Engineering Applications of Artificial Intelligence 26 (2013) 1301–1321

Contents lists available at SciVerse ScienceDirect

Engineering Applications of Artificial Intelligence

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

Performance evaluation of an improved harmony search algorithm for numerical optimization: Melody Search (MS)

S.M. Ashrafi*, A.B. Dariane

Department of Civil Engineering, KN Toosi University of Technology, No. 1346, Vali-Asr St., Mirdamad Intersection P.C. 1996715433, Tehran, Iran

ARTICLE INFO

Article history:
Received 5 May 2012
Received in revised form
9 August 2012
Accepted 21 August 2012
Available online 13 September 2012

Keywords:
Melody Search algorithm
Alternative improvisation procedure
Harmony search
Numerical optimization
Stochastic search methods

ABSTRACT

Melody Search (MS) Algorithm as an innovative improved version of Harmony Search optimization method, with a novel Alternative Improvisation Procedure (AIP) is presented in this paper. MS algorithm mimics performance processes of the group improvisation for finding the best succession of pitches within a melody. Utilizing different player memories and their interactive process, enhances the algorithm efficiency compared to the basic HS, while the possible range of variables can be varied going through the algorithm iterations. Moreover, applying the new improvisation scheme (AIP) makes algorithm more capable in optimizing shifted and rotated unimodal and multimodal problems than the basic MS.

In order to demonstrate the performance of the proposed algorithm, it is successfully applied to various benchmark optimization problems. Numerical results reveal that the proposed algorithm is capable of finding better solutions when compared with well-known HS, IHS, GHS, SGHS, NGHS and basic MS algorithms. The strength of the new meta-heuristic algorithm is that the superiority of the algorithm over other compared methods increases when the dimensionality of the problem or the entire feasible range of the solution space increases.

1. Introduction

Nature is inspiring researchers to develop effective and powerful optimization methods (Karaboga and Akay, 2009). In the past, many optimization methods were adopted to solve various real-world optimization problems which were constrained by the complexities of non-linearity in the model formulation and affected by the increase in the number of constraints and decision variables (Kumar and Reddy, 2006).

Nowadays evolutionary stochastic search methods are very popular for solving optimization problems in the research arena of computational intelligence (Karaboga and Basturk, 2007). The routine feature of meta-heuristic algorithms is the point that they often employ combinations of rules and randomness to imitate natural processes (Lee and Geem, 2005). Although the algorithms do not always ensure the global optimum solution, quite good results in a reasonable computation time are achieved. This is why many researchers have been eager to develop newer techniques and improve existing methods over the past years (Kumar and Reddy, 2006).

Many meta-heuristic algorithms, such as genetic algorithm, particle swarm optimization, tabu search, ant colony optimization, bees' algorithm, artificial immune system and simulated annealing have been extensively employed for various science and engineering problems. One of the major disadvantages of these algorithms is the fact that most of the meta-heuristic algorithms are successful for solving some certain class of problems. Furthermore, in some cases, although the algorithms show superior performance on low dimensional problems, they cannot preserve their superior performance on high dimensional cases (Karaboga and Akay, 2009).

The concepts of Genetic Algorithms (GAs) were originally explained by Holland (1975) and further developed by Goldberg (1989). GA-based algorithms are global search methods found on concepts from natural genetics and the Darwinian survival-of-the-fittest code. During the past two decades, GA has been studied extensively by many researchers to solve difficult and complicated real-world and engineering optimization problems. GAs are generally capable in finding good solutions in reasonable amounts of time. However, applying to harder and bigger problems increases the time required to find adequate solutions. Several papers, book chapters, special issues and books have surveyed GAs literature (e.g. Cheng et al., 1999; Coello et al., 2007; Nicklow et al., 2010).

Tabu Search (TS) as a gradient-descent search method with memory was originally suggested by Glover (1977). Details about tabu search can also be found in Glover (1989, 1990), Hertz et al.

* Corresponding author. Tel.: +98 21 88779623; fax: +98 21 88779476.
E-mail addresses: mashrafi@mail.kntu.ac.ir (S.M. Ashrafi), borhani@kntu.ac.ir (A.B. Dariane).

0952-1976/$ - see front matter
http://dx.doi.org/10.1016/j.engappai.2012.08.005

(1997) and Glover and Laguna (1997). In the TS method the solution space is explored by moving from a solution to the best solution in a subset of its neighborhood trough different iterations while, to avoid cycling, recently explored solutions are temporarily declared tabu or forbidden. Many different models based on TS have been developed to improve the efficiency of the algorithm and applied to solve optimization problems (e.g. Griinert, 2002; Glover and Kochenberger, 2003; Gendreau, 2002).

The Simulated Annealing (SA) algorithm has been modeled on the annealing of solids in nature. The SA initially, proposed by Kirkpatrick et al. (1983) as a method/tool for solving single objective combinatorial problems. However, recently there are many reports of SA applications in different fields of engineering problems containing continuous and discrete spaces (Suman and Kumar, 2006).

An important approach which is based on ants behavior, called Ant Colony Optimization (ACO), proposed by Dorigo et al. (1991). The approach has been studied by many researchers, while several new variants have been proposed and applied to solve optimization problems in different areas. Numerous publications related with the applications of ACO models have been presented to the literature (Blum, 2005; Dorigo and Blum, 2005; Pedemonte et al., 2011; Chandra Mohan and Baskaran, 2012).

Inspired by the social behavior of the birds flocking or fish schooling, Kennedy and Eberhat (1995) introduced an evolutionary computational method named Particle Swarm Optimization (PSO) algorithm. PSO, similar to the other evolutionary computational algorithms like GA, is a population-based interactive method. As the most important swarm intelligence paradigms (Kennedy et al., 2001), PSO is popular and useful to solve various kinds of real-world optimization problems (Eberhart and Shi, 2001); however, a number of PSO variants have been developed to overcome some weaknesses which have restricted wider applications of the standard-PSO. Several survey papers regarding the PSO variants applications have been presented (Eberhart and Shi, 2004; Reyes-Sierra and Coello, 2006; AlRashidi and El-Hawary, 2009).

Artificial Bee Colony (ABC) presented by Karaboga (2005) inspired by the intelligent behavior of honey bees for seeking a quality food source in nature. The algorithm was originally proposed for solving numerical problems; however, the success of the algorithm as a single-objective optimizer has motivated researchers to extend its application to other study areas (Karaboga et al., 2012).

Recently, a new meta-heuristic technique, namely Harmony Search (HS) algorithm has been proposed by Geem et al. (2001), which simulates the improvisation process of musicians. In HS algorithm, solution vectors correspond to the harmony in music and the local and global search schemes correspond to the musician's improvisations (Lee and Geem, 2005). HS is a stochastic search technique without the need of derivative information, and with reduced memory requirement. In comparison with other meta-heuristic methods, HS is computationally effective and easy to implement for solving various kinds of engineering optimization problems (Mahdavi et al., 2007; Omran and Mahdavi, 2008).

The algorithm is capable for identifying the high performance regions of solution space in a reasonable run time; however, it is not successful in performing local search in numerical optimization applications (Mahdavi et al., 2007). Moreover, in the case of problems with large space of decision variables, the likelihood of obtaining the global optimum is considerably reduced. Karaboga and Akay (2009) presented a comparative investigation of the performance of basic Harmony Search, Bees algorithms, and Artificial Bee Colony algorithm. It was concluded that HS algorithm is less efficient than the ABC in solving different optimization problems.

The initial development of HS Algorithm was conducted by Geem (2000), during his Ph.D. studies. Design of water distribution networks was the main aim, while the study covered benchmark optimization, parameter estimation, and the traveling salesman problem (TSP) (Ingram and Zhang, 2009). Since then, a variety of HS models have been adopted to diverse field of problems, such as structural design, Sudoku puzzles, musical composition, medical imaging, heat exchanger design, course timetabling, web page clustering, robotics, water network design, dam scheduling, vehicle routing, energy system dispatch, cell phone network, satellite heat pipe design, and medical physics.

There are several attempts to improve the performance of basic HS algorithm for enhancing solution accuracy and convergence rate, like Improved Harmony Search algorithm (IHS) (Mahdavi et al., 2007), global best Harmony Search algorithm (GHS) (Omran and Mahdavi, 2008), self-adaptive global best harmony search algorithm (SGHS) (Pan et al., 2010b), novel global harmony search (NGHS) (Zou et al., 2010). All improvements are categorized into two classes by Alia and Mandava (2011); the first one is improvement of HS in terms of parameter setting, and the second one is improvement in terms of hybridizing HS components with other meta-heuristic algorithms. Ingram and Zhang (2009) have classified various modifications of HS in seven categories and briefly explained each category.

Geem et al. (2005), proposed a multiple pitch adjusting rate (*PAR*) strategy in solving the so called Generalized Oriented Problem. Using three *PAR*'s for moving rates to the nearest, second nearest, and third nearest cities was proposed in their study. Mahdavi et al. (2007) developed an Improved HS algorithm, denoted as IHS, by introducing a method to dynamically adjust the algorithm computational parameters (i.e. *PAR* and *bw*). Their algorithm was applied to solve four engineering and four mathematical optimization problems. According to their comparative investigation between obtained results and those from other techniques in the literature, they remarked that the algorithm can find better solutions. Omran and Mahdavi (2008) presented a Global-best Harmony Search algorithm, (GHS) GHS algorithm, by borrowing the concepts from swarm intelligence. They studied the sensitivity of the HS parameters and compared the performance of HS, IHS and GHS on ten continuous optimization functions and six integer programming problems. Both IHS and GHS algorithms could find better solutions, compared with the basic HS algorithm. Coelho and Mariani (2009), proposed an improved harmony search (IHS) algorithm based on exponential distribution, for solving Economic Dispatch Problems, which updated the *PAR* parameter dynamically. The application of HS and IHS for solving thirteen thermal units of generation with the valve-point effects was reported there. Numerical results show that IHS the algorithm converged to more reasonable results compared with basic HS algorithm. Another new improvement to HS, named (DHS) which was inspired by mutation operator of Differential Evolution (DE) is proposed by Chakraborty et al. (2009). They replaced the pitch adjustment operation in basic HS with a mutation strategy borrowed from DE algorithm. Inspired by the local version of the particle swarm optimization algorithm, Pan et al. (2010a) proposed the local-best harmony search algorithm with dynamic subpopulations (DLHS) for solving the bound-constrained continuous optimization problems. In DLHS method the Harmony Memory (*HM*) is divided into many sub-harmony memories. New harmonies are independently generated in these small-sized sub-harmony memories, which are regrouped frequently by using a regrouping schedule. A recent variant of HS algorithm was proposed by Wang, Huang (2010). The model totally replaces bandwidth parameter (*bw*) parameter with a new concept based on using the maximal and minimal values in *HM*. While the search process is going on, *PAR* values are dynamically adapted using the modification proposed by Mahdavi et al. (2007). Kattan et al. (2010), applied HS for feed-forward artificial neural networks (ANN) training. They proposed a new stopping criterion that is based on the best-to-worst (BtW) harmony ratio in the current harmony memory, and applied the ratio within the existing improved version of HS (proposed by Mahdavi et al. 2007). They remarked that the modification is more proper for ANN training since parameters and termination depend on the quality of the obtained solutions (Alia and Mandava, 2011). A self-adaptive global best harmony search algorithm denoted as (SGHS), was proposed by Pan et al. (2010a) for solving continuous optimization problems. In SGHS, the algorithm parameters, Harmony Memory Considering Rate (*HMCR*) and *PAR*, are self-adaptive by a learning mechanism and *bw* bandwidth parameter (*bw*), is dynamically decreased with increasing the generation number. Numerical experiments revealed that SGHS algorithm is more effective in finding better solutions than HS, IHS and GHS algorithms. Zou et al. (2010) developed a novel global harmony search method, called NGHS, for solving unconstrained problems. A novel location updating strategy is designed which makes the algorithm easier to converge. The experimental results showed better performance for NGHS algorithm for solving most of unconstrained problems compared with other harmony search methods (i.e. HS, IHS, and SGHS). For enhancing the performance of HS, Al-Betar et al. (2010) proposed eight procedures instead of using one *PAR* value, to solve the Course Timetabling Problem, which were controlled by their certain *PAR* value ranges.

There are several hybrid models of HS with other meta-heuristic methods in the literature. Alia and Mandava (2011) categorized this hybridization into two classes. The first class consists of models which are the integration of some components of the other meta-heuristic algorithms within HS, while the second class consists of methods which integrate some HS components within other meta-heuristic algorithms.

More recently, inspired by basic concepts applied in HS algorithm, an innovative improved version of HS algorithm is presented by authors (Ashrafi and Dariane 2011). The algorithm named Melody Search (MS) is designed in accordance with the concept of melody instead of harmony. This algorithm is based on musical performance processes and interactive relations occurred between members of a group of musicians attempting to find better and better series of pitches within a melodic line. In such a group, the music players can improvise the melody differently and lead each other to achieve the best subsequence of pitches. Furthermore, a novel improvisation scheme is introduced and applied in this study through which the efficiency of the algorithm for solving shifted and rotated optimization problems would be increased.

In order to demonstrate the performance of the proposed algorithm, MS with the novel improvisation scheme is applied to various benchmark problems and the results are compared with those of the basic HS, IHS, GHS, SGHS, NGHS and basic MS algorithms.

The remaining of this paper is organized as follows: basic concepts of HS algorithm are explained in Section 2. Some variants of HS algorithms including IHS, GHS, SGHS and NGHS are briefly described in Sections 3 and 4. describes the proposed MS algorithm, and the basic differences between HS and MS methods. Some experimental studies regarding numerical benchmark problems, along with their analysis and discussions are summarized in Section 5. Finally Section 6 provides a brief summary and conclusion.

2. Harmony search (HS) algorithm

Harmony search was developed by Geem et al. (2001), based on mimicking music improvisation process where music players improvise the pitches of their instruments to obtain better harmony.

Main steps of the algorithm are given below:

1. initialize the problem and algorithm parameters,
2. initialize the harmony memory with the random solution vectors,
3. improvise a new harmony vector,
4. update the harmony memory and
5. check stopping criterion and repeat steps 3–4.

In Step 1, initializing optimization problems, the algorithm parameters are specified; including the harmony memory size (*HMS*) (i.e. the number of solution vectors in harmony memory), harmony memory considering rate (*HMCR*) where $HMCR \in [0,1]$, pitch adjusting rate (*PAR*) where $PAR \in [0,1]$, bandwidth distance (*bw*) for problems with continuous space of variables or neighboring index (m) for problems with discrete space of variables, and Maximum number of improvisation (termination criterion, *MaxImp*) (Lee and Geem, 2005).

In Step 2, the harmony memory (*HM*) matrix is filled with randomly generated harmony vectors and relevant objective function values as can be seen in Eqs. (1) and (2). As a harmony vector, $X_i=\{x_i(1), x_i(2),\ldots, x_i(D), Fitness_i\}$ represents a solution of the optimization problem (D is the number of decision variables);

$$x_i(j) = LB(j) + rand() \times (UB(j) - LB(j)) \quad \text{for } j = 1,\ldots,D \quad \text{and} \quad i = 1,\ldots,HMS \tag{1}$$

$$HM = \begin{bmatrix} X_1 \\ X_2 \\ \ldots \\ X_{HMS} \end{bmatrix} \tag{2}$$

where, $rand(\ )$ is a random number between 0 and 1, and $[LB(j), UB(j)]$ is the entire feasible range of jth variable.

In Step 3, a new harmony vector is produced and evaluated against the worst harmony in the memory. Improvising a new harmony is conducted considering the following three rules:

1. memory consideration,
2. pitch adjustment
3. random selection.

In the memory consideration, values of the new harmony vector are chosen randomly from the existing vectors in the *HM* with a probability of *HMCR*. In the randomization, decision variable values are randomly chosen according to their feasible range with a probability of $(1-HMCR)$ (Lee and Geem, 2005). The operators are illustrated in Eq. (3).

$$x_{new}(j) \leftarrow \begin{cases} x_{new}(j) \in (x_1(j),\ x_2(j),\ldots,x_{HMS}(j)) & \text{with probability} \quad HMCR \\ x_{new}(j) \in [LB(j), UB(j)] & \text{with probability} \quad 1-HMCR \end{cases} \tag{3}$$

Pitches can be adjusted with a probability of $HMCR \times PAR$ in the pitch adjusting operation as follows;

$$x_{new} = x_{old} + bw \times \varepsilon \tag{4}$$

where, x_{old} is the existing pitch, chosen from *HM*, x_{new} is the new pitch after the pitch adjusting operation, *bw* is the bandwidth distance for continuous problem, and ε is a random number in the range of $[-1,1]$. Pitch adjusting and randomization increase the diversity of the solution in HS algorithm (Alia and Mandava, 2011).

In step 4, if the objective function value of the new harmony is better than the worst one in the Harmony memory, the new harmony is included in the *HM* and the worst harmony is set aside.

