, New Delhi, India

ARTICLE INFO

Article history:
Received 17 April 2019
Received in revised form 30 September 2019
Accepted 6 October 2019
Available online xxxx

Keywords:
Parameter tuning
Gravitational search algorithm (GSA)
Gravitational constant
Meta-heuristics

ABSTRACT

These days meta-heuristic algorithms are gaining lot of popularity. The performance of the meta-heuristics depends upon the suitable selection of user dependent parameters. Finding the most suitable values for the parameters (fine tuning) is a challenging problem. This paper proposes a generalized strategy to find the most suitable value of any parameter for a meta-heuristic algorithm. The approach is based on the relation between algorithm's performance and functional landscape. The proposed approach is evaluated by applying it to a recent meta-heuristic algorithm, Gravitational Search Algorithm (GSA). The parameter α which plays a vital role in the convergence of GSA search process, is fine tuned using the proposed strategy. Obtained values of α, change the nature of gravitational coefficient G from monotonic to non-monotonic for a cluster free diversified search. The proposed strategy is tested over CEC-2015 test suite. Various statistical tests have been applied to compare the obtained results with recent variants of GSA and other state-of-the-art meta-heuristics. Results confirm that the parameters obtained using proposed approach significantly improve the results.

1. Introduction

To achieve the optimal parameter setting as per the search requirement is the most challenging yet not significantly explored research area in the context of the meta-heuristic algorithms. The parameter tuning is the most crucial factor, which is responsible for the efficiency of any meta-heuristic algorithm. An algorithm having properly tuned parameters converge to the global position more swiftly, which further improves the efficiency of an algorithm.

In the literature, various approaches have been used to tune the parameters of different meta-heuristic algorithms. Akay et al. [1] analyzed the behavior of the control parameters on the performance of artificial bee colony (ABC). To obtain the parameter fine tuning of an evolutionary algorithm, Smit et al. [2] used a method called REVAC (Relevance Estimation and Value Calibration Method). Iwasaki et al. [3] tuned the parameters of particle swarm optimization (PSO) by using a dynamic parameter tuning method. To find the optimal parameters of simulated annealing (SA), Bartz-Beielstein [4] proposed a method based on regression analysis, statistical design of experiments, and tree-based regression. Vafadarnikjoo et al. [5] tuned the parameters of ABC by using full factorial design of experiments approach. Tavana et al. [6] also used full factorial experimental design approach to tune the parameters of artificial immune algorithm (AIA). To tune the parameters of genetic algorithm (GA), Yu et al. [7] used design of experiments approach. Amoozegar et al. [8] also used design of experiments approach to tune the GSA parameters. To find the optimal parameter setting of Intelligent Water Drops (IWD) algorithm, Kayvanfar et al. [9] employed the full factorial experiments approach. Detail description about the tuning strategies of meta-heuristic algorithms can be found in [10].

This paper proposes a general strategy for fine tuning of parameters for any meta-heuristic algorithm and its application to fine tune the parameter α of Gravitational Search Algorithm (GSA). Therefore the latest research in fine tuning methodologies of GSA parameter α is reviewed here. In GSA, α is a parameter which plays a vital role in the convergence ability of GSA model. Throughout the evolutionary process, all agents participate in the search movement with the constant value of α. The fixed α defines the gravitational constant (G) as a monotonic decreasing function. Since G scales the step size of the agent, therefore this monotonic behavior of G emphasizes exploitation over exploration in the middle phase of the evolutionary process, which results, a rapid loss of diversity and premature convergence. To address this limitation, a number of α-adjusting strategies have been proposed. To avoid the possibilities of premature convergence, A. Sombra et al. [11] used a fuzzy strategy to adjust the parameter α. Chaoshun Li et al. [12] introduced a hyperbolic function to model α as a variable entity with respect to iteration. This

☆ No author associated with this paper has disclosed any potential or pertinent conflicts which may be perceived to have impending conflict with this work. For full disclosure statements refer to https://doi.org/10.1016/j.knosys.2019.105094.

* Corresponding author.
E-mail addresses: sushil4843@gmail.com, susheeljoshi@students.sau.ac.in (S.K. Joshi), jcbansal@sau.ac.in (J.C. Bansal).

https://doi.org/10.1016/j.knosys.2019.105094
0950-7051/

variability of α reduces the chance of premature convergence. To prevent the possibilities of premature convergence, Saeidi-Khabisi et al. [13] introduced an adaptive α strategy with the help of fuzzy logic controller. To alleviate the premature problem, Sun et al. [14] proposed a self adaptive α which is guided by the variation of an agent's position and its fitness.

In addition, there is another strong research trend to develop GSA by embedding new learning strategies into it. Sarafrazi et al. [15] improved the exploration and exploitation ability of GSA by using disruption operator. To prevent the premature convergence and improve the convergence characteristic of GSA, Doraghinejad et al. [16] used the application of Black Hole Principle. In [17], the agents move their position under the influence of the Gbest solution (best solution obtained so far). This influential movement improves the exploitation ability of the search process. To improve the convergence speed of GSA, Shaw et al. [18] used the opposition-based learning. To improve the exploitation, Chen et al. [19] proposed a local search operator as a multi-type local improvement scheme. To improve the exploitation ability of GSA, Susheel et al. [20] introduced the encircling behavior of gray wolves in GSA. To overcome the premature convergence, Li et al. [21] hybridized differential evolution (DE) with GSA. In GSA, the concept of the dynamic (adaptive) parameter is proposed by Seyedali Mirjalili et al. [22]. In the proposed variant, the gravitational constant (G) adapts the chaotic behavior using 10 different chaotic maps. For a fix chaotic map, G follows a fix chaotic nature throughout the search process. To provide better tradeoff between exploration and exploitation, Zhang et al. [23] used a dynamic neighborhood-based learning strategy. To overcome stagnation and improve the convergence speed of GSA, Bansal et al. [24] introduced a dynamic gravitational constant which varies according to the fitness of the agents.

To overcome premature convergence and to avoid local optima, Han et al. [25] introduced a non linear dynamic behavior in GSA model. Gao et al. [26] proposed two versions of GSA based on two chaotic behavior. In the first version, the random sequences are replaced by chaotic sequences, while the second version uses the chaotic operator as a local search tool. To improve the local search capacity of GSA, Shang et al. [27] used the neighborhood crossover operator (NCO). In an another study [28], the orthogonal crossover is used to improve the exploitation ability of the GSA model. Guvenc et al. [29] proposed a novel escape velocity operator for GSA in which the worst fitted agent uses its escape velocity to improve its position in the search space. To maintain the balance between exploration and exploitation, Sarafrazi et al. [30] introduced the Kepler operator in GSA. Seyedali Mirjalili et al. [31] assigned a memory to each agent of GSA to improve its search ability. To make GSA as a memory based algorithm, the similar studies [32–37] has been done in which the idea of particle swam optimization is employed on GSA through different approaches. To improve the convergence speed of GSA, Shams et al. [38] used a clustering method to reduce the number of agents during the iterations. In [39,40], the application of quantum mechanics theories are used in GSA to improve its performance in preventing premature convergence to local optima. In [19], simulated annealing is used to improve the local search ability of GSA. Khajooei et al. [41] proposed anti-gravity force in GSA. This force restricts the agent to approach towards worst solution. In a similar study, Zandevakili et al. [42] introduced both attractive and repulsive force in GSA. Apart from the continuous search space, GSA has a large list of variants for binary, discrete, and mixed (real and binary both) search space also. Detailed description about these GSA variants can be found in [43,44]. Additionally, the applicability of GSA and its various variants in engineering optimization problems is described in [43]. Besides engineering optimization problems, there are other areas where GSA can be applied. In case of EPQ models [45], sustainable transportation planning [46], maintenance scheduling [47], supply chain network [48–52], EOQ models [53], the problems can be modeled as optimization problems. GSA with proposed parameter tuning strategy can be a good approach for these kind of optimization models.

Recently, Peio Loubière et al. [54] used morris method as a tool of sensitivity analysis to find the most influential direction for the furtherance of the evolutionary process in the uni-perturbed search strategy of ABC. In Morris-ABC [54], a matrix of elementary effects with N rows (population size) and D columns (dimension) is created, in which all entries are initialized to 1. After the algorithm is looping, the elementary effect matrix (EE) is updated its $(i, j)th$ entry by the step size (uni dimensional offset) of the ith agent in the jth dimension. The linear or non linear impact of jth dimension on the model is equivalent to the corresponding weight of the jth column in EE matrix which is formulated as a function of its mean (for linear impact) and standard deviation (for non-linear impact).

In the proposed strategy of parameter tuning, the elementary effect matrix (EE) is used as elementary weight matrix (EW) to find the most suitable value of the parameter as per the search requirement of the meta-heuristic algorithm. EW updates its jth column by N weights of N candidate solutions under the influence of a selected value a_j of the parameter. Finally, the mean of the corresponding column decides the influence of the selected value a_j on the fitness landscape f. The N weights of solutions for the particular evolutionary state depend upon the topological behavior of the functional landscape f of the problem. Further, the proposed method is used to tune α for G in GSA. This tuning provides those values of α which have the high impact on the functional landscape f. These values generate a self adaptive G, which produces a cluster free diversified search mechanism in GSA.

The remaining paper is organized as follows: A detailed description of proposed method of parameter tuning is presented in Section 2. Section 3 provides a brief overview of basic GSA. In Section 4, the proposed method of parameter tuning is implemented on GSA. Section 5 describes the experiment results, comparative study and managerial implications of the study. Finally the paper is concluded in Section 6.

