Engineering with Computers (2022) 38 (Suppl 4):S3539–S3562
https://doi.org/10.1007/s00366-021-01467-8

ORIGINAL ARTICLE

Enhanced harmony search algorithm with non-linear control parameters for global optimization and engineering design problems

Shubham Gupta[1]

Received: 6 March 2021 / Accepted: 30 June 2021 / Published online: 28 July 2021

Abstract
Harmony search algorithm (HSA), inspired by the behaviors of music improvisation process, is a widely used metaheuristic to solve global optimization problems arises in various fields. The reason of its popularity is its simplicity of algorithm structure and good performance. However, the conventional harmony search algorithm (HSA) experiences prone towards the local optima, tedious task of tuning parameters, and premature convergence. To overcome all these drawbacks of conventional HSA and further improve the precision of numerical results, a new variant of the HSA called modified harmony search algorithm (MHSA) is proposed in the present study. This MHSA utilizes the valuable information stored in the harmony memory and modifies the search strategy to make an efficient search procedure by adopting a new formulation to the pitch adjustment process, randomization process, harmony memory considering rate (HMCR), and pitch adjustment rate (PAR). The experimental validation and comparative performance study with conventional HSA, variants of HSA such as adaptive harmony search with best based search strategy (ABHS), enhanced self-adaptive global-best harmony search (ESGHS), novel self-adaptive harmony search (NSHS), parameter adaptive harmony search (PAHS), Gaussian global-best harmony search algorithm (GGHS) and other metaheuristics such as sine cosine algorithm (SCA), grey wolf optimizer (GWO), comprehensive learning particle swarm optimization (CLPSO), gbest-guided artificial bee colony (GABC), and covariance matrix adaptation evolution strategy (CMA-ES) is conducted on a set of 23 well-known benchmark problems. In addition to this benchmarking, the proposed MHSA is also used to solve three structural engineering design problem. The statistical test and convergence behaviour analysis are used to analyze the quality of search and significance of improved accuracy. The comparison illustrates the superior search efficiency of the proposed MHSA than other algorithms as a global optimizer.

Keywords Optimization · Harmony search algorithm · Exploration and exploitation

1 Introduction

Meta-heuristic algorithms (MAs) inspired by the phenomena of biological or physical are trending tools to solve complex optimization problem. Some well-known examples of metaheuristics are genetic algorithm [43], particle swarm optimization [19], ant colony optimization [8], artificial bee colony algorithm [18], and so on. In the field of metaheuristics, the No Free Lunch theorem [44] plays an important role and allows the development of new algorithms by claiming the fact that there does not exist and even not possible to design a single optimizer, which can solve all the optimization problems. Some recently developed but efficient optimization algorithms are grey wolf optimizer (GWO) [34], monarch butterfly optimization (MBO) [41], slime mould algorithm (SMA) [27], moth search algorithm (MSA) [40], hunger games search (HGS) [46], Runge–Kutta optimizer (RUN) [1], and Harris Hawks optimization (HHO) [15].

The concept of evolution in nature has been mimicked by many researchers to develop optimization algorithms, which can solve the real-world problems where the conventional approaches of optimization fail. Harmony search algorithm (HSA) is one of the well-known evolutionary algorithms developed by Geem et al. [11] from the inspiration of music improvisation process. In this algorithm, the population of search agents is referred by harmony memory and candidate solutions are referred by harmony. HSA has very easy implementation process, which involves the memory

✉ Shubham Gupta
shubham.gupta@ntu.edu.sg

[1] School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore 639798, Singapore

consideration, pitch adjustment, and random process of generating new harmony. In the literature, HSA has shown its impressive performance in terms of solution quality and convergence rate on several benchmarks and real-life problems [38, 42]. Among these benefits, HSA is having relatively few and easy mathematical equations, which utilizes all the existing harmonies while producing a new harmony.

However, the HSA confronts several challenging and serious issues. One issue is parameter tuning of its control parameters that remarkably affects its search performance. The second and main issue is its prone towards the local optima for multimodal problems during the search procedure, which is the cause of inappropriate balance between exploitation (intensification) and exploration (diversification). The exploration or diversification is a process of discovering new and promising search regions of the search space, while the exploitation or intensification refers to the process of extracting useful information from the discovered search areas.

To overcome the issues of getting trapped at local optima and to minimize the efforts of tuning the parameters for the search procedure, the present article proposes a comparatively efficient and alternative variant of the HSA, called modified harmony search algorithm (MHSA). In this proposed variant, the non-linear functions are used to update the parameters HMCR and PAR. The parameter HMCR is designed based on the dimension of the problem and the parameter PAR is chosen as a decreasing exponential function over the growth of iterations/function evaluations. In the MHSA, both the pitch adjustment process and process of generating new random memory are modified. The proposed study of generating a harmony, which replaces the concept of bandwidth, discovers the search space from a wider range of search region to a narrower range. The opposition-based learning and random generation using uniform distribution is used to generate a new harmony, when the HMCR disallow to use the harmony memory. This randomization is used to propagate the search in partially opposite regions of the search space. It can be noticed from the framework of the proposed variant of the HSA that it is not destroying the original structure of the algorithm. In this proposal, it has been tried to keep the structure of the algorithm simple because of the fact that the practitioners are not experts in programming and their aim is to apply a simple and efficient algorithm for their optimization purposes [7]. Overall, major contribution of the proposed study can be summarized as follows:

- The modified HSA is proposed by adopting non-linear nature of the parameters HMCR and PAR, which avoids the tedious task of tuning them and maintain the balance between exploitation and exploration during the search process.
- The pitch adjustment process is modified by inspiring the search mechanism of the GWO. This integration effectively improves the harmony by providing a balanced transition from exploration to exploitation.
- In the end, the concept of opposite numbers is used to generate a opposite harmony, which explores broader area of the search space and helps to speed up the convergence rate.

The rest of the paper is organized as follows: Sect. 2 summarizes the conventional HSA and review some important developments of the HSA. In Sect. 3, a new modified variant of the HSA called MHSA is presented. Section 4 analyzes the performance of the proposed MHSA and compares it with developed variants of the HSA and other metaheuristic optimization algorithms. This section also analyzes the convergence behaviour of the MHSA and sensibility to the harmony memory size. Furthermore, some real structural engineering design problems are also solved using the proposed MHSA. Finally, the present work concludes in Sect. 5 with some future research directions.

2 Preliminaries

2.1 Harmony search algorithm

In this section, the basic version of the HS algorithm is introduced. Readers may review the more details of the algorithm from Geem et al. [11]. Similar to other evolutionary algorithms, the HS algorithm is also a population-based stochastic algorithm, which involves a simple strategy of evolving the candidate solution. Its search procedure starts with an initialization of harmony memory (HM) using the search Eq. (1)

$$\mathrm{HM}_{i,j} = \mathrm{HM}_j^{\min} + \mathrm{rand}(0,1) \times (\mathrm{HM}_j^{\max} - \mathrm{HM}_j^{\min}). \tag{1}$$

Here, rand(0, 1) is a uniformly distributed random number from the interval (0, 1), and $\mathrm{HM}_j^{\max}$ and $\mathrm{HM}_j^{\min}$ are the upper and lower boundary limits for the jth component of harmony memory vector. The index i runs over the size of the harmony (HMS), and j runs over the dimension (d) of the problem or number of components in any harmony, i.e., $i \in 1, 2, \ldots, \mathrm{HMS}$ and $j \in 1, 2, \ldots, d$.

After the initialization of harmony, the HS executes its search procedure under iterative process. The search operators, which play a major role in search procedure, are the memory consideration, pitch adjustment, and random process of generating the harmony. In each iteration of the HSA, a new harmony is generated by performing these three operations. This newly generated harmony replaces the worst harmony of the memory if its fitness is better than

that fitness; otherwise, it is discarded. This search procedure of generating the new harmony is repeated iteratively until the termination criteria is not met or maximum number of iterations are not reached.

During the generation of new harmony, the memory consideration operation is executed with a probability of harmony memory considering (accepting) rate (HMCR), while the random process of generating the harmony has the probability (1 − HMCR). After the harmony memory consideration process, the pitch adjustment process is performed with probability called pitch adjustment rate (PAR). In detail, these processes are described as follows:

In the memory consideration process, a random harmony from the current harmony memory is selected as follows:

$$\mathrm{HM}_{\mathrm{rand},j} = \mathrm{HM}_{r_1,j}, \quad j = 1, 2, \ldots, d, \tag{2}$$

where r_1 is an integer selected randomly from [1, HMS]. During the process of pitch adjustment, the randomly selected harmony component is adjusted as follows:

$$u_j = \mathrm{HM}_{\mathrm{rand},j} + \mathrm{rand}(-1, 1) \times \mathrm{BW}, \quad j = 1, 2, \ldots, d, \tag{3}$$

where the variable BW is known as bandwidth and it determines the step size taken during the search procedure, and rand(−1, 1) is a variable drawn randomly form the interval (−1, 1).

In the random process of generating the harmony, a new harmony is generated randomly within the search space as follows:

$$u_j = u_j^{\min} + \mathrm{rand}(0, 1) \times (u_j^{\max} - u_j^{\min}), \tag{4}$$

where $u_j^{\min}$ and $u_j^{\max}$ are the allowed lower and upper limits for the jth component of the harmony u.

The pseudo-code of the conventional HSA based on the above description is provided in Algorithm 1.

Algorithm 1 Pseudo-code of conventional HSA

Inputs: The harmony size HMS, maximum function evaluations FEV, harmony memory considering rate $HMCR$ and pitch adjustment rate PAR
Output: The best memory HM_{best}
Initialize the harmony memory $HM_i (i = 1, 2, \ldots, HMS)$ randomly and within the provided search space
while $(n < FEV)$ **do**
  for $j = 1, 2, \ldots d$ **do**
    if $rand() < HMCR$ **then**
      $HM_{rand} = HM_{r_1,j}$
      if $rand() < PAR$ **then**
        $u_j = HM_{rand} + rand(-1, 1) \times BW$
    else
      $u_j = u_j^{min} + rand(0, 1) \times (u_j^{max} - u_j^{min})$
  Evaluate the fitness of harmony u
  $n = n + 1$
  if $f(u) \leq f(HM_{worst})$ **then**
    $HM_{worst} = u$
  Update the best harmony HM_{best}
Return the best harmony HM_{best}

2.2 Previous work

The HSA has gained wide attention by the researchers due to its simplicity, faster computation, and efficiency. This algorithm has been used widely in various fields [2, 5, 37]. However, similar to other metaheuristics, the HSA also feels the problem of getting trapped at local optima during the search procedure, and therefore, many researchers have attempted to improve its search efficiency.

2.2.1 Fine tuning of control parameters

It was clearly explain by the developers of the HSA [11] that the parameter HMCR supports to the exploration and the parameters PAR and BW helps in exploiting the search

space. Therefore, these parameters are crucial and responsible for the better performance of the HSA. Mahdavi et al. [32] have modified the search mechanism of the HSA by setting up a new formulation of the parameters PAR and BW given by

$$\mathrm{PAR}_t = \mathrm{PAR}^{\min} + \frac{t}{T} \times (\mathrm{PAR}^{\max} - \mathrm{PAR}^{\min}), \tag{5}$$

$$\mathrm{BW}_t = \mathrm{BW}^{\max} \times e^{\frac{t}{T} \ln\left(\frac{\mathrm{BW}^{\min}}{\mathrm{BW}^{\max}}\right)}, \tag{6}$$

where t indicate the current iteration and T is the maximum number of iterations. $\mathrm{PAR}^{\min}$ and $\mathrm{PAR}^{\max}$ are the minimum and maximum values of the parameter PAR, and $\mathrm{BW}^{\min}$ and $\mathrm{BW}^{\max}$ are the minimum and maximum values for the parameter BW, respectively. Kumar et al. [26] have introduced both linear as well as non-linear settings for the parameters HMCR and PAR as follows:

$$\mathrm{HMCR}_t^{\text{linear}} = \mathrm{HMCR}^{\min} + \frac{t}{T} \times (\mathrm{HMCR}^{\max} - \mathrm{HMCR}^{\min}), \tag{7}$$

$$\mathrm{HMCR}_t^{\text{non-linear}} = \mathrm{HMCR}^{\min} \times e^{-\frac{t}{T} \times \ln\left(\frac{\mathrm{HMCR}^{\min}}{\mathrm{HMCR}^{\max}}\right)}, \tag{8}$$

$$\mathrm{PAR}_t^{\text{linear}} = \mathrm{PAR}^{\min} + \frac{T-t}{T} \times (\mathrm{PAR}^{\max} - \mathrm{PAR}^{\min}), \tag{9}$$

$$\mathrm{PAR}_t^{\text{non-linear}} = \mathrm{PAR}^{\min} \times e^{\frac{t}{T} \times \ln\left(\frac{\mathrm{PAR}^{\min}}{\mathrm{PAR}^{\max}}\right)}, \tag{10}$$

where $\mathrm{PHMCR}^{\min}$ and $\mathrm{HMCR}^{\max}$ are the minimum and maximum values of the parameter HMCR. In Khalili et al. [25], the parameter HMCR is updated using Eq. (7)

$$\mathrm{HMCR}_t = 0.9 + 0.2\sqrt{\frac{t-1}{t-1} \times \left(1 - \frac{t-1}{T-1}\right)}. \tag{11}$$

Several other researchers also tried to improve the search performance of the HSA by fine-tuning these control parameters [3, 14, 16, 18–23, 29, 36]. Luo et al. [30] have tried to modify the search strategy of the HSA by introduced an modified variant of the HSA, where the self-adaptive and parameter-free approaches are used to fine-tune the parameters HMCR and PAR. To improvise the harmony in pitch adjustment process, the bandwidth parameter has been replaced by additional term called exponential term with some constant factor. The random process of generating the harmony is also modified and the Gaussian distribution is used to generate a new harmony component. This work motivates us to develop a new and alternative variant of the HSA which is more efficient in determining the better solution quality, convergence rate and sufficient enough to avoid the prone of solutions towards the local optima.