3. IHS, GHS, SGHS and NGHS Algorithms

In order to perform a good comparison between proposed algorithm and HS variants, algorithms including IHS, GHS, SGHS and NGHS are briefly described in this section.

3.1. Improved harmony search (IHS)

Mahdavi et al. (2007) introduced an improved variant of HS denoted as IHS. In this algorithm *PAR* and *bw* parameters, are updated dynamically with increasing generation number, as described in Eqs. (5), (6) and (7);

$$PAR(gn) = PAR_{\min} + \frac{(PAR_{\max} - PAR_{\min})}{Max\,\mathrm{Im}\,p} \times gn \tag{5}$$

$$bw(gn) = bw_{max}\exp(c.gn), \tag{6}$$

$$c = \frac{Ln(bw_{\min}/bw_{\max})}{Max\,\mathrm{Im}\,p} \tag{7}$$

where *MaxImp* is the maximum number of improvisation, PAR_{min} and PAR_{max} are the minimum and maximum values of pitch adjusting rate, $PAR(gn)$ is the calculated pitch adjusting rate for the *gn*th generation, bw_{min} and bw_{max} are the minimum and maximum values of distance bandwidth and $bw(gn)$ is the obtained distance bandwidth for the *gn*th generation. Other steps of IHS are the same as the basic HS algorithm.

3.2. Global-best harmony search (GHS)

GHS method, inspired by the particle swarm optimization was proposed by Omran and Mahdavi (2008). Pitch adjustment operator of HS algorithm is modified in GHS, while the new improvised harmony can consider the best harmony in the memory. Additionally, *PAR* parameter is determined with a dynamic updating procedure. Except the Improvisation step all other steps of the GHS are the same as the basic HS algorithm. The modified improvisation step is as follows:

For *each* $i \in [1, \ldots, D]$ **do**
 If $rand(\ ) \le HMCR$ **then**
  $X^i_{new} = X^i_j$ *where* $j \in U(1, \ldots, HMS)$,
  If $rand1(\ ) \le PAR$ **then**
   $X^i_{new} = X^k_{Best}$ *where* $k \in [1,\ldots,D]$ *and Best indicates the best harmony in the memory*
  End if
 Else
  $X^i_{new} = LB_i + rand2(\ ) \times (UB_i \_ LB_i)$
  End if
Done

3.3. Self-adaptive global-best harmony search (SGHS)

Inspired by the GHS algorithm, Pan et al. (2010b) proposed (SGHS) algorithm, which adopts a new improvisation scheme and an adaptive parameter tuning method. According to the applied pitch adjustment rule in the new improvisation scheme, X^i_{new} is assigned to the corresponding decision variable (X^i_{Best}) of the best harmony. In order to avoid getting trapped into local optimum solution, a modified memory consideration operator is used in the algorithm.

Furthermore, *HMCR* and *PAR* parameters are dynamically updated to a suitable range by recording their historical values corresponding to generated harmonies entering the *HM*. The value of *bw* parameter is decreased with increasing generations by a dynamic method.

The computational procedure of the SGHS improvisation can be summarized as follows:

For *each* $i \in [1, \ldots, D]$ **do**
 If $rand(\ ) \le HMCR$ **then**
  $X^i_{new} = X^i_j \pm rand1(\ ) \times bw$ *where* $j \in U(1, \ldots, HMS)$,
  If $rand2(\ ) \le PAR$ **then**
   $X^i_{new} = X^i_{Best}$ *where Best indicates the best harmony in the memory*
   End if
 Else
  $X^i_{new} = LB_i + rand3(\ ) \times (UB_i - LB_i)$
 End if
Done

3.4. Novel global harmony search (NGHS)

Inspired by the swarm intelligence, Zou et al. (2010) proposed a different variation of HS named, novel global harmony search. Harmony memory consideration and pitch adjustment are excluded from NGHS and genetic mutation with a low probability is included in the NGHS. Furthermore a position updating technique is proposed and applied in NGHS. The improvisation procedure of the NGHS can be summarized as follows:

For *each* $i \in [1, \ldots, D]$ **do**
 $X_R = 2 \times X^i_{best} - X^i_{wors}$
  If $X^i > UB_i$ **then**
   $X^i = UB_i$
 Else if $X^i < LB_i$ **then**
   $X^i = LB_i$
 End if
 $X^i_{new} = X^i_{worst} + rand1(\ ) \times (X_R - X^i_{worst})$ *Position Updating,*
 If $rand2(\ ) \le P_m$ **then**
   $X^i_{new} = LB_i + rand3(\ ) \times (UB_i – LB_i)$ *Genetic Mutation*
 End if
Done

where, "best" and "worst" indicate the best harmony and the worst harmony in the harmony memory, respectively. Note that in the NGHS algorithm, the worst harmony (X_{worst}) of *HM* is replaced with the new harmony (X_{new}), even if X_{new} is worse than X_{worst}.

4. Proposed algorithm: melody search (MS)

In music, harmony is the use of simultaneous pitches or chords and is often referred to the vertical aspect of music space; while, melodic line refers to the horizontal aspect (Sturman, 1983), as illustrated in Fig. 1. Melody might be defined as a linear succession of individual pitches, one followed by another one in order,

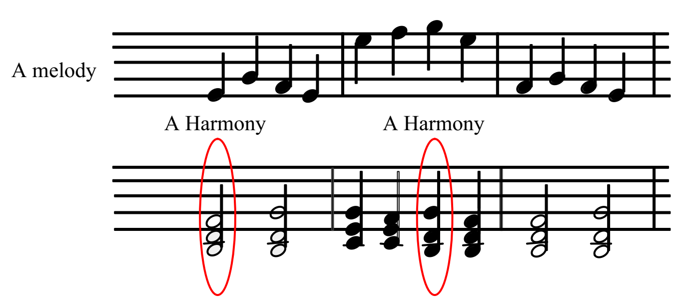

Fig. 1. Melody and Harmony.

where the composite order of applied pitches forms a single entity. Melodic pitches are not randomly ordered; however, they are subject to basic principles of musical design (Schoenberg, 1982).

In order to enhance the efficiency of HS algorithm introduced by Geem et al. (2001), a novel improved optimization technique, called Melody Search (MS) was proposed by authors (Ashrafi and Dariane, 2011). MS algorithm mimics the musical performance processes and interactive relations occurred between members of a musicians group; while, they are looking for the best series of pitches within a melodic line. In such a group, the existing several music players—with different tastes, ideas and experiences—can lead in achieving the best subsequence of pitches faster.

Fig. 2 demonstrates the analogy between melodies which are played by music players, and components of the optimization problem. In Melody Search algorithm, each melodic pitch is replaced by a decision variable of the real problem and each melody is replaced by a solution of the optimization problem. Each music player sounds a series of subsequent pitches within their possible ranges, if the succession of pitches makes a good melody, that experience is stored in the player memory.

Although the new algorithm adopts the basic concepts of HS, the structure is quite different. Unlike HS that uses a single harmony memory, MS algorithm employs several memories named Player Memory (*PM*). Memories interact to each other like musicians performance in a group. Furthermore, in the MS

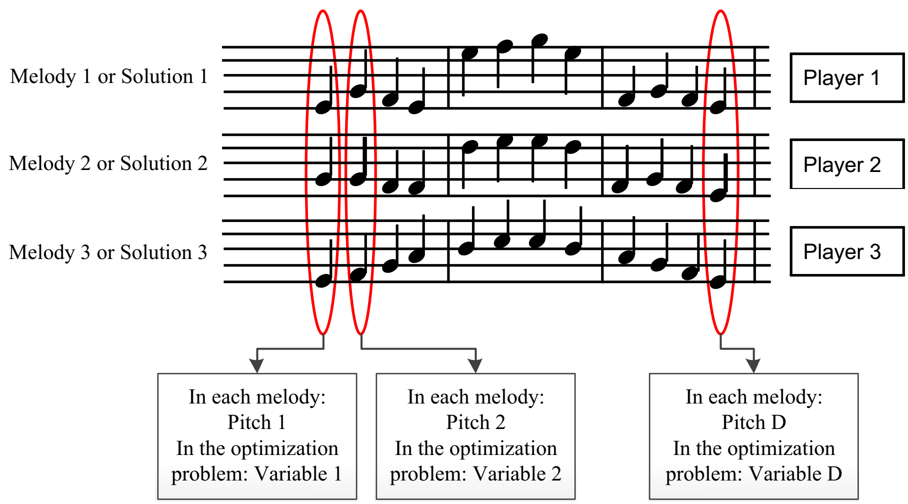

Fig. 2. Analogy between melodies and components of the optimization problem.

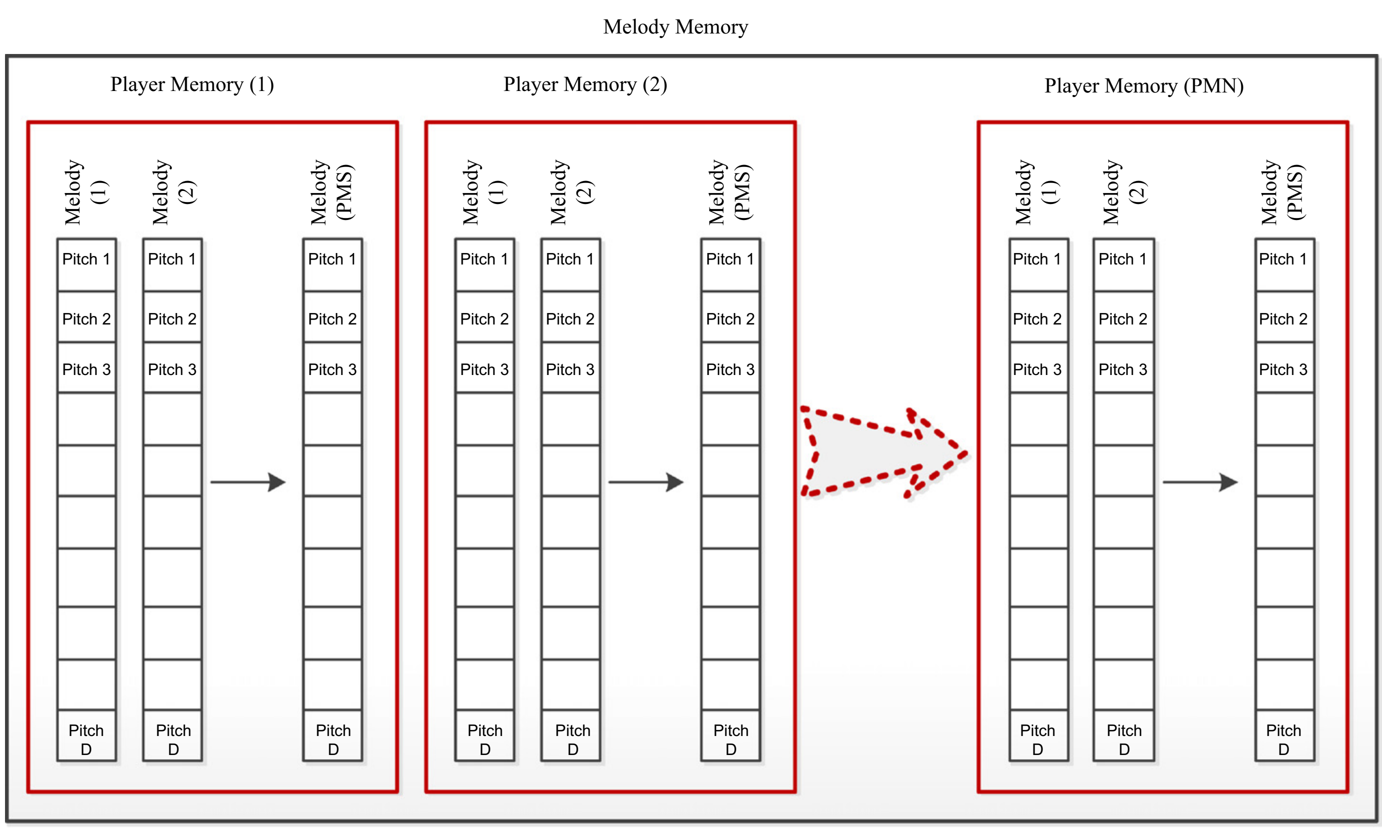

Fig. 3. Structure of Melody Memory in MS algorithm.

algorithm the possible range of each pitch for random selection can be changed going through different iterations. Fig. 3 shows the structure of Melody Memory (*MM*), identified in MS algorithm.

Main body of the algorithm consists of two different computational phases. In the initial phase, each music player can improvise his/her melody without the influence of others; while in the second phase, the algorithm acts as a group performance. Existing different melodies in the group of musicians can lead players to select better random pitches and the possibility of composing a better melody is increased in the next step.

Main steps of MS algorithm are as follows;

Step 1. Initializing the optimization problem and adopting algorithm parameters
Step 2. Initial Phase;
- 2.1. Initialize player memories (*PM*)
- 2.2. Improvise a new melody from each *PM*
- 2.3. Update each *PM*
- 2.4. Repeat sub-steps 2.2 and 2.3 until the criterion for stopping the initial phase is satisfied

Step 3. Second phase;
- 3.1. Improvise a new melody from each *PM* according to the possible range of pitches
- 3.2. Update each *PM*
- 3.3. Determine the possible ranges of pitches for next improvisation (just for randomization)
- 3.4. Repeat sub-steps 3.1, 3.2 and 3.3 until the criterion for stopping the algorithm is satisfied (i.e. maximum number of iterations: *NI*)

The three described steps forms a framework for solving problems through MS algorithm. There are seven major parameters defined in MS algorithm, including number of player memories (*PMN*), player memory size (*PMS*), maximum number of iterations (*NI*), maximum number of iterations for the initial phase (*NII*), distance bandwidth (*bw*), player memory considering rate (*PMCR*) which is identical to *HMCR* in HS algorithm and pitch adjusting rate. The mentioned parameters are adopted in the first step.

Searching for the best arrangement of pitches in the melody is carried out separately by any of the music players in the second step (the initial phase). In the primary part of the second step the player memories are initialized. In MS algorithm, melody memory (*MM*) consists of several player memories. The player memories matrixes are generated with random initial melodies; as described in Eqs. (8)–(10).

$$MM = [PM_1, PM_2, \ldots, PM_{PMN}] \tag{8}$$

$$PM_i = \begin{bmatrix} x_{i,1}^1 & x_{i,1}^2 & \ldots & x_{i,1}^D & Fit_i^1 \\ x_{i,2}^1 & x_{i,2}^2 & \ldots & x_{i,2}^D & Fit_i^2 \\ \ldots & & & & \\ x_{i,PMS}^1 & x_{i,PMS}^2 & \ldots & x_{i,PMS}^D & Fit_i^{PMS} \end{bmatrix} \tag{9}$$

$$x_{i,j}^k = LB_k + U(0,1)\times(UB_k - LB_k) \quad \text{for } i = 1,\ldots,PMN, \quad j = 1,\ldots, PMS,\ k = 1,\ldots,D \tag{10}$$

where, D is the number of pitches of melodic line or decision variables of optimization problem, and $[LB(k), UB(k)]$ is the entire feasible range of the kth pitch or variable which is not changed in the initial phase (Step 2) but it can be changed in the second phase (Step 3).

In the secondary part of Step 2, a new melody is improvised for each *PM*. In the proposed algorithm, the new melodic line or vector of decision variables from each *PM*, $X_{i,new} = (x_{i,new}^1, x_{i,new}^2, \ldots, x_{i,new}^D)$, are generated using a novel Alternative Improvisation Procedure (AIP) based on the main concepts of harmony improvisation.

During sub-Step 2.3, the player memories are updated. If the objective function value of the new melody in each *PM* is better than the worst objective function value, the new melody is included in the specified *PM* and the existing worst melody is excluded. The latter two steps are repeated while the iteration number is smaller than the maximum iteration number of initial phase.