2. The proposed method of parameter tuning

In this Section, we propose a novel approach of parameter tuning based on the topological characteristics of the optimization problem i.e., its functional landscape f. The objective of this approach is to find the most influential values of any parameter say, α of an algorithm within the pre-defined range. These values of α make perfect informative relation between landscape f and the algorithm's performance.

To find the most influential values of parameter α, first, we create a $N \times M$ matrix of elementary weights $[e_{ij}]_{N\times M}$ (named as EW matrix), where N is the population size used in the given algorithm and M represents the distinct number of possible values of parameter α. Also $e_{ij} \in \mathbb{R}, \forall i = 1 : N, \forall j = 1 : M$.

Initially, we take $e_{ij} = 1; \forall i = 1 : N, \forall j = 1 : M$. If $\alpha \in [a_1, a_M]$, we consider that the possible values of α are a_1, a_2, ...a_M. Here $a_j's$ $(j = 1 : M)$ are equally spaced. The objective of the proposed strategy is to select one of the $a_j's$ as the most influential value of α. Let $\tau = \{a_1, a_2, \ldots, a_M\}$. Now consider that a_j is corresponding to the jth column of EW matrix.

In the proposed strategy, α can attain any particular a_j, based on the mean value $(\mu_j = \frac{1}{N}\sum_{i=1}^{N} e_{ij}; \forall j = 1, 2, \ldots, M)$ of the entries corresponding to jth column. A small value of μ_j indicates the low impact of corresponding value of α on f while a large

Please cite this article as: S.K. Joshi and J.C. Bansal, Parameter tuning for meta-heuristics, Knowledge-Based Systems (2019) 105094, https://doi.org/10.1016/j.knosys.2019.105094.

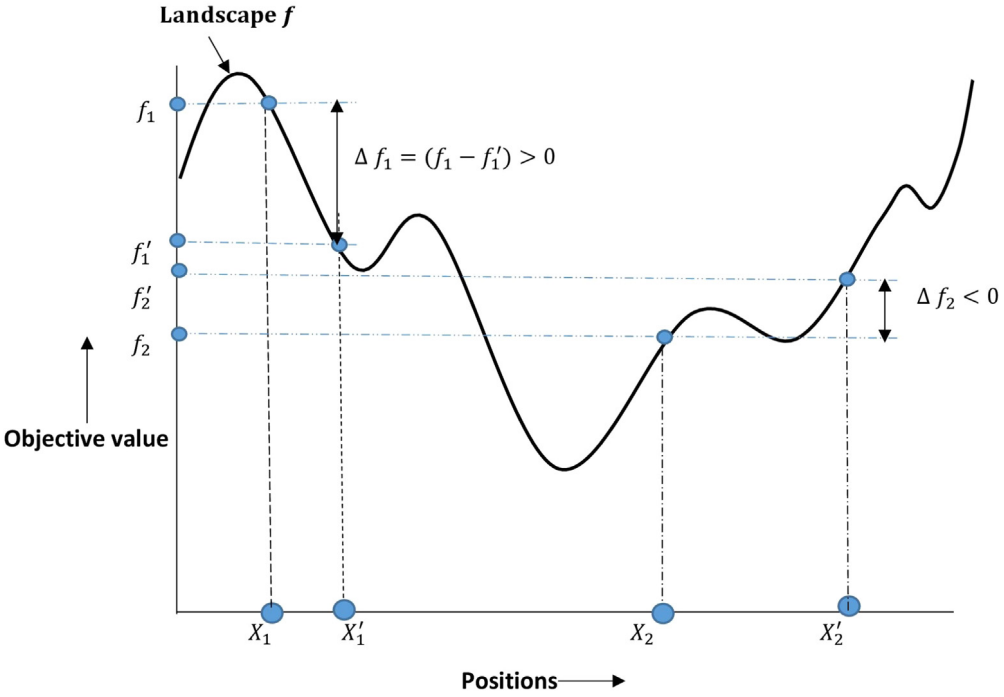


Fig. 1. 2-D presentation of an evolutionary process in the span of its two consecutive evolutionary states.

value of μ_j admits its high impact on f. Therefore, $\mu_j \geq \mu_t,\ \forall\ t = 1 : M$ then $\alpha = a_j$. Ties are broken arbitrarily. Significance of this strategy for selection of α and to calculate the value of α in next iteration, the column having maximum mean of entries μ_j (call it, most influential column) is modified as described below.

The optimization problems can have different fitness landscapes f with respect to their topological characteristics like uni-modal, multi-modal, shifted or biased or both. Under the supervision of meta-heuristic algorithm, the objective values of the candidate solutions move on these landscapes according to their movement in the search space. Fig. 1 illustrates an evolutionary process in the span of its two consecutive evolutionary states. In this Figure, the movement of two candidate solutions and their associated objective values are described under the environment of a multi-modal landscape f (in the minimization case). The first (second) candidate solution changes its position from X_1 to X_1' (X_2 to X_2') and the corresponding objective value changes from f_1 to f_1' (f_2 to f_2'). It is clear from Fig. 1 that the first solution's movement is in the better direction since it reduces the objective value ($\Delta f_1 > 0$), whereas the second solution increases the objective value ($\Delta f_2 < 0$). Δf_1 provides the extra information about the topological region between f_1 and f_1' while the topological knowledge about the region between f_2 and f_2' is evaluated by Δf_2. Therefore the evolutionary process between $(t-1)th$ and tth evolutionary states (iterations) consists a set of objective value differences which can be defined as:

$$OD(t) = \{\Delta f_i : i = 1 : N\} \tag{1}$$

where t is the iteration (evolutionary state).

Fig. 2 illustrates the behavior of the OD set in case of two solutions (i= 2) under the uni-modal landscape (sphere function). For every span of two consecutive evolutionary states, there exists a set OD having two values (Δf_1 and Δf_2). In general, the population of N candidate solutions have its own OD having N values which collectively provides the extra information about the N regions of the landscape through their magnitudes and signs. Magnitude approximates the topological behavior of the landscape covered by the solution in this movement, while sign indicates solution's direction with respect to the optima. With respect to magnitude and sign, the set OD can be classified into the following six categories.

1. All Δf_i in OD have large magnitude (LM) with positive sign, i.e. solutions have been improved largely.
2. All Δf_i in OD have small magnitude (SM) with positive sign, i.e. minor improvement in the solutions.
3. All Δf_i in OD have large magnitude (LM) with negative sign, i.e. solutions became worse.
4. All Δf_i in OD have small magnitude (SM) with negative sign, i.e. solutions became minor worse than that of previous.
5. Δf_i in OD have mixed (positive and negative both) magnitudes with mixed sign, i.e. few solutions have been improved while few became worse.
6. All Δf_i in OD have equal magnitude with same sign, i.e. no change in the solutions.

The first category helps to achieve the fast convergence in the early evolutionary states while the second category may help in the exploitation in the terminal states of the search process. But due to the same sign, both categories are not suitable for multi-modal landscapes. In the multi-modal landscape, these two categories lead the whole population of solutions either towards left (in minimization case) or the right (in maximization case) region of the landscape which further can result stagnation. Third and fourth categories are not suitable for any kind of landscape (uni-modal or multi-modal) due to the negative signs while the fifth category helps to maintain the diversity of the search space. Sixth category indicates that either the whole population of solutions has achieved the optimal point or it has stagnated somewhere.

On the account of the impact of these categories on the landscape, we have assigned different weights for all six categories by three activation functions defined in Algorithm 1. Fig. 3 shows the distribution of weights for first five categories calculated based on Algorithm 1. The first two categories are positively normalized by the dotted region of Fig. 3(a), while the △ region of Fig. 3(a)

Please cite this article as: S.K. Joshi and J.C. Bansal, Parameter tuning for meta-heuristics, Knowledge-Based Systems (2019) 105094, https://doi.org/10.1016/j.knosys.2019.105094.

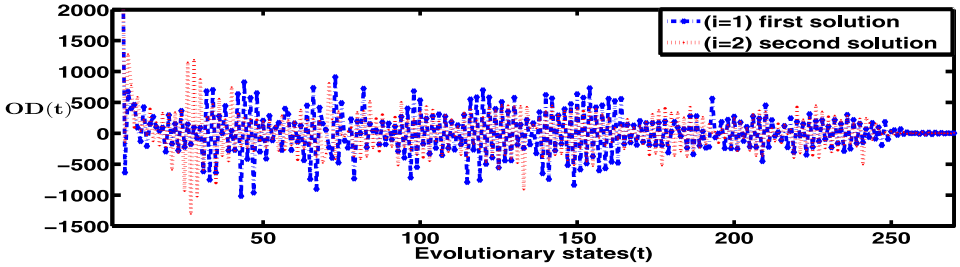
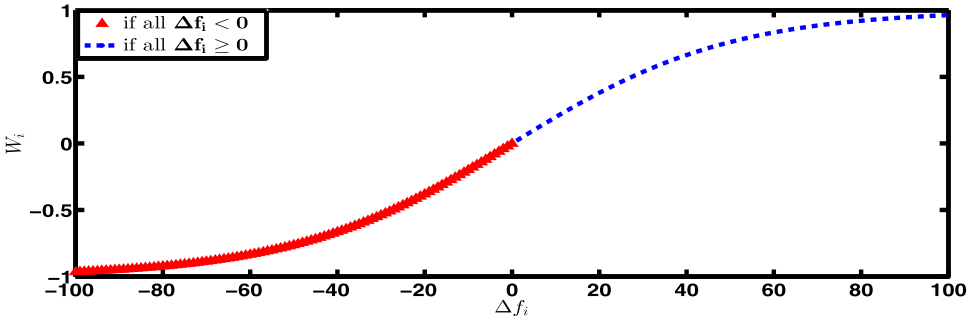
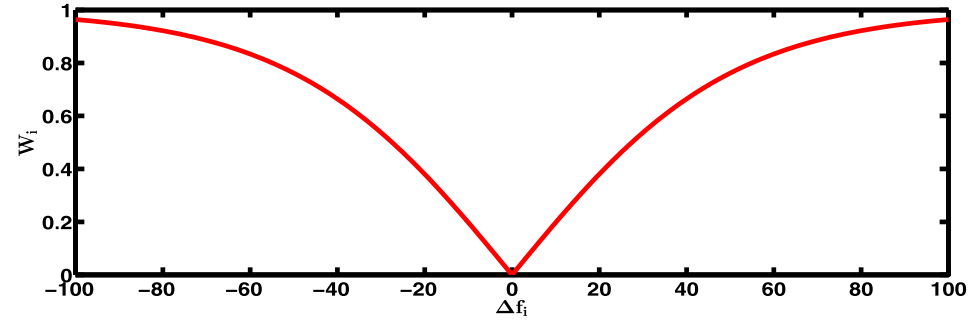


Fig. 2. *OD* of the two candidate solutions (i=1 and 2) for sphere function (unimodal landscape).



