Soft Comput
DOI 10.1007/s00500-015-1868-1

METHODOLOGIES AND APPLICATION

A novel harmony search algorithm with gaussian mutation for multi-objective optimization

Xiangshan Dai[1] · **Xiaofang Yuan**[1] · **Lianghong Wu**[2]

Abstract This paper proposes a novel harmony search algorithm with gaussian mutation (GMHS) for multi-objective optimization. Harmony search (HS) algorithm has shown many advantages in solving global optimization problems; however, it also has some shortcomings, such as poor ability of escaping from local optimum and poor convergence. In view of the weaknesses of HS algorithm, several important improvements are employed in the proposed GMHS, including (a) the memory consideration rule is modified to improve convergence, (b) two bandwidths in pitch adjustment are designed to obtain better exploration and exploitation, (c) chaotic maps are utilized in the GMHS to enhance global search ability and (d) a gaussian mutation operator is employed in the GMHS to speed up convergence rate and to jump out the local optimum. To solve multi-objective optimization problems, the GMHS uses fast non-dominated sorting and crowding distance method to update harmony memory. For the purpose of preserving non-dominated solutions found during the entire search process, an external archive has been adopted. To demonstrate the effectiveness of the GMHS, it is tested with benchmark problems. The experimental results show that the GMHS is competitive in convergence and diversity performance, compared with other multi-objective evolutionary algorithms. Finally, the impact of two key parameters on the performance of GMHS is also analyzed.

Keywords Multi-objective optimization · Harmony search (HS) · Gaussian mutation · External archive · Self-adaptive parameter setting

Communicated by V. Loia.

This work was supported in part by the National Natural Science Foundation of China (No. 61573133, No. 61203309) and Hunan Provincial Natural Science Foundation of China (No. 2015JJ3053).

✉ Xiaofang Yuan
yuanxiaofang126@126.com

[1] College of Electrical and Information Engineering, Hunan University, Changsha 410082, China

[2] College of Information and Electrical Engineering, Hunan University of Science and Technology, Xiangtan 411201, China

1 Introduction

In the real word, many engineering optimization problems usually have multiple objectives which conflict with each other, and problems of this kind are called multi-objective optimization problems (MOPs). MOPs are different from single objective optimization as these competing objectives must be optimized simultaneously. Instead of finding a single optimal solution, the purpose of MOPs is to find a set of optimal solutions, largely known as Pareto optimal set (Deb et al. 2002). Since most classical optimization algorithms can typically obtain only a Pareto optimal solution in one run, the efficiency of these optimization algorithms is low for finding multiple solutions. To acquire a Pareto optimal set, these optimization algorithms have to be run many times.

Over the past few decades, numerous effective multi-objective evolutionary algorithms (MOEAs) have been developed, such as NSGA-II (Deb et al. 2002), PESA-II (Corne et al. 2001), SPEA2 (Zitzler et al. 2001), M-PAES (Knowles et al. 2000), NSGA-III (Deb and Jain 2014), MOEA/D (Zhang and Li 2007). These MOEAs are able to find multiple Pareto optimal solutions in just one single execution and they are all population-based algorithms. What is more, several significant techniques, such as the use of a second population (or an archive) (Obayashi et al. 1998; Parks and

Miller 1998), have proved to be able to greatly improve the performance of MOEAs.

Harmony search (HS) algorithm is a new meta-heuristic algorithm proposed by Geem et al. (2001), which is inspired by the improvisation processes of music players. The HS algorithm has been successfully applied to many engineering optimization problems (Vasebi et al. 2007; Diao and Shen 2012; Landa et al. 2012; Kulluk et al. 2012) because of its own advantages and potentials. The main advantages of HS include the following: (a) it has few control parameters; (b) it does not require the initial value settings of the decision variables; (c) it does not rely on the rigid mathematical characteristics of optimization problems, such as differentiability, continuity; and (d) it is easy to be understood and implemented.