In the third step (the second phase), the best arrangement of pitches in the melody is searched through an interactive process

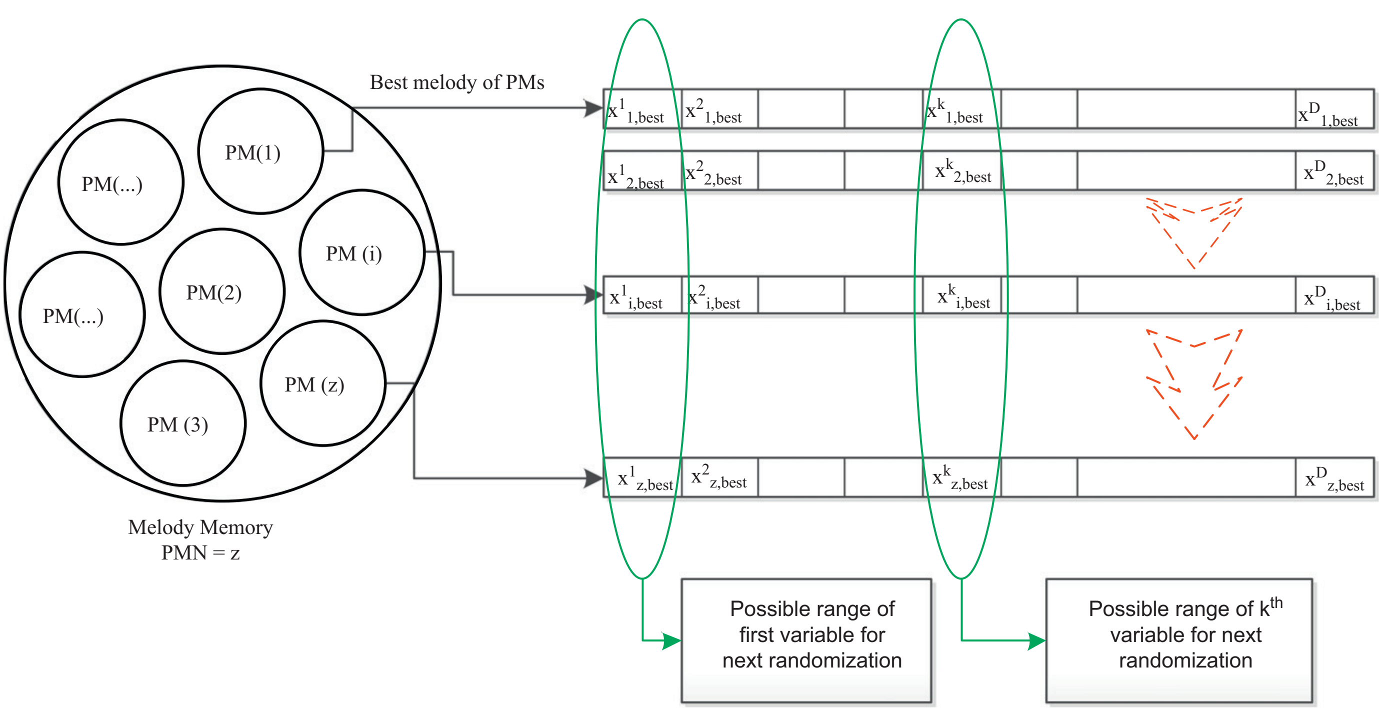

Fig. 4. Determining the possible range of variables.

between the music players, while the possible range of pitches are being updated. A new melody is improvised by AIP procedure in sub-Step 3.1 from each *PM* according to the possible range of pitches which can be varied here through different iterations. Again, the player memories are updated, as done during sub-Step 2.3. Sub-step 3.3 is a main part of MS algorithm and there is a major difference between MS and HS here. In the second phase, the best melody variables of each *PM* are stored and the new possible variable ranges can be calculated for the next randomization, as described below;

For *each* $k \in [1, \ldots, D]$ *do*
$LB_k = \min\,(x^k_{i,\ best},\ i = 1, \ldots, PMN)$
$UB_k = \max\,(x^k_{i,\ best},\ i = 1, \ldots, PMN)$
Done

The *best*-subscript stands for the best melody in any certain player memory. Fig. 4 graphically shows the contributing processes in Sub-step 3.3. The latter two sub-steps are repeated while the iteration number is smaller than the maximum number of iterations (*NI*). A flowchart for Melody Search algorithm is presented in Fig. 5.

4.1. The novel alternative improvisation procedure (AIP)

A novel improvisation scheme for generating new melodies is proposed in this section. In MS algorithm, melody improvisation is performed for each *PM* separately. In the memory consideration process, two different rules are alternatively applied while each rule forms a linear combination of a chosen variable from current *PM* and a proportion of *bw*. In the first rule, value of each new variable ($x^k_{i,new}$ for kth variable in ith player memory) is generated using a corresponding decision variable from *PM* (i.e. $x^k_{i,l}$, $l \in U(1, 2, \ldots, PMS)$) while a random variable amongst any one of the decision variables of *PM* (i.e. $x^h_{i,l}$, $l \in U(1,2,\ldots,PMS)$, $h \in U(1,2,\ldots,D)$), is applied in the second one. Although using the

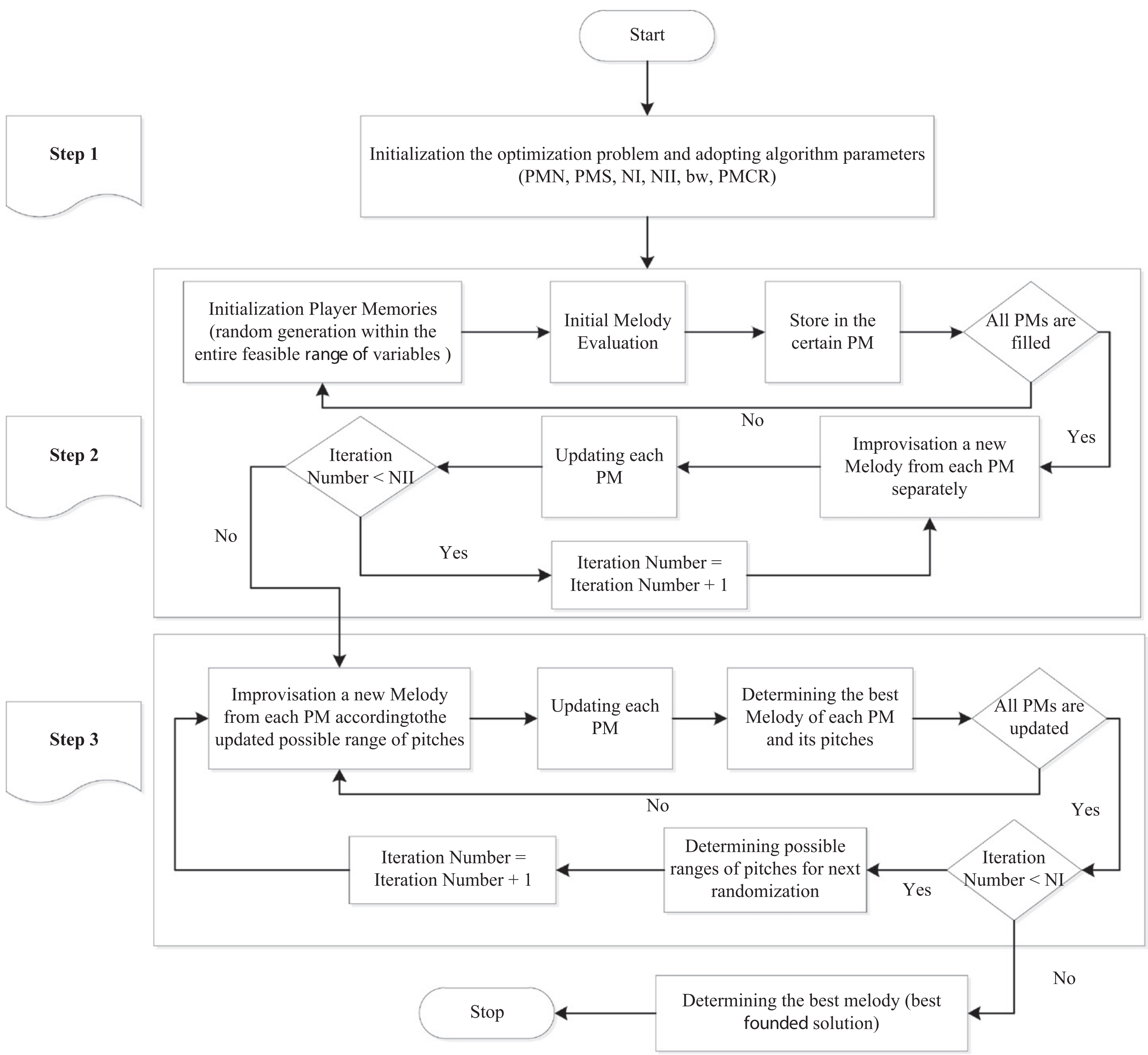

Fig. 5. Flowchart of MS algorithm.

first rule increases the algorithm convergence, second rule enhances the diversity of generated solutions. Furthermore, in order to avoid getting trapped in a local optimal solution a proportion of *bw* is adopted in the rules.

Inspired by the modified pitch adjustment rule proposed by Pan et al. (2010b) for SGHS algorithm, variables of the best melody in the current player memory are applied exactly in the pitch adjustment process. Finally, random generation is performed in the possible ranges of variables which can be varied through different iterations.

The function of proposed Alternative Improvisation Procedure for MS algorithm can be summarized as follows:

For *each* $i \in [1, \ldots, PMN]$ **do**
  For *each* $k \in [1, \ldots, D]$ **do**
    If $rand1(\ ) \le PMCR$ **then** *(memory consideration)*
      If *Iteration Number is odd* **then**
        $X^k_{i,\ new} = X^k_{i,\ L} \pm rand2(\ ) \times bw(k)$ *where* $L \in U(1, \ldots, PMS)$
      Else
        $X^k_{i,\ new} = X^h_{i,\ L} \pm rand3(\times bw(k)$ *where* $L \in U(1, \ldots, PMS)$
        and $h \in U(1, \ldots, D)$
      End if
        If $rand4(\ ) \le PAR_t$ **then** *pitch adjustment)*
        $X^k_{i,\ new} = X^k_{i,\ best}$
      End if
    Else *(randomization)*
        $X^k_{i,\ new} = LB_k + rand5(\ ) \times (UB_k - LB_k)$
      End if
  Done
Done

Here, $randn(\ )$ is a uniform random number between 0 and 1, $x^k_{i,best}$ is the kth variable of the best melody in the ith *PM*. Parameter PAR_t is the pitch adjusting rate of the tth iteration, which is determined by the following equation:

$$PAR_t = PAR_{min} + \frac{PAR_{max} - PAR_{min}}{NI} \times t \tag{11}$$

where, PAR_{min} and PAR_{max} are the minimum and maximum adjusting rates, respectively. The described method of dynamically updating *PAR* was proposed by Mahdavi et al. (2007) for IHS algorithm. $bw(k)$ is the specified distance bandwidth for kth variable which is dynamically determined as follows:

$$bw(k) = \frac{(UB_k - LB_k)}{200} \tag{12}$$

Note that, in the proposed MS algorithm the distance bandwidth of each variable can be varied through different iterations while the upper and lower bounds of variables are varied.

5. Experiments

Eighteen classical benchmark functions are considered to evaluate the performance of proposed algorithm. Results of the MS algorithm with new improvisation scheme (AIP_MS) are compared with the results obtained from basic HS, IHS, GHS, SGHS, NGHS and basic MS algorithms.

5.1. Test functions

The first function is Sphere function, defined as

$$f_1(\overrightarrow{x}) = \sum_{i=1}^{D} x_i^2 \tag{13}$$

where the global minimum of the function is $f_1^{Opt}(X^*) = 0.0$ and $X^* = (0.0, 0.0, \ldots, 0.0)$. The initial range for each variable is $[-100, 100]$.

The second function is Griewank function, defined as

$$f_2(\overrightarrow{x}) = \frac{1}{4000}\sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1 \tag{14}$$

where the global minimum of the function is $f_2^{Opt}(X^*) = 0.0$ and $X^* = (0.0, 0.0, \ldots, 0.0)$. The initial range for each variable is $[-600, 600]$.

The third function is Rastrigin function, defined as

$$f_3(\overrightarrow{x}) = \sum_{i=1}^{D} (x_i^2 - 10\cos(2\pi x_i) + 10) \tag{15}$$

where the global minimum of the function is $f_3^{Opt}(X^*) = 0.0$ and $X^* = (0.0, 0.0, \ldots, 0.0)$. The initial range for each variable is $[-5.12, 5.12]$.

The fourth function is Rosenbrock function, defined as

$$f_4(\overrightarrow{x}) = \sum_{i=1}^{D-1} 100(x_i^2 - x_{i+1})^2 + (1 - x_i)^2 \tag{16}$$

where, the global minimum of this function is $f_4^{Opt}(X^*) = 0.0$ and $X^* = (1, 1, \ldots, 1)$. The initial range for each variable is $[-30, 30]$.

The fifth function is Ackley function, defined as

$$f_5(\overrightarrow{x}) = 20 + e - 20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi\, x_i)\right) \tag{17}$$

where, the global minimum of this function is $f_5^{Opt}(X^*) = 0.0$ and $X^* = (0.0, 0.0, \ldots, 0.0)$. The initial range for each variable is $[-32, 32]$.

The sixth function is Schwefel function 2.22, defined as

$$f_6(\overrightarrow{x}) = \sum_{i=1}^{D} |x_i| + \prod_{i=1}^{D} x_i| \tag{18}$$

where the global minimum of the function is $f_6^{Opt}(X^*) = 0.0$ and $X^* = (0.0, 0.0, \ldots, 0.0)$. The initial range for each variable is $[-10, 10]$.

The seventh function is Schaffer function f6, defined as

$$f_7(\overrightarrow{x}) = 0.5 + \frac{\sin^2\left(\sqrt{\sum_{i=1}^{D} x_i^2}\right) - 0.5}{\left(1 + 0.001\sum_{i=1}^{D} x_i^2\right)^2} \tag{19}$$

where the global minimum of the function is $f_7^{Opt}(X^*) = 0.0$ and $X^* = (0.0, 0.0, \ldots, 0.0)$. The initial range for each variable is $[-100, 100]$.

The eighth function is Shifted Sphere function, defined as (Qin et al., 2009):

$$f_8(\overrightarrow{x}) = \sum_{i=1}^{D} z_i^2 \tag{20}$$

where $z = \mathrm{x} - \mathrm{o}$ and $O = \{o(1), o(2), \ldots, o(D)\}$ is the shifted global optimum. The global minimum of this function is $f_8^{Opt}(X^*) = 0.0$ and $X^* = O$. The initial range for each variable is $[-100, 100]$.

The ninth function is Shifted Schwefel function 1.2, defined as

$$f_9(\overrightarrow{x}) = \sum_{i=1}^{D}\left(\sum_{j=1}^{i} z_j\right)^2 \tag{21}$$

where $z = x - o$and $O = \{o(1), o(2), \ldots, o(D)\}$ is the shifted global optimum. The global minimum of this function is $f_9^{Opt}(X^*) = 0.0$ and $X^* = O$. The initial range for each variable is $[-100, 100]$.

The 10th function is Shifted Rosenbrock function, defined as

$$f_{10}(\vec{x}) = \sum_{i=1}^{D-1}(100(z_{i+1} - z_i{}^2)^2 + (z_i - 1)^2) \tag{22}$$

where $z = x - o$ and $O = \{o(1), o(2), \ldots, o(D)\}$ is the shifted global optimum. The global minimum of this function is $f_{10}^{Opt}(X^*) = 0.0$ and $X^* = O$. The initial range for each variable is $[-100,100]$.

The 11th function is Shifted Rastrigin function, defined as

$$f_{11}(\vec{x}) = \sum_{i=1}^{D}(z_i^2 - 10\cos(2\pi z_i) + 10) \tag{23}$$

where $z = x - o$ and $O = \{o(1), o(2), \ldots, o(D)\}$ is the shifted global optimum. The global minimum of this function is $f_{11}^{Opt}(X^*) = 0.0$ and $X^* = O$. The initial range for each variable is $[-5.0,5.0]$.

The 12th function is Shifted Griewank function, defined as

$$f_{12}(\vec{x}) = \frac{1}{4000}\sum_{i=1}^{D} z_i^2 - \prod_{i=1}^{D}\cos\left(\frac{z_i}{\sqrt{i}}\right) + 1 \tag{24}$$

where $z = x - o$and $O = \{o(1), o(2), \ldots, o(D)\}$ is the shifted global optimum. The global minimum of this function is $f_{12}^{Opt}(X^*) = 0.0$ and $X^* = O$. The initial range for each variable is $[-600,600]$.

The 13th function is Shifted Ackley function, defined as

$$f_{13}(\vec{x}) = 20 + e - 20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} z_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi z_i)\right) \tag{25}$$

where $z = x - o$ and $O = \{o(1), o(2), \ldots, o(D)\}$ is the shifted global optimum. The global minimum of this function is $f_{13}^{Opt}(X^*) = 0.0$ and $X^* = O$. The initial range for each variable is $[-32,32]$.

The 14th function is Shifted Rotated Rastrigin function, defined as

$$f_{14}(\vec{x}) = \sum_{i=1}^{D}(z_i^2 - 10\cos(2\pi z_i) + 10) \tag{26}$$

where $z = (x - o) \times M$, $O = \{o(1), o(2), \ldots, o(D)\}$ is the shifted global optimum, M is linear transformation matrix and condition number=2. The global minimum of this function is $f_{14}^{Opt}(X^*) = 0.0$ and $X^* = O$. The initial range for each variable is $[-5.0, 5.0]$.

The 15th function is Shifted Rotated Ackley function, defined as

$$f_{15}(\vec{x}) = 20 + e - 20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} z_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi z_i)\right) \tag{27}$$

where $z = (x - o) \times M$, $O = \{o(1), o(2), \ldots, o(D)\}$ is the shifted global optimum, M is linear transformation matrix and condition number=1. The global minimum of this function is $f_{15}^{Opt}(X^*) = 0.0$ and $X^* = O$. The initial range for each variable is $[-32,32]$.