(a) Activation function for all $\Delta f_i < 0$ or all $\Delta f_i \geq 0$



(b) Activation function for some $\Delta f_i < 0$ and some $\Delta f_i \geq 0$ both

Fig. 3. Different weights for different categories of *OD*.

Algorithm 1 N-dimensional Weight:

if (all $\Delta f_i \geq 0$ or all $\Delta f_i < 0$) **then**
  Weight$_{a_j}(X_i) = tanh(\frac{\Delta f_i}{N})$, $\forall i \in 1:N$
else
  if (some $\Delta f_i \geq 0$ and some $\Delta f_i < 0$) **then**
    Weight$_{a_j}(X_i) = |tanh(\frac{\Delta f_i}{N})|$, $\forall i \in 1:N$
  else
    if (all Δf_i are equal with same sign) **then**
      Weight$_{a_j}(X_i) = -2$, $\forall i \in 1:N$
    end if
  end if
end if

Algorithm 2 Column selection algorithm:

Find mean vector $MV = [\mu_1, \mu_2, ..., \mu_M]$
k ← The column number of *EW* having maximum mean value μ_k in *MV*
if Two or more columns have equal maximum mean values **then**
  Select any one of them randomly
end if

provides negative weights for the third and fourth categories. Since the fifth category of *OD* is responsible for maintaining the diversity in the search space, therefore equal and positive weights for both negative and positive values of *OD* are provided by Fig. 3(b). The sixth category is responsible for stagnation, hence a large negative weight −2 is assigned for every value of *OD* of that category through the third activation function of Algorithm 1.

Therefore, each selected value a_j for the parameter α have an N dimensional weight corresponding to its *OD* set. This N dimensional weight represents the impact of the selected a_j on the given landscape f. Thus, the performance of an evolutionary algorithm is increased by providing those value of a_j which have maximum N dimensional weight.

The proposed approach for fine tuning of a parameter can be applied to any meta-heuristic algorithm. The objective of the proposed fine tuning methodology is to maintain the diversified search of the meta-heuristic algorithm. Further, the purposed approach has been verified by applying it to a recent and popular meta-heuristic, Gravitational Search Algorithm (GSA). Next

Please cite this article as: S.K. Joshi and J.C. Bansal, Parameter tuning for meta-heuristics, Knowledge-Based Systems (2019) 105094, https://doi.org/10.1016/j.knosys.2019.105094.

sections introduce the basic GSA and the implementation of proposed strategy in GSA.

3. Basic GSA

Gravitational Search Algorithm (GSA) is a new meta-heuristic technique for optimization developed by Rashedi et al. [55]. This algorithm is inspired by the law of gravity and the law of motion.

The GSA algorithm can be described as follows:

Consider the population of N agents (candidate solutions), in which each agent X_i in the search space $\mathbb{S}$ is defined as:

$$X_i = (x_i^1, \ldots.., x_i^d, \ldots.., x_i^n), \quad \forall\, i = 1, 2, \ldots.., N \tag{2}$$

Here, X_i shows the position of ith agent in n-dimensional search space $\mathbb{S}$. The mass of each agent depends upon its fitness value calculated as below:

$$q_i(t) = \frac{fit_i(t) - worst(t)}{best(t) - worst(t)} \tag{3}$$

$$M_i(t) = \frac{q_i(t)}{\sum_{j=1}^{N} q_j(t)}, \quad \forall\, i = 1, 2, \ldots.., N \tag{4}$$

Here,
$fit_i(t)$ is the fitness value of agent X_i at iteration t,
$M_i(t)$ is the mass of agent X_i at iteration t.
Worst(t) and best(t) are worst and best fitness of the current population, respectively.

The acceleration of ith agent in dth dimension is denoted by $a_i^d(t)$ and defined as:

$$a_i^d(t) = \frac{F_i^d(t)}{M_i(t)} \tag{5}$$

Where $F_i^d(t)$ is the total force acting on the ith agent by a set of Kbest heavier masses in dth dimension at iteration t. $F_i^d(t)$ is calculated as:

$$F_i^d(t) = \sum_{j \in KBEST, j \neq i} rand_j \times F_{ij}^d(t) \tag{6}$$

Here, KBEST is the set of first K agents with the best fitness values (say Kbest agents) and biggest masses and $rand_j$ is a uniform random number between 0 and 1. Cardinality of KBEST, i.e. Kbest is a linearly decreasing function of time. The value of Kbest will reduce in each iteration and at the end only one agent will apply force to the other agents. At the tth iteration, the force applied on agent i by agent j in the dth dimension is defined as:

$$F_{ij}^d(t) = G(t)\frac{M_i(t)M_j(t)}{R_{ij} + \epsilon}(x_i^d(t) - x_j^d(t)) \tag{7}$$

Here, $R_{ij}(t)$ is the Euclidean distance between two agents, i and j. ϵ ($\epsilon > 0$) is a small number. Finally, the acceleration of an agent in dth dimension is calculated as:

$$a_i^d(t) = \sum_{j \in KBEST, j \neq i} rand_j G(t)\frac{M_j(t)}{R_{ij} + \epsilon}(x_i^d(t) - x_j^d(t)), \tag{8}$$

$d = 1, 2, \ldots, n$ and $i = 1, 2, \ldots, N$. $G(t)$ is called gravitational constant and is a decreasing function of time (iteration):

$$G(t) = G_0 e^{-\alpha \frac{t}{T}} \tag{9}$$

G_0 and α are constants and set to 100 and 20, respectively. T is the total number of iterations.

The velocity update equation of an agent X_i in dth dimension is given below:

$$v_i^d(t+1) = rand_i \times v_i^d(t) + a_i^d(t) \tag{10}$$

Based on the velocity calculated in Eq. (10), the position of an agent X_i in dth dimension is updated using position update equation as follow:

$$x_i^d(t+1) = x_i^d(t) + v_i^d(t+1) \tag{11}$$

where $v_i^d(t)$ and $x_i^d(t)$ represent the velocity and position of agent X_i in dth dimension, respectively. $rand_i$ is uniformly distributed random number in the interval [0, 1].

4. Implementation of the proposed parameter tuning strategy in GSA

In GSA, an agent's movement is guided by the net gravitational force exerted on it by all the agents having best fitnesses (Kbest agents). The net gravitational force is scaled under the supervision of an exponential monotonic decreasing function G of time (iteration), which determines the step size of an agent for its next move. Although the monotonic decreasing nature of G provides an effective way to balance exploration and exploitation, but the fix value of α in G does not produce the significant change in G with time, resulting, a rapid loss of diversity and premature convergence.

To address the drawback of fixed α, the proposed method of parameter tuning (section:2) is used to tune α for G within the predefined range [5, 70]. The GSA integrated with the proposed approach, called PTGSA (parameter tuning in gravitational search algorithm) is described in Algorithm 3. For the better understanding, the flowchart of PTGSA is also added in Fig. 4.

Algorithm 3 PTGSA algorithm:

1: Initialize the position and velocities of agents
2: Initialize the elementary weight matrix (EW) along with M distinct values of α from the set $\tau = \{a_1, a_2, ..., a_M\}$
3: Find column wise mean μ's of EW
4: **while** Termination criteria is not satisfied **do**
5: Evaluate agents's fitness
6: find best and worst fitness
7: **if** Iteration > 1 **then**
8: $\Delta f_i = f(X_i(\text{iteration})) - f(X_i(\text{iteration-1})),\ \forall i = 1 : N$
9: Find the N-dimensional weight vector (using Algorithm 1)
10: Update the Elementary weight Matrix $EW($ k^{th} column, N-dimensional weight of a_k)
11: Calculate new μ's for EW
12: **end if**
13: Apply k ← Column Selection Algorithm (using Algorithm 2) for α
14: Calculate masses
15: Select k^{th} value of α
16: Calculate G
17: Find acceleration
18: Update velocities and positions of agents
19: **end while**

Figs. 5, 6, 7, and 8 present the graphical analyses of the proposed variant PTGSA on f_1 (Unimodal function), f_4 (Multimodal function), f_6 (Hybrid function), and f_{12} (Composite function) (refer Section 5.1). In each Figure, the first subgraph (a) presents the sensitive behavior of the parameter α according to the model's requirement. The second subgraph (b) presents a comparative study of G based on the sensitive or non-sensitive behavior of α. In these Figures, we can observe that the sensitive α (refer subgraphs (a)) and proposed G (refer subgraphs (b)) have opposite topological behavior throughout the search space. It proves that α is inversely proportional to G (refer Eq. (9)). These best tuned values of α

Please cite this article as: S.K. Joshi and J.C. Bansal, Parameter tuning for meta-heuristics, Knowledge-Based Systems (2019) 105094, https://doi.org/10.1016/j.knosys.2019.105094.