2.2.2 Modifying the pitch adjustment operation

To improvise the harmony, several variants by modifying the control parameter BW are proposed in the literature, which show the improvement on global optimization problems. Chakraborty et al. [6] have introduced a mutation strategy, which is used in the DE algorithm, during the pitch adjustment operation. In this, harmony is improvised using the following equation:

$$u_j = \mathrm{HM}_{r_1,j} + \mathrm{rand}(0,1) \times (\mathrm{HM}_{r_2,j} - \mathrm{HM}_{r_3,j}), \tag{12}$$

where $\mathrm{HM}_{r_1,j}$, $\mathrm{HM}_{r_1,j}$, and $\mathrm{HM}_{r_3,j}$ are the jth components of the randomly selected but different harmonies from the current harmony memory. Although, this strategy enhances the exploration ability, but sometimes, this leads to the stagnation and long perturbation. By inspiring this mutation scheme, Guo et al. [12] have introduced DE/Best/1 scheme, given by Eq. (13), to improvise the harmony

$$u_j = \mathrm{HM}_{\mathrm{best},j} + \mathrm{rand}(0,1) \times (\mathrm{HM}_{r_2,j} - \mathrm{HM}_{r_3,j}), \tag{13}$$

where $\mathrm{HM}_{\mathrm{best},j}$ is the jth component of the best harmony from the current harmony memory. This adjustment weakens the diversification of search space when the perturbation is very low or the algorithm stagnate at local optima. In some cases, this greedy direction of improvisation also leads to the premature convergence. El-Abd [9] has proposed a new adjustment to improvise the harmony. In his scheme, a transition from the exploration to exploitation is tried to maintain by improvising the harmony initially around the random harmony and later around the best harmony. The proposed scheme is explained by Eqs. (14) and (15)

$$u_j = \mathrm{HM}_{\mathrm{rand},j} + \mathrm{Gaussian}(0,1) \times \mathrm{BW}, \tag{14}$$

$$u_j = \mathrm{HM}_{\mathrm{best},j} + \mathrm{rand}(-1,1) \times \mathrm{BW}, \tag{15}$$

where Gaussian(0, 1) is Gaussian distributed random number with mean 0 and variance 1. rand(−1, 1) is a uniformly distributed random number from the interval (−1, 1). In this algorithm, parameter PAR has been linearly decreased and the parameter BW is exponentially decreased over the course of iterations of the search procedure. In Zou et al. [48], an improved HSA is proposed by inspiring the PSO mechanism. The parameters PAR and HMCR are excluded in this variant and a genetic mutation probability (p_m) is introduced. To improvise the harmony, the global best and the worst harmony are utilized with the help of Eq. (16)

$$u_j = \mathrm{HM}_{\mathrm{worst},j} + \mathrm{rand}(0,1) \times (\mathrm{HM}_{R,j} - \mathrm{HM}_{\mathrm{worst},j}), \tag{16}$$

where $\mathrm{HM}_{R,j} = (2\mathrm{HM}_{\mathrm{best},j} - \mathrm{HM}_{\mathrm{worst},j})$.

In Wang and Huang [39], a self-adaptive approach is proposed to modify the pitch adjustment process. In this

approach, the bandwidth parameter is modified with the help of maximum and minimum values, namely HM_j^{max} and HM_j^{min} of the harmony variables. This modification is presented by Eqs. (17) and (18)

$$u_j = HM_{rand,j} + rand(0,1) \times (HM_j^{max} - HM_{rand,j}), \tag{17}$$

$$u_j = HM_{rand,j} - rand(0,1) \times (HM_{rand,j} - HM_j^{min}). \tag{18}$$

Gao et al. [10] have used the concept of opposition-based leaning at the initialization of the harmony to reach the far points of the solution space, which may have a greater fitness. The parameter bandwidth is also modified in this variant based on the following equation:

$$BW_j = \sqrt{\gamma \cdot \bar{x_j}}, \tag{19}$$

where $\bar{x_j} = \frac{1}{HMS} \sum_{k=1}^{HMS} x_{k,j}$.

In addition to these variants, some other variants of the HSA are also proposed in the literature, where not only the algorithm parameters, but the search mechanism is also modified. For example, Nehdi et al. [35] have introduced a new variant of HSA called dynamical self-adjusted harmony search optimization (DSAHS) to dynamically adopt the algorithm parameters such as HMCR, PAR, and BW as follows:

$$HMCR_t = 0.95 + 0.1 \times \beta_t \sqrt{\frac{t}{T}}, \tag{20}$$

$$PAR_t = 0.5 + 0.4 \times (1 - \beta_t), \tag{21}$$

$$BW_{j,t} = \frac{HM_j^M - HM_j^m + 0.001}{10} \times \exp\left(-10\frac{t}{T}\right), \tag{22}$$

where HM_j^M and HM_j^m are maximum and minimum unknown coefficients. The value of β_t can be determined as follows:

$$\beta_t = \sqrt{1 - \frac{t}{T}}. \tag{23}$$

Based on the above modified parameters, the new harmony memory for unknown coefficients is determined as follows:

$$\hat{HM}_{i,j} = \begin{cases} HM_{i,j} + (2 \times rand(0,1) - 1) \times \beta_t \times (HM^M - HM^m) & \text{with probability } HMCR_t \\ HM_j^{min} + rand(0,1) \times (HM_j^{max} - HM_j^{min}) & \text{with probability } (1 - HMCR_t). \end{cases} \tag{24}$$

After that, the new harmony is adjusted using the following equation:

$$\hat{HM}_{i,j} = \begin{cases} \hat{HM}_{i,j} + (2 \times rand(0,1) - 1) \times \beta_t \times BW_{j,t} & \text{with probability } PAR_t \\ \hat{HM}_{i,j} & \text{with probability } (1 - PAR_t). \end{cases} \tag{25}$$

In Keshtegar and Sadeq [24], Gaussian global-best harmony search algorithm (GGHS) is introduced to deal with complex optimization problems. This algorithm is an enhanced version of the El-Abd [9], where the Gaussian distributed random numbers are used to update the harmony. In the GGHS, the harmony is updated in two stages. In the first stage, the new harmony is obtained using the following equations:

$$HM_{i,j}^{new} = \begin{cases} HM_{i,j} + Gaussian(0,1) \times BW_{j,t} & \text{with probability } HMCR_t \\ HM_j^{min} + rand(0,1) \times (HM_j^{max} - HM_j^{min}) & \text{with probability } (1 - HMCR_t). \end{cases} \tag{26}$$

where Gaussian(0, 1) is a Gaussian distributed random number with mean 0 and variance 1. The parameter bandwidth $BW_{j,t}$ is updated as follows:

$$BW_{j,t} = \frac{|HM_j^M - HM_j^m + 0.0001|}{10} \times \exp\left(-\frac{10t}{T}\right), \tag{27}$$

where HM_j^M and HM_j^m are the maximum and minimum values of memory component HM_j. In the second stage of harmony update process, the best harmony is adjusted to obtain new harmonies as follows:

$$HM_{i,j}^{new} = \begin{cases} HM_{best,j} + \gamma_t \times Gaussian(0, \beta_j) & \text{with probability } PAR_t \\ HM_{i,j} & \text{with probability } (1 - PAR_t). \end{cases} \tag{28}$$

where $Gaussian(0, \beta_j)$ is a Gaussian distributed random number with mean 0 and variance β_j. The value of β_j and γ_t is obtained as follows:

$$\beta_j = \delta \times \left(1 - \frac{t}{T}\right)^{\delta}, \tag{29}$$

$$\delta = |\mathrm{HM}_j^M - \mathrm{HM}_j^m + 0.0001|, \tag{30}$$

$$\gamma_t = \sqrt{1 - \frac{t}{T}}. \tag{31}$$

By analyzing all these variants of the HSA, an alternative variant of the HSA has been proposed in the present study with an aim of achieving better quality of transition from the exploration to the exploitation with less number of parameters. In our approach, the parameter tuning of the control parameters of HSA is not needed by the users except the step-size control parameter of newly proposed pitch adjustment scheme. However, in our experiments, it has been tried to provide better transition scheme, so that no extra efforts have to be performed by the user. Hence, this makes the algorithm very convenient to use for optimization purpose.

3 Proposed modified harmony search algorithm (MHSA)

The search strategy of the conventional HSA is affected by the three process, namely, harmony memory consideration, pitch adjustment, and random generation of new harmony. In our proposal, each process has been modified by either making them parameter independent or providing a new efficient search procedure.

3.1 Parameter setting free control parameters

First, a parameter harmony memory considering rate (HMCR) has been adopted by a normal random number, which is mathematically stated in Eq. (32) [30]

$$\mathrm{HMCR}_t = N\left(\frac{d}{1+d}, \frac{1}{1+d}\right), \tag{32}$$

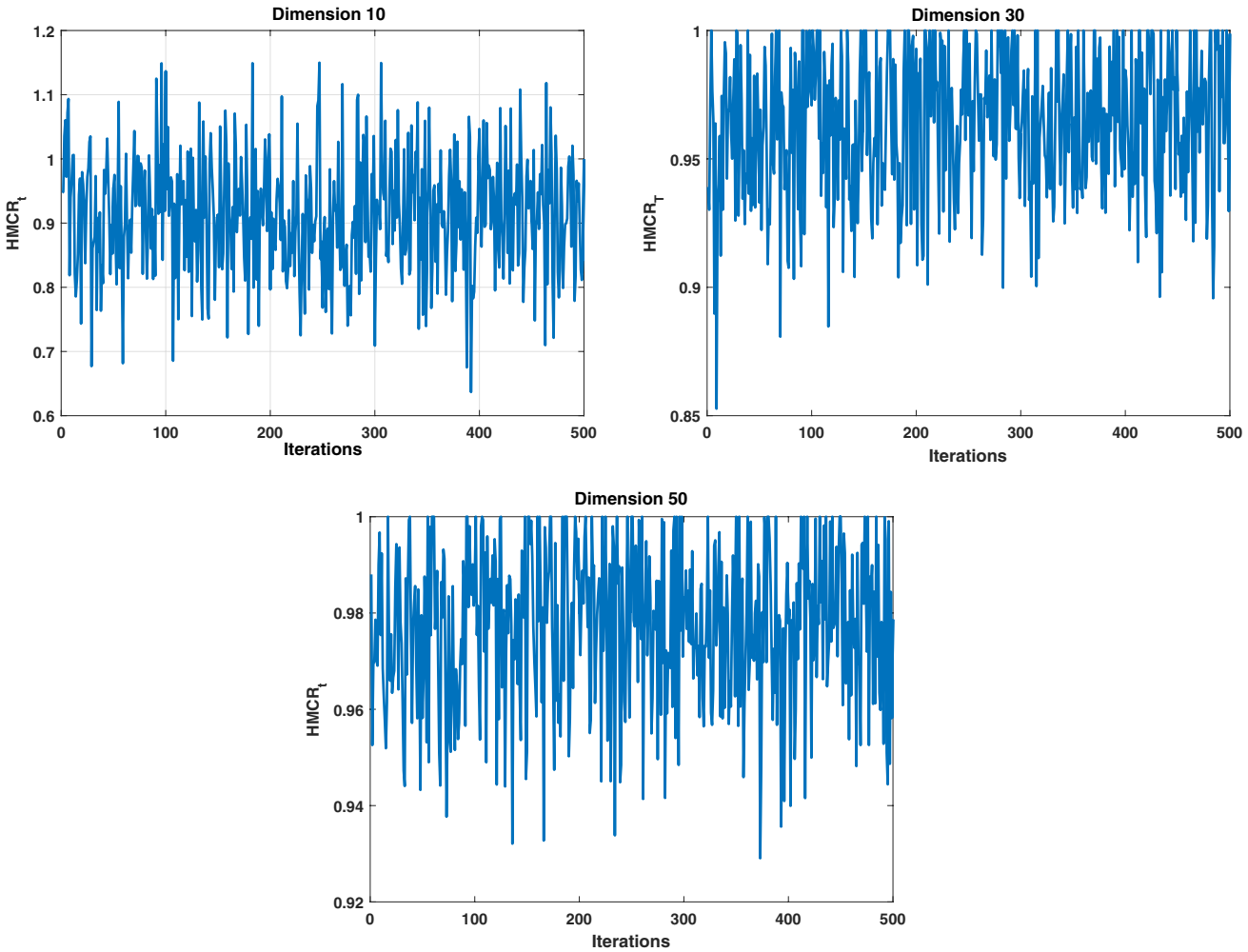

Fig. 1 Distribution of the parameter HMCR for the dimension 10, 30, and 50, respectively

where $N(\mu, \sigma^2)$ indicate the Gaussian distribution with mean μ and variance σ. During the search procedure, if the value of the HMCR exceeds the range [0, 1], it should be truncated. The dynamic change can be visualized in Fig. 1 for different iterations and for the dimension 10, 30, and 50. In this figure, the sampling values of the parameter HMCR are shown. From this figures, it can be seen that when the dimension increases, the values of the parameter approach to 1. One of the main advantages of this setting is that it avoids the burden of tuning parameter HMCR. It also follows the suggestion of bigger values of this parameter, so that the chance of getting a good harmony from the memory by improvisation. On the other hand, this parameter also allows the random generation of harmony by the occasional exceeding of the uniform distributed random number from the parameter value of HMCR.