The 16th function is Shifted Rotated Griewank function, defined as

$$f_{16}(\vec{x}) = \frac{1}{4000}\sum_{i=1}^{D} z_i^2 - \prod_{i=1}^{D}\cos\left(\frac{z_i}{\sqrt{i}}\right) + 1 \tag{28}$$

where $z = (x - o) \times M$, $O = \{o(1), o(2), \ldots, o(D)\}$ is the shifted global optimum, M is linear transformation matrix and condition number=3. The global minimum of this function is $f_{16}^{Opt}(X^*) = 0.0$ and $X^* = O$. The initial range for each variable is $[-600,600]$.

The 17th function is Shifted Expanded Griewank plus Rosenbrock function, defined as (Suganthan et al., 2005).

$$f_{17}(\vec{x}) = f_4(f_2(z_1,z_2)) + f_4(f_2(z_2,z_3)) + \ldots + f_4(f_2(z_{D-1},z_D)) + f_4(f_2(z_D,z_1)) \tag{29}$$

f_4 : Griewank function

f_2 : Rosenbrock function

where $z = x - o + 1$ and $O = \{o(1), o(2), \ldots, o(D)\}$ is the shifted global optimum. The global minimum of this function is $f_{17}^{Opt}(X^*) = 0.0$ and $X^* = O$. The initial range for each variable is $[-3.0,1.0]$.

The 18th function is Shifted Rotated Expanded Schaffer f6 function, defined as

$$f_{18}(\vec{x}) = f_9(z_1, z_2) + f_9(z_2, z_3) + \ldots + f_9(z_{D-1}, z_D) + f_9(z_D, z_1) \tag{30}$$

f_9 : Schaffer $f6$ function

where $z = (x - o) \times M$, $O = \{o(1), o(2), \ldots, o(D)\}$ is the shifted global optimum, M is linear transformation matrix and condition number=3. The global minimum of this function is $f_{18}^{Opt}(X^*) = 0.0$ and $X^* = O$. The initial range for each variable is $[-100,100]$.

Among the abovementioned benchmark problems, Sphere function and Schwefel's problem 2.22, are unimodal. Rosenbrock function is a unimodal function with a narrow parabolic shaped valley from the perceived local optima to the global optimum. Some researchers believe that this function can be also treated as a multimodal function (Liang et al., 2006). Rastrigin, Ackley, Schaffer f6 and Griewank functions are well-known as difficult nonlinear multimodal functions (Digalakis and Margaritis, 2002). The main characteristic of these functions is that the number of local optima increases exponentially with the problem dimension (Pan et al., 2010b). Functions f_8 to f_{18} are shifted functions while the global optimums are shifted to a random position; consequently, the global optimum positions have different numerical values for different dimensions (Liang et al., 2005). Functions f_{14}, f_{15}, f_{16} and f_{18} are shifted Rotated functions; while f_{17} and f_{18} are expanded ones. Table 1 briefly shows the characteristics of the functions.

5.2. Common experimental settings

In order to show the performance of the proposed algorithm, all benchmark functions are tested with dimensions 30 and 50. The computational parameters of HS, IHS and GHS algorithms are set to values recommended in (Omran and Mahdavi, 2008). The harmony memory size (HMS) is equal to 5, $HMCR$=0.9, PAR=0.3 for HS, $PAR_{\min}$=0.01 and $PAR_{\max}$=0.99 for IHS and GHS methods, bw=0.01 for HS and $bw_{\min}$=0.0001 and $bw_{\max}=(UB-LB)/20$ for IHS algorithm. For SGHS, according to the values which are

Table 1
Characteristics of the test functions.

Test functions		Functions characteristics
f_1	Sphere	Unimodal
f_2	Griewank	Multimodal
f_3	Rastrigin	Multimodal
f_4	Rosenbrock	Unimodal treats as a multimodal
f_5	Ackley	Multimodal
f_6	Schwefel 2.22	Unimodal
f_7	Schaffer f6	Multimodal
f_8	Shifted Sphere	Unimodal and shifted
f_9	Shifted Schwefel 1.2	Unimodal and shifted
f_{10}	Shifted Rosenbrock	Multimodal and shifted
f_{11}	Shifted Rastrigin	Multimodal and shifted
f_{12}	Shifted Griewank	Multimodal and shifted
f_{13}	Shifted Ackley	Multimodal and shifted
f_{14}	Shifted rotated Rastrigin	Multimodal, shifted and rotated
f_{15}	Shifted rotated Ackley	Multimodal, shifted and rotated
f_{16}	Shifted rotated Griewank	Multimodal, shifted and rotated
f_{17}	Shifted expanded Griewank plus Rosenbrock	Expanded, multimodal and shifted
f_{18}	Shifted rotated expanded Schaffer f6	Expanded, multimodal, shifted and rotated

recommended in (Pan et al., 2010b), main parameters of the algorithm are set as follows; $HMS=5$, $HMCRm=0.98$, $PARm=0.9$, $bw_{\min}=0.0005$ and $bw_{\max}=(UB\text{-}LB)/10$. For NGHS, the algorithm parameter values applied by Zou et al. (2010) are adopted here and NGHS parameters are set as $HMS=5$, $P_m=0.005$. In order to equalize the maximum number of evaluations performed in all algorithms, the maximum number of iteration (NI) is 50000 for all test cases. Parameters of basic MS algorithm are adopted as; $PMS=5$, $PMN=5$, $PMCR=0.98$, $PAR_{\min}=0.01$, $PAR_{\max}=0.99$, $bw=0.0$ in initial phase and $bw=5\text{E}-5$ in second phase (Ashrafi and Dariane, 2011).

In MS algorithm the PMN is equal to the number of evaluations performed in the each iteration of the algorithm. Therefore, the maximum numbers of iterations (NI) of proposed algorithm (AIP_MS) MS algorithm for different PMN can be calculated as follows;

$$NI=\frac{\text{Total Evaluations}}{PMN} \tag{31}$$

The values of the other parameters of the algorithm are chosen to be equal to $PMN=5$, $PMS=5$, $PMCR=0.98$, $PAR_{\min}=0.01$, $PAR_{\max}=0.99$, and $bw_k=(UB_k\text{-}LB_k)/200$ for the kth variable. NII parameter is determined by sensitivity analysis carried out for each test function.

5.3. Experimental results and discussion

For each test case, 30 trials were carried out using different random seeds and the average optimal values, standard deviations, maximum and minimum values of the results as well as success rates are presented in Tables 2–5. The best results are specified in bold. In the results illustrations, Basic_MS refers to the primary MS algorithm introduced by Ashrafi and Dariane (2011) and the AIP_MS refers to the proposed algorithm in this study. The obtained results of centered functions (i.e. f_1 to f_7) are summarized in Tables 2 and 4 for dimensions 30 and 50 respectively. In most real-world problems, numerical values of variables in global optimum are not the same in different dimensions. In order to evaluate the performance of proposed algorithm in such problems, eleven shifted functions (Suganthan et al., 2005) are applied in this study while the obtained results are compared with those of HS, IHS, GHS, SGHS, NGHS and basic MS algorithms. Tables 3 and 5 present the results of Shifted functions for dimensions 30 and 50, respectively.

Table 2
Results for centered problems with 30 dimensions ($D=30$).

Functions	HS	IHS	GHS	SGHS	NGHS	Basic_MS	AIP_MS
f_1							
Mean	5.4173E+00	6.6886E+00	1.2334E−02	1.6297E−09	3.6910E−14	4.3430E−17	**3.6204E−123**
Std.	2.7958E+00	3.4287E+00	1.9797E−02	8.6970E−10	1.0503E−13	2.0347E−16	**7.6392E−123**
Success rate	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
Max	1.2069E+01	2.0586E+01	8.4177E−02	3.6440E−09	5.8343E−13	1.1036E−15	3.0308E−122
Min	1.8882E+00	1.9424E+00	3.8369E−08	3.0687E−10	5.1518E−16	1.2583E−196	5.7263E−130
f_2							
Mean	1.0777E+00	1.0860E+00	3.5078E−02	6.5408E−02	6.8302E−02	3.7007E−18	**0.0000E+00**
Std.	2.7869E−02	2.2196E−02	7.3063E−02	4.0092E−02	5.8552E−02	2.0270E−17	**0.0000E+00**
Success rate	0.00%	0.00%	0.00%	0.00%	3.33%	96.67%	100.00%
Max	1.1293E+00	1.1333E+00	2.9983E−01	1.6278E−01	2.6130E−01	1.1102E−16	0.0000E+00
Min	1.0360E+00	1.0478E+00	1.8235E−08	4.3710E−04	0.0000E+00	0.0000E+00	0.0000E+00
f_3							
Mean	4.6053E+00	7.0379E+00	2.7938E−02	1.6897E−01	1.3266E+01	8.5539E−03	**0.0000E+00**
Std.	2.1954E+00	2.2701E+00	4.8197E−02	3.7611E−01	3.4399E+01	2.8549E−02	**0.0000E+00**
Success rate	0.00%	0.00%	0.00%	0.00%	0.00%	90.00%	100.00%
Max	1.1102E+01	1.2935E+01	1.7667E−01	9.9511E−01	9.9492E+01	1.3615E−01	0.0000E+00
Min	1.1502E+00	2.3932E+00	7.6333E−07	9.1138E−08	2.3093E−14	0.0000E+00	0.0000E+00
f_4							
Mean	3.1143E+03	4.7040E+03	3.4164E+02	5.8959E+02	7.2139E+02	2.4378E+00	**2.1130E+00**
Std.	2.9117E+03	2.9777E+03	1.3702E+03	1.8626E+03	1.7810E+03	6.1901E+00	**4.7047E+00**
Success rate	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
Max	1.2924E+04	1.3046E+04	7.5689E+03	8.7849E+03	7.9918E+03	2.8288E+01	2.1935E+01
Min	5.2250E+02	9.9251E+02	1.0653E−01	1.1427E−01	7.3642E−04	3.5753E−03	3.3723E−03
f_5							
Mean	9.6269E−01	1.3967E+00	1.6227E−02	2.3888E−05	2.3760E+00	**2.9606E−15**	3.4343E−15
Std.	4.9985E−01	3.9074E−01	2.0575E−02	5.5631E−06	5.4184E+00	**1.6383E−15**	1.1363E−15
Success rate	0.00%	0.00%	0.00%	0.00%	0.00%	20.00%	6.67%
Max	1.9577E+00	2.0137E+00	9.3381E−02	3.7779E−05	1.6170E+01	7.1054E−15	7.1054E−15
Min	2.5460E−02	4.9367E−01	7.6296E−06	1.5493E−05	7.6989E−09	0.0000E+00	0.0000E+00
f_6							
Mean	4.5425E+00	3.3077E+00	5.0645E−01	4.1831E−02	2.5538E−07	4.0674E−03	**9.0420E−78**
Std.	1.5568E+00	1.4538E+00	5.4573E−01	1.0598E−01	5.0247E−07	2.2278E−02	**2.1160E−77**
Success rate	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
Max	6.9744E+00	6.8599E+00	2.0215E+00	5.0352E−01	2.5321E−06	1.2202E−01	9.8618E−77
Min	1.2332E+00	5.4710E−01	1.5020E−02	8.5163E−05	3.4158E−08	6.3627E−117	8.3871E−82
f_7							
Mean	3.7164E−01	3.7383E−01	4.6615E−02	2.1850E−01	3.7484E−01	1.4426E−02	**1.1553E−02**
Std.	3.6623E−02	4.6349E−02	3.4846E−02	3.7970E−02	7.7901E−02	1.4841E−02	**3.8825E−03**
Success rate	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
Max	4.2973E−01	4.4191E−01	1.2699E−01	2.7274E−01	4.9954E−01	7.8189E−02	1.7224E−02
Min	2.7274E−01	2.7274E−01	9.7159E−03	1.2699E−01	2.2769E−01	5.5960E−06	7.2241E−03

Table 3
Results for shifted and rotated problems with 30 dimensions (D=30).

Functions	HS	IHS	GHS	SGHS	NGHS	Basic_MS	AIP_MS
f_8							
Mean	5.2945E+00	8.0465E+00	1.9265E+03	3.3217E−04	**3.1699E−14**	2.6849E+03	2.1185E−02
Std.	2.6239E+00	3.9048E+00	4.9355E+02	1.8194E−03	**1.2183E−13**	4.7140E+02	1.5160E−02
Success rate	0.00%	0.00%	0.00%	0.00%	63.33%	0.00%	0.00%
Max	1.2813E+01	1.8861E+01	2.9759E+03	9.9651E−03	6.3665E−13	3.5328E+03	6.9236E−02
Min	9.2610E−01	3.5380E+00	9.0528E+02	4.0355E−10	0.0000E+00	1.4827E+03	6.0161E−03
f_9							
Mean	4.0564E+03	4.0239E+03	2.1554E+04	**7.2317E+01**	3.4382E+02	2.5357E+04	3.1672E+03
Std.	1.4699E+03	1.1589E+03	5.3862E+03	**3.8188E+01**	1.8134E+02	4.9194E+03	1.1453E+03
Success rate	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
Max	7.5881E+03	6.9973E+03	3.2141E+04	1.6802E+02	7.0737E+02	3.4100E+04	5.1254E+03
Min	1.8334E+03	2.4551E+03	1.1665E+04	1.9538E+01	1.1609E+02	1.5595E+04	1.0911E+03
f_{10}							
Mean	2.5187E+03	4.6512E+03	2.9647E+07	6.6414E+02	8.9862E+02	5.4858E+07	**1.0355E+02**
Std.	1.8544E+03	3.6528E+03	1.7917E+07	2.2298E+03	2.9882E+03	2.8019E+07	**9.1995E+01**
Success rate	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
Max	7.8098E+03	1.4360E+04	9.3178E+07	1.2081E+04	1.1956E+04	1.2445E+08	5.3001E+02
Min	5.4843E+02	1.1350E+03	7.2905E+06	1.3846E+01	1.9780E−02	1.1306E+07	4.2259E+01
f_{11}							
Mean	9.5140E−01	2.4779E+00	6.5082E+01	1.4096E+00	2.4762E+01	8.5868E+01	**3.8445E−01**
Std.	8.7488E−01	1.5898E+00	1.0734E+01	1.4940E+00	2.7917E+01	9.3486E+00	**7.2196E−01**
Success rate	0.00%	0.00%	0.00%	0.00%	3.33%	0.00%	0.00%
Max	3.1468E+00	6.8254E+00	9.6411E+01	5.9698E+00	1.1781E+02	1.0453E+02	3.0547E+00
Min	1.2139E−01	5.8752E−02	4.7038E+01	1.3059E−07	0.0000E+00	6.7456E+01	2.7147E−04
f_{12}							
Mean	1.0766E+00	1.0876E+00	1.7643E+01	6.2682E−02	9.6056E−02	2.5129E+01	**4.1634E−02**
Std.	2.9414E−02	3.4841E−02	4.1709E+00	4.4721E−02	1.0605E−01	5.4020E+00	**3.1300E−02**
Success rate	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
Max	1.1258E+00	1.1764E+00	2.7124E+01	1.8122E−01	5.9945E−01	3.7883E+01	1.4413E−01
Min	1.0290E+00	1.0310E+00	1.2087E+01	8.0206E−03	7.3960E−03	1.6300E+01	9.3695E−03
f_{13}							
Mean	9.3437E−01	1.4773E+00	9.6350E+00	**2.2360E−05**	8.5089E+00	1.0631E+01	3.0369E−01
Std.	5.2735E−01	4.3973E−01	8.8069E−01	**4.8064E−06**	7.1262E+00	7.3218E−01	2.7880E−01
Success rate	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
Max	1.9014E+00	2.2532E+00	1.1815E+01	3.4308E−05	1.9367E+01	1.2143E+01	1.0999E+00
Min	3.7681E−02	2.7944E−01	8.1196E+00	1.3818E−05	−8.2548E−08	8.7766E+00	5.8265E−02
f_{14}							
Mean	9.6782E+01	8.6289E+01	2.8903E+02	**1.2012E+02**	5.7227E+02	2.8669E+02	2.1284E+02
Std.	3.3349E+01	2.0991E+01	2.3619E+01	**3.5172E+01**	1.2556E+02	5.4329E+01	3.8259E+01
Success rate	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
Max	1.9316E+02	1.3432E+02	3.2979E+02	2.0795E+02	9.4828E+02	3.6467E+02	2.8972E+02
Min	4.9074E+01	5.3294E+01	2.3108E+02	5.9697E+01	3.0048E+02	2.0336E+02	1.6537E+02
f_{15}							
Mean	6.0100E+00	5.9818E+00	1.0075E+01	7.4583E+00	1.9792E+01	1.1894E+01	**5.5007E+00**
Std.	1.5878E+00	5.7412E+00	8.0460E−01	9.2410E+00	1.5124E−01	1.3110E+00	**2.0312E+00**
Success rate	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
Max	9.7392E+00	1.8997E+01	1.2033E+01	2.0925E+01	1.9961E+01	1.5958E+01	9.0717E+00
Min	4.0310E+00	1.0324E−01	8.6406E+00	2.6701E−04	1.9400E+01	9.1119E+00	2.0178E+00
f_{16}							
Mean	3.1850E+00	1.0876E+00	9.8388E+01	4.9828E−01	**4.8668E−02**	2.9291E+02	3.2360E−01
Std.	1.2344E+00	3.4841E−02	4.1793E+01	2.2441E−01	**3.8342E−02**	7.5748E+01	8.7042E−02
Success rate	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
Max	6.6908E+00	6.4660E+00	2.2590E+02	9.6579E−01	2.0101E−01	3.5684E+02	5.6021E−01
Min	1.6268E+00	1.0822E+00	4.3985E+01	2.3514E−01	4.0215E−03	2.6376E+02	1.7366E−01
f_{17}							
Mean	5.5093E+00	2.3473E+00	1.1272E+01	2.1256E+00	1.4934E+00	2.8528E+02	**1.4253E+00**
Std.	9.6863E−01	5.4180E−01	1.8211E+00	3.4831E−01	3.9305E−01	5.7007E+01	**3.3879E−01**
Success rate	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
Max	7.5151E+00	4.0346E+00	1.4452E+01	2.9676E+00	2.5556E+00	3.8526E+02	2.2366E+00
Min	2.8896E+00	1.6680E+00	7.3277E+00	1.6139E+00	8.5180E−01	2.0373E+02	9.5564E−01
f_{18}							
Mean	1.2993E+01	1.3053E+01	1.3638E+01	1.3044E+01	1.3648E+01	2.9920E+02	**1.2604E+01**
Std.	4.8794E−01	4.2032E−01	1.8089E−01	5.9219E−01	3.0831E−01	7.7679E+01	**7.0435E−01**
Success rate	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
Max	1.3571E+01	1.3746E+01	1.3905E+01	1.3993E+01	1.4039E+01	3.7660E+02	1.3432E+01
Min	1.1615E+01	1.2034E+01	1.3247E+01	1.1436E+01	1.2947E+01	2.1606E+02	1.0029E+01

Table 4
Results for centered problems with 50 dimensions ($D=50$).