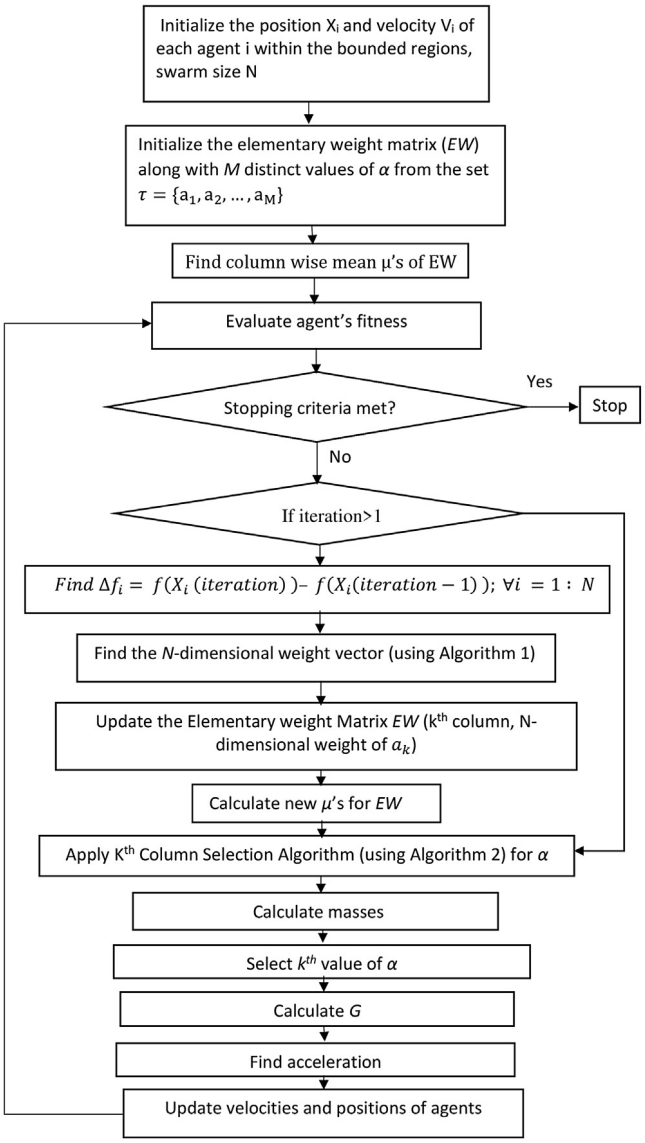


Fig. 4. Flowchart of the proposed PTGSA.

produce a non-monotonic behavior in G which can switch from exploitation to exploration and vise-versa according to the requirement of the evolutionary process. The sudden changes in these graphs avoid the possibility of the premature convergence and provide a cluster free diversified search mechanism.

5. Results and discussion

5.1. Test bed under consideration

In this section, the proposed PTGSA is tested over 15 unconstrained continuous test functions of CEC 2015 test suite [56]. According to the different topological characteristics, these test functions are categorized into four groups : unimodal functions (f_1 and f_2), simple multimodal functions (f_3, f_4, and f_5), hybrid functions (f_6, f_7, and f_8), and composite functions (f_9, f_{10}, f_{11}, f_{12}, f_{13}, f_{14}, and f_{15}). The dimension (D) and the range of the search space are 30 and $[-100, 100]$, respectively.

5.2. Experimental setting

In order to validate the effectiveness and robustness of proposed algorithm, PTGSA is compared with basic GSA along with some state-of-the-art algorithms like Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [57], Biogeography-based optimization (BBO) [58], Disruption in biogeography-based optimization (DBBO) [59], and Differential evolution (DE) [60]. PTGSA is also compared with some recent α adjusting GSA variants like MGSA-α [12], Fuzzy gravitational search algorithm (FGSA) [13], $FS\alpha$(Increase) [11], $FS\alpha$(Decrement) [11], and SCAA [14]. All the comparisons have been done over the CEC 2015 test suite with the following parameter setting, which is as per recommendations of CEC 2015 test set:

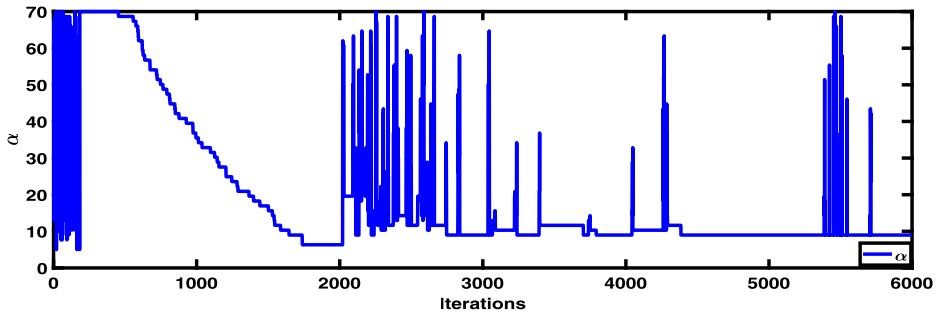
- The number of simulations/run =51,
- Swarm size=50,
- The maximum number of function evaluations for the stopping criteria of the algorithms are set to be $10,000 \times D$,
- Parameters for the algorithms GSA [55], CMA-ES [57], BBO [58], DBBO [59], and DE [60] are considered from the corresponding resources while the results of all α adjusting GSA variants are reproduced from SCAA [14].

5.3. Comparison of PTGSA with state-of-the-art algorithms

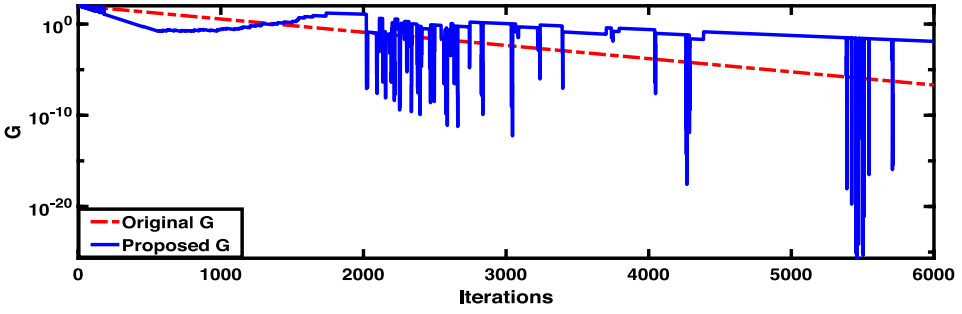
Following the experimental setting explained in Section 5.2, the searching behavior of the proposed PTGSA is compared with some state-of-the-art algorithms. The experimental results of fitness errors are summarized in Table 1. The Table 1 lists the three metrics of fitness error: Mean error(Mean), Standard Deviation of error (*SD*), and Wilcoxon Signed-Rank Test (h-value) [61].

The fitness error is the absolute difference between the best fitness value obtained by the algorithm and the actual global optimum of the optimization problem. The Mean and *SD* of these results validate the searching accuracy of the algorithms, while Wilcoxon Signed-Rank Test checks whether the results obtained by PTGSA and other considered algorithms are significantly different or not. This non-parametric statistical test is performed on these results at 5% level of significance with the null hypothesis, 'There is no significant difference between the results' obtained by PTGSA and other considered algorithms. In Table 1, the '+' (or '-') h-value indicates that PTGSA is significantly better (or worse) than the other considered algorithms, while '=' h-value stands for similar performance between PTGSA and others. The bold entries indicate the best results. As shown in Table 1, PTGSA outperforms other algorithms in term of mean value for 10 test functions including one unimodal (f_1), one multimodal (f_3), all three hybrid functions (f_6, f_7, and f_8), and five composite functions (f_{10}, f_{11}, f_{12}, f_{14}, and f_{15}). Among all metrics of comparison PTGSA confirms its efficacy on 5 test functions (f_1, f_6, f_8, f_{10}, and f_{12}). While for f_5, f_9, and f_{13}, PTGSA performs inferior than others. Here f_5 is multimodal function, while f_9 and f_{13} are composite functions. Since PTGSA has shown superior performance over 5 composite functions out of 7 and over 3 multi-modal functions out of 5. Therefore, the inferior behavior of PTGSA as compare to other existing algorithms over f_5, f_9, and f_{13} does not show that it is not a good choice for composite and multi-modal functions. Furthermore, 56 '+' h-value out of 75 comparison confirms that the proposed PTGSA is significantly better algorithm than other considered algorithms.

Please cite this article as: S.K. Joshi and J.C. Bansal, Parameter tuning for meta-heuristics, Knowledge-Based Systems (2019) 105094, https://doi.org/10.1016/j.knosys.2019.105094.