Many researchers have applied HS for single objective optimization problems. However, so far just few works are concerned with MOPs. Geem (2010) has used HS to solve multi-objective time-cost optimization problem. This work only considers dominated-based comparison and ignores diversity comparison which results in the algorithm showing poor diversity performance. A multi-objective harmony search algorithm, developed by Sivasubramani and Swarup (2011), has the capability to converge to Pareto optimal solutions. However, it requires a large number of iterations, which means it has low convergence rate. This low convergence rate can be attributed to the use of an improved harmony search (IHS) algorithm (Mahdavi et al. 2007) whose convergence rate is far from fast. Two detailed proposals for applying HS to solve MOPs have been proposed by Ricart et al. (2011). Since the performance of the two proposals are only tested on ZDT functions (Zitzler et al. 2000), the performance of these two proposals is unknown for tri-objective problems and more complicated MOPs, such as CEC 2009 instances (Zhang et al. 2009). Afterwards, Pavelski et al. (2012) has investigated the performance of four variants of HS algorithm for solving CEC 2009 problems. However, these four variants cannot generate true and well distributed Pareto optimal solutions. Overall, the approximated Pareto fronts obtained by the aforementioned multi-objective HS algorithms are all far from uniformly distributed mainly because these algorithms don't utilize an external archive to maintain the Pareto optimal set. In addition, these multi-objective HS algorithms have shown poor convergence and poor global search ability. Nevertheless, gaussian mutation is a useful technique for improving convergence because new mutated value generated by this mutation follows gaussian distribution instead of rand distribution.

In view of the above-mentioned drawbacks of HS algorithm for MOPs, a novel harmony search algorithm with gaussian mutation (GMHS), which has better exploration and exploitation ability, is proposed for solving MOPs in this paper. Some crucial improvements proposed in GMHS are as follows: (a) in the modified memory consideration phrase, each harmony stored in harmony memory is selected with different probabilities for the purpose of improving convergence of HS algorithm; (b) chaotic maps are integrated with GMHS for parameter adaption, which can enhance the global search ability of HS algorithm and solve the difficult problem of parameter adjustment; (c) in pitch adjustment phrase, a decreasing bandwidth and an adaptive bandwidth are designed to improve the global search ability and local search ability during the entire search process; (d) a gaussian mutation operator is utilized after improvisation in the GMHS for the purpose of speeding up convergence rate and preventing the algorithm from trapping in local optimum. Finally, a fast non-dominated sorting and crowding distance method (Deb et al. 2002) is employed to update the HM more effectively, and an external archive is utilized in the proposed GMHS for MOPs to preserve non-dominated solutions.

The rest of this paper is organized as follows. In Sect. 2, some basic concepts of MOPs are briefly presented. Section 3 describes the basic HS algorithm. Thereafter, in Sect. 4, the detailed improvements and framework of the GMHS algorithm are presented. Benchmark functions simulations show the performance of the GMHS for MOPs in Sect. 5. In Sect. 6, finally, we give the conclusion and discussion.

2 Basic concepts of MOPs

Formally, a MOP can be mathematically defined as follows: (Zhang and Li 2007):

$$\begin{cases} \text{minimize } \ \boldsymbol{F}(\boldsymbol{x}) = (f_1(\boldsymbol{x}), f_2(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x})) \\ \text{subject to } \ \boldsymbol{x} \in S, \end{cases} \tag{1}$$

where S is the decision variable space, $F : S \longrightarrow R^m$ consists of m real-valued objective functions and R^m is the objective space. The decision vector $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$ belongs to the search space S and S can be described by

$$S = \{\boldsymbol{x} | h_j(\boldsymbol{x}) \leq 0, \ j = 1, 2, \ldots, q\}, \tag{2}$$

where $h_j(\boldsymbol{x})$ is the set of inequality constraints and equality constraints.

To better understand multi-objective optimization, the brief introduction of the basic Pareto concepts is necessary. Therefore, Pareto dominance, Pareto optimality, Pareto optimal set and Pareto front (PF) are defined as follows (Van Veldhuizen and Lamont 2000):

Definition 2.1 (Pareto Dominance): a vector $\boldsymbol{u} = (u_1, u_2, \ldots, u_m)$ dominates another vector $\boldsymbol{v} = (v_1, v_2, \ldots, v_m)$ (denoted by $\boldsymbol{u} \preceq \boldsymbol{v}$) if and only if $\boldsymbol{u}$ is partially less than $\boldsymbol{v}$, i.e.,, $\forall i \in \{1, 2, \ldots, m\}, u_i \leq v_i \wedge \exists i \in \{1, 2, \ldots, m\} : u_i < v_i$.

Definition 2.2 (Pareto Optimality): a solution $\boldsymbol{x}^* \in S$ is Pareto optimal solution if there is not another $\boldsymbol{x} \in S$ satisfies with $\boldsymbol{F}(\boldsymbol{x}) \preceq \boldsymbol{F}(\boldsymbol{x}^*)$.

Definition 2.3 (Pareto Optimal Set): for a given MOP $\boldsymbol{F}(\boldsymbol{x})$, the Pareto optimal set is defined as the set of all non-dominated solutions with respect to S, i.e., $P^* = \{\boldsymbol{x} \in S | \neg\exists \boldsymbol{x}' \in S : \boldsymbol{F}(\boldsymbol{x}') \preceq \boldsymbol{F}(\boldsymbol{x})\}$.