Functions	HS	IHS	GHS	SGHS	NGHS	Basic_MS	AIP_MS
f_1							
Mean	5.1054E+02	5.5924E+02	2.3819E+00	5.0688E−08	5.2782E−07	6.8208E−16	**1.7270E−124**
Std.	1.1920E+02	1.2656E+02	3.7155E+00	1.1309E−07	8.5435E−07	2.0835E−16	**5.2616E−124**
Success rate	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
Max	8.4728E+02	7.8674E+02	1.3917E+01	6.0040E−07	4.8281E−06	1.1092E−15	2.1715E−123
Min	2.9474E+02	3.2022E+02	1.3142E−04	7.5862E−09	9.8383E−08	3.0885E−16	2.1517E−130
f_2							
Mean	5.6142E+00	5.5038E+00	6.7826E−01	1.1601E−01	3.9735E−02	1.1332E−03	**0.0000E+00**
Std.	1.2167E+00	9.8849E−01	3.2482E−01	1.8058E−01	4.7654E−02	6.2069E−03	**0.0000E+00**
Success rate	0.00%	0.00%	0.00%	0.00%	0.00%	90.00%	100.00%
Max	8.7217E+00	8.0226E+00	1.1378E+00	1.0462E+00	1.7505E−01	3.3996E−02	0.0000E+00
Min	3.4590E+00	3.8055E+00	5.4037E−05	3.0415E−02	9.8571E−08	0.0000E+00	0.0000E+00
f_3							
Mean	7.3486E+01	7.5762E+01	1.4929E+00	4.4783E+00	1.7518E+01	2.3429E−09	**0.0000E+00**
Std.	1.1024E+01	1.0652E+01	2.4238E+00	2.2448E+00	3.7689E+01	1.2831E−08	**0.0000E+00**
Success rate	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	100.00%
Max	9.8795E+01	9.2355E+01	8.5711E+00	9.7178E+00	1.0148E+02	7.0280E−08	0.0000E+00
Min	5.0530E+01	5.7138E+01	2.7789E−03	9.9496E−01	3.8720E−05	1.0481E−13	0.0000E+00
f_4							
Mean	2.4582E+06	2.5824E+06	2.4221E+03	1.1098E+03	4.8091E+02	1.4338E+01	**1.3659E+01**
Std.	1.2094E+06	1.0450E+06	4.9672E+03	2.0902E+03	1.4647E+03	1.6817E+01	**2.5424E+01**
Success rate	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
Max	5.6163E+06	4.7575E+06	2.0432E+04	8.3891E+03	5.8845E+03	5.0044E+01	1.0092E+02
Min	7.5985E+05	9.5242E+05	3.8865E+00	8.8120E+01	2.8900E+00	2.4611E−11	7.6312E−03
f_5							
Mean	5.2769E+00	5.1836E+00	3.1024E−01	5.0353E−05	2.0704E−05	1.5628E−08	**5.3291E−15**
Std.	3.4206E−01	3.9437E−01	4.1571E−01	9.4438E−06	4.7272E−04	2.6065E−09	**2.0334E−15**
Success rate	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	3.33%
Max	5.9702E+00	6.1354E+00	1.6490E+00	8.0018E−05	1.4449E+01	2.0070E−08	7.1054E−15
Min	4.6452E+00	4.2507E+00	1.5036E−02	3.6857E−05	7.6252E−05	1.0379E−08	0.0000E+00
f_6							
Mean	6.3168E+01	6.5670E+01	2.1681E+00	2.9727E−01	2.6752E−03	4.5392E−10	**8.4158E−75**
Std.	8.1157E+00	9.6298E+00	2.0023E+00	1.7519E−01	1.4763E−03	9.2025E−11	**3.4518E−74**
Success rate	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
Max	8.9311E+01	8.1341E+01	6.8593E+00	8.2942E−01	6.1073E−03	6.1609E−09	1.8877E−73
Min	4.5810E+01	4.2489E+01	9.9540E−02	8.9269E−02	8.9361E−04	3.6632E−11	1.4990E−78
f_7							
Mean	4.7778E−01	4.7813E−01	1.3942E−01	3.9111E−01	4.8181E−01	6.5696E−02	**4.0852E−02**
Std.	1.1135E−02	7.2418E−03	7.4415E−02	3.6797E−02	2.3408E−02	1.0794E−01	**1.6433E−02**
Success rate	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
Max	4.9151E−01	4.9150E−01	3.7329E−01	4.4191E−01	4.9965E−01	3.7224E−01	7.8189E−02
Min	4.4191E−01	4.5978E−01	3.7225E−02	3.1210E−01	4.1467E−01	9.7159E−03	9.7159E−03

Tables 2 and 4 indicate that although the performance of basic MS algorithm is slightly better than the proposed algorithm in optimizing Ackley function with dimension 30, AIP_MS algorithm shows much better performance for all centered test functions with dimensions 30 and 50. For Sphere, Griewank, Rastrigin and Schwefel 2.22 functions, the obtained results from AIP_MS algorithm are significantly better than the ones from other methods. For Griewank and Rastrigin functions the proposed algorithm is capable for finding the global optimum.

Based on the results presented in Table 3 for dimensions 30, it is clear that AIP_ MS algorithm produced better solutions for f_{10}, f_{11}, f_{12}, f_{15}, f_{17} and f_{18} functions as compared with the other algorithms; where, the entire feasible ranges of decision variables for Shifted Griewank (f_{12}) are wider than those for other functions. For f_9, f_{13} and f_{14} with 30 dimensions, the best results were found by SGHS algorithm while SGHS cannot preserve the accuracy of the results for dimension 50 of f_9 and f_{14}. The NGHS algorithm could produce the best results for f_8 and f_{16} with 30 dimensions.

The results presented in Table 5 for dimension 50; show that the proposed algorithm is superior over other abovementioned algorithms for all test functions except for Shifted Sphere and Shifted Ackley functions. For Shifted Sphere function the best result was found by NGHS and for Shifted Ackley function, SGHS could produce the best result; however, AIP_MS algorithm could find appropriate results. Therefore, capability of MS algorithm for solving multimodal optimization problems with wide range of variables is presumable, especially in high dimensional problems. The results also indicate that the performance of AIP_MS algorithm is less sensitive to the increase of problem dimensions as compared to the other methods. Figs. 6 and 7 present typical solution history graphs along fitness evaluations for $f_1, f_4, f_5, f_6, f_{10}$, $f_{11}, f_{12}, f_{13}, f_{14}$ and f_{16} functions with 50 dimension.

Maximum number of iterations for the initial phase, (parameter *NII*) for all test functions and all dimensions was determined by sensitive analysis. The best *NII* values for centered test cases were obtained between 3% and 15% of maximum number of iterations and for shifted and rotated test cases were obtained between 45% and 92% of maximum number of iterations. Although determining the exact range or mathematical relationship for this parameter requires further investigations, the obtained results show that the maximum number of iterations in the initial phase for AIP_MS algorithm must be increased for finding appropriate results for shifted and rotated functions. The mean of functions' fitness values versus different values of maximum iteration numbers in the initial phases are shown in

Table 5
Results for shifted and rotated problems with 50 dimensions ($D=50$).

Functions		HS	IHS	GHS	SGHS	NGHS	Basic_MS	AIP_MS
f_8	Mean	5.6640E+02	5.5927E+02	1.5928E+04	3.2044E−04	**2.0119E−07**	1.6444E+04	1.3584E−01
	Std.	1.2264E+02	1.2389E+02	2.9820E+03	1.7246E−03	**1.7393E−07**	1.8506E+03	6.7357E−02
	Success rate	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	Max	8.1509E+02	8.3434E+02	2.0998E+04	9.4503E−03	7.5387E−07	1.9642E+04	2.9019E−01
	Min	3.1978E+02	3.4302E+02	1.0568E+04	1.1553E−08	3.9479E−08	1.2404E+04	4.4971E−02
f_9	Mean	2.9922E+04	3.2850E+04	9.7194E+04	1.8990E+03	4.7277E+03	9.9648E+04	**1.8208E+03**
	Std.	7.0723E+03	8.0712E+03	1.9478E+04	5.4127E+02	1.6261E+03	1.2164E+04	**5.9322E+02**
	Success rate	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	Max	4.5069E+04	4.7155E+04	1.4019E+05	3.1171E+03	7.9632E+03	1.1979E+05	2.8183E+03
	Min	1.2376E+04	1.7275E+04	6.0252E+04	8.9339E+02	2.2651E+03	7.3286E+04	1.0773E+03
f_{10}	Mean	2.5210E+06	2.3166E+06	1.6062E+09	1.1340E+03	1.0619E+03	1.3908E+07	**3.0795E+02**
	Std.	1.2964E+06	7.3263E+05	5.2497E+08	2.8072E+03	3.6149E+03	2.8719E+06	**9.9375E+01**
	Success rate	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	Max	6.0687E+06	3.7787E+06	3.5469E+09	1.4357E+04	1.6311E+04	1.8776E+07	5.9655E+02
	Min	1.1706E+06	1.2787E+06	7.9859E+08	8.1019E+01	1.0571E+00	8.1083E+06	1.8808E+02
f_{11}	Mean	3.6765E+01	4.2881E+01	2.3666E+02	4.2868E+00	6.0165E+01	2.4733E+02	**2.2488E+00**
	Std.	4.7886E+00	4.9695E+00	2.2205E+01	1.6567E+00	4.0244E+01	1.9772E+01	**1.4399E+00**
	Success rate	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	Max	4.5117E+01	5.3588E+01	3.0053E+02	8.9547E+00	1.4948E+02	2.8208E+02	6.8566E+00
	Min	2.6697E+01	3.1061E+01	1.9845E+02	9.9499E−01	3.1169E+00	2.0142E+02	3.3034E−01
f_{12}	Mean	5.9451E+00	6.5416E+00	1.4741E+02	8.4691E−02	5.7817E−02	1.4261E+02	**4.8811E−02**
	Std.	1.1345E+00	1.3592E+00	2.3710E+01	4.1155E−02	6.6703E−02	2.4004E+01	**3.5757E−02**
	Success rate	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	Max	7.7768E+00	9.1935E+00	2.0011E+02	2.0239E−01	3.3219E−01	1.9507E+02	1.3788E−01
	Min	3.6168E+00	4.5619E+00	9.1311E+01	3.1732E−02	2.0650E−08	8.9546E+01	1.0065E−02
f_{13}	Mean	5.3607E+00	5.2751E+00	1.4859E+01	**4.7326E−05**	1.4651E+01	1.5131E+01	1.1696E+00
	Std.	3.5957E−01	4.1694E−01	4.9037E−01	**8.8534E−06**	5.8037E+00	6.0818E−01	3.9617E−01
	Success rate	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	Max	6.0094E+00	6.1185E+00	1.5580E+01	6.5672E−05	1.9068E+01	1.6570E+01	2.4901E+00
	Min	4.7364E+00	4.3623E+00	1.3352E+01	3.2486E−05	3.9579E−05	1.4389E+01	2.9076E−01
f_{14}	Mean	4.5860E+02	4.7070E+02	5.9682E+02	2.4491E+02	1.0726E+03	5.9781E+02	**2.2825E+02**
	Std.	5.2066E+01	3.2927E+01	3.3556E+01	5.8355E+01	2.0238E+02	3.8558E+01	**1.0678E+01**
	Success rate	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	Max	5.3817E+02	5.2089E+02	6.7252E+02	3.3331E+02	1.4586E+03	6.6601E+02	4.7670E+02
	Min	3.3844E+02	3.7347E+02	5.1372E+02	1.4924E+02	7.3725E+02	5.2447E+02	8.0244E+01
f_{15}	Mean	9.1714E+00	9.0851E+00	1.5535E+01	1.3270E+01	1.9821E+01	1.6330E+01	**8.8373E+00**
	Std.	1.8237E+00	1.9273E+00	8.6483E−01	9.0301E+00	1.0101E−01	9.8470E−01	**1.0340E+00**
	Success rate	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	Max	1.3187E+01	1.3013E+01	1.6735E+01	2.0603E+01	1.9986E+01	1.9551E+01	1.0741E+01
	Min	6.5625E+00	7.0055E+00	1.3688E+01	5.5883E−04	1.9503E+01	1.4213E+01	6.7951E+00
f_{16}	Mean	4.1373E+01	4.7256E+01	6.3883E+02	1.0508E+00	1.0428E+00	5.4519E+02	**1.0341E+00**
	Std.	1.3758E+01	1.5849E+01	1.7350E+02	1.4829E−01	2.1147E−02	1.0026E+02	**1.6974E−01**
	Success rate	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	Max	7.7142E+01	8.4105E+01	1.0979E+03	1.7149E+00	1.0724E+00	8.6918E+02	1.3652E+00
	Min	2.3024E+01	2.5120E+01	3.3599E+02	7.9771E−01	1.0011E+00	3.4229E+02	5.1441E−01
f_{17}	Mean	1.8341E+01	1.4337E+01	4.4428E+01	4.2185E+00	2.7260E+00	3.2749E+01	**2.5200E+00**
	Std.	3.7765E+00	1.7375E+00	4.0261E+00	6.8054E−01	3.9343E−01	3.6506E+00	**8.3096E−01**
	Success rate	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	Max	2.5232E+01	1.4569E+01	5.3676E+01	5.7562E+00	3.6325E+00	3.9929E+01	4.4003E+00
	Min	1.1137E+01	7.1826E+00	3.7780E+01	3.0784E+00	2.0714E+00	3.0235E+01	1.0507E+00
f_{18}	Mean	2.3330E+01	2.3271E+01	2.3472E+01	2.2563E+01	2.3340E+01	2.3438E+01	**2.2465E+01**
	Std.	2.3189E−01	1.9227E−01	2.2929E−01	5.6482E−01	3.6090E−01	2.2158E−01	**3.5177E−01**
	Success rate	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
	Max	2.3849E+01	2.3581E+01	2.3902E+01	2.3540E+01	2.3977E+01	2.3463E+01	2.3222E+01
	Min	2.2843E+01	2.2839E+01	2.2856E+01	2.0830E+01	2.2543E+01	2.2298E+01	2.1785E+01

Figs. 8 and 9 for centered (i.e. f_1, f_4, f_5 and f_7) and shifted (i.e. f_8, f_9, f_{10}, f_{11}, f_{12} and f_{13}) functions ($D=50$), respectively.

5.4. Statistical tests and effects of parameters

Mann–Whitney *U* tests are conducted to show whether the proposed algorithm results are statistically different from those obtained by the other aforementioned algorithms. The Mann–Whitney *U* test (Mann and Whitney, 1947) is a non-parametric statistical rank-based test for considering differences between independent populations.