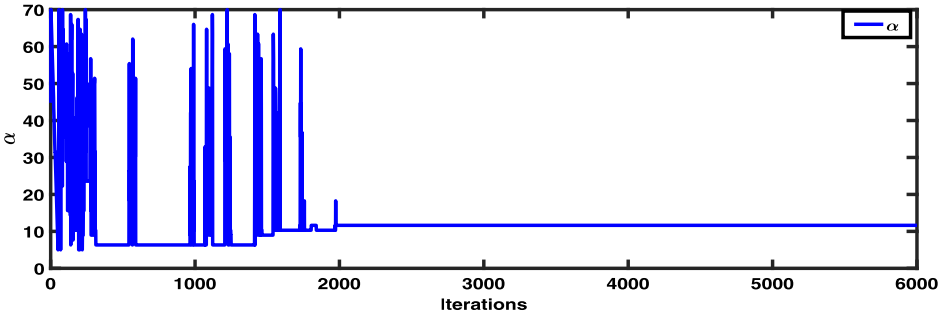


(a) Nature of α with iterations

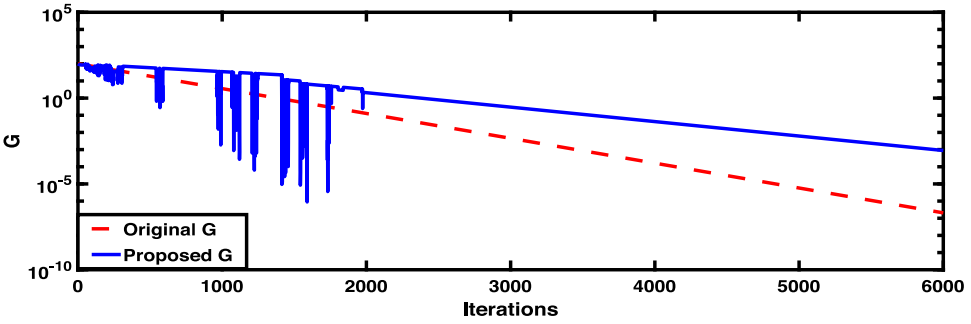


(b) Behaviour of different G with iterations

Fig. 5. Performance analysis of PTGSA for f_1.



(a) Nature of α with iterations



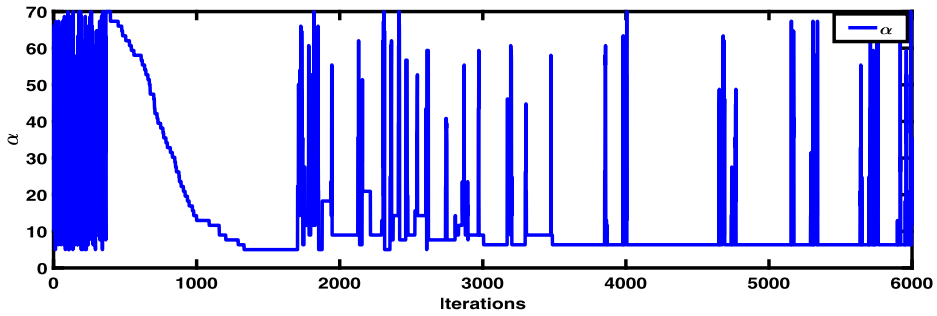
(b) Behaviour of different G with iterations

Fig. 6. Performance analysis of PTGSA for f_4.

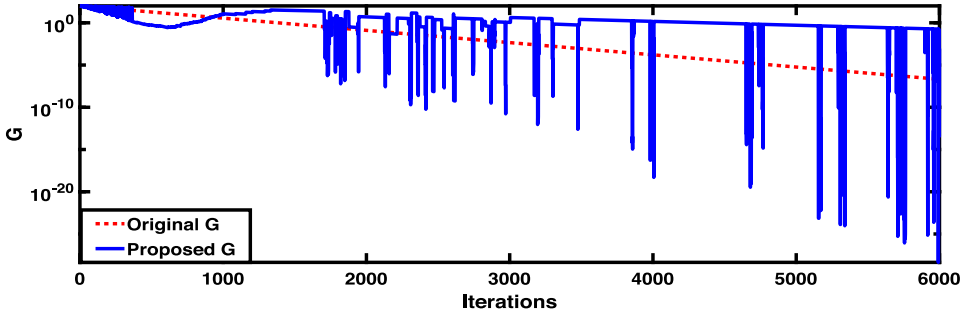
5.4. Comparison of PTGSA with α-adjusting GSA variants

Since the proposed PTGSA is based on the auto adjusting behavior of α, therefore it is essential to compare the performance of PTGSA with other α adjusting GSA variants. To do so, five recent GSA variants with α adjusting strategies: MGSA-α [12], Fuzzy GSA [13], *FS*α(Increase) [11], *FS*α(Decrement) [11], and SCAA [14] are considered for comparison. Table 2 presents the experimental results of fitness errors with two metrics: Mean and SD. The bold entries indicate the best results. Except result of PTGSA, other results are reproduced from SCAA [14]. As per the results shown in Table 2, PTGSA provides better accuracy for

Please cite this article as: S.K. Joshi and J.C. Bansal, Parameter tuning for meta-heuristics, Knowledge-Based Systems (2019) 105094, https://doi.org/10.1016/j.knosys.2019.105094.

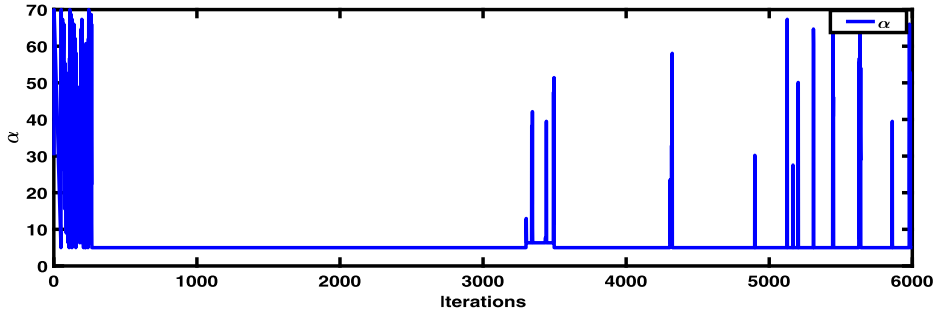


(a) Nature of α with iterations

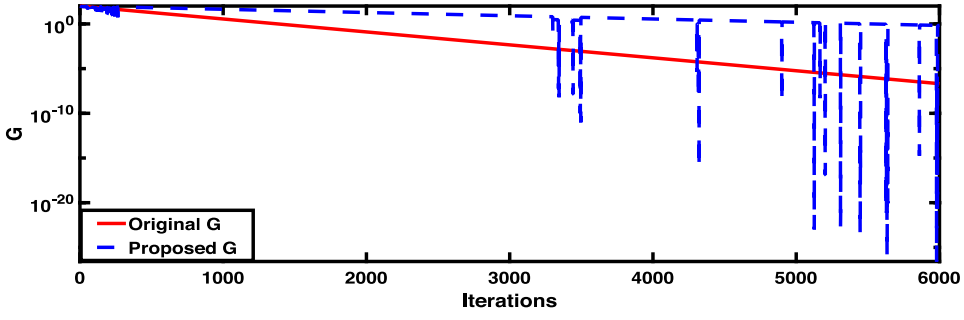


(b) Behaviour of different G with iterations

Fig. 7. Performance analysis of PTGSA for f_6.



(a) Nature of α with iterations



(b) Behaviour of different G with iterations

Fig. 8. Performance analysis of PTGSA for f_{12}.

one unimodal (f_1), one multimodal (f_3), all three hybrid (f_6, f_7, and f_8), and six composite (f_{10}-f_{15}) functions over other considered variants. For six test functions (f_1, f_6, f_7, f_8, f_{10}, and f_{12}) PTGSA outperforms others in both metrics of comparison. f_2, f_4, f_5, and f_9 are the problems for which PTGSA is not better in either criteria. However only for f_2, FSα (Decrement) is better than PTGSA. While for f_5, MGSA-α is better than PTGSA. For other problems no single algorithm is better than PTGSA in both criteria. Therefore, overall PTGSA is an optimal choice among all α-adjusting variants of GSA. Based on the comparison of PTGSA with state-of-the-art algorithms and the recent variants of GSA, it is clear that PTGSA

Please cite this article as: S.K. Joshi and J.C. Bansal, Parameter tuning for meta-heuristics, Knowledge-Based Systems (2019) 105094, https://doi.org/10.1016/j.knosys.2019.105094.

Table 1
Fitness errors of PTGSA along with the considered state-of-the-art algorithms on CEC2015 test suite at 30-D (**TP** denotes Test Problem under consideration and **SD** stands for Standard Deviation).