In the second modification of the MHSA, a parameter PAR is modified using a non-linear decreasing exponential function, which is given by

$$\mathrm{PAR}_t = \exp\left(\frac{-t^2}{(k \cdot T)^2}\right), \tag{33}$$

where k is parameter that decides that how much iterations are devoted to the exploration and how much for the exploitation. In our algorithm, we have fixed this to 0.6 to perform exploration and exploitation equally. This newly proposed parameter can be visualized in Fig. 2, where this has been compared with linear adaptation. This non-linear PAR does not change its value suddenly as compared to linear one and allows slow rate of change to simulate the non-linear process of search in the MHSA. In the first half iterations of the MHSA, the value of PAR is higher than the linear PAR, which allows comparatively better diversification in the MHSA. After the half of the iterations, the value of the PAR is lower than the linear one, which allows more intensification of the discovered promising harmonies. Moreover, at the end of the maximum number of iterations, the value of the proposed PAR is not approaching to zero as compared to the linear PAR and this selection allows to the improvisation of the harmony, when the HMCR allows.

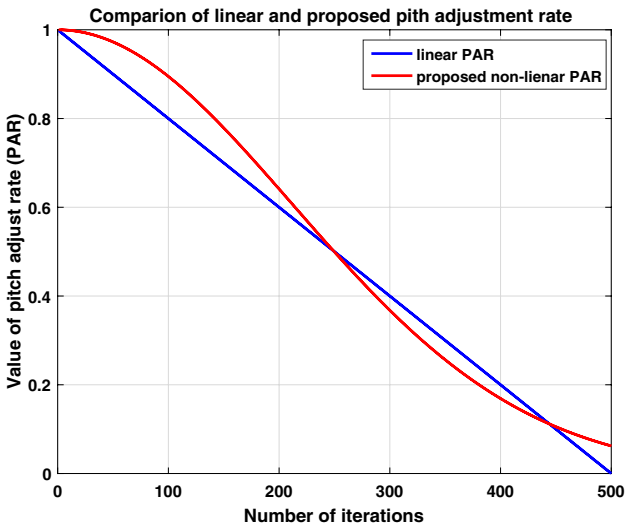

Fig. 2 Proposed non-linear PAR

These newly proposed parameter values for the HMCR and PAR are decreased over the course of iterations, which follows the realistic nature of the metaheuristic algorithms.

3.2 Modification in the pitch adjustment process and random generation of harmony

In the third modification of the MHSA, the pitch adjustment process is modified, which can be demonstrated by Eq. (34)

$$u_j = \mathrm{HM}_{\mathrm{rand},j} + A_j \times |C_j \cdot \mathrm{HM}_{\mathrm{best},j} - \mathrm{HM}_{\mathrm{rand},j}|, \tag{34}$$

where $\mathrm{HM}_{\mathrm{rand},j}$ is the jth component of the harmony selected by the harmony memory consideration process. The scalars A_j and C_j are defined as follows:

$$A_j = 2 \cdot \alpha \cdot \mathrm{rand}(0, 1) - \alpha, \tag{35}$$

$$C_j = 2 \cdot \mathrm{rand}(0, 1), \tag{36}$$

where rand(0, 1) is a random number selected from the interval (0,1) and α is a parameter which decreases linearly to provide an appropriate transition from the exploration to the exploitation. Beside this parameter, the coefficient C_j also provides an exploration and exploitation during the search. One of the main advantages of this C_j is that it provides a diversification of the search space even when the parameter A_j fails. This new equation of pitch adjustment process is inspired by the encircling behaviour organized by the grey wolves in nature [34]. This equation has shown its outperform ability of search to explore, exploit, and in maintaining an appropriate balance between them.

In the fourth and last modification, the opposite numbers are used to generate an opposite harmony, which helps in generating a new random harmony. The advantage of this opposite harmony is to perform the search far from the current search region and to discover more promising regions in opposite directions of the current harmonies. In this randomization, a new harmony is generated based on the hybridization of opposite harmonies and the conventional process of the HSA. This can be understood as follows:

$$u_j = \begin{cases} u_j^{\min} + u_j^{\max} - u_{\mathrm{rand},j} & \mathrm{rand}(0, 1) < 0.5 \\ u_j^{\min} + \mathrm{rand}(0, 1) \times (u_j^{\max} - u_j^{\min}) & \text{otherwise,} \end{cases} \tag{37}$$

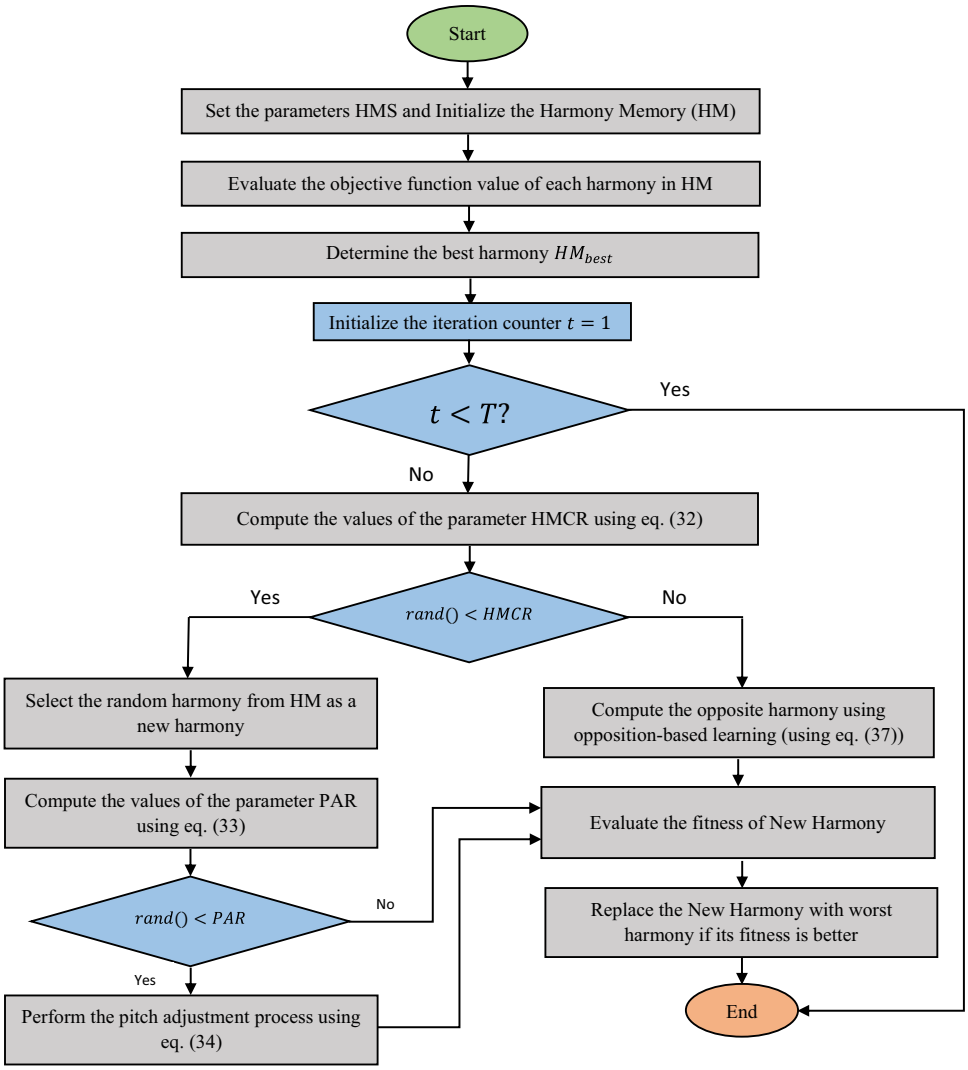

Fig. 3 Flowchart of the proposed MHSA

where $u_j^{\max}$ and $u_j^{\min}$ are the allowed upper and lower bounds for the jth component of the harmony u, and $u_{\text{rand},j}$ is a harmony component of a random harmony selected from current harmony memory.

In this way, the proposed MHSA updates the harmony. First, it initializes the harmony memory and parameters, and then repeats the process of improving the harmony and memorizing the best harmony until the maximum number of iterations are not reached or the termination criteria are not fulfilled. The complete search procedure of the proposed MHSA can be understood by Algorithm 2. The flowchart for the proposed MHSA is provided in Fig. 3.

From this pseudo-code, the complexity of the proposed MHSA can be calculated easily. The complexity of the improvisation phase is $O(d)$ and for the updating process it is $O(\text{HMS})$. Hence, the overall complexity is equal to $O\big((d + \text{HMS}) \cdot T\big)$. This complexity of the MHSA is same as the complexity of the conventional HSA, because this does not needed ant extra or complicated process during the search.

Algorithm 2 Pseudo-code of the proposed MHSA

Inputs: The harmony size HMS, maximum function evaluations FEV.
Outputs: The best memory HM_{best}
Initialize the harmony memory $HMi(i = 1, 2, \ldots, HMS)$ randomly and within the provided search space
while $(n < FEV)$ **do**
  Set the value of parameters $HMCR$ and PAR by equations (32) and (33), respectively.
  for $j = 1, 2, \ldots d$ **do**
    if $rand() < HMCR$ **then**
      $HM_{rand} = HM_{r_1,j}$
      if $rand() < PAR$ **then**
        $u^j = HM^j_{rand} + A^j \times |C_j \cdot HM_{best} - HM_{rand}|$
    else
      select a random number k uniformly from the interval (0,1)
      if $k < 0.5$ **then**
        $u_j = u^{min}_j + u^{max}_j - u_j$
      else
        $u_j = u^{min}_j + rand(0, 1) \times (u^{max}_j - u^{min}_j)$
  evaluate the fitness of harmony u
  $n = n + 1$
  if $f(u) \leq f(HM_{worst})$ **then**
    $HM_{worst} = u$
  update the best harmony HM_{best}
Return the best harmony HM_{best}

Table 1 Unimodal benchmark functions

Function name	Formula	Dim	Search range	$f_{\min}$
Sphere	$F1(x) = \sum_{i=1}^{d} x_i^2$	10, 30, 50	[−100, 100]	0
Sum squares	$F2(x) = \sum_{i=1}^{d} ix_i^2$	10, 30, 50	[−10, 10]	0
Schwefel's 2.22	$F3(x) = \sum_{i=1}^{d} \lvert x_i\rvert + \prod_{i=1}^{d} \lvert x_i\rvert$	10, 30, 50	[−10, 10]	0
Rotated hyper-ellipsoid	$F4(x) = \sum_{i=1}^{d} \left(\sum_{j-1}^{i} x_j\right)^2$	10, 30, 50	[−100, 100]	0
Schewefel 2.21	$F5(x) = \max_i \left\{\lvert x_i\rvert, 1 \leq i \leq d\right\}$	10, 30, 50	[−100, 100]	0
Rosenbrock	$F6(x) = \sum_{i=1}^{d-1} \left[100\left(x_{i+1} - x_i^2\right)^2 + \left(x_i - 1\right)^2\right]$	10, 30, 50	[−30, 30]	0
Step	$F7(x) = \sum_{i=1}^{d} \left(\left[x_i + 0.5\right]\right)^2$	10, 30, 50	[−100, 100]	0
Quartic	$F8(x) = \sum_{i=1}^{d} ix_i^4$	10, 30, 50	[−1.28, 1.28]	0
Noise	$F9(x) = \sum_{i=1}^{d} ix_i^4 + \text{random}[0, 1)$	10, 30, 50	[−1.28, 1.28]	0
Sum-power	$F10(x) = \sum_{i=1}^{d} \lvert x_i\rvert^{i+1}$	10, 30, 50	[−1, 1]	0

Table 2 Multimodal benchmark functions

Function name	Formula	Dim	Search range	$f_{\min}$
Schwefel 2.26	$F11(x) = \sum_{i=1}^{d} -x_i \sin\left(\sqrt{\lvert x_i \rvert}\right)$	10, 30, 50	$[-500, 500]$	$-418.9829 \times d$
Rastrigin	$F12(x) = \sum_{i=1}^{d} \left[x_i^2 - 10\cos\left(2\pi x_i\right) + 10\right]$	10, 30, 50	$[-5.12, 5.12]$	0
Ackley	$F13(x) = -20\exp\left(-0.2\sqrt{\frac{1}{d}\sum_{i=1}^{d} x_i^2}\right) - \exp\left(\frac{1}{d}\sum_{i=1}^{d}\cos\left(2\pi x_i\right)\right) + 20 + e$	10, 30, 50	$[-32, 32]$	0
Griewank	$F14(x) = \frac{1}{4000}\sum_{i=1}^{d} x_i^2 - \prod_{i=1}^{d}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	10, 30, 50	$[-600, 600]$	0
Penalized	$F15(x) = \frac{\pi}{d}\left\{10\sin\left(\pi y_1\right) + \sum_{i=1}^{d-1}(y_i - 1)^2\left[1 + 10\sin^2(\pi y_{i+1})\right] + (y_d - 1)^2\right\} + \sum_{i=1}^{n} u(x_i, 10, 100, 4)$, where $y_i = 1 + \frac{x_i + 1}{4}$ and $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & x_i > a \\ 0 - a & < x_i < a \\ k(-x_i - a)^m & x_i < -a \end{cases}$	10, 30, 50	$[-50, 50]$	0
Penalized 2	$F16(x) = 0.1\left\{\sin^2(3\pi x_1) + \sum_{i=1}^{d}\left(x_i - 1\right)^2\left[1 + \sin^2(3\pi x_i + 1)\right] + \right\} + 0.1\left\{(x_d - 1)^2\left[1 + \sin^2(2\pi x_d)\right]\right\} + \sum_{i=1}^{d} u(x_i, 5, 100, 4)$	10, 30, 50	$[-50, 50]$	0
Alpine	$F17(x) = \sum_{i=1}^{d} \lvert x_i \sin(x_i) + 0.1x_i \rvert$	10, 30, 50	$[-10, 10]$	0
Inverted cosine mixture	$F18(x) = \sum_{i=1}^{d} 0.1n - \left(0.1\sum_{i=1}^{d}\cos(5\pi x_i) - \sum_{i=1}^{d} x_i^2\right)$	10, 30, 50	$[-1, 1]$	0
Stretched V-sine	$F19(x) = \sum_{i=1}^{d-1}(x_i^2 + 2x_{i+1}^2)^{0.25} \times [1 + \sin(50(x_i^2 + x_{i+1}^2)^{0.1})^2$	10, 30, 50	$[-10, 10]$	0
Elliptic	$F20(x) = \sum_{i=1}^{d}(10^6)^{(i-1)/(d-1)} x_i^2$	10, 30, 50	$[-100, 100]$	0
Easom	$F21(x) = (-1)^{d+1}\prod_{i=1}^{d}\cos(x_i) \times \exp(-\sum_{i=1}^{d}(x_i - \pi)^2)$	10, 30, 50	$[-100, 100]$	0
Salomon	$F22(x) = 1 - \cos\left(2\pi\sqrt{(\sum_{i=1}^{d} x_i^2)}\right) + 0.1\sqrt{(\sum_{i=1}^{d} x_i^2)}$	10, 30, 50	$[-100, 100]$	0
Schafer	$F23(x) = 0.5 + \frac{\sin^2\left(\sqrt{(\sum_{i=1}^{d} x_i^2)}\right) - 0.5}{1 + 0.001\left(\sum_{i=1}^{d} x_i^2\right)^2}$	10, 30, 50	$[-100, 100]$	0

4 Experimental results

In this section, the performance of the proposed MHSA is evaluated and analyzed on a set of 23 benchmark test functions, which are unimodal and multimodal in nature. This variety of difficulty level will help to analyze the exploration and exploitation abilities of the proposed algorithm. The list of benchmark test problems is presented in Tables 1 and 2 with the search range and optimal solution. In this study, the performance comparison of the MHSA is performed with the conventional HSA, variants of MHSA, and some other algorithms. Hence, the comparison section is divided into two parts. In the first part, the MHSA is compared with conventional HSA on 10-, 30-, and 50-dimensional problems. In the second part, the variants of the HSA which are developed in the literature and other algorithms are compared with the MHSA.