Definition 2.4 (Pareto Front): for a MOP $\boldsymbol{F}(\boldsymbol{x})$ and Pareto optimal set P^*, the PF is defined as the following set: $PF^* = \{\boldsymbol{F}(\boldsymbol{x}) | \boldsymbol{x} \in P^*\}$.

Most MOPs may have many or even infinite Pareto optimal solutions. In most cases, it is impossible to obtain an analytical expression of the line or surface that contains these points (Coello et al. 2004). However, it is necessary to obtain a manageable number of Pareto optimal vectors which are evenly distributed along the PF, and thus good representative of the entire PF (Zhang and Li 2007). In order to obtain the approximated PF, the normal procedure is to compute the feasible domain S and their corresponding $\boldsymbol{f}(S)$ by applying specific algorithms.

Algorithm 1: The pseudo-code of the basic HS

```
Set $HMS, HMCR, bw, PAR, NI$
$x_{i,j} = LB_j + rand() \cdot (UB_j - LB_j), \forall i = 1, 2, \ldots, HMS$ and $\forall j = 1, 2, \ldots, n$ //initialize the HM
Calculate($f(X_i)$), $\forall i = (1, 2, \ldots, HMS)$; $X_i = (x_{i,1}, x_{i,2}, \ldots, x_{i,n})$
Sort(HM)
$it = 0$
while $it \le NI$ do
  $X_{new} = \varnothing$
  for $j = 1, \ldots, n$ do
    if $rand() \le HMCR$ then
      $x_{new,j} \in \{x_{1,j}, x_{2,j}, \ldots, x_{HMS,j}\}$ // memory consideration
      if $rand() \le PAR$ then
        $x_{new,j} = x_{new,j} \pm rand() \cdot bw$ //pitch adjustment
      end if
    else
      $x_{new,j} = LB_j + rand() \cdot (UB_j - LB_j)$ // random selection
    end if
  end for
  if $f(X_{new}) < f(X_{worst})$ then
    Include $X_{new}$ to HM and exclude $X_{worst}$ from HM // update the HM
  end if
  $it = it + 1$
end while
```

3 The basic HS algorithm

In the basic HS algorithm, each solution is called a "harmony" and expressed by an n-dimension real vector (Geem et al. 2001). First, an initial population of harmony vectors is generated randomly and stored in a harmony memory (HM). Second, a new harmony is improvised based on all harmonies stored in the HM by applying an improvisation scheme. Finally, the HM will be updated if the fitness value of the newly generated harmony is better than that of the worst one in the HM. The above process is repeated until a predefined stopping condition is met. The procedure of the basic HS algorithm is shown in Algorithm 1 and it can be delineated as follows:

Step 1: Initialize the problem and HS parameters. The global optimization problem is formulated as follows:

$$\text{Minimize} \quad f(\boldsymbol{x}) \quad \text{subject to} \quad x_j \in [\text{LB}_j, \text{UB}_j], \quad j = 1, 2, \ldots, n, \tag{3}$$

where $f(\boldsymbol{x})$ is the objective function; $\boldsymbol{x} = (x_1, x_2, \ldots, x_n)$ is the set of each decision variable; n denotes the number of decision variables; LB_j and UB_j denote the lower and upper limits for x_j, respectively. What is more, the following parameters of HS algorithm are set in this step: harmony memory size (HMS), harmony memory considering rate (HMCR), distance bandwidth (bw), pitch adjusting rate (PAR) and the number of improvisations (NI).

Step 2: Initialize the harmony memory.
In this step, the HM matrix is filled with the HMS randomly generated harmony vectors as follows:

$$\text{HM} = \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_i \\ \vdots \\ X_{\text{HMS}} \end{bmatrix} = \begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,j} & \cdots & x_{1,n} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,j} & \cdots & x_{2,n} \\ \vdots & \vdots & \cdots & \vdots & \cdots & \vdots \\ x_{i,1} & x_{i,2} & \cdots & x_{i,j} & \cdots & x_{i,n} \\ \vdots & \vdots & \cdots & \vdots & \cdots & \vdots \\ x_{\text{HMS},1} & x_{\text{HMS},2} & \cdots & x_{\text{HMS},j} & \cdots & x_{\text{HMS},n} \end{bmatrix} \tag{4}$$