TP	metrics	GSA	CMA-ES	BBO	DBBO	DE	PTGSA
f_1	Mean	8.12E+05	2.04E+07	8.63E+06	3.77E+06	1.52E+07	**3.52E+05**
	SD	3.42E+05	7.39E+06	4.51E+06	2.31E+06	3.14E+06	**1.79E+05**
	h value	(+)	(+)	(+)	(+)	(+)	
f_2	Mean	**5.58E+02**	1.26E+04	3.76E+05	9.16E+03	4.22E+03	4.28E+03
	SD	**7.01E+02**	1.40E+04	1.48E+05	9.72E+03	2.59E+03	6.89E+03
	h value	(−)	(+)	(+)	(+)	(=)	
f_3	Mean	**2.00E+01**	2.10E+01	2.01E+01	**2.00E+01**	2.07E+01	**2.00E+01**
	SD	6.80E−05	5.22E−02	2.96E−02	**1.44E−07**	5.20E−02	1.01E−02
	h value	(−)	(+)	(+)	(=)	(+)	
f_4	Mean	2.12E+02	1.16E+02	**6.09E+01**	8.55E+01	1.20E+02	2.12E+02
	SD	1.93E+01	6.62E+01	1.36E+01	2.31E+01	**1.06E+01**	2.39E+01
	h value	(−)	(−)	(−)	(−)	(−)	
f_5	Mean	3.82E+03	7.51E+03	**2.24E+03**	2.48E+03	5.01E+03	3.84E+03
	SD	5.19E+02	3.26E+02	4.84E+02	4.69E+02	**2.95E+02**	4.47E+02
	h value	(−)	(+)	(+)	(+)	(+)	
f_6	Mean	1.33E+05	2.64E+06	4.14E+06	9.93E+05	1.47E+06	**2.36E+04**
	SD	5.22E+04	1.46E+06	3.10E+06	1.01E+06	7.05E+05	**2.46E+04**
	h value	(+)	(+)	(+)	(+)	(+)	
f_7	Mean	1.54E+01	8.67E+00	1.47E+01	1.72E+01	1.27E+01	**8.61E+00**
	SD	9.09E+00	9.11E−01	1.32E+01	1.93E+01	**6.27E−01**	1.89E+00
	h value	(+)	(−)	(+)	(+)	(+)	
f_8	Mean	2.41E+04	1.90E+06	2.12E+06	3.30E+05	2.87E+05	**1.52E+04**
	SD	9.84E+03	1.22E+06	2.13E+06	5.69E+05	1.10E+05	**3.77E+03**
	h value	(+)	(+)	(+)	(+)	(+)	
f_9	Mean	1.37E+02	1.52E+02	1.05E+02	**1.03E+02**	**1.03E+02**	1.50E+02
	SD	1.02E+02	9.38E+01	6.08E−01	2.47E−01	**1.88E−01**	1.10E+02
	h value	(+)	(+)	(+)	(+)	(+)	
f_{10}	Mean	3.98E+05	2.54E+06	2.04E+06	2.85E+05	4.66E+05	**2.99E+04**
	SD	1.49E+05	1.58E+06	1.50E+06	8.33E+05	1.94E+05	**1.53E+04**
	h value	(+)	(+)	(+)	(+)	(+)	
f_{11}	Mean	3.40E+02	6.29E+02	7.86E+02	7.51E+02	6.81E+02	**3.30E+02**
	SD	1.46E+02	4.13E+02	**9.26E+01**	1.00E+02	1.18E+02	1.23E+02
	h value	(+)	(+)	(+)	(+)	(+)	
f_{12}	Mean	1.04E+02	1.92E+02	1.09E+02	1.85E+02	1.07E+02	**1.02E+02**
	SD	8.45E−01	2.56E+01	1.59E+00	3.38E+01	6.24E−01	**5.95E−01**
	h value	(+)	(+)	(+)	(+)	(+)	
f_{13}	Mean	1.38E+03	6.95E−03	3.58E−02	**6.14E−03**	2.59E−02	1.23E+03
	SD	1.26E+03	**9.72E−05**	4.00E−03	2.48E−04	2.22E−04	1.46E+03
	h value	(−)	(−)	(−)	(−)	(−)	
f_{14}	Mean	**1.00E+02**	1.02E+04	3.34E+04	6.75E+03	3.35E+04	**1.00E+02**
	SD	**9.63661E−08**	7.72E+03	1.05E+03	9.55E+03	2.96E+02	7.93E−02
	h value	(+)	(+)	(+)	(+)	(+)	
f_{15}	Mean	**1.00E+02**	**1.00E+02**	**1.00E+02**	**1.00E+02**	**1.00E+02**	**1.00E+02**
	SD	1.34832E−10	1.41E−13	2.98E−02	1.12E−03	**1.25E−13**	4.45E−04
	h value	(=)	(=)	(+)	(+)	(=)	

have an excellent search mechanism for unimodal, multi-modal, hybrid, and composite test functions.

To statistically compare the performance of all the above algorithms simultaneously, a two-stage method (i.e., the statistical Friedman test and then a post-hoc test) is used. Table 3 presents the mean Friedman ranks of algorithms for errors. The Friedman test p-Value is 2.275E−05, that clearly indicates the significant difference between the performance of the algorithms even at 1% level of significance. According to Friedman test results, a post-hoc statistical analysis is needed. In this study, for pairwise comparisons, we also reported the adjusted p-values achieved by four post-hoc test procedures, namely Hochberg, Hommel, Benjamin–Hochberg (BH), and Fdr. All these procedures are implemented in R [62,63]. Table 4 presents the p-Values for each comparison which involves the proposed algorithm. From Table 4, the following observations are made:

- For all considered post-hoc test procedures, the proposed PTGSA is significantly better than FuzzyGSA, FSα(Increase), and FSα(Decrement)
- All p-Values are greater than 0.1 for MGSA-α and SCAA, which indicates that the proposed PTGSA performs equally to MGSA-α and SCAA.
- Based on the multiple comparison analysis, the proposed PTGSA is an overall better algorithm as compare to other considered α-adjusting GSA variants.

5.4.1. Managerial implications of the proposed strategy

The study proposes a generic framework to tune a parameter of any meta-heuristic algorithm. The parameter α of GSA has been taken as a case study to show the applicability of the proposed approach. The proposed parameter tuning framework will help the researchers, users to fine tune the parameters of other well known meta-heuristics, like PSO [64], BBO [65], GWO [66],

Table 2
Fitness errors of PTGSA along with α adjusting GSA variants on CEC2015 at 30-D (**TP** denotes Test Problem under consideration and **SD** stands for Standard Deviation).

TP	metrics	MGSA-α	FuzzyGSA	FSα(Increase)	FSα(Decrement)	SCAA	PTGSA
f_1	Mean	1.012E+06	2.707E+06	2.046E+06	1.428E+07	4.093E+05	**3.520E+05**
	SD	3.907E+05	5.063E+06	1.297E+06	8.813E+06	2.405E+05	**1.790E+05**
f_2	Mean	7.445E+02	7.169E+02	7.634E+02	**4.701E+02**	5.328E+02	4.280E+03
	SD	9.367E+02	9.480E+02	1.117E+03	**6.262E+02**	8.585E+02	6.890E+03
f_3	Mean	**2.000E+01**	**2.000E+01**	**2.000E+01**	**2.000E+01**	2.094E+01	**2.000E+01**
	SD	6.594E−05	8.113E−05	1.109E−04	**6.150E−05**	5.840E−02	1.010E−02
f_4	Mean	**1.963E+02**	2.194E+02	2.084E+02	2.363E+02	2.173E+02	2.120E+02
	SD	2.811E+01	**1.924E+01**	2.023E+01	2.261E+01	2.223E+01	2.390E+01
f_5	Mean	**3.625E+03**	3.981E+03	3.773E+03	4.156E+03	3.810E+03	3.840E+03
	SD	**4.456E+02**	4.533E+02	5.325E+02	4.548E+02	4.548E+02	4.470E+02
f_6	Mean	3.553E+05	6.856E+05	9.472E+05	1.704E+06	5.587E+04	**2.360E+04**
	SD	1.725E+05	2.962E+05	3.593E+05	6.204E+05	2.566E+04	**2.460E+04**
f_7	Mean	1.524E+01	2.236E+01	2.441E+01	6.397E+01	9.779E+00	**8.610E+00**
	SD	9.643E+00	1.949E+01	2.058E+01	2.388E+01	3.133E+00	**1.890E+00**
f_8	Mean	2.388E+04	3.050E+04	5.557E+04	1.007E+05	2.154E+04	**1.520E+04**
	SD	7.509E+03	1.152E+04	3.344E+04	1.141E+05	8.329E+03	**3.770E+03**
f_9	Mean	1.265E+02	**1.151E+02**	1.262E+02	2.025E+02	1.358E+02	1.500E+02
	SD	8.221E+01	1.218E+02	**7.943E+01**	1.627E+02	1.016E+02	1.100E+02
f_{10}	Mean	6.936E+05	9.961E+05	1.280E+06	2.485E+06	1.921E+05	**2.990E+04**
	SD	2.310E+05	3.994E+05	6.108E+05	1.004E+06	5.998E+04	**1.530E+04**
f_{11}	Mean	3.343E+02	3.480E+02	3.473E+02	3.849E+02	3.226E+02	**3.300E+02**
	SD	1.121E+02	1.557E+02	1.422E+02	2.021E+02	**9.132E+01**	1.230E+02
f_{12}	Mean	1.036E+02	1.053E+02	1.047E+02	1.449E+02	1.034E+02	**1.020E+02**
	SD	8.215E−01	1.104E+00	9.304E−01	2.771E+01	7.031E−01	**5.950E−01**
f_{13}	Mean	4.759E+03	1.673E+03	1.602E+03	2.100E+03	1.550E+03	**1.230E+03**
	SD	3.987E+03	**1.083E+03**	1.571E+03	1.121E+03	1.296E+03	1.460E+03
f_{14}	Mean	**1.000E+02**	**1.000E+02**	**1.000E+02**	2.821E+04	**1.000E+02**	**1.000E+02**
	SD	8.716E−13	6.668E−10	0.000E+00	7.551E+03	**3.565E−13**	7.930E−02
f_{15}	Mean	**1.000E+02**	**1.000E+02**	**1.002E+02**	1.232E+02	**1.000E+02**	**1.000E+02**
	SD	4.295E−13	2.422E−10	**1.435E−13**	7.414E+00	**1.435E−13**	4.450E−04

Table 3
Friedman ranks of PTGSA and other considered α adjusting GSA variants based on errors.

Algorithms	MGSA−α	FuzzyGSA	FSα(Increase)	FSα(Decrement)	SCAA	PTGSA
Mean of ranks	3.0667	3.8000	3.8667	5.1467	2.7200	**2.4000**

Table 4
p-Values for comparison of PTGSA with the considered α adjusting GSA variants.