After calculating the test statistic U for a pair data series, the smaller value of U_1 and U_2 is the one used when consulting significance tables (Zou et al., 2010). Six groups of Mann–Whitney *U* tests are performed applying all eighteen test functions with 50 dimensions, while 30 independent experiments are carried out in each case, and the results are presented in Table 6.

From Table 6, it can be concluded that the values of $U_{\mathrm{AIP\_MS}}$ for all test functions are significantly smaller than the U_{HS}, U_{IHS} and

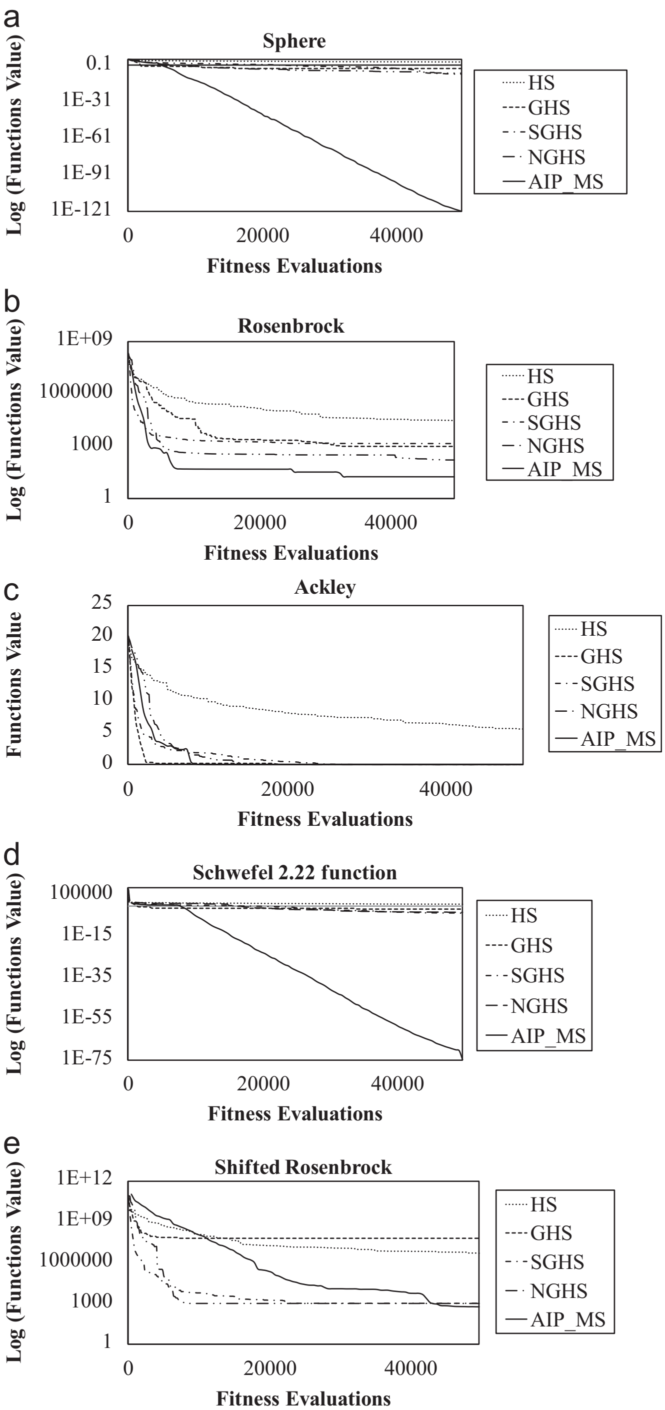

Fig. 6. Typical convergence curves (D=50). (a) f_1: Sphere function, (b) f_4: Rosenbrock function, (c) f_5: Ackley function, (d) f_6: Schwefel function 2.22 and (e) f_{10}: shifted Rosenbrock function.

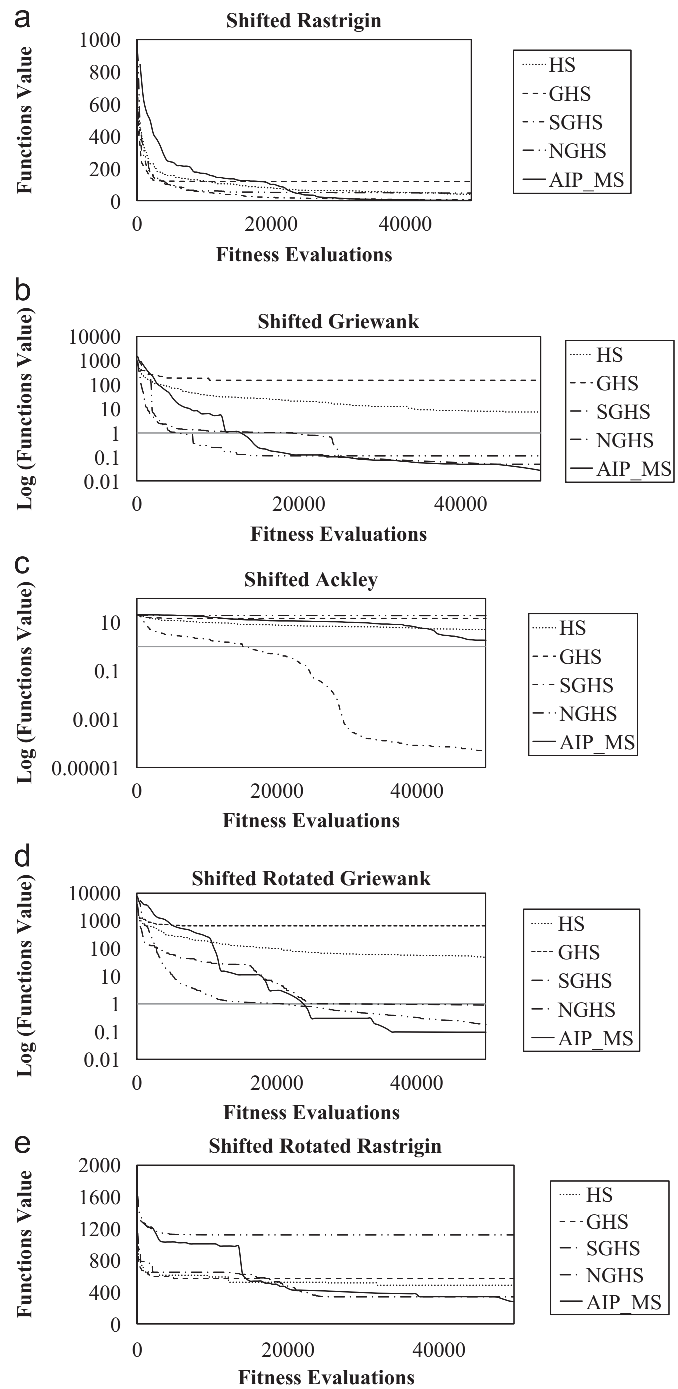

Fig. 7. Typical convergence curves (D=50). (a) f_{11}: shifted Rastrigin function, (b) f_{12}: shifted Griewank function, (c) f_{13}: shifted Ackley function, (d) f_{16}: shifted Rotated Griewank function and (e) f_{14}: shifted Rotated Rastrigin function.

U_{GHS} values. Thus it is safely concluded that the performance of AIP_MS is statistically better than HS, IHS and GHS algorithms. Table 6 shows that the values of $U_{\text{AIP\_MS}}$ are smaller than those of U_{SGHS} for all test functions, except for Shifted Sphere and Shifted Ackley functions. It is consequently revealed that the proposed algorithm is more effective than SGHS for solving all test cases except those two functions. Although the efficiency of AIP_MS in case of Shifted Schwefel 1.2 and Shifted Rotated Griewank functions is better than the SGHS, there are no statistically significant differences between results of two compared algorithms. As can be seen from Table 6, the values of $U_{\text{AIP\_MS}}$ are clearly smaller than those of U_{NGHS} except for Shifted Sphere function. In other words, AIP_MS has obtained better solutions compared to the NGHS algorithm which indicates that the proposed algorithm outperforms NGHS. For centered functions (f_1–f_7) the numbers of AIP_MS solutions beaten by HS, HIS, GHS, SGHS and NGHS solutions are 0. It can be seen that the proposed method produces significantly better results than basic MS for all test cases. For Schaffer function f6, there are no significant differences between results of AIP_MS and basic MS algorithm.

In order to evaluate the influence of *PMN* value on the performance of AIP_MS algorithm, four *PMN* values (i.e. 2, 5, 10,

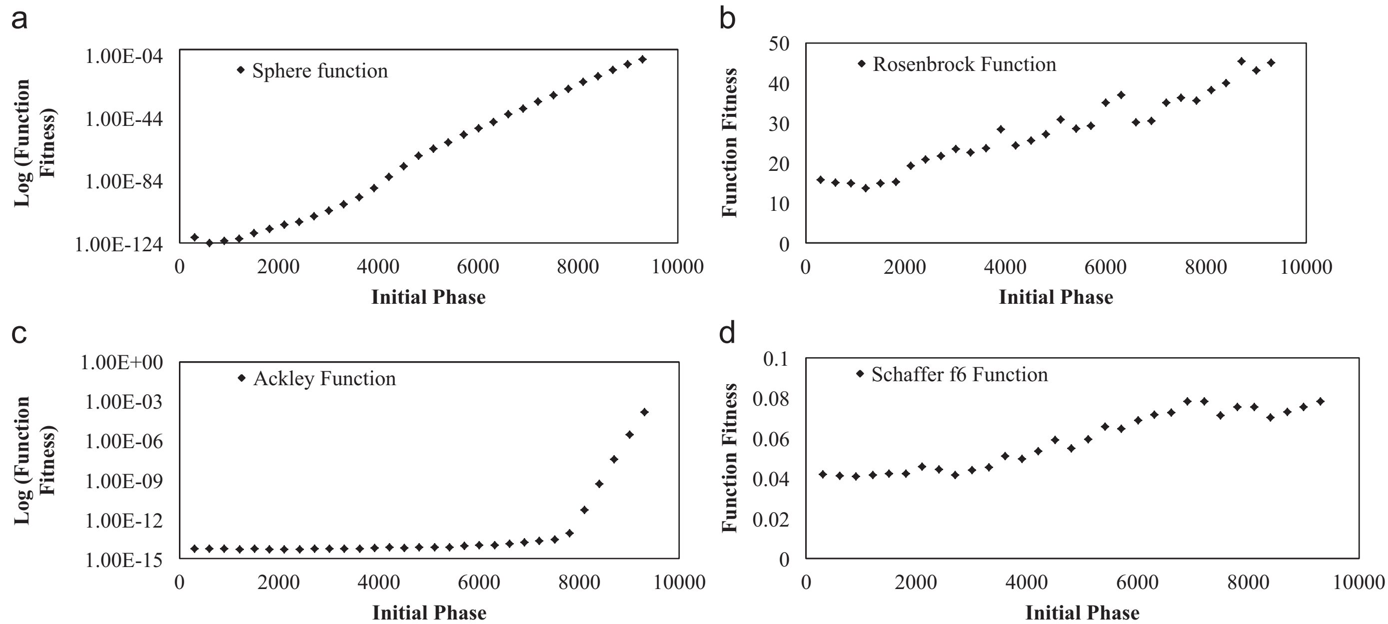

Fig. 8. Median values of function fitness versus values of Initial phases (50 *D*). (a) f_1: Sphere function, (b) f_4: Rosenbrock function, (c) f_5: Ackley function and (d) f_7: Schaffer function f6.

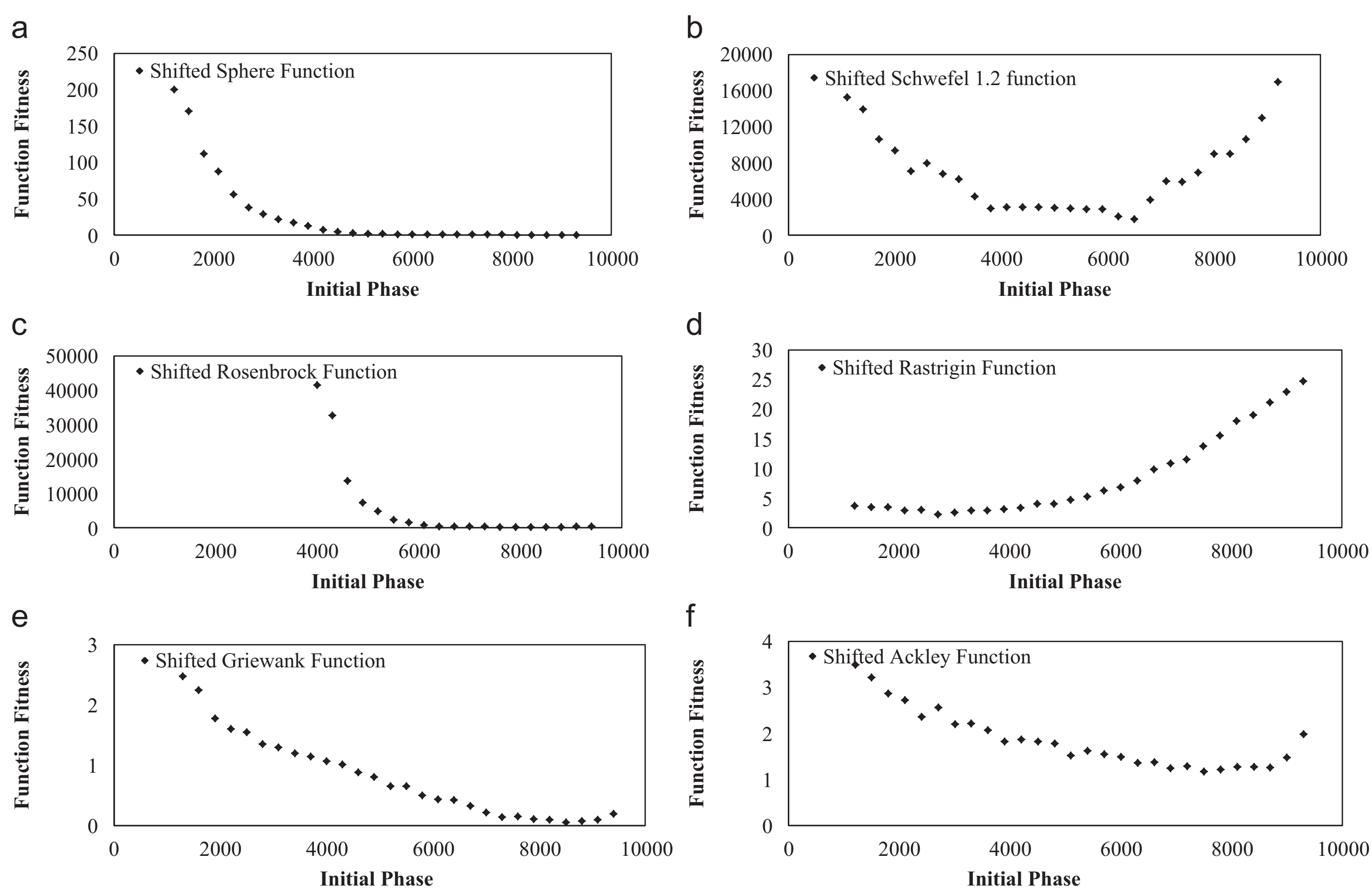

Fig. 9. Median values of function fitness versus values of Initial phases (50 *D*). (a) f_8: shifted Sphere function, (b) f_9: shifted Schwefel function 1.2, (c) f_{10}: shifted Rosenbrock function. (d) f_{11}: shifted Rastrigin function, (e) f_{12}: Shifted Griewank function and (f) f_{13}: shifted Ackley function.

and 20) were adopted for each test case with 50 dimensions. Table 7 represents the results generated by using different *PMN* values for all test cases with a specified maximum number of iteration ($NI=10{,}000$). Although for centered functions the best results are obtained by using medium values of this parameter (e.g. $PMN=5$ or 10), for shifted functions the bigger *PMN* values reach better results. Note that, in these experiments, increasing the *PMN* value increases the total number of function evaluations

Table 6
Results of statistical tests (Mann–Whitney Wilcoxon) with $D=50$.