4.1 Comparison of the MHSA with conventional HSA

This section compares the results of the MHSA with the conventional HSA on 10-, 30-, and 50-dimensional problems, which are given in Tables 1 and 2. The results are obtained by conducting 30 independent trials of each algorithm with $10^4 \times d$ function evaluations. Size of the harmony memory is fixed to 5 for the proposed MHSA and conventional HSA. In this experiment, the mean and standard deviation value of the set of objective function values obtained over 30 runs are

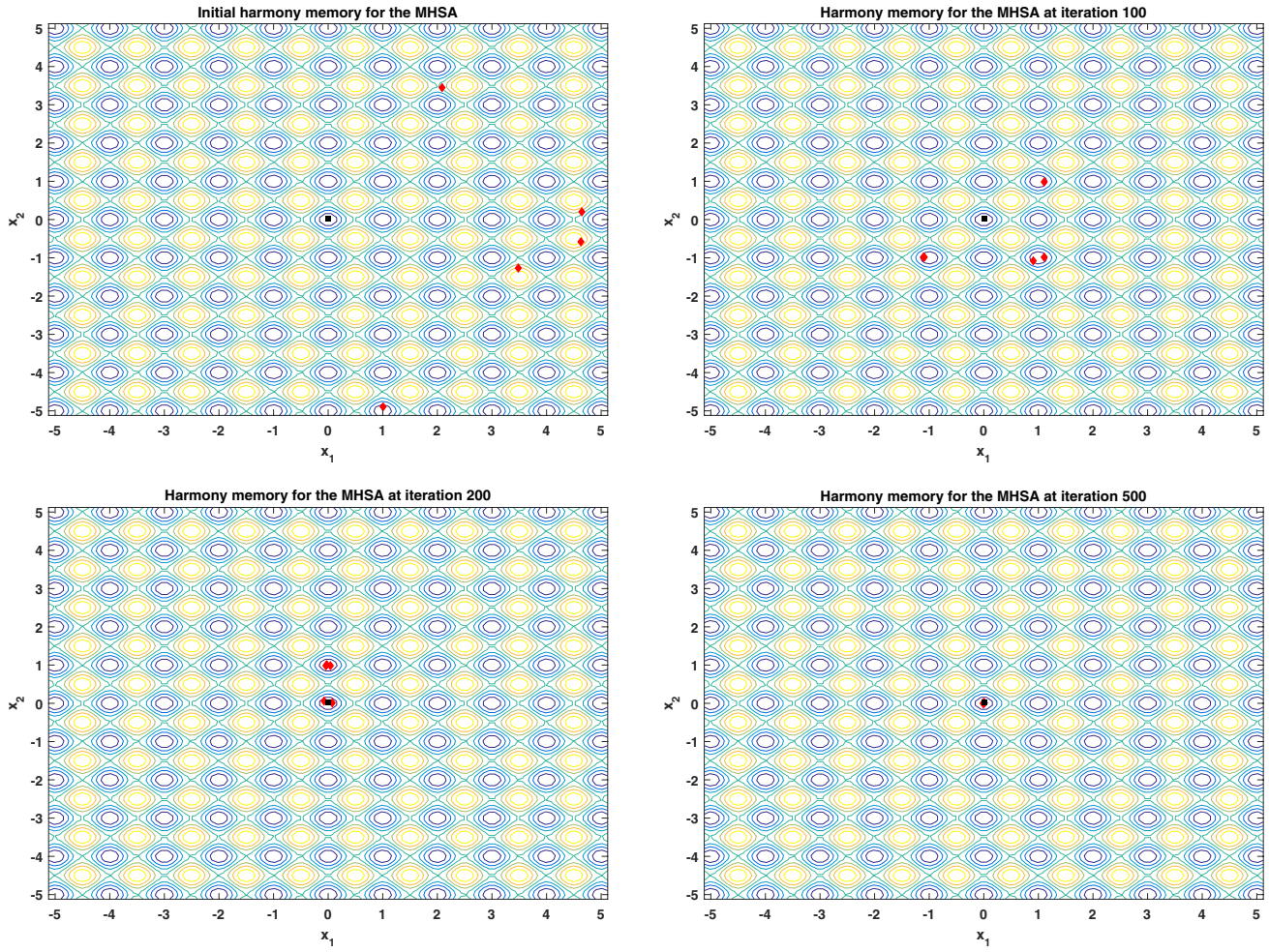

Fig. 4 Search history of the proposed MHSA, while solving the Rastrigin problem (optima : ■, HM : ◊)

Table 3 Comparison of results on 10, 30 and 50-dimensional unimodal benchmark problems

Test function	Result	Dim 10			Dim 30			Dim 50		
		HSA	MHSA	Stat. out.	HSA	MHSA	Stat. out.	HSA	MHSA	Stat. out.
F1	Mean	1.43E−08	**0**	1.21E−12	8.28E−05	**0**	1.21E−12	8.17E+00	**0**	1.212E−12
	Std	9.02E−09	**0**	+	1.48E−05	**0**	+	2.84E+00	**0**	+
F2	Mean	6.46E−08	**0**	1.21E−12	9.14E−04	**0**	1.21E−12	2.89E−02	**0**	1.212E−12
	Std	3.87E−08	**0**	+	1.77E−04	**0**	+	5.83E−03	**0**	+
F3	Mean	2.59E−04	**0**	1.21E−12	2.86E−02	**0**	1.21E−12	2.07E−01	**0**	1.212E−12
	Std	7.27E−05	**0**	+	3.00E−03	**0**	+	7.14E−02	**0**	+
F4	Mean	1.08E+00	**1.75E−203**	3.02E−11	9.06E+02	**5.61E−62**	3.02E−11	7.13E+03	**1.54E−29**	3.02E−11
	Std	2.49E+00	**0**	+	3.00E+02	**3.07E−61**	+	1.60E+03	**5.26E−29**	+
F5	Mean	2.66E−04	**4.88E−226**	3.02E−11	1.29E+00	**6.37E−181**	3.02E−11	6.89E+00	**1.38E−137**	3.02E−11
	Std	8.03E−05	**0**	+	2.29E−01	**0**	+	7.17E−01	**7.53E−137**	+
F6	Mean	7.30E+00	**3.68E+00**	2.34E−01	6.34E+01	**2.23E+01**	4.64E−03	3.32E+02	**4.26E+01**	3.02E−11
	Std	7.23E+00	**4.62E−01**	≈	4.24E+01	**1.35E+00**	+	1.28E+02	**2.11E+00**	+
F7	Mean	1.20E−08	**4.50E−09**	9.53E−07	8.73E−05	**1.67E−02**	8.48E−09	7.84E+00	**9.99E−02**	3.02E−11
	Std	5.90E−09	**4.09E−09**	+	1.39E−05	**6.34E−02**	–	3.13E+00	**1.41E−01**	+

Table 4 Comparison of results on 10, 30 and 50-dimensional multimodal benchmark problems

Test function	Result	Dim 10			Dim 30			Dim 50		
		HSA	MHSA	Stat. out.	HSA	MHSA	Stat. out.	HSA	MHSA	Stat. out.
F8	Mean	3.07E−16	**0**	1.21E−12	7.44E−09	**0**	1.21E−12	5.21E−07	**0**	1.21E−12
	Std	3.77E−16	**0**	+	2.00E−09	**0**	+	1.48E−07	**0**	+
F9	Mean	2.94E−03	**1.80E−04**	3.02E−11	1.77E−02	**2.14E−04**	3.02E−11	9.64E−02	**2.57E−04**	3.02E−11
	Std	1.11E−03	**1.34E−04**	+	5.14E−03	**8.45E−05**	+	2.35E−02	**9.27E−05**	+
F10	Mean	2.45E−12	**0**	1.21E−12	8.58E−12	**0**	1.21E−12	1.91E−11	**0**	1.21E−12
	Std	3.44E−12	**0**	+	1.36E−11	**0**	+	2.33E−11	**0**	+
F11	Mean	**−4.19E+03**	**−4.19E+03**	3.02E−11	**−1.26E+04**	**−1.26E+04**	5.57E−10	−2.09E+04	**−2.09E+04**	1.86E−06
	Std	**1.94E−09**	4.46E−04	–	1.02E−02	**1.59E−03**	+	**8.44E+00**	4.91E+01	+
F12	Mean	2.67E−06	**0**	1.21E−12	1.53E−02	**0**	1.21E−12	1.14E+00	**0**	1.21E−12
	Std	2.12E−06	**0**	+	3.29E−03	**0**	+	7.66E−01	**0**	+
F13	Mean	1.42E−04	**2.43E−15**	1.41E−11	6.61E−03	**4.56E−15**	1.72E−12	5.72E−01	**4.56E−15**	1.72E−12
	Std	3.51E−05	**1.79E−15**	+	6.13E−04	**6.49E−16**	+	3.49E−01	**6.49E−16**	
F14	Mean	5.42E−02	**0**	1.21E−12	3.28E−02	**0**	1.21E−12	1.10E+00	**0**	1.21E−12
	Std	2.30E−02	**0**	+	2.97E−02	**0**	+	2.91E−02	**0**	+
F15	Mean	**1.90E−09**	3.18E−09	1.91E−01	5.74E−07	**1.55E−09**	3.02E−11	5.38E−03	**6.60E−09**	3.02E−11
	Std	**4.28E−09**	7.30E−09	≈	1.13E−07	**6.44E−10**	+	2.10E−03	**1.96E−09**	+
F16	Mean	**3.66E−04**	**3.66E−04**	5.01E−02	5.14E−03	**3.16E−08**	3.02E−11	2.90E−01	**7.80E−03**	3.02E−11
	Std	**2.01E−03**	**2.01E−03**	≈	5.57E−03	**2.02E−08**	+	9.31E−02	**2.17E−02**	+
F17	Mean	2.47E−04	**1.97E−09**	2.37E−12	1.04E−02	**0**	1.21E−12	3.16E−01	**0**	1.21E−12
	Std	1.58E−04	**9.74E−09**	+	8.05E−03	**0**	+	1.05E−01	**0**	+
F18	Mean	1.65E−07	**0**	1.21E−12	8.76E−04	**0**	1.21E−12	6.46E−03	**0**	1.21E−12
	Std	9.21E−08	**0**	+	1.13E−04	**0**	+	7.77E−04	**0**	+
F19	Mean	1.52E−06	**8.93E−223**	3.02E−11	1.37E−01	**3.06E−93**	3.02E−11	2.27E+01	**2.29E−57**	3.02E−11
	Std	7.63E−07	**0**	+	1.53E−01	**9.58E−93**	+	5.60E+00	**1.25E−56**	+
F20	Mean	1.14E−03	**0**	1.21E−12	2.10E+01	**0**	1.21E−12	4.12E+03	**0**	1.21E−12
	Std	1.60E−03	**0**	+	1.00E+01	**0**	+	1.75E+03	**0**	+
F21	Mean	**0**	**0**	≈	**0**	**0**	≈	**0**	**0**	≈
	Std	**0**	**0**	NA	**0**	**0**	NA	**0**	**0**	NA
F22	Mean	5.30E−01	**9.99E−02**	2.79E−11	1.23E+00	**1.03E−01**	3.02E−11	2.39E+00	**1.47E−01**	3.02E−11
	Std	1.34E−01	**8.31E−10**	+	1.49E−01	**1.83E−02**	+	2.43E−01	**5.07E−02**	+
F23	Mean	8.61E−02	**9.72E−03**	2.60E−11	2.91E−01	**1.06E−02**	3.02E−11	4.34E−01	**2.16E−02**	3.02E−11
	Std	3.38E−02	**5.72E−11**	+	3.55E−02	**5.02E−03**	+	2.00E−02	**1.39E−02**	+

calculated and presented in Tables 3 and 4. In these tables, the better results are highlighted in bold face.