Post-hoc procedure	MGSA−α	FuzzyGSA	FSα(Increase)	FSα(Decrement)	SCAA
Hochberg	0.34234	0.00062	0.00079	8.6E−12	0.88188
Hommel	0.34234	0.00056	0.00079	8.6E−12	0.88188
Benjamin–Hochberg(BH)	0.10698	0.00015	0.00019	8.6E−12	0.47244
Fdr	0.10698	0.00015	0.00019	8.6E−12	0.47244

SMO [67] etc and their respective variants. Additionally, this approach can be explored in place of presented approaches of parameter setting in the recent meta-heuristics, like Symbiotic Organism Search (SOS) [68–72], Heat Transfer Search (HTS) [73, 74], Passing Vehicle Search (PVS) [75] etc, and their applications in structural optimization problems.

6. Conclusion

This paper proposes a generic parameter tuning methodology for meta-heuristic algorithms which is based on the topological characteristics of the given optimization problem. For any given iteration, this approach selects the most suitable value among the different pre-assigned values of the parameter based on the relation between f and algorithm's performance. In that iteration, This most suitable value of the parameter prevents the algorithm from the useless evaluations of the objective function and therefore the efficiency of the algorithm increases.

Next, the proposed approach of parameter tuning is demonstrated to tune the parameter α for G as per the search requirement of GSA. These tuned values of α produce a self adaptive G which further avoids the chances of stagnation, resulting a proper diversified search mechanism. The performance of the proposed variant is compared with some state-of-the-art algorithms along with some recent variants of GSA over CEC 2015 test suite. Based on the comparisons, the proposed variant has proved its excellent search ability, specially for hybrid and composite test functions. To enhance the proposed strategy in future, the activation functions can further be explored for getting more information about the landscapes. The proposed strategy can be used to tune the parameter(s) of other well known meta-heuristics.

Please cite this article as: S.K. Joshi and J.C. Bansal, Parameter tuning for meta-heuristics, Knowledge-Based Systems (2019) 105094, https://doi.org/10.1016/j.knosys.2019.105094.

Acknowledgments

The first author acknowledges the funding from South Asian University New Delhi, India to carry out this research.

We would like to show our gratitude to Dr. Ritu Gupta (Curtin University, Western Australia) for sharing her statistical wisdom during the statistical analysis (Friedman test) of this research.

References

[1] B. Akay, D. Karaboga, Parameter tuning for the artificial bee colony algorithm, in: N.T. Nguyen, R. Kowalczyk, S.-M. Chen (Eds.), Computational Collective Intelligence. Semantic Web, Social Networks and Multiagent Systems, Springer Berlin Heidelberg, Berlin, Heidelberg, 2009, pp. 608–619.

[2] S.K. Smit, A.E. Eiben, Comparing parameter tuning methods for evolutionary algorithms, in: Proceedings of the Eleventh Conference on Congress on Evolutionary Computation, CEC'09, IEEE Press, Piscataway, NJ, USA, 2009, pp. 399–406.

[3] N. Iwasaki, K. Yasuda, G. Ueno, Dynamic parameter tuning of particle swarm optimization, IEEJ Trans. Electr. Electron. Eng. 1 (4) (2006) 353–363.

[4] T. Bartz-Beielstein, S. Markon, Tuning Search Algorithms for Real-World Applications: A Regression Tree Based Approach, Universitätsbibliothek Dortmund, 2004.

[5] A. Vafadarnikjoo, S.M.A.K. Firouzabadi, M. Mobin, A. Roshani, A meta-heuristic approach to locate optimal switch locations in cellular mobile networks, in: Proceedings of the International Annual Conference of the American Society for Engineering Management, American Society for Engineering Management (ASEM), 2015, p. 1.

[6] M. Tavana, M.R. Kazemi, A. Vafadarnikjoo, M. Mobin, An artificial immune algorithm for ergonomic product classification using anthropometric measurements, Measurement 94 (2016) 621–629.

[7] A.J. Yu, J. Seif, Minimizing tardiness and maintenance costs in flow shop scheduling by a lower-bound-based ga, Comput. Ind. Eng. 97 (2016) 26–40.

[8] M. Amoozegar, E. Rashedi, Parameter tuning of gsa using doe, in: 2014 4th International Conference on Computer and Knowledge Engineering, ICCKE, 2014, pp. 431–436, http://dx.doi.org/10.1109/ICCKE.2014.6993390.

[9] V. Kayvanfar, M. Zandieh, E. Teymourian, An intelligent water drop algorithm to identical parallel machine scheduling with controllable processing times: a just-in-time approach, Comput. Appl. Math. 36 (1) (2017) 159–184.

[10] M. Mobin, S.M. Mousavi, M. Komaki, M. Tavana, A hybrid desirability function approach for tuning parameters in evolutionary optimization algorithms, Measurement 114 (2018) 417–427.

[11] A. Sombra, F. Valdez, P. Melin, O. Castillo, A new gravitational search algorithm using fuzzy logic to parameter adaptation, in: 2013 IEEE Congress on Evolutionary Computation, 2013, pp. 1068–1074.

[12] C. Li, H. Li, P. Kou, Piecewise function based gravitational search algorithm and its application on parameter identification of avr system, Neurocomputing 124 (2014) 139–148.

[13] F. Saeidi-Khabisi, E. Rashedi, Fuzzy gravitational search algorithm, in: 2012 2nd International eConference on Computer and Knowledge Engineering, ICCKE, 2012, pp. 156–160.

[14] G. Sun, P. Ma, J. Ren, A. Zhang, X. Jia, A stability constrained adaptive alpha for gravitational search algorithm, Knowl.-Based Syst. 139 (2018) 200–213.

[15] S. Sarafrazi, H. Nezamabadi-Pour, S. Saryazdi, Disruption: a new operator in gravitational search algorithm, Sci. Iran. 18 (3) (2011) 539–548.

[16] M. Doraghinejad, H. Nezamabadi-pour, Black hole: A new operator for gravitational search algorithm, Int. J. Comput. Intell. Syst. 7 (5) (2014) 809–826.

[17] S. Mirjalili, A. Lewis, Adaptive gbest-guided gravitational search algorithm, Neural Comput. Appl. 25 (7–8) (2014) 1569–1584.

[18] B. Shaw, V. Mukherjee, S. Ghoshal, A novel opposition-based gravitational search algorithm for combined economic and emission dispatch problems of power systems, Int. J. Electr. Power Energy Syst. 35 (1) (2012) 21–33.

[19] H. Chen, S. Li, Z. Tang, Hybrid gravitational search algorithm with random-key encoding scheme combined with simulated annealing, IJCSNS 11 (6) (2011) 208.

[20] S. Joshi, J.C. Bansal, Grey wolf gravitational search algorithm, in: Computational Intelligence (IWCI), International Workshop on, IEEE, 2016, pp. 224–231.

[21] X. Li, M. Yin, Z. Ma, Hybrid differential evolution and gravitation search algorithm for unconstrained optimization, Int. J. Phys. Sci. 6 (25) (2011) 5961–5981.

[22] S. Mirjalili, A.H. Gandomi, Chaotic gravitational constants for the gravitational search algorithm, Appl. Soft Comput. 53 (2017) 407–419.

[23] A. Zhang, G. Sun, J. Ren, X. Li, Z. Wang, X. Jia, A dynamic neighborhood learning-based gravitational search algorithm, IEEE Trans. Cybern. 48 (1) (2016) 436–447.

[24] J.C. Bansal, S.K. Joshi, A.K. Nagar, Fitness varying gravitational constant in gsa, Appl. Intell. 48 (10) (2018) 3446–3461, http://dx.doi.org/10.1007/s10489-018-1148-8.

[25] X. Han, X. Chang, A chaotic digital secure communication based on a modified gravitational search algorithm filter, Inform. Sci. 208 (2012) 14–27.

[26] S. Gao, C. Vairappan, Y. Wang, Q. Cao, Z. Tang, Gravitational search algorithm combined with chaos for unconstrained numerical optimization, Appl. Math. Comput. 231 (2014) 48–62.

[27] Z. Shang, Neighborhood crossover operator: a new operator in gravitational search algorithm, Int. J. Comput. Sci. Issues IJCSI 10 (5) (2013) 116.

[28] M. Khatibinia, S. Khosravi, A hybrid approach based on an improved gravitational search algorithm and orthogonal crossover for optimal shape design of concrete gravity dams, Appl. Soft Comput. 16 (2014) 223–233.

[29] U. Güvenc, F. Katırcıoğlu, Escape velocity: a new operator for gravitational search algorithm, Neural Comput. Appl. 31 (1) (2019) 27–42.

[30] S. Sarafrazi, H. Nezamabadi-pour, S.R. Seydnejad, A novel hybrid algorithm of gsa with kepler algorithm for numerical optimization, J. King Saud Univ.-Comput. Inf. Sci. 27 (3) (2015) 288–296.

[31] S. Mirjalili, S.Z.M. Hashim, A new hybrid gsa algorithm for function optimization, in: Computer and Information Application (ICCIA), 2010 International Conference on, IEEE, 2010, pp. 374–377.

[32] P. Li, H. Duan, Path planning of unmanned aerial vehicle based on improved gravitational search algorithm, Sci. China Technol. Sci. 55 (10) (2012) 2712–2719.

[33] B. Gu, F. Pan, Modified gravitational search algorithm with particle memory ability and its application, Int. J. Innovative Comput. Inf. Control 9 (11) (2013) 4531–4544.