Functions	U_{AIP_MS}	U_{HS}	U_{AIP_MS}	U_{IHS}	U_{AIP_MS}	U_{GHS}	U_{AIP_MS}	U_{SGHS}	U_{AIP_MS}	U_{NGHS}	U_{AIP_MS}	U_{Basic_MS}
f_1	0	900	0	900	0	900	0	900	0	900	0	900
f_2	0	900	0	900	0	900	0	900	0	900	0	900
f_3	0	900	0	900	0	900	0	900	0	900	0	900
f_4	0	900	0	900	0	900	0	900	0	900	0	900
f_5	0	900	0	900	0	900	0	900	0	900	0	900
f_6	0	900	0	900	0	900	0	900	0	900	0	900
f_7	0	900	0	900	0	900	0	900	0	900	448	452
f_8	0	900	0	900	0	900	900	0	900	0	0	900
f_9	0	900	0	900	0	900	417	483	18	882	0	900
f_{10}	0	900	0	900	0	900	332	568	96	804	0	900
f_{11}	0	900	0	900	0	900	148	752	9	891	0	900
f_{12}	0	900	0	900	0	900	215	685	444	456	0	900
f_{13}	0	900	0	900	0	900	900	0	90	810	0	900
f_{14}	37	863	18	882	0	900	380	520	0	900	0	900
f_{15}	143	757	264	636	0	900	210	690	0	900	0	900
f_{16}	0	900	0	900	0	900	405	495	342	558	0	900
f_{17}	0	900	0	900	0	900	54	846	263	637	0	900
f_{18}	12	888	17	883	9	891	340	560	49	851	18	882

Table 7
The effect of PMN values ($D=50$, $NI=10000$).

Functions	*PMN* *NoFE*	**2** **20,000**	**5** **50,000**	**10** **100,000**	**20** **200,000**
f_1	Mean	7.0233E−06	1.7270E−124	1.3597E−110	2.5725E−100
	Std.	3.8140E−05	5.2616E−124	6.1709E−110	4.0578E−100
	Success rate	0.00%	0.00%	0.00%	0.00%
f_2	Mean	2.7342E−04	0.0000E+00	0.0000E+00	0.0000E+00
	Std.	1.0322E−03	0.0000E+00	0.0000E+00	0.0000E+00
	Success rate	70.00%	100.00%	100.00%	100.00%
f_3	Mean	1.1891E−06	0.0000E+00	0.0000E+00	0.0000E+00
	Std.	6.5090E−06	0.0000E+00	0.0000E+00	0.0000E+00
	Success rate	86.67%	100.00%	100.00%	100.00%
f_4	Mean	3.3587E+01	1.3659E+01	8.1088E−01	2.4408E−02
	Std.	2.4853E+01	2.5424E+01	3.8146E+00	5.1653E−03
	Success rate	0.00%	0.00%	0.00%	0.00%
f_5	Mean	4.7634E−05	5.3291E−15	4.7370E−15	4.3817E−15
	Std.	1.3958E−04	2.0334E−15	1.7034E−15	1.5283E−15
	Success rate	0.00%	3.33%	0.00%	0.00%
f_6	Mean	2.7080E−05	8.4158E−75	6.6341E−64	2.7365E−58
	Std.	1.1532E−04	3.4518E−74	9.6616E−64	3.4278E−58
	Success rate	0.00%	0.00%	0.00%	0.00%
f_7	Mean	6.3035E−02	4.0852E−02	3.6307E−02	2.5304E−02
	Std.	3.0377E−02	1.6433E−02	5.0223E−03	1.3864E−02
	Success rate	0.00%	0.00%	0.00%	0.00%
f_8	Mean	3.4487E+00	1.3584E−01	4.8026E−03	3.0314E−04
	Std.	1.0174E+00	6.7357E−02	1.0734E−02	7.8001E−05
	Success rate	0.00%	0.00%	0.00%	0.00%
f_9	Mean	1.2222E+04	1.8208E+03	1.4711E+03	1.0831E+03
	Std.	2.3545E+03	5.9322E+02	4.3267E+02	4.1539E+02
	Success rate	0.00%	0.00%	0.00%	0.00%
f_{10}	Mean	4.0114E+03	3.0795E+02	2.2009E+02	1.9609E+02
	Std.	3.0501E+03	9.9375E+01	6.6931E+01	3.6560E+01
	Success rate	0.00%	0.00%	0.00%	0.00%
f_{11}	Mean	2.2858E+01	2.2488E+00	1.2137E+00	3.8672E−01
	Std.	4.9934E+00	1.4399E+00	9.5814E−01	9.9976E−02
	Success rate	0.00%	0.00%	0.00%	0.00%
f_{12}	Mean	9.1410E−01	4.8811E−02	3.6460E−02	1.3184E−02
	Std.	8.6186E−02	3.5757E−02	3.9704E−02	3.4467E−03
	Success rate	0.00%	0.00%	0.00%	0.00%
f_{13}	Mean	4.3241E+00	1.1696E+00	1.7717E−01	1.9216E−01
	Std.	6.0444E−01	3.9617E−01	5.3394E−01	6.0719E−01
	Success Rate	0.00%	0.00%	0.00%	0.00%
f_{14}	Mean	1.8203E+03	2.2825E+02	2.0788E+02	1.0934E+02
	Std.	3.3377E+03	1.0678E+02	2.4465E+02	3.2913E+02

Table 7 (*continued*)

Functions	*PMN* *NoFE*	**2** **20,000**	**5** **50,000**	**10** **100,000**	**20** **200,000**
	Success rate	0.00%	0.00%	0.00%	0.00%
f_{15}	Mean	1.1378E+01	8.8373E+00	8.6722E+00	8.5259E+00
	Std.	1.9974E+00	1.0340E+00	1.0883E+00	1.0695E+00
	Success rate	0.00%	0.00%	0.00%	0.00%
f_{16}	Mean	1.6657E+00	1.0341E+00	8.7701E−01	8.4105E−01
	Std.	5.9516E−01	1.6974E−01	1.6089E−01	1.3098E−01
	Success rate	0.00%	0.00%	0.00%	0.00%
f_{17}	Mean	8.0888E+00	2.5200E+00	2.4924E+00	2.4248E+00
	Std.	1.7631E+00	8.3096E−01	8.6741E−01	8.9607E−01
	Success rate	0.00%	0.00%	0.00%	0.00%
f_{18}	Mean	3.0106E+01	2.2465E+01	2.2159E+01	2.2075E+01
	Std.	2.6708E+00	3.5177E−01	3.9617E−01	3.1108E−01
	Success rate	0.00%	0.00%	0.00%	0.00%

Table 8
The effect of PMN values $D=50$, $NoFE=50000$.

Functions	*PMN* *NI*	**2** **25,000**	**5** **10,000**	**10** **5000**	**20** **2500**
f_1	Mean	6.1339E−06	1.7270E−124	8.5735E−54	6.0244E−23
	Std.	2.3425E−05	5.2616E−124	2.4951E−53	4.0061E−23
	Success rate	0.00%	0.00%	0.00%	0.00%
f_2	Mean	3.4862E−04	0.0000E+00	0.0000E+00	1.6061E−15
	Std.	1.9087E−03	0.0000E+00	0.0000E+00	1.9599E−15
	Success rate	73.33%	100.00%	100.00%	0.00%
f_3	Mean	2.2822E−06	0.0000E+00	0.0000E+00	0.0000E+00
	Std.	1.0610E−05	0.0000E+00	0.0000E+00	0.0000E+00
	Success rate	63.33%	100.00%	100.00%	100.00%
f_4	Mean	3.8909E+01	1.3659E+01	3.2321E+00	5.4990E−01
	Std.	3.7170E+01	2.5424E+01	1.2007E+01	1.1006E+00
	Success rate	0.00%	0.00%	0.00%	0.00%
f_5	Mean	1.6815E−04	5.3291E−15	5.3291E−15	3.4498E−12
	Std.	8.9313E−04	2.0334E−15	1.8067E−15	1.2914E−12
	Success rate	0.00%	3.33%	0.00%	0.00%
f_6	Mean	5.5203E−05	8.4158E−75	4.0624E−32	9.0646E−15
	Std.	3.0224E−04	3.4518E−74	3.7291E−32	6.1724E−15
	Success rate	0.00%	0.00%	0.00%	0.00%
f_7	Mean	4.5397E−02	4.0852E−02	3.3556E−02	3.2151E−02
	Std.	2.1728E−02	1.6433E−02	9.5108E−03	2.0197E−02
	Success rate	0.00%	0.00%	0.00%	0.00%
f_8	Mean	1.8165E+00	1.3584E−01	2.0997E+00	3.5464E+01
	Std.	4.3218E−01	6.7357E−02	1.5278E+00	6.4691E+00
	Success rate	0.00%	0.00%	0.00%	0.00%
f_9	Mean	1.0412E+04	1.8208E+03	4.7786E+03	6.3368E+03
	Std.	3.0436E+03	5.9322E+02	7.6602E+02	9.5956E+02
	Success rate	0.00%	0.00%	0.00%	0.00%
f_{10}	Mean	9.5295E+02	3.0795E+02	2.2813E+03	1.1245E+05
	Std.	1.6165E+03	9.9375E+01	1.9733E+03	3.7537E+04
	Success rate	0.00%	0.00%	0.00%	0.00%
f_{11}	Mean	1.4439E+01	2.2488E+00	1.6971E+01	4.7007E+01
	Std.	4.1673E+00	1.4399E+00	2.9392E+00	5.2194E+00
	Success rate	0.00%	0.00%	0.00%	0.00%
f_{12}	Mean	6.7527E−01	4.8811E−02	1.2049E−01	1.0516E+00
	Std.	1.0076E−01	3.5757E−02	8.0795E−02	1.3144E+00
	Success rate	0.00%	0.00%	0.00%	0.00%
f_{13}	Mean	1.1192E+00	1.1696E+00	1.7597E+00	6.8430E+00
	Std.	5.3604E−01	3.9617E−01	5.4426E−01	9.7268E−01
	Success rate	0.00%	0.00%	0.00%	0.00%
f_{14}	Mean	3.0749E+03	2.2825E+02	3.1597E+02	9.7731E+02
	Std.	2.9397E+03	1.0678E+02	4.0063E+02	7.0716E+02

Table 8 (*continued*)

Functions	*PMN* *NI*	**2** **25,000**	**5** **10,000**	**10** **5000**	**20** **2500**
	Success rate	0.00%	0.00%	0.00%	0.00%
f_{15}	Mean	9.8509E+00	8.8373E+00	1.3329E+01	2.2979E+01
	Std.	7.8074E+00	1.0340E+00	2.2633E+00	2.6248E+01
	Success rate	0.00%	0.00%	0.00%	0.00%
f_{16}	Mean	1.4039E+00	1.0341E+00	1.4020E+01	2.2127E+02
	Std.	2.0115E−01	1.6974E−01	3.4697E+01	3.6174E+02
	Success rate	0.00%	0.00%	0.00%	0.00%
f_{17}	Mean	5.4353E+00	2.5200E+00	3.2877E+00	5.8404E+00
	Std.	6.2138E+01	8.3096E−01	2.6331E+00	5.2781E+00
	Success rate	0.00%	0.00%	0.00%	0.00%
f_{18}	Mean	2.7684E+01	2.2465E+01	2.3743E+01	2.5002E+01
	Std.	2.7756E+01	3.5177E−01	3.0843E+00	3.0281E+00
	Success rate	0.00%	0.00%	0.00%	0.00%

Table 9
The effect of PMCR values $D=50$, $NI=10000$.

Functions	*PMCR*	**0.5**	**0.7**	**0.9**	**0.95**	**0.98**
f_1	Mean	2.0230E−01	9.4173E−13	1.1925E−169	8.4917E−144	1.7270E−124
	Std.	8.2109E−01	5.1581E−12	2.3455E−170	2.9524E−143	5.2616E−124
	Success rate	0.00%	0.00%	13.33%	0.00%	0.00%
f_2	Mean	4.3481E−02	7.7067E−09	0.0000E+00	0.0000E+00	0.0000E+00
	Std.	1.8528E−01	3.8333E−08	0.0000E+00	0.0000E+00	0.0000E+00
	Success rate	0.00%	80.00%	100.00%	100.00%	100.00%
f_3	Mean	2.6437E+01	4.7954E+01	0.0000E+00	0.0000E+00	0.0000E+00
	Std.	5.6075E+01	3.7252E+01	0.0000E+00	0.0000E+00	0.0000E+00
	Success rate	3.33%	10.00%	100.00%	100.00%	100.00%
f_4	Mean	5.6133E+01	5.3696E+01	4.9513E+01	1.3318E+01	1.3659E+01
	Std.	3.0545E+01	2.0424E+01	1.0572E+01	2.5943E+01	2.5424E+01
	Success rate	0.00%	0.00%	0.00%	0.00%	0.00%
f_5	Mean	1.6779E−02	1.3651E−07	6.9870E−15	6.1580E−15	5.3291E−15
	Std.	5.4890E−02	4.1751E−07	6.4863E−16	1.5979E−15	2.0334E−15
	Success rate	0.00%	0.00%	0.00%	0.00%	3.33%
f_6	Mean	1.3264E−02	4.7490E−14	4.8535E−90	2.5409E−80	8.4158E−75
	Std.	3.4055E−02	2.0048E−13	1.4007E−89	7.9497E−80	3.4518E−74
	Success rate	0.00%	0.00%	3.33%	0.00%	0.00%
f_7	Mean	4.5417E−02	3.7224E−02	4.1321E−02	3.9955E−02	4.0852E−02
	Std.	1.6666E−02	2.9178E−08	1.2500E−02	1.0393E−02	1.6433E−02
	Success rate	0.00%	0.00%	0.00%	0.00%	0.00%
f_8	Mean	2.3679E+02	1.7605E+02	4.0343E+01	1.6288E+01	1.3584E−01
	Std.	1.4037E+02	1.3007E+02	1.2227E+01	1.1990E+00	6.7357E−02
	Success rate	0.00%	0.00%	0.00%	0.00%	0.00%
f_9	Mean	3.3003E+04	2.6747E+04	3.6779E+03	2.0665E+03	**1.8208E+03**
	Std.	6.7275E+03	7.2597E+03	9.8127E+02	6.3481E+02	**5.9322E+02**
	Success rate	0.00%	0.00%	0.00%	0.00%	0.00%
f_{10}	Mean	5.5268E+05	3.1939E+05	2.8701E+04	1.0911E+03	**3.0795E+02**
	Std.	7.3183E+05	2.6572E+05	2.4566E+04	1.7045E+03	**9.9375E+01**
	Success rate	0.00%	0.00%	0.00%	0.00%	0.00%
f_{11}	Mean	7.5416E+01	6.5365E+01	3.1360E+01	4.8833E+00	**2.2488E+00**
	Std.	1.6843E+01	1.5388E+01	8.8972E+00	2.1574E+00	**1.4399E+00**
	Success rate	0.00%	0.00%	0.00%	0.00%	0.00%
f_{12}	Mean	4.1875E+01	2.2981E+01	5.1868E+00	1.2155E−01	**4.8811E−02**
	Std.	1.8361E+01	4.6669E+00	1.9884E+00	7.8663E−02	**3.5757E−02**
	Success rate	0.00%	0.00%	0.00%	0.00%	0.00%
f_{13}	Mean	7.1500E+00	6.5637E+00	3.9033E+00	2.1462E+00	1.1696E+00
	Std.	1.2007E+00	8.1399E−01	3.9662E−01	3.1340E−01	3.9617E−01
	Success rate	0.00%	0.00%	0.00%	0.00%	0.00%
f_{14}	Mean	3.7305E+02	3.5977E+02	2.4923E+02	2.3234E+02	2.2825E+02
	Std.	6.5773E+01	4.9571E+01	8.8992E+02	5.6917E+02	1.0678E+02
	Success rate	0.00%	0.00%	0.00%	0.00%	0.00%

Table 9 (*continued*)

Functions	*PMCR*	0.5	0.7	0.9	0.95	0.98
f_{15}	Mean	1.3747E+01	1.1744E+01	9.5362E+00	9.4527E+00	8.8373E+00
	Std.	2.0619E+00	2.0340E+00	1.6446E+00	1.1370E+00	1.0340E+00
	Success rate	0.00%	0.00%	0.00%	0.00%	0.00%
f_{16}	Mean	7.1550E+01	3.7037E+01	1.1639E+01	4.8029E+00	1.0341E+00
	Std.	2.3331E+01	9.7563E+00	8.5812E+00	2.4661E+00	1.6974E−01
	Success rate	0.00%	0.00%	0.00%	0.00%	0.00%
f_{17}	Mean	1.3472E+01	1.2473E+01	1.1747E+01	6.8343E+00	2.5200E+00
	Std.	1.4329E+01	1.3892E+01	5.8581E+00	9.1912E−01	8.3096E−01
	Success rate	0.00%	0.00%	0.00%	0.00%	0.00%
f_{18}	Mean	2.7569E+01	2.4915E+01	2.3916E+01	2.2909E+01	2.2465E+01
	Std.	1.5177E+01	8.5678E+00	2.5056E+00	9.2829E−01	3.5177E−01
	Success rate	0.00%	0.00%	0.00%	0.00%	0.00%

Table 10
The effect of PMS, D=50, NI=10000, PMN=5, PMCR=0.98.