In Table 3, the results are shown for the unimodal problems, and in Table 4, the results on multimodal problems are presented. These results are also compared by Wilcoxon rank sum test at 5% significance level to analyze the significant difference between the conventional HSA and the proposed MHSA. These statistical results are indicated by 'stat. out.' which provide the p value and the outcome of the results. The outcomes '+/ = /−' indicate that the proposed MSA is significantly better, equal, or worst than the conventional HSA. By comparing the results on unimodal test functions, it can be analyzed that in all the problems with varying dimension size, the proposed MHSA has significantly outperformed the conventional HSA except for 10-dimensional F6. In this problem, the proposed MHSA provides better value of mean and standard deviation of objective function values than the conventional HSA, but this improvement is not statistically significant. The low value of standard deviations shows the robustness of results in all the problems by the proposed MHSA. Moreover, on the functions F1, F2, and F3, the MHSA is able to provide the optima irrespective of the dimension, while the conventional HSA is unable to locate the optima even for a single problem. On multimodal problems F8, F10, F12, F14, F17, F18, F20, and F21, the proposed MHSA locate the global optima, while the conventional HSA is able to do this only for F21. Obviously, these results demonstrate the better exploration ability by the MHSA as compared to the conventional HSA. The statistical outcomes demonstrate that in most of the problems, the MHSA has obtained significantly better results than the conventional HSA. As an example to show the search history of the MHSA, Fig. 4 is plotted. In this figure, the harmonies of the HM are shown for 2-dimensional Rastrigin function (F12). The figures show that, initially, the harmonies are diversified and after some iterations, they try to converge the optima and finally at iteration 500, and they all converge to the global optima (0) of the problem.

Overall analysis conclude that the proposed MHSA improves the exploitation as well as the exploration ability of the conventional HSA using the modified search mechanism. On some cases like on problem F6, which is complex and have massive local optima, the algorithm is unable to achieve the global optima. Although the results are better than the conventional HSA, but the proposed approaches are not sufficient enough to determine the near optimal solution. One reason for this may be the proposed pitch adjustment process, which is inspired by the encircling behavior of the GWO. In this approach, at later iterations of the algorithm, the coefficient A is unable to explore the search space, and therefore, when the best solution trapped at local optima, then there are high chances that the whole harmony memory may stuck at local optima. On some highly non-linear problems, this reason may affect the performance of the algorithm, and therefore, in future research work, this shortcoming can be reduced using other evolutionary operators like mutation.

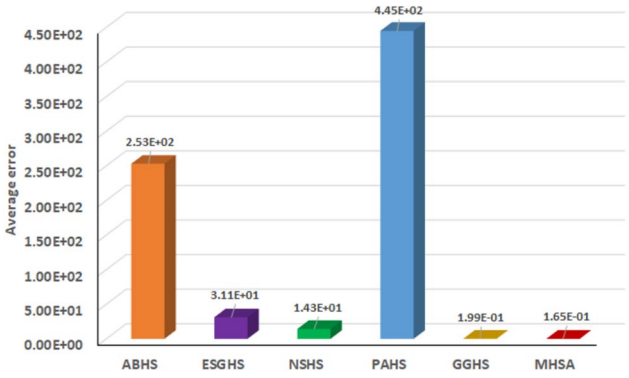

Fig. 5 Comparison of average error values between proposed MHSA and other variants of HSA for 10-dimensional problems

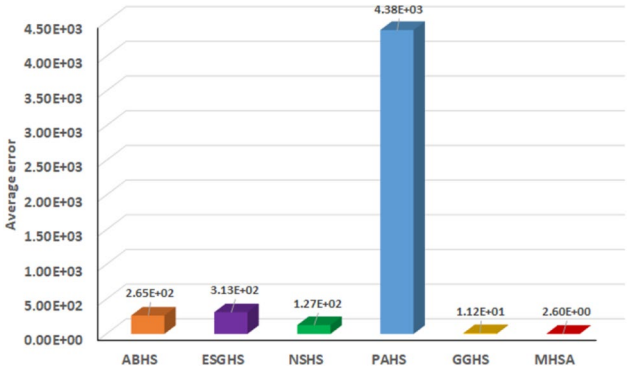

Fig. 6 Comparison of average error values between proposed MHSA and other variants of HSA for 50-dimensional problems

To validate the capacity of the proposed MHSA, the next subsection provides a comparison of the MHSA to the other variants of the HSA developed in the literature.

4.2 Comparison of the MHSA with other variants of the HSA

In this section, different variants of the HSA developed in the literature are used to compare the performance of the proposed MHSA. The comparison is performed on the same set of benchmark problems and same setting of function evaluations as fixed in the previous subsection. Table 5 reports the mean and standard deviation of objective function values yields by the MHSA and other variants of HSA such as adaptive harmony search with best based search strategy (ABHS) [12], enhanced self-adaptive global-best harmony search (ESGHS) [30], novel self-adaptive harmony search (NSHS)

Table 5 Comparison of mean and standard deviation values obtained by the proposed MHSA and other variants of the HSA for 30-dimensional benchmark problems

Test function	ABHS			ESGHS			NSHS			PAHS			GGHS			MHSA	
	Mean	Std	Stat. out.	Mean	Std	Stat. out.	Mean	Std	Stat. out.	Mean	Std	Stat. out.	Mean	Std	Stat. out.	Mean	Std
F1	3.90E−02	2.94E−02	+	1.14E−168	**0.00E+00**	+	8.54E−05	5.33E−05	+	5.12E−01	1.74E−01	+	2.88E−12	8.55E−13	+	**0**	**0**
F2	6.79E−03	6.86E−03	+	3.87E−16	1.78E−16	+	1.74E−06	5.36E−07	+	7.73E−02	3.51E−02	+	5.92E−11	1.39E−11	+	**0**	**0**
F3	5.08E−02	1.77E−02	+	2.03E−08	4.06E−09	+	2.60E−03	4.84E−04	+	2.57E−01	4.80E−02	+	6.48E−06	1.44E−06	+	**0**	**0**
F4	1.15E+03	6.99E+02	+	1.61E−02	7.05E−03	+	3.55E−02	1.46E−02	+	1.26E+03	4.24E+02	+	4.06E−03	7.27E−03	+	**5.61E−62**	**3.07E−61**
F5	1.26E+00	4.61E−01	+	2.84E−03	7.50E−04	+	1.25E−02	3.24E−03	+	2.00E+00	3.97E−01	+	4.53E−01	1.05E−01	+	**6.37E−181**	**0**
F6	6.70E+01	3.88E+01	+	4.53E+01	2.78E+01	+	7.63E+01	8.88E+01	+	1.17E+02	3.05E+01	+	3.14E+01	4.26E+01	+	**2.23E+01**	**1.35E+00**
F7	4.12E−02	2.83E−02	+	**2.47E−33**	**5.01E−33**	−	8.28E−05	4.56E−05	−	5.18E−01	1.63E−01	+	5.20E−12	1.36E−12	−	1.67E−02	6.34E−02
F8	4.49E−10	9.48E−10	+	4.55E−12	4.37E−12	+	1.71E−15	4.66E−16	+	1.09E−08	1.44E−08	+	3.01E−23	1.13E−23	+	**0**	**0**
F9	1.66E−02	1.03E−02	+	5.75E−03	2.01E−03	+	1.59E−03	5.60E−04	+	2.65E−02	9.00E−03	+	9.01E−03	3.40E−03	+	**2.14E−04**	**8.45E−05**
F10	1.76E−08	3.25E−08	+	1.89E−09	3.21E−09	+	3.97E−09	1.13E−09	+	4.53E−08	6.56E−08	+	2.86E−20	2.92E−20	+	**0**	**0**
F11	−1.26E+04	8.91E−02	+	−8.48E+03	5.07E+02	+	−1.26E+04	2.16E−04	−	−1.26E+04	5.33E−01	+	**−1.26E+04**	**1.29E−12**	−	−1.26E+04	1.59E−03
F12	1.84E−02	1.19E−02	+	1.75E−07	7.72E−08	+	1.68E−05	2.04E−06	+	2.13E−01	8.21E−02	+	**8.56E−10**	**1.67E−10**	+	**0**	**0**
F13	5.34E−02	2.65E−02	+	1.57E−04	2.22E−04	+	9.10E−04	2.02E−04	+	2.43E−01	5.86E−02	+	1.57E−06	1.88E−07	+	**4.56E−15**	**6.49E−16**
F14	9.70E−02	5.57E−02	+	5.33E−03	8.64E−03	+	4.49E−03	4.44E−03	+	5.38E−01	1.30E−01	+	1.28E−02	4.04E−03	+	**0**	**0**
F15	1.95E−04	2.46E−04	+	6.91E−03	2.63E−02	+	5.30E−10	6.94E−11	−	3.00E−03	4.05E−03	+	**9.74E−14**	**9.13E−14**	−	1.55E−09	6.44E−10
F16	3.93E−03	4.53E−03	+	**1.35E−32**	**5.57E−48**	−	3.66E−04	2.01E−03	+	2.22E−02	1.10E−02	+	5.86E−13	4.71E−13	−	3.16E−08	2.02E−08
F17	3.24E−03	1.53E−03	+	8.13E−09	4.16E−09	+	5.01E−02	2.40E−02	+	1.09E−02	2.05E−03	+	3.76E−06	9.61E−07	+	**0**	**0**
F18	5.56E−05	4.46E−05	+	2.01E−05	8.14E−06	+	6.40E−07	7.73E−08	+	3.33E−04	1.91E−04	+	5.16E−11	1.89E−11	+	**0**	**0**
F19	1.18E+01	1.08E+01	+	1.28E−10	4.52E−11	+	1.97E−07	4.82E−08	+	7.73E+00	1.99E+00	+	2.05E−09	1.74E−09	+	**3.06E−93**	**9.58E−93**
F20	3.32E+03	3.09E+03	+	1.78E−04	1.95E−04	+	2.21E+03	8.42E+02	+	2.17E+04	1.64E+04	+	8.07E−04	1.69E−03	+	**0**	**0**
F21	**0**	**0**	≈	**0**	**0**		**0**	**0**	≈	**0**	**0**	≈	**0**	**0**	≈	**0**	**0**
F22	1.19E+00	2.24E−01	+	3.13E−01	3.46E−02	+	2.27E−01	4.50E−02	+	1.36E+00	2.23E−01	+	3.20E−01	8.37E−02	+	**1.03E−01**	**1.83E−02**
F23	3.06E−01	6.63E−02	+	9.31E−02	2.55E−02	+	5.00E−01	**7.16E−05**	+	3.30E−01	3.31E−02	+	1.17E−01	2.18E−02	+	**1.06E−02**	5.02E−03
Rank sum	107			63			81			128			58			31	
Rank	5			3			4			6			2			1	

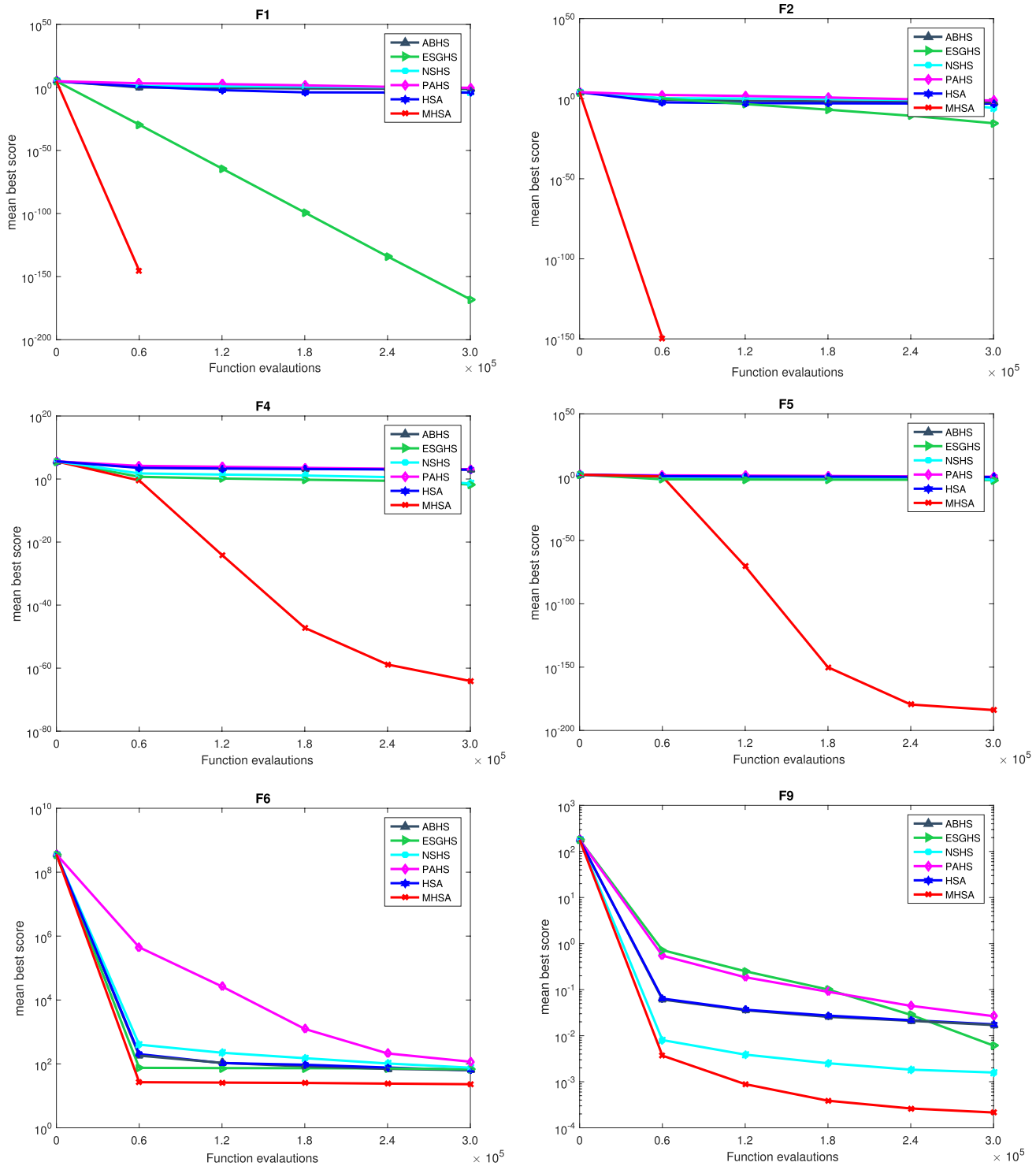

Fig. 7 Convergence curves for unimodal test functions

[29], parameter adaptive harmony search (PAHS) [26], and Gaussian global-best harmony search algorithm (GGHS) [24] on 30-dimensional test problems. In the table, the better results are highlighted in bold face. In the same table, the statistical outcomes obtained by applying the Wilcoxon rank sum test between the proposed MHSA and its competitive algorithm are also presented. The symbols '+/ = /−' are used to demonstrate that the MHSA is better, equal, or worse than its competitive algorithm. Moreover, the bottom part of the table provides the rank sum and overall rank (rank)