Step 3: Improvise a new harmony.
A new harmony vector $X_{new} = (x_{new,1}, x_{new,2}, \ldots, x_{new,n})$, is generated by using three rules: (a) memory consideration, (b) pitch adjustment and (c) random

selection. Generating a new harmony is typically called "improvisation" (Lee and Geem 2005).
In the memory consideration, the value of each decision variable $x_{ncw,j}$ $(j \in \{1, 2, \ldots, n\})$ for the new harmony is chosen randomly from any harmony vector $x_{i,j}$ $(i \in \{1, 2, \ldots, \text{HMS}\})$ with the probability of HMCR, while with the probability of $1 - \text{HMCR}$, $x_{ncw,j}$ is generated randomly from the feasible range. The procedure of generating new decision variable $x_{new,j}$ can be mathematically formulated as follows:

$$x_{new,j} \leftarrow \begin{cases} x_{new,j} \in \{x_{1,j},\ x_{2,j},\ \ldots,\ x_{\text{HMS},j}\}; \\ \quad \text{with probability HMCR} \\ x_{new,j} = \text{LB}_j + rand() \cdot (\text{UB}_j - \text{LB}_j); \\ \quad \text{with probability } 1 - \text{HMCR}, \end{cases} \tag{5}$$

where rand() is a random number from a uniform distribution of [0, 1]. Additionally, every $x_{new,j}$ achieved by the memory consideration is adjusted through the pitch adjustment rule with the probability of PAR, while with the proportion of $1 - \text{PAR}$, the value of $x_{new,j}$ remains unchanged. In the pitch adjustment rule, $x_{new,j}$ is adjusted as follows:

$$x_{new,j} = \begin{cases} x_{new,j} \pm rand() \cdot bw; & \text{with probability PAR} \\ x_{new,j}; & \text{with probability } 1 - \text{PAR} \end{cases} \tag{6}$$

Each variable of a new harmony is obtained using the aforementioned three rules in turn.
Step 4: Update the harmony memory.
If the newly generated harmony X_{new} is better than the worst harmony stored in the HM, judged by the objective function value, the HM will be updated. It means that the new harmony is included into the HM and the worst harmony is excluded from the HM.
Step 5: Check the stopping criterion.
If the stopping criterion is satisfied, algorithm computation is terminated. Otherwise, go to Step 3.

4 The proposed GMHS for multi-objective optimization

When the basic HS algorithm is applied to solve MOPs, the algorithm shows poor performance in escaping from local PF and poor convergence rate. In view of this, a GMHS for solving MOPs is proposed in this paper. The proposed GMHS algorithm is different from the basic HS algorithm in improvisation step and the HM updating. The modifications of improvisation process mainly include the following: (a) a newly introduced probability in the memory consideration is used to consider the corresponding harmony and other harmonies with diverse probabilities and this modification is able to enhance the convergence of the proposed algorithm; (b) chaotic maps based parameter adaptation is applied to prevent the GMHS from getting stuck on local optimum; (c) a decreasing bandwidth is used to improve the local search ability, and an adaptive bandwidth is used to improve the global search ability in the pitch adjustment phase. So another new probability is introduced to allow the algorithm to switch between exploration and exploitation over the entire search process; (d) when the improvisation is completed, a gaussian mutation operator is added into the proposed GMHS algorithm for speeding up convergence rate and avoiding getting stuck on local optimum. As for HM updating, the HM will be updated after generating HMS new harmonies instead of a harmony at each iteration. The details of the proposed algorithm are described below.

4.1 Modified memory consideration

A small HMS value is a good choice for single objective optimization (Chen et al. 2012), while a relatively big HMS is more proper for solving MOPs because multi-objective optimization is to acquire a Pareto optimal set instead of a single optimal solution (Ricart et al. 2011). As each harmony is selected with a low probability when the HMS value is big, the information of each harmony stored in HM cannot be well applied. In view of this, a modified memory consideration is proposed for better using the information of each harmony allocated in HM. The modified memory consideration can generate some promising solutions, so the convergence of HS algorithm can be improved effectively. The details of the modified memory consideration are shown as below.