Test function	PMS=2	PMS=5	PMS=10	PMS=20	PMS=30
f_1					
Mean	7.23E−136	1.73E−124	1.33E−108	4.75E−88	3.83E−77
Std.	2.71E−133	5.26E−124	5.36E−106	1.51E−85	1.39E−74
Success rate	0.00%	0.00%	0.00%	0.00%	0.00%
f_2					
Mean	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
Std.	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
Success rate	100%	100%	100%	100%	100%
f_3					
Mean	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
Std.	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00
Success rate	100%	100%	100%	100%	100%
f_4					
Mean	9.85E+00	1.37E+01	1.03E+01	1.36E+01	2.15E+01
Std.	2.53E+01	2.54E+01	2.52E+01	3.64E+01	3.59E+01
Success rate	0.00%	0.00%	0.00%	0.00%	0.00%
f_5					
Mean	5.33E−15	5.33E−15	6.18E−15	6.66E−15	8.36E−15
Std.	2.03E−15	2.03E−15	3.18E−15	2.37E−15	4.61E−15
Success rate	0.00%	3.33%	0.00%	0.00%	0.00%
f_6					
Mean	3.81E−80	8.42E−75	7.65E−67	9.83E−56	1.07E−49
Std.	1.08E−79	3.45E−74	2.47E−66	2.61E−55	4.40E−49
Success rate	6.67%	0.00%	0.00%	0.00%	0.00%
f_7					
Mean	4.09E−02	4.09E−02	4.03E−02	4.14E−02	4.14E−02
Std.	1.39E−02	1.64E−02	1.45E−02	1.31E−02	1.52E−02
Success rate	0.00%	0.00%	0.00%	0.00%	0.00%
f_8					
Mean	1.72E−01	1.36E−01	1.35E−01	1.28E−01	1.14E−01
Std.	8.12E−03	6.74E−02	6.07E−03	6.77E−03	8.52E−03
Success rate	0.00%	0.00%	0.00%	0.00%	0.00%
f_9					
Mean	1.64E+03	1.82E+03	2.11E+03	2.27E+03	2.63E+03
Std.	6.73E+02	5.93E+02	7.69E+02	4.73E+02	7.45E+02
Success rate	0.00%	0.00%	0.00%	0.00%	0.00%
f_{10}					
Mean	2.91E+02	3.08E+02	3.47E+02	3.55E+02	4.63E+02
Std.	9.90E+01	9.94E+01	1.69E+02	1.59E+02	3.18E+02
Success rate	0.00%	0.00%	0.00%	0.00%	0.00%
f_{11}					
Mean	5.53E+00	2.25E+00	2.24E+00	2.60E+00	2.65E+00
Std.	1.76E+00	1.44E+00	1.68E+00	1.68E+00	1.29E+00
Success rate	0.00%	0.00%	0.00%	0.00%	0.00%
f_{12}					
Mean	5.75E−02	4.88E−02	4.64E−02	5.39E−02	5.69E−02
Std.	4.15E−02	3.58E−02	4.56E−02	7.71E−02	5.68E−02
Success rate	0.00%	0.00%	3.33%	0.00%	0.00%

Table 10 (*continued*)

Test function	PMS=2	PMS=5	PMS=10	PMS=20	PMS=30
f_{13}					
Mean	1.07E+00	1.17E+00	1.17E+00	2.36E+00	2.66E+00
Std.	9.87E−02	3.96E−01	5.24E−01	9.13E−01	1.24E+00
Success rate	0.00%	0.00%	0.00%	0.00%	0.00%
f_{14}					
Mean	4.54E+03	2.28E+02	2.29E+02	2.74E+02	2.72E+02
Std.	9.24E+03	1.07E+02	3.62E+02	2.89E+02	2.70E+02
Success rate	0.00%	0.00%	0.00%	0.00%	0.00%
f_{15}					
Mean	1.37E+01	8.84E+00	9.22E+00	1.39E+01	9.49E+00
Std.	5.24E+00	1.03E+00	5.63E−01	1.18E+00	1.12E+00
Success rate	0.00%	0.00%	0.00%	0.00%	0.00%
f_{16}					
Mean	5.24E+00	1.03E+00	1.57E+00	5.24E+00	9.66E+00
Std.	2.13E+00	1.70E−01	9.33E−02	2.37E−01	8.33E−01
Success rate	0.00%	0.00%	0.00%	0.00%	0.00%
f_{17}					
Mean	5.64E+00	2.52E+00	2.67E+00	1.01E+01	5.13E+01
Std.	7.48E+00	8.31E−01	5.04E+00	7.23E+00	4.92E+00
Success rate	0.00%	0.00%	0.00%	0.00%	0.00%
f_{18}					
Mean	2.42E+01	2.25E+01	2.33E+01	2.41E+01	2.45E+01
Std.	1.72E+01	3.52E−01	1.55E+01	1.77E+01	1.71E+01
Success rate	0.00%	0.00%	0.00%	0.00%	0.00%

(*NoFE*). The results generated by using different *PMN* values and a specified number of function evaluations (*NoFE*=50,000) for all test cases are presented in Table 8. In these cases increasing the *PMN* value decreases the total number of iterations (*NI*). Totally it is presumable that, the best results are produced applying medium values (e.g. *PMN*=5) for this parameter. From Tables 7 and 8 it is presumable that the algorithm is not significantly influenced by this parameter for solving f_{17} and f_{18} functions. From the obtained results using medium values (e.g. 5 or 10) for *PMN* parameter is recommended. It can be seen that choosing low values for this parameter (i.e. *PMN*=2) reduces the algorithm performance for all test cases. Actually, since the algorithm simulates the group performances and *PMN* is the number of the group members, using too small *PMN* would not be a good choice.

Table 9 summarizes the calculated results with different values of *PMCR* parameter using *PMN*=5, *PMS*=5 and *NI*=10,000 for all functions with 50 dimensions. The *NII* values are determined by sensitive analysis for each test case. From the calculated results it is revealed that an increase in *PMCR* value enhances the algorithm performance for all test functions except for Sphere and Schwefel 2.22 functions which found the best results with *PMCR*=0.9. Griewank and Rastrigin functions are capable to produce global optimum using *PMCR* values greater than 0.9. Finally using large values (i.e. > 0.9) for *PMCR* is recommended generally.

Table 10 presents the results generated by using different *PMS* values for all test functions with 50 dimensions. For all test cases it is assumed that *NI*=10,000, *PMN*=5, *PMCR*=0.98 and *NII* values are equal to the best values determined by sensitivity analysis. For Shifted Rotated Rastrigin function using *PMS*=2 might not be a good choice. For Rastrigin and Griewank functions, all values of *PMS* result in the global optimum solution. Calculated results of f_4, f_5, f_7, f_8, f_9, f_{10}, f_{11}, f_{12} and f_{18} functions applying different values of *PMS* are in the same order of accuracy. Generally it may be concluded from Table 10 that using middle values of *PMS* (i.e. 5 and 10) is preferred to using other values for all benchmark problems; while reducing space requirements is beneficial. Since *PMS* resembles the capacity of musician's short term memory, it is reasonable to adopt small values for *PMS*, but not too small.

6. Conclusion

This paper introduced a novel improved version of Harmony Search optimization algorithm called Melody Search (MS). Principles of the new algorithm were described and similarities and differences were explained comparing with basic HS. Moreover, a novel alternative improvisation procedure (AIP) was proposed in this study. In order to demonstrate the performance of MS algorithm with the new improvisation scheme (AIP_MS), it was tested using eighteen high dimensional numerical well-known functions. The obtained results were compared with those of basic HS, IHS, GHS, SGHS, NGHS and basic MS algorithms. Based on the experimental results, it is concluded that the proposed algorithm is more effective in finding better solutions comparing with the aforementioned algorithms. Furthermore, the main advantage of AIP_MS algorithm is that the algorithm can better preserve the accuracy of the results comparing with other described methods in the case that the dimensionality of the problem or the entire feasible range of the search space is increased.

Further research on issues, such as the investigation of the control parameters' effects on the performance of MS algorithm and the convergence speed of the algorithm in different conditions, is sought in our future research.

References

Al-Betar, M., Khader, A., Liao, I., 2010. A harmony search with multi-pitch adjusting rate for the university course timetabling. In: Geem, Z.W. (Ed.), Recent Advances in Harmony Search Algorithm. Springer-Verlag, Berlin, Heidelberg, pp 147–161.

Alia, O.M., Mandava, R., 2011. The variants of the harmony search algorithm: an overview. Artif Intell Rev 36, 49–68.

AlRashidi, M.R., El-Hawary, M.E., 2009. A survey of particle swarm optimization applications in electric power systems. IEEE Trans. Evol. Comput. 13 (4), 913–918.

Ashrafi, S.M., Dariane, A.B., 2011. A novel and effective algorithm for numerical optimization: melody search. In: 11th International Conference on Hybrid Intelligent Systems (HIS 2011), Malacca, Malaysia, 5–8 December, pp. 109–114.
Blum, C., 2005. Ant colony optimization: introduction and recent trends. Phys. Life Rev. 2 (4), 353–373.
Chakraborty, P., Roy, G.G., Das, S., Jain, D., Abraham, A., 2009. An improved harmony search algorithm with differential mutation operator. Fundam. Inform. 95, 1–26.
Chandra Mohan, B., Baskaran, R., 2012. A survey: ant colony optimization based recent research and implementation on several engineering domain. Expert Syst. Appl. 39, 4618–4627.
Cheng, R., Genb, M., Tsujimurab, Y., 1999. A tutorial survey of job-shop scheduling problems using genetic algorithms, part II: hybrid genetic search strategies. Comput. Ind.l Eng. 36, 343–364.
Coelho, L.D.S., Mariani, V.C., 2009. An improved harmony search algorithm for power economic load dispatch. Energy Convers. Manage. 50, 2522–2526.
Coello, C.A.C., Lamont, G.B., Veldhuizen, D.A.V., 2007. Evolutionary Algorithms for Solving Multi-objective Problems, second edition Springer.
Digalakis, J.G., Margaritis, K.G., 2002. An experimental study of benchmarking functions for genetic algorithms. Int. J. Comput. Math. 79 (4), 403–416.
Dorigo, M., Blum, C., 2005. Ant colony optimization theory: a survey. Theor. Comput. Sci. 344 (2–3), 243–278.
Dorigo, M., Colorni, A., Maniezzo, V., 1991. Positive feedback as a search strategy. Technical report 91-016. Dipartimento di Elettronica, Politecnico di Milano, Milan, Italy.
Eberhart, R.C., Shi, Y.H., 2001. Particle swarm optimization: developments applications and resources. In Proceedings of the IEEE Congress on Evolutionary Computation, Seoul, Korea, pp. 81–86.
Eberhart, R.C., Shi, Y.H., 2004. Particle swarm optimization. IEEE Trans. Evol. Comput. 8 (3), 201–203. (special issue: guest editorial).
Geem, Z.W., 2000. Optimal design of water distribution networks using harmony search. Ph.D. Thesis. Korea University.
Geem, Z.W., Kim, J., Loganathan, G., 2001. A new heuristic optimization algorithm: harmony search. Simulation 76 (2), 60–68.
Geem, Z.W., Tseng, C.L., Park, Y., 2005. Harmony search for generalized orienteering problem: best touring in china. In: Wang, L., Chen, K., Ong, Y. (Eds.), Advances in Natural Computation. Springer, Berlin, pp 741–750.
Gendreau, M., 2002. Recent advances in tabu search. In: Ribeiro, C.C., Hansen, R. (Eds.), Essays and Surveys in Metaheuristics. Kluwer, Dordrecht, pp. 369–377.
Glover, F., 1977. Heuristic for integer programming using surrogate constraints. Decision Sci. 8 (1), 156–166.
Glover, F., 1989. Tabu search part I. J. Comput. 1, 190–206.
Glover, F., 1990. Tabu search part II. J. Comput. 2, 4–32.
Glover, F., Kochenberger, G.A. (Eds.), 2003. Handbook of Metaheuristics. Kluwer, Boston.
Glover, F., Laguna, M., 1997. Tabu Search. Kluwer Academic Publishers.
Goldberg, D.E., 1989. Genetic Algorithms in Search, Optimization and Machine Learning. Addison Wesley, Boston, MA.
Griinert, T., 2002. Lagrangean tabu search. In: Ribeiro, C.C., Hansen, P. (Eds.), Essays and Surveys in Metaheuristics. Kluwer, Dordrecht, pp. 379–397.
Hertz, A., Taillard, E., De Werra, D., 1997. Tabu Search. In: Aarts, E., Lenstra, J.K. (Eds.), Local Search in Combinatorial Optimization. John Wiley, pp. 121–136.
Holland, J.H., 1975. Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor, MI.
Ingram, G., Zhang, T., 2009. Overview of applications and developments in the harmony search algorithm. In: Geem, Z.W. (Ed.), Music-inspired harmony search algorithm. Springer, Berlin, Heidelberg, pp 15–37.
Karaboga, D., 2005. An idea based on honey bee swarm for numerical optimization. Technical report TR06. Engineering Faculty, Computer Engineering Department, Erciyes University, Turkey.
Karaboga, D., Akay, B., 2009. Artificial bee colony (ABC), harmony search and bees algorithms on numerical optimization. In: Proceedings of the IPROMS 2009 Innovative Production Machines and systems, Virtual Conference, Cardiff, UK.
Karaboga, D., Basturk, B., 2007. A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. J. Glob. Optim. 39, 459–471.
Karaboga, D., Gorkemli, B., Ozturk, C., Karaboga, N., 2012. A comprehensive survey: artificial bee colony (ABC) algorithm and applications. Artif. Intell. Rev. pp. 1–37.
Kattan, A., Abdullah, R., Salam, R.A., 2010. Harmony search based supervised training of artificial neural networks, In: International Conference on Intelligent Systems, Modelling and Simulation (ISMS), pp 105–110.
Kennedy, J., Eberhat, R.C., 1995. Particle swarm optimization. In: Proceedings of IEEE International Conference on Neural Networks, No. IV, IEEE Service Center, Piscataway, NJ, pp. 1942–1948.
Kennedy, J., Eberhart, R.C., Shi, Y.H., 2001. Swarm Intelligence. Morgan Kaufmann, San Mateo, CA.
Kirkpatrick, S., Gelatt, C., Vecchi, M., 1983. Optimization by simulated annealing. Science 220 (4598), 671–680.
Kumar, D.N., Reddy, M.J., 2006. Ant colony optimization for multi-purpose reservoir operation. Water Resour. Manag. 20, 879–898.
Lee, K., Geem, Z., 2005. A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice. Comput. Method Appl. M. 194, 3902–3933.
Liang, J.J., Qin, A.K., Suganthan, P.N., Baskar, S., 2006. Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. IEEE Trans. Evol. Comput. 10 (3), 281–295.
Mahdavi, M., Fesanghary, M., Damangir, E., 2007. An improved harmony search algorithm for solving optimization problems. Appl. Math. Comput. 188, 1567–1579.
Mann, H.B., Whitney, D.R., 1947. On a test of whether one of two random variables is stochastically larger than the other. Ann. Math. Stat. 18, 50–60.
Nicklow, J., Reed, P., Savic, D., Dessalegne, T., et al., 2010. State of the art for genetic algorithms and beyond in water resources planning and management. J. Water Res. Pl.—ASCE , http://dx.doi.org/10.1061/(ASCE)WR.1943-5452.0000053.
Omran, M.G.H., Mahdavi, M., 2008. Global-best harmony search. Appl. Math. Comput. 198, 643–656.
Pan, Q.K., Suganthan, P.N., Liang, J.J., Tasgetiren, M.F., 2010a. A local-best harmony search algorithm with dynamic subpopulations. Eng. Optim. 42 (2), 101–117.
Pan, Q.K., Suganthan, P.N., Tasgetiren, M.F., Liang, J.J., 2010b. A self-adaptive global best harmony search algorithm. Appl. Math. Comput. 216, 830–848.
Pedemonte, M., Nesmachnow, S., Cancela, H., 2011. A survey on parallel ant colony optimization. Appl. Soft Comput. 11, 5181–5197.
Qin, A.K., Huang, V.L., Suganthan, P.N., 2009. Differential evolution algorithm with strategy adaptation for global numerical optimization. IEEE Trans. Evol. Comput. 13 (2), 398–417.
Reyes-Sierra, M., Coello, C.A.C., 2006. Multi-objective particle swarm optimizers: A survey of the state-of-theart. Int. J. Comput. Intell. Res. 2 (3), 287–308.
Schoenberg, A., 1982. Fundamentals of Musical Composition. Faber & Faber Press, London.
Sturman, P., 1983. Harmony, Melody and Composition. Cambridge University Press, England.
Suganthan, P.N., Hansen, N., Liang J.J., et al., 2005. Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization. Technical report. Nanyang Technological University, Singapore, May 2005 (KanGal Report # 2005005, IIT Kanpur, India).
Suman, B., Kumar, P., 2006. A survey of simulated annealing as a tool for single and multiobjective optimization. J. Oper. Res. Soc. 57, 1143–1160.
Wang, C.M., Huang, Y.F., 2010. Selfadaptive harmony search algorithm for optimization. Expert Syst. Appl. 37 (4), 2826–2837.
Zou, D., Gao, L., Wu, J., Li, S., 2010. Novel global harmony search algorithm for unconstrained problems. Neurocomputing 73, 3308–3318.