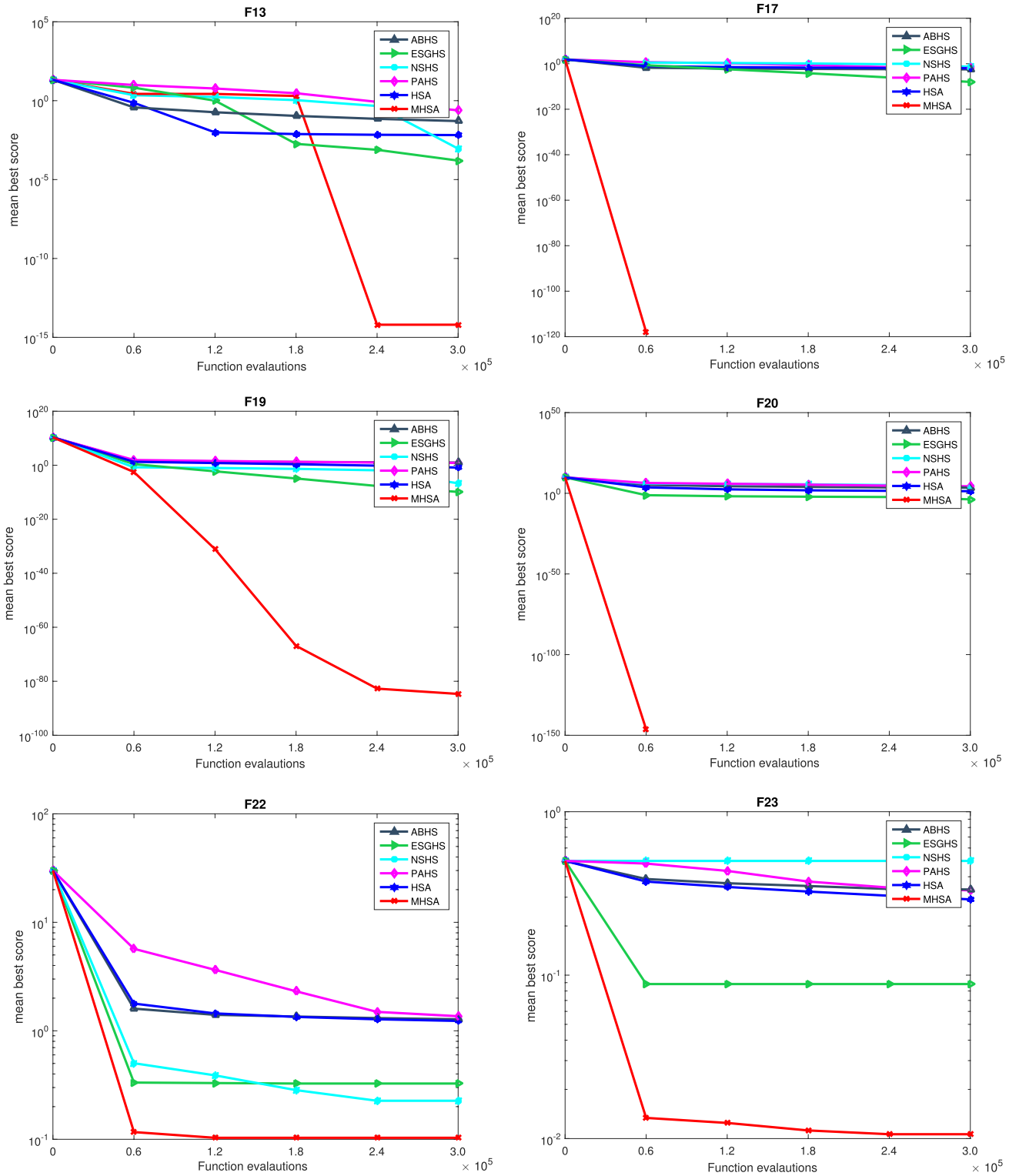

Fig. 8 Convergence curves for multimodal test functions

of the algorithms to pick up the best performer algorithm. The rank sum is the number which denotes the sum of the rank for each test problem obtained by ascending order of the objective function values.

It can also be seen from Table 5 that the proposed MHSA is significantly better than the ABHS on all of the test problems except F21. In this problem, all the algorithms MHSA, ABHS, ESGHS, NSHS, PAHS, and GGHS provide the global optimal solution. The comparison between the ESGHS and MHSA shows that the proposed MHSA provides better results than the ESGHS in all the problems except F7 and F16. The NSHS algorithm is better than the MHSA on F7 and F15. When the comparison is performed between the PAHS and MHSA, the outperform search ability of the MHSA can be verified. The comparison with the GGHS shows that the proposed MHSA is better in all the problems except F7, F11, F15, and F16. The statistical results also validate this improvement in the search strategy of the proposed MHSA as compared to the other algorithms. The ranking of the algorithms shows that the proposed MHSA is the best performer algorithm than the ABHS, ESGHS, NSHS, PAHS, and GGHS. The methods next to the MHSA are GGHS, ESGHS, NSHS, ABHS, and PAHS, respectively.

To observe the impact of the proposed MHSA in solving the benchmark problems with dimensions 10 and 50, and to compare it with the other variants of HSA, Figs. 5 and 6 are presented. The experiments are carried out with same parameter setting as used for solving 30-dimensional problems. In Figs. 5 and 6, the average error has been reported for each of the optimization method over 23 problems. Also, the obtained average error data have been shown with the bars in the figure, which clearly demonstrate that the proposed MHSA is superior to all other variants of the HSA.

To analyze and compare the convergence rate among the algorithms ABHS, ESGHS, NSHS, PAHS, conventional HSA, and the proposed MHSA, the convergence curves are plotted in Figs. 7 and 8 corresponding to unimodal and multimodal benchmark problems. These variants are used for comparison of convergence rate because of their similar in structure. In these figures, each chart corresponds to one test functions that is used in the experiments. The horizontal axis represents the number of function evaluations and the vertical axis represents the the best value of the objective function obtained so far. Figure 7 clearly demonstrates that in most of the test functions, the ABHS, conventional HSA, PAHS, and NSHS algorithms exhibit similar search behavior during the search procedure. In problem F1 and F23, the convergence behavior of the ESGHS is better than ABHS, NSHS, conventional HSA, and PAHS. On the other hand, in all of the problems, the convergence behavior of the proposed MHSA is better than all other variants of the HSA and this shows a better global search performance of the MHSA. In problems F1, F2, F17, and F20, the proposed MHSA shows outperformed convergence rate and locates the optima within $1/5^{th}$ of the total number of function evaluations. Hence, the convergence behavior analysis demonstrates the better convergence rate of the MHSA not only than conventional HSA but than other variants such as ABHS, ESGHS, NSHS, and PAHS.

4.3 Effect of the harmony memory size

To analyze the performance of the proposed MHSA on varying the harmony memory size HMS, the results corresponding to the harmony memory size 5, 10, 20, and 50 are calculated on benchmark test problems given in Tables 1 and 2. This experiment is conducted by repeating the proposed MHSA 30 times independently on 30-dimensional problems. The Wilcoxon rank sum test at 5% significance level is used to signify the difference in results. These statistical results are shown in symbols '+/ = /−' to indicate that the MHSA with 5 harmony memory size is better, equal, or worse than the same algorithm but with harmony memory sizes 10, 20, or 50. From Table 6, it can be observed that on almost all of the test functions, the proposed HMSA is either performing equal or significantly outperforming other cases with different harmony sizes. The statistical outcomes show that increment in the harmony memory size of the MHSA degrades its solution accuracy in most of the test problems. The average rank and overall rank calculated based on sorting the mean objective function values also demonstrate that the HMSA with harmony memory size 5 is superior, while the next are MHSA with 10, 20, and 50 harmony memory sizes, respectively.

4.4 Comparison of the MHSA with other metaheuristics

In this section, the performance of the MHSA is compared with some other metaheuristic algorithms such as sine cosine algorithm (SCA) [33], grey wolf optimizer (GWO) [34], comprehensive learning particle swarm optimization (CLPSO) [28, 31], gbest-guided artificial bee colony (GABC) [47], and covariance matrix adaptation evolution strategy (CMA-ES) [13]. The SCA and GWO are two relatively new metaheuristic algorithms. SCA is inspired from the mathematical features of sine and cosine trigonometric functions and the GWO is inspired by the social and hunting behavior of grey wolves in nature. The original versions of these algorithms are used in this section to compare the performance of the proposed MHSA. The GABC is an extended version of the artificial bee colony algorithm, when the best solution obtained so far is used to guide the search procedure. The CLPSO is an improved version of the conventional particle swarm optimization (PSO), where the best positions of other particles are used to update the position of current particle. The reason of employing this concept in the PSO was to enhance the exploration ability of particles by learning from others particles and to avoid the particles to prone

Table 6 Results of the proposed HMSA on different harmony memory size 5, 10, 15 and 50

Test function	HMS = 5		HMS = 10			HMS = 20			HMS = 50		
	Mean	Std	Mean	Std	Stat. out.	Mean	Std	Stat. out.	Mean	Std	Stat. out.
F1	0	0	0	0	≈	0	0	≈	5.39E − 174	0	+
F2	0	0	0	0	≈	0	0	≈	4.16E − 177	0	+
F3	0	0	0	0	≈	7.67E − 257	0	+	6.03E − 129	1.62E − 128	+
F4	5.61E − 62	3.07E − 61	1.38E − 23	6.63E − 23	+	1.03E − 10	3.17E − 10	+	3.79E − 03	1.05E − 02	+
F5	6.37E − 181	0	6.29E − 80	3.45E − 79	+	2.15E − 42	9.42E − 42	+	7.77E − 21	1.11E − 20	+
F6	2.23E + 01	1.35E + 00	2.19E + 01	6.96E − 01	≈	2.27E + 01	2.86E − 01	+	2.42E + 01	7.02E − 02	+
F7	1.67E − 02	6.34E − 02	8.33E − 03	4.56E − 02	+	1.74E − 07	5.76E − 08	+	7.77E − 07	1.92E − 07	+
F8	0	0	0	0	≈	0	0	≈	9.52E − 231	0	+
F9	2.14E − 04	8.45E − 05	2.48E − 04	1.18E − 04	≈	4.46E − 04	1.92E − 04	+	6.48E − 04	2.65E − 04	+
F10	0	0	0	0	≈	0	0	≈	1.51E − 304	0	≈
F11	− 1.26E + 04	1.59E − 03	− 1.26E + 04	4.83E + 01	+	− 1.26E + 04	8.62E − 03	+	− 1.26E + 04	2.67E − 02	+
F12	0	0	0	0	≈	0	0	≈	0	0	≈
F13	4.56E − 15	6.49E − 16	1.28E − 14	1.09E − 14	+	1.44E − 14	9.89E − 15	+	4.27E − 14	7.59E − 14	+
F14	0	0	0	0	≈	8.21E − 04	3.38E − 03	≈	0	0	≈
F15	1.55E − 09	6.44E − 10	2.83E − 09	1.81E − 09	+	5.14E − 09	1.38E − 09	+	2.15E − 08	7.35E − 09	+
F16	3.16E − 08	2.02E − 08	4.73E − 08	4.62E − 08	+	7.33E − 04	2.79E − 03	+	3.59E − 07	9.63E − 08	+
F17	0	0	0	0	≈	2.39E − 240	0	+	1.51E − 118	8.14E − 118	+
F18	0	0	0	0	≈	0	0	≈	0	0	≈
F19	3.06E − 93	9.58E − 93	3.09E − 44	1.53E − 43	+	2.13E − 21	6.46E − 21	+	6.21E − 08	7.01E − 08	+
F20	0	0	0	0	≈	0	0	≈	5.10E − 174	0	+
F21	0	0	0	0	≈	0	0	≈	0	0	≈
F22	1.03E − 01	1.83E − 02	1.13E − 01	3.46E − 02	+	1.03E − 01	1.83E − 02	+	9.99E − 02	5.71E − 10	−
F23	1.06E − 02	5.02E − 03	1.15E − 02	6.98E − 03	+	1.26E − 02	8.38E − 03	+	9.72E − 03	4.80E − 11	−
Rank sum	29		40			49			58		
Rank	1		2			3			4		

Table 7 Parameter setting for algorithms

Algorithm	Parameters
CLPSO	$N = 30, \text{FEV} = 10^4 \times D, P_{C_i} = 0.05 + 0.45 \times \frac{\exp\left(\frac{10(i-1)}{N-1}\right)-1}{\exp(10)-1}$
GWO	$N = 30, \text{FEV} = 10^4 \times D, a = 2 - t\left(\frac{2}{T}\right)$
CMA-ES	$N = 30, \text{FEV} = 10^4 \times D$
GABC	$N = 30, \text{FEV} = 10^4 \times D, \text{limit} = D * (N/2)$
SCA	$N = 30, \text{FEV} = 10^4 \times D, r_1 = a - t\left(\frac{a}{T}\right)$

towards the local optima during the search procedure. The CMA-ES is an adaptive method of the evolution strategy. This method is well popular among the researchers because of their search abilities.