[34] S. Mirjalili, S.Z.M. Hashim, H.M. Sardroudi, Training feedforward neural networks using hybrid particle swarm optimization and gravitational search algorithm, Appl. Math. Comput. 218 (22) (2012) 11125–11137.

[35] S. Mallick, S. Ghoshal, P. Acharjee, S. Thakur, Optimal static state estimation using improved particle swarm optimization and gravitational search algorithm, Int. J. Electr. Power Energy Syst. 52 (2013) 254–265.

[36] R.-E. Precup, R.-C. David, A.-I. Stinean, M.-B. Radac, E.M. Petriu, Adaptive hybrid particle swarm optimization-gravitational search algorithm for fuzzy controller tuning, in: 2014 IEEE International Symposium on Innovations in Intelligent Systems and Applications (INISTA) Proceedings, IEEE, 2014, pp. 14–20.

[37] S. Jayaprakasam, S.K.A. Rahim, C.Y. Leow, Psogsa-explore: A new hybrid metaheuristic approach for beampattern optimization in collaborative beamforming, Appl. Soft Comput. 30 (2015) 229–237.

[38] M. Shams, E. Rashedi, A. Hakimi, Clustered-gravitational search algorithm and its application in parameter optimization of a low noise amplifier, Appl. Math. Comput. 258 (2015) 436–453.

[39] M. Soleimanpour-moghadam, H. Nezamabadi-pour, An improved quantum behaved gravitational search algorithm, in: 20th Iranian Conference on Electrical Engineering, ICEE2012, IEEE, 2012, pp. 711–715.

[40] M. Soleimanpour-Moghadam, H. Nezamabadi-Pour, M.M. Farsangi, A quantum inspired gravitational search algorithm for numerical function optimization, Inform. Sci. 267 (2014) 83–100.

[41] F. Khajooei, E. Rashedi, A new version of gravitational search algorithm with negative mass, in: 2016 1st Conference on Swarm Intelligence and Evolutionary Computation, CSIEC, IEEE, 2016, pp. 1–5.

[42] H. Zandevakili, E. Rashedi, A. Mahani, Gravitational search algorithm with both attractive and repulsive forces, Soft Comput. 23 (3) (2019) 783–825.

[43] E. Rashedi, E. Rashedi, H. Nezamabadi-pour, A comprehensive survey on gravitational search algorithm, Swarm Evol. Comput. 41 (2018) 141–158.

[44] H. Nezamabadi-Pour, F. Barani, Gravitational search algorithm: concepts, variants, and operators, in: Handbook of Research on Modern Optimization Algorithms and Applications in Engineering and Economics, IGI Global, 2016, pp. 700–750.

[45] A. Gharaei, S.A. Hoseini Shekarabi, M. Karimi, Modelling and optimal lot-sizing of the replenishments in constrained, multi-product and bi-objective epq models with defective products: Generalised cross decomposition, Int. J. Syst. Sci.: Oper. Logist. (2019) 1–13.

[46] R. Sayyadi, A. Awasthi, A simulation-based optimisation approach for identifying key determinants for sustainable transportation planning, Int. J. Syst. Sci.: Oper. Logist. 5 (2) (2018) 161–174.

[47] C. Duan, C. Deng, A. Gharaei, J. Wu, B. Wang, Selective maintenance scheduling under stochastic maintenance quality with multiple maintenance actions, Int. J. Prod. Res. 56 (23) (2018) 7160–7178.

[48] Y.-C. Tsao, Design of a carbon-efficient supply-chain network under trade credits, Int. J. Syst. Sci.: Oper. Logist. 2 (3) (2015) 177–186.

[49] S.A. Hoseini Shekarabi, A. Gharaei, M. Karimi, Modelling and optimal lot-sizing of integrated multi-level multi-wholesaler supply chains under the shortage and limited warehouse space: generalised outer approximation, Int. J. Syst. Sci.: Oper. Logist. 6 (3) (2019) 237–257.

Please cite this article as: S.K. Joshi and J.C. Bansal, Parameter tuning for meta-heuristics, Knowledge-Based Systems (2019) 105094, https://doi.org/10.1016/j.knosys.2019.105094.

[50] A. Gharaei, M. Karimi, S.A.H. Shekarabi, An.integrated. multi product, An integrated multi-product multi-buyer supply chain under penalty, green, and quality control polices and a vendor managed inventory with consignment stock agreement: The outer approximation with equality relaxation and augmented penalty algorithm, Appl. Math. Model. 69 (2019) 223–254.
[51] M. Rabbani, S.A.A. Hosseini-Mokhallesun, A.H. Ordibazar, H. Farrokhi-Asl, A hybrid robust possibilistic approach for a sustainable supply chain location–allocation network design, Int. J. Syst. Sci.: Oper. Logist. (2018) 1–16.
[52] A. Gharaei, M. Karimi, S.A. Hoseini Shekarabi, Joint economic lot-sizing in multi-product multi-level integrated supply chains: generalized benders decomposition, Int. J. Syst. Sci.: Oper. Logist. (2019) 1–17.
[53] N. Kazemi, S.H. Abdul-Rashid, R.A.R. Ghazilla, E. Shekarian, S. Zanoni, Economic order quantity models for items with imperfect quality and emission considerations, Int. J. Syst. Sci.: Oper. Logist. 5 (2) (2018) 99–115.
[54] P. Loubière, A. Jourdan, P. Siarry, R. Chelouah, A sensitivity analysis method for driving the artificial bee colony algorithm's search process, Appl. Soft Comput. 41 (2016) 515–531.
[55] E. Rashedi, H. Nezamabadi-Pour, S. Saryazdi, Gsa: a gravitational search algorithm, Inform. Sci. 179 (13) (2009) 2232–2248.
[56] J. Liang, B. Qu, P. Suganthan, Q. Chen, Problem Definitions and Evaluation Criteria for the Cec 2015 Competition on Learning-Based Real-Parameter Single Objective Optimization, Technical Report201411A, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore.
[57] N. Hansen, The cma evolution strategy: a comparing review, in: Towards a New Evolutionary Computation, Springer, 2006, pp. 75–102.
[58] D. Simon, Biogeography-based optimization, IEEE Trans. Evol. Comput. 12 (6) (2008) 702–713.
[59] J.C. Bansal, P. Farswan, A novel disruption in biogeography-based optimization with application to optimal power flow problem, Appl. Intell. 46 (3) (2017) 590–615.
[60] R. Storn, K. Price, Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces, J. Glob. Optim. 11 (4) (1997) 341–359.
[61] J. Derrac, S. García, D. Molina, F. Herrera, A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms, Swarm Evol. Comput. 1 (1) (2011) 3–18.
[62] R Core Team, R: A Language and Environment for Statistical Computing, R Foundation for Statistical Computing, Vienna, Austria, 2018, https://www.R-project.org/.
[63] T. Pohlert, The Pairwise Multiple Comparison of Mean Ranks Package (PMCMR), R Package, 2014.
[64] R. Eberhart, J. Kennedy, A new optimizer using particle swarm theory, in: MHS'95 Proceedings of the Sixth International Symposium on Micro Machine and Human Science, IEEE, 1995, pp. 39–43.
[65] D. Simon, Biogeography-based optimization, IEEE Trans. Evol. Comput. 12 (6) (2008) 702–713.
[66] S. Mirjalili, S.M. Mirjalili, A. Lewis, Grey wolf optimizer, Adv. Eng. Softw. 69 (2014) 46–61.
[67] J.C. Bansal, H. Sharma, S.S. Jadon, M. Clerc, Spider monkey optimization algorithm for numerical optimization, Memet. Comput. 6 (1) (2014) 31–47.
[68] G.G. Tejani, N. Pholdee, S. Bureerat, D. Prayogo, Multiobjective adaptive symbiotic organisms search for truss optimization problems, Knowl.-Based Syst. 161 (2018) 398–414.
[69] G.G. Tejani, V.J. Savsani, V.K. Patel, S. Mirjalili, Truss optimization with natural frequency bounds using improved symbiotic organisms search, Knowl.-Based Syst. 143 (2018) 162–178.
[70] G.G. Tejani, V.J. Savsani, V.K. Patel, Adaptive symbiotic organisms search (sos) algorithm for structural design optimization, J. Comput. Des. Eng. 3 (3) (2016) 226–249.
[71] S. Kumar, G.G. Tejani, S. Mirjalili, Modified symbiotic organisms search for structural optimization, Eng. Comput. (2018) 1–28.
[72] G.G. Tejani, N. Pholdee, S. Bureerat, D. Prayogo, A.H. Gandomi, Structural optimization using multi-objective modified adaptive symbiotic organisms search, Expert Syst. Appl. 125 (2019) 425–441.
[73] G.G. Tejani, S. Kumar, A.H. Gandomi, Multi-objective heat transfer search algorithm for truss optimization, Eng. Comput. (2019) 1–22.
[74] G.G. Tejani, V.J. Savsani, V.K. Patel, S. Mirjalili, An improved heat transfer search algorithm for unconstrained optimization problems, J. Comput. Des. Eng. 6 (1) (2019) 13–32.
[75] G.G. Tejani, V.J. Savsani, S. Bureerat, V.K. Patel, P. Savsani, Topology optimization of truss subjected to static and dynamic constraints by integrating simulated annealing into passing vehicle search algorithms, Eng. Comput. 35 (2) (2019) 499–517.

Please cite this article as: S.K. Joshi and J.C. Bansal, Parameter tuning for meta-heuristics, Knowledge-Based Systems (2019) 105094, https://doi.org/10.1016/j.knosys.2019.105094.