All these algorithms compare the performance of the MHSA by comparing the mean and standard deviation of objective function values. The parameter setting used by all the algorithms is presented in Table 7. The stopping criteria for each algorithm are fixed to $10^4 \times d$ function evaluations (FEV). All the algorithms are executed 30 times independently, and the mean and standard deviation of objective function values is recorded, which are presented in Table 8. In this table, the best results are highlighted in bold face. The statistical analysis of results is also performed to signify the difference in results and the ranking of all the algorithms is also done to select the best performer algorithm among all the compared methods. From the table, it can be seen that the SCA performs very poor as compared to the MHSA, because it has not provided significantly better results on any of the problem. The CLPSO has performed better than the MHSA only on F6, F7, and F15, the GWO has performed better than the MHSA on F4, F9, F19, and F22. The CMA-ES performs significantly better than the MHSA for F4, F6, F7, F15, and F19. The GABC provides better results than the MHSA on F6, F7, F11, F15, and F16 due to their global-best guidance ability. The ranking of the algorithms demonstrates that the proposed MHSA is overall best performer algorithm, while the GWO, CMA-ES, CLPSO, SCA, and GABC are the next methods, respectively.

To analyze the performance of the proposed MHSA and to compare it with other metaheuristics for other dimensions of the problems, the experiments are carried out on 10- and 50-dimensional benchmark problems. The parameter settings are adopted same as used for solving 30-dimensional problems. The experimental results are obtained in terms of average error over all the 23 benchmark problems. These results are shown in Figs. 9 and 10, where the average error is indicated on each bar of the algorithms. From these figures, it can be easily concluded that the proposed MHSA provides better optimization results on the considered benchmark set. Hence, the proposed MHSA can be considered as a better optimizer than other comparative algorithms.

4.5 Illustrative examples

In this section, to evaluate the performance of the proposed MHSA, three well-known benchmark structural engineering design problems are used. For this, different variants of the HSA and conventional HSA are simultaneously compared with the proposed MHSA to analyze its search performance. The harmony size for all the problems is fixed to 5 as suggested in Subsect. 4.3. One of the interesting features of this section is the employed constraint handling mechanism, where neither the penalty approach nor any other constraint handling is applied. In this approach, first, the value of the objective function and constraints are evaluated on harmonies, and then, the violation is recorded corresponding to each harmony. In this way, the better harmony can be recognized easily, which will be the one that having lesser constraint violation value. In the situation, where more than one harmonies are recognized as feasible one, i.e., having zero value of constraint violation, then the one having better objective fitness is picked as a best harmony of the memory. In this way, the constraints are tackled in the proposed MHSA, conventional HSA, and other variants. Obviously, this mechanism of handling the constraints does not involve any parameters. This mechanism evaluates the search strategy of the MHSA directly to the constrained problems by just picking the best one in terms of constraint violation value. For a general form of the optimization problem

$$\min \quad F(X), \quad X = (x_1, x_2, \ldots, x_d) \in \mathbb{R}^d, \tag{38}$$

$$\begin{aligned} \text{s.t.} \quad & g_j(X) \le 0, \quad j = 1, 2, \ldots, J \\ & h_k(X) = 0, \quad k = 1, 2, \ldots, K \\ & x_i^{\min} \le x_i \le x_i^{\max}, \end{aligned} \tag{39}$$

the constraint violation corresponding to the harmony $\hat{X}$ can be calculated as follows:

$$\text{viol}_{\hat{X}} = \sum_{j=1}^{J} G_j(\hat{X}) + \sum_{k=1}^{K} H_k(\hat{X}), \tag{40}$$

where

$$G_j(\hat{X}) = \begin{cases} g_j(X) & g_j(X) > 0 \\ 0 & \text{otherwise}, \end{cases} \tag{41}$$

$$H_k(\hat{X}) = \begin{cases} \left|h_k(\hat{X})\right| & \left|h_k(\hat{X})\right| - \epsilon > 0 \\ 0 & \text{otherwise}, \end{cases} \tag{42}$$

Table 8 Comparison of mean and standard deviation values obtained by the proposed MHSA and other metaheuristic algorithms

Test function	CLPSO			GWO			CMA-ES			GABC			SCA			MHSA	
	Mean	Std	Stat. out.	Mean	Std	Stat. out.	Mean	Std	Stat. out.	Mean	Std	Stat. out.	Mean	Std	Stat. out.	Mean	Std
F1	3.80E−49	1.31E−48	+	0	0	≈	0	0	≈	3.44E−25	2.21E−25	+	5.30E−53	2.74E−52	+	0	0
F2	2.71E−50	3.82E−50	+	0	0	≈	0	0	≈	4.70E−26	2.75E−26	+	1.11E−55	5.74E−55	+	0	0
F3	9.96E−31	9.61E−31	+	0	0	≈	1.93E−223	0	+	1.68E−13	3.94E−14	+	1.21E−57	5.71E−57	+	0	0
F4	3.32E+02	1.55E+02	+	5.2E−179	0	−	0	0	−	1.19E+04	1.84E+03	+	3.29E+01	9.80E+01	+	5.61E−62	3.07E−61
F5	5.16E+01	7.96E+00	+	2.8E−152	8.1E−152	+	5.08E−162	8.01E−162	+	1.16E+01	1.12E+00	+	4.63E−03	1.97E−02	+	6.37E−181	0
F6	1.33E+01	2.13E+01	−	26.26529	0.723004	+	3.04E−27	6.43E−28	−	1.96E−01	2.19E−01	−	2.74E+01	7.72E−01	+	2.23E+01	1.35E+00
F7	0	0	−	0.416822	0.289171	+	2.87E−30	6.60E−31	−	3.45E−25	2.30E−25	−	3.62E+00	2.85E−01	+	1.67E−02	6.34E−02
F8	5.39E−80	2.95E−79	+	0	0	≈	0	0	≈	3.96E−54	5.82E−54	+	3.90E−58	2.14E−57	+	0	0
F9	8.07E−03	4.25E−03	+	6.89E−05	4.56E−05	−	9.56E−02	3.32E−02	+	3.29E−02	7.92E−03	+	1.93E−03	1.84E−03	+	2.14E−04	8.45E−05
F10	5.41E−86	2.96E−85	+	0	0	≈	3.87E−38	9.52E−38	+	5.86E−32	9.98E−32	+	4.20E−77	2.30E−76	+	0	0
F11	−1.23E+04	1.36E+02	+	−5989.06	763.574	+	−1.18E+02	4.08E−14	+	−1.26E+04	4.76E−12	−	−4.46E+03	2.51E+02	+	−1.26E+04	1.59E−03
F12	4.64E−01	6.78E−01	+	0	0	≈	1.86E+01	4.32E+00	+	1.89E−15	1.04E−14	+	3.39E−01	1.86E+00	+	0	0
F13	2.18E−01	4.73E−01	+	7.52E−15	1.23E−15	+	7.76E−15	9.01E−16	+	2.49E−12	6.99E−13	+	1.40E+01	8.43E+00	+	4.56E−15	6.49E−16
F14	2.37E−03	9.46E−03	≈	0.00025	0.001368	≈	0	0	≈	7.34E−12	3.42E−11	+	1.40E−15	7.68E−15	+	0	0
F15	1.57E−32	5.57E−48	−	0.029883	0.011313	+	1.61E−31	5.51E−32	−	1.47E−26	1.28E−26	−	3.43E−01	7.24E−02	+	1.55E−09	6.44E−10
F16	1.10E−03	3.35E−03	+	0.377554	0.185824	+	3.66E−04	2.01E−03	+	6.84E−25	8.82E−25	−	1.97E+00	1.22E−01	+	3.16E−08	2.02E−08
F17	1.25E−07	2.19E−07	+	0	0	≈	1.72E−16	6.50E−17	+	1.13E−06	2.93E−06	+	2.36E−31	1.06E−30	+	0	0
F18	1.48E−17	8.11E−17	+	0	0	≈	6.40E−02	1.00E−01	+	0	0	≈	0	0	≈	0	0
F19	1.95E−01	8.96E−02	+	7E−224	0	−	0	0	−	2.33E+02	2.61E+01	+	5.54E−06	2.98E−05	+	3.06E−93	9.58E−93
F20	3.35E−44	1.80E−43	+	0	0	≈	0	0	≈	2.31E−20	3.12E−20	+	1.78E−53	9.11E−53	+	0	0
F21	0	0	≈	0	0	≈	0	0	≈	0	0	≈	0	0	≈	0	0
F22	3.90E−01	5.47E−02	+	0.099873	2.45E−13	−	3.97E−01	4.14E−02	+	9.33E−01	1.03E−01	+	1.13E−01	3.46E−02	+	9.99E−02	4.01E−11
F23	1.17E−01	3.03E−02	+	0.010633	0.005022	+	9.72E−03	1.69E−17	−	3.29E−01	3.49E−02	+	1.15E−02	6.98E−03	+	1.06E−02	5.02E−03
Rank sum	79			52			56			87			81			40	
Rank	4			2			3			6			5			1	

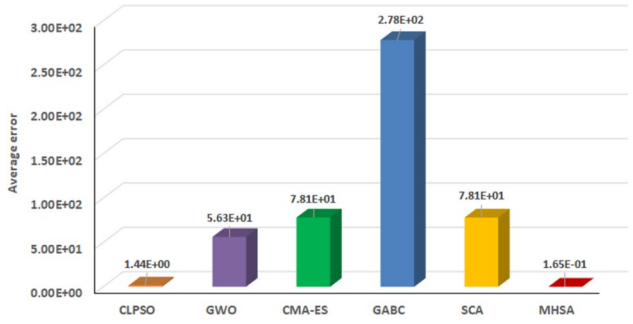

Fig. 9 Comparison of average error values between proposed MHSA and other metaheuristics for 10-dimensional problems

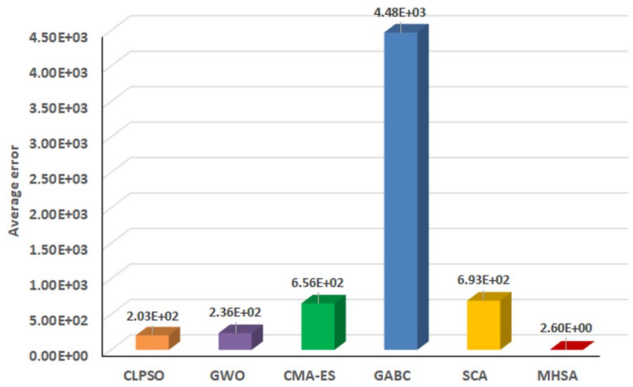

Fig. 10 Comparison of average error values between proposed MHSA and other metaheuristics for 50-dimensional problems

where $x_i^{\min}$ and $x_i^{\max}$ are the lower and upper bounds for a harmony component x_i of X. Symbols j and k indicate the number of inequality and equality constraints, respectively, in the optimization problem. The symbol ϵ is predefined tolerance parameter, which has been fixed to 10^{-4} in the present study.

The description of the constrained engineering design problems is presented as follows.

4.5.1 Compression spring design

The objective of this problem [4] is to minimize the weight of a tension/compression spring with some constraints such as surge frequency, shear stress, and minimum deflection. In this problem, three decision variables, namely, wire diameter (d), mean coil diameter (D), and the number of active coils (N), are involved. Mathematically, the problem can be stated as follows:

$$\min \quad f_1(X) = (x_3 + 2)x_1^2 x_2, \tag{43}$$

where $X = (x_1, x_2, x_3, x_4) = (d, D, N) \in \mathbb{R}^3$

$$\text{s.t.} \quad 1 - \frac{x_2^3 x_3}{71785 x_1^4} \leq 0, \tag{44}$$

$$\frac{4x_2^2 - x_1 x_2}{12566 x_1^3 x_2 - x_1^4} - \frac{1}{5108 x_1^2} \leq 0, \tag{45}$$

$$1 - \frac{140.45 x_1}{x_2^2 x_3} \leq 0, \tag{46}$$

$$\frac{x_1 + x_2}{1.5} - 1 \leq 0, \tag{47}$$

where $0.05 \leq x_1 \leq 2.00$, $0.25 \leq x_2 \leq 1.30$, and $2.00 \leq x_3 \leq 15.00$.

The numerical results in terms of mean, median, minimum (best), and maximum (worst) and standard deviation values of the weights recorded over 30 trails are presented in Table 9 for the proposed MHSA. In this table, the results of the conventional HSA, ABHS, ESGHS, PAHS and GGHS are also reported and the best results are highlighted in bold face in order to compare the search efficacy of the MHSA. The statistical outcomes along with the p values are also shown in the same table, which clearly demonstrate the superior and significantly better search ability if the MHSA than other variants of the HSA and conventional HSA.

4.5.2 Pressure vessel design

In this problem [17], the goal is to minimize the cost of cylindrical pressure vessel, which is closely related to material, structure, and welding. The ends of the pressure vessel are covered. In this problem, "the thickness of the shell (T_{SH})", "head (T_{HD})", "inner radius (R)", and "range of cross-section minus head (L)" are needed to be optimized. Mathematically, the problem is stated as follows:

$$\min \quad f_2(X) = 0.6224 x_1 x_3 x_4 + 1.7781 x_1^2 x_3 + 3.1661 x_1^2 x_4 + 19.84 x_1^2 x_3, \tag{48}$$

where $X = (x_1, x_2, x_3, x_4) = (T_{\text{SH}}, T_{\text{HD}}, R, L) \in \mathbb{R}^4$

$$\text{s.t.} \quad -x_1 + 0.0193 x_3 \leq 0, \tag{49}$$

$$-x_3 + 0.00954 x_3 \leq 0, \tag{50}$$

$$-\pi x_3^2 x_4 - \frac{4}{3}\pi x_3^3 + 12,96,000 \leq 0, \tag{51}$$

$$x_4 - 240 \leq 0, \tag{52}$$

where $0 \leq x_1, x_2 \leq 99, 10 \leq x_3, x_4 \leq 200$.

Table 9 Comparison of results on constrained engineering design problems

Algorithm	Mean	Median	Best	Worst	Std	FEV	p value	Stat. out.
Compression spring design problem								
MHSA	**0.0128345**	**0.0127995**	**0.0126861**	**0.0136802**	**0.0001732**	40000	NA	NA
HSA	0.0145712	0.0142970	0.0127946	0.0171970	0.0013578	40000	4.20E−10	+
ABHS	0.0159747	0.0159368	0.0129420	0.0197169	0.0018282	40000	5.49E−11	+
ESGHS	0.0159423	0.0163954	0.0127165	0.0183882	0.0021504	40000	2.83E−08	+
PAHS	0.0163342	0.0171540	0.0128944	0.0187377	0.0021084	40000	8.99E−11	+
GGHS	0.0142752	0.0142138	0.0127946	0.0160065	0.0008740	40000	3.75E−09	+
Pressure vessel design problem								
MHSA	**6308.2549**	**6111.8340**	6064.0094	7544.9743	443.6219	40000	NA	NA
HSA	7210.9737	7354.4882	6429.4540	7557.6150	**359.8187**	40000	2.60E−08	+
ABHS	7121.8395	7343.9307	6173.6463	7588.1223	475.5281	40000	2.19E−08	+
ESGHS	6944.4932	7333.1461	**6062.1890**	**7544.5357**	575.1279	40000	6.34E−05	+
PAHS	7113.2841	7328.2965	6173.0881	7567.3171	427.8528	40000	6.01E−08	+
GGHS	6860.3532	6746.6684	6454.9921	7820.2118	386.4812	40000	2.27E−06	+
Three-bar truss design problem								
MHSA	**263.95377**	**263.93247**	263.89631	**264.11574**	**0.06459**	20000	NA	NA
HSA	264.40546	264.01413	263.89731	266.57837	0.69138	20000	3.00E−03	+
ABHS	264.68805	264.15548	263.90636	268.69102	1.10957	20000	1.25E−05	+
ESGHS	263.96838	263.94919	**263.89610**	264.33365	0.08816	20000	5.90E−01	≈
PAHS	264.26629	264.23222	263.93320	265.27965	0.32957	20000	9.83E−08	+
GGHS	264.17496	264.25206	263.93999	264.32097	0.14624	20000	7.65E−06	+

The numerical results obtained by the MHSA are presented in Table 9 along with the results of other algorithms. In the table, better results are highlighted in bold face. It can be observed from this table that the proposed MHSA provides significantly better results as compared to the conventional HSA and other variants of the HSA. The obtained p values and statistical outcomes by the Wilcoxon rank sum test validate this fact. The less standard value of the MHSA than other algorithms also verifies the robustness of the results. Therefore, to minimize the cost of pressure vessel, the proposed MHSA can be preferred over other algorithms.

4.5.3 Three-bar truss design

This problem, which considers a three-bar planner truss structure, was introduced by Nowcki [45], where the volume of a bar truss is minimized with constraints applied on stress of each truss member. This can be achieved by optimizing the cross-sectional area which is formulated mathematically as follows:

$$\min \quad f_3(x_1,x_2) = L \times (2\sqrt{x}_1 x_1 + x_2), X = (x_1,x_2) = (A_1,A_2), \tag{53}$$

$$\text{s.t.} \quad \frac{\sqrt{2}x_1 + x_2}{\sqrt{2}x_1^2 + 2x_1x_2} P - \sigma \le 0, \tag{54}$$

$$\frac{x_2}{\sqrt{2}x_1^2 + 2x_1x_2} P - \sigma \le 0, \tag{55}$$

$$\frac{1}{x_1 + \sqrt{2}x_2} P - \sigma \le 0, \tag{56}$$

where $0 \le x_1, x_2 \le 1\, L = 100\,\text{cm}, \sigma, P = 2\,\text{KN/cm}^2$.

The optimization results are presented in Table 9, where the better results are highlighted in bold face. In the same table, the comparison of results obtained from the MHSA is performed with the conventional HSA, other variants of the HSA such as ABHS, ESGHS, PAHS, and GGHS with same parameter setting, and same constraint handling technique used for the MHSA. The obtained p values and statistical outcomes by the Wilcoxon rank sum test verify that the proposed MHSA performs significantly superior than the conventional HSA, ABHS, PAHS, and GGHS. The comparison between the MHSA and ESGHS shows that both the algorithms are significantly similar to provide the optimum value of the cross-sectional area for three-bar truss structure.

5 Conclusions and future research directions

This paper proposes a modified variant of the harmony search algorithm, called MHSA, for solving global optimization problems. In the direction of improvement; first, a parameter-free approach has been proposed for the harmony memory considering rate (HMCR) and pitch adjustment rate (PAR), which are considered base factors for the performance of the HSA. These parameters are adopted as non-linear in nature to mimic the non-linearity of the search process. The main advantage of using this approach is that the user does not need to worry about the fine-tune setting of these core control parameters. Second, the improvisation process is modified, based on inspiring the encircling mechanism of the well-known grey wolf optimizer, to effectively increase the diversification and intensification of the search space by the harmony, and to provide a better transition from exploration to exploitation. In the last step, the random generation of harmony process is modified to maintain the randomness and to speed up the convergence rate by the help of partial opposite harmonies. The impact of all these strategies in improving the search efficiency of the conventional HSA is validated through a standard and well-known collection of benchmark set of 23 problems and three structural real engineering design problems. The statistical and convergence analysis has verified the significant improvement in the search procedure of the proposed MHSA and introduced it as a superior global optimizer than the conventional HSA, other variants such as ABHS, ESGHS, NSHS, PAHS, and GGHS, and other metaheuristics such as CLPSO, GWO, CMA-ES, GABC, and SCA. The engineering design problems also demonstrate the significantly superior performance of the proposed MHSA than conventional HSA and other variants of the HSA.

The present study can be extended for binary, discrete, multi-objective, and many-objective optimization tasks by integrating necessary operators. Future research may also include the validation on other complex benchmark problems and real-world application problems like vehicle scheduling, aircraft streamline modeling problems, feature selection, and some others.

References

1. Ahmadianfar I, Heidari AA, Gandomi AH, Chu X, Chen H (2021). RUN beyond the metaphor: An efficient optimization algorithm based on Runge Kutta method. Expert Syst Appl 181:115079. https://doi.org/10.1016/j.eswa.2021.115079
2. Al-Betar MA, Awadallah MA, Khader AT, Abdalkareem ZA (2015) Island-based harmony search for optimization problems. Expert Syst Appl 42(4):2026–2035
3. Alatas B (2010) Chaotic harmony search algorithms. Appl Math Comput 216(9):2687–2699
4. Arora JS (1989). Introduction to optimum design. New York: McGraw-Hill
5. Assad A, Deep K (2016) Applications of harmony search algorithm in data mining: a survey. In: Proceedings of fifth international conference on soft computing for problem solving. Springer, pp 863–874
6. Chakraborty P, Roy GG, Das S, Jain D, Abraham A (2009) An improved harmony search algorithm with differential mutation operator. Fundam Inform 95(4):401–426
7. Das S, Suganthan PN (2010) Differential evolution: a survey of the state-of-the-art. IEEE Trans Evol Comput 15(1):4–31
8. Dorigo M, Birattari M, Stutzle T (2006) Ant colony optimization. IEEE Comput Intell Mag 1(4):28–39
9. El-Abd M (2013) An improved global-best harmony search algorithm. Appl Math Comput 222:94–106
10. Gao L-Q, Li S, Kong X, Zou D-X et al (2014) On the iterative convergence of harmony search algorithm and a proposed modification. Appl Math Comput 247:1064–1095
11. Geem ZW, Kim JH, Loganathan GV (2001) A new heuristic optimization algorithm: harmony search. Simulation 76(2):60–68
12. Guo Z, Yang H, Wang S, Zhou C, Liu X (2018) Adaptive harmony search with best-based search strategy. Soft Comput 22(4):1335–1349
13. Hansen N, Ostermeier A (2001) Completely derandomized self-adaptation in evolution strategies. Evol Comput 9(2):159–195
14. Hasanipanah M, Keshtegar B, Thai DK, Troung NT (2020) An ANN-adaptive dynamical harmony search algorithm to approximate the flyrock resulting from blasting. Eng Comput 1–13. https://doi.org/10.1007/s00366-020-01105-9
15. Heidari AA, Mirjalili S, Faris H, Aljarah I, Mafarja M, Chen H (2019) Harris hawks optimization: algorithm and applications. Future Gen Comput Syst 97:849–872
16. Jaberipour M, Khorram E (2010) Two improved harmony search algorithms for solving engineering optimization problems. Commun Nonlinear Sci Numer Simul 15(11):3316–3331
17. Kannan BK, Kramer SN (1994). An augmented Lagrange multiplier based method for mixed integer discrete continuous optimization and its applications to mechanical design. 405–411. https://doi.org/10.1115/1.2919393
18. Karaboga D, Basturk B (2007) A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. J Glob Optim 39(3):459–471
19. Kennedy J, Eberhart R (1995) Particle swarm optimization. In: Proceedings of ICNN'95-international conference on neural networks, vol 4, pp 1942–1948
20. Keshtegar B, Etedali S (2018) Nonlinear mathematical modeling and optimum design of tuned mass dampers using adaptive dynamic harmony search algorithm. Struct Control Health Monit 25(7):e2163
21. Keshtegar B, Hao P, Wang Y, Hu Q (2018) An adaptive response surface method and gaussian global-best harmony search algorithm for optimization of aircraft stiffened panels. Appl Soft Comput 66:196–207
22. Keshtegar B, Hao P, Wang Y, Li Y (2017a) Optimum design of aircraft panels based on adaptive dynamic harmony search. Thin Walled Struct 118:37–45
23. Keshtegar B, Ozbakkaloglu T, Gholampour A (2017b) Modeling the behavior of FRP-confined concrete using dynamic harmony search algorithm. Eng Comput 33(3):415–430
24. Keshtegar B, Sadeq MO (2017) Gaussian global-best harmony search algorithm for optimization problems. Soft Comput 21(24):7337–7349

25. Khalili M, Kharrat R, Salahshoor K, Sefat MH (2014) Global dynamic harmony search algorithm: GDHS. Appl Math Comput 228:195–219
26. Kumar V, Chhabra JK, Kumar D (2014) Parameter adaptive harmony search algorithm for unimodal and multimodal optimization problems. J Comput Sci 5(2):144–155
27. Li S, Chen H, Wang M, Heidari AA, Mirjalili S (2020) Slime mould algorithm: a new method for stochastic optimization. Future Gen Comput Syst 111:300–323
28. Liang JJ, Qin AK, Suganthan PN, Baskar S (2006) Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. IEEE Trans Evol Comput 10(3):281–295
29. Luo K (2013) A novel self-adaptive harmony search algorithm. J Appl Math 2013. https://doi.org/10.1155/2013/653749
30. Luo K, Ma J, Zhao Q (2019) Enhanced self-adaptive global-best harmony search without any extra statistic and external archive. Inf Sci 482:228–247
31. Mahadevan K, Kannan P (2010) Comprehensive learning particle swarm optimization for reactive power dispatch. Appl Soft Comput 10(2):641–652
32. Mahdavi M, Fesanghary M, Damangir E (2007) An improved harmony search algorithm for solving optimization problems. Appl Math Comput 188(2):1567–1579
33. Mirjalili S (2016) SCA: a sine cosine algorithm for solving optimization problems. Knowl Based Syst 96:120–133
34. Mirjalili S, Mirjalili SM, Lewis A (2014) Grey wolf optimizer. Adv Eng Softw 69:46–61
35. Nehdi ML, Keshtegar B, Zhu SP (2019)Nonlinear modeling for bar bond stress using dynamical self-adjusted harmony search optimization. EngComput. 1–12. https://doi.org/10.1007/s00366-019-00831-z
36. Ouyang H-B, Gao L-Q, Li S, Kong X-Y, Wang Q, Zou D-X (2017) Improved harmony search algorithm: LHS. Appl Soft Comput 53:133–167
37. Salcedo-Sanz S, Pastor-Sánchez A, Del Ser J, Prieto L, Geem Z-W (2015) A coral reefs optimization algorithm with harmony search operators for accurate wind speed prediction. Renew Energy 75:93–101
38. Shahraki A, Ebrahimi SB (2015) A new approach for forecasting enrollments using harmony search algorithm. J Intell Fuzzy Syst 28(1):279–290
39. Wang C-M, Huang Y-F (2010) Self-adaptive harmony search algorithm for optimization. Expert Syst Appl 37(4):2826–2837
40. Wang G-G (2018) Moth search algorithm: a bio-inspired metaheuristic algorithm for global optimization problems. Memet Comput 10(2):151–164
41. Wang G-G, Deb S, Cui Z (2019) Monarch butterfly optimization. Neural Comput Appl 31(7):1995–2014
42. Wang Y, Liu Y, Feng L, Zhu X (2015) Novel feature selection method based on harmony search for email classification. Knowl Based Syst 73:311–323
43. Whitley D (1994) A genetic algorithm tutorial. Stat Comput 4(2):65–85
44. Wolpert DH, Macready WG (1997) No free lunch theorems for optimization. IEEE Trans Evol Comput 1(1):67–82
45. Nowcki H. Optimization in pre-contract ship design. In: Fujita Y, Lind K, Williams TJ (eds) Computer Applications in the Automation of Shipyard Operation and Ship Design, vol 2. North-Holland, Elsevier, New York, pp 327–338
46. Yang Y, Chen H, Heidari AA, Gandomi AH (2021) Hunger games search: visions, conception, implementation, deep analysis, perspectives, and towards performance shifts. Expert Syst Appl 177:114864
47. Zhu G, Kwong S (2010) Gbest-guided artificial bee colony algorithm for numerical function optimization. Appl Math Comput 217(7):3166–3173
48. Zou D, Gao L, Wu J, Li S (2010) Novel global harmony search algorithm for unconstrained problems. Neurocomputing 73(16–18):3308–3318