Unlike the basic HS where all harmonies are chosen with same probability in the memory consideration, the corresponding harmony and other harmonies are selected with different probability by introducing a cross probability P_c in the GMHS. When generate i th harmony vector, its corresponding harmony in HM is allocated in the i th position of HM. In the modified memory consideration, the value of each decision variable $x_{new,j}$ for the new harmony is generated randomly from the possible range of values, with a probability of $1 - \text{HMCR}$, while with the probability of HMCR, $x_{new,j}$ is chose from any harmony vector $x_{i,j}$ in HM with different probabilities instead of same probability. The process of producing candidate decision variable $x_{new,j}$ can be described as follows:

$$x_{new,j} \leftarrow \begin{cases} x'_{new,j}; \\ \quad \text{with probability HMCR} \\ x_{new,j} = \text{LB}_j + rand() \cdot (\text{UB}_j - \text{LB}_j); \\ \quad \text{with probability } 1 - \text{HMCR} \end{cases} \tag{7}$$

$$x'_{new,j} \leftarrow \begin{cases} x'_{new,j} \in \{x_{1,j}, x_{2,j}, \ldots, x_{\text{HMS},j}\} \\ \quad \text{with probability } P_c \\ x_{corresponding,j} \\ \quad \text{with probability } 1 - P_c, \end{cases} \quad (8)$$

where the subscripts 'new' of $x_{new,j}$ implies that $x_{new,j}$ is newth newly generated harmony, $new \in \{1, 2, \ldots, \text{HMS}\}$. In Eq. (8), the parameter P_c is a number in the region of [0, 1]. It is the probability of randomly choosing one value from the existing values stored in the HM, while $1 - P_c$ is the probability of selecting the corresponding value stored in the HM with respect to position. It should be noted that the subscript 'corresponding' is equal to the subscript 'new' of $x_{new,j}$.

Remark 1 After the introduction of P_c, the probability of all elements in ith harmony stored in HM being selected to generate newth harmony is formulated as Eq. (9), while without P_c, this probability can be formulated as Eq. (10). It can be observed from the two equations that P_i^{HS} does not depend on P_c, while P_i^{GMHS} does. What is more, P_i^{GMHS} is almost equal to P_i^{HS} when P_c has a value close to 1 and $new \neq i$, and the P_i^{GMHS} is always much bigger than P_i^{HS} when $new = i$. It means that each harmony stored in HM has one chance to be highlighted over the process of generating the HMS harmonies at each iteration. In order to graphically illustrate the impact of P_c, Fig. 1 shows the variations of P_i^{GMHS} and P_i^{HS} over improvisations when HMS = 50, $P_c = 0.9$, $n = 1$ and $i = 10$. It can be clearly seen from Fig. 1 that the tenth harmony in HM is highlighted when performing tenth improvisations. Therefore, each harmony in HM can be better highlighted over improvisations process after the introduction of P_c.

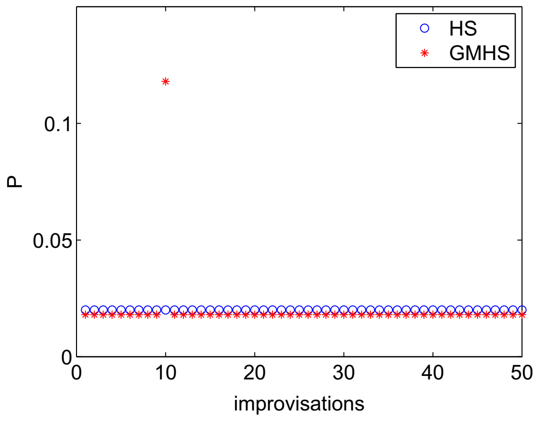


Fig. 1 Variations of P_i^{GMHS} and P_i^{HS} over improvisations when HMS = 50, $P_c = 0.9$, $n = 1$ and $i = 10$

$$P_i^{\text{GMHS}} = \begin{cases} \left(1 - P_c + \frac{P_c}{\text{HMS}}\right)^n; & new = i \\ \left(\frac{P_c}{\text{HMS}}\right)^n; & \text{otherwise} \end{cases} \quad (9)$$

$$P_i^{\text{HS}} = \left(\frac{1}{\text{HMS}}\right)^n, \quad new = 1, 2, \ldots, \text{HMS} \quad (10)$$

4.2 Chaotic map based parameter adaption for PAR

Since the parameter PAR relies on the characteristics of problem and it is really hard to adjust, it is a good choice to select an adaptive PAR. Many chaotic maps have the following main properties: ergodicity, stochastic and certainty; these properties are helpful for a global search (Yuan et al. 2014). Therefore, the PAR is updated by a chaotic map in the proposed algorithm. On the one hand, this adaptive PAR is able to improve the ability of escaping from local optimum. On the other hand, the problem of the setting of PAR has been solved.

Several well-known one-dimensional chaotic maps are presented in Yuan et al. (2014); here only two easy but typical chaotic maps used in the pitch adjustment phase of GMHS for parameter adaption are listed below:

(a) ICMIC map: iterative chaotic map with infinite collapses (ICMIC) (He et al. 2001) is formally defined by the following equation:

$$x_{n+1} = \sin\left(\frac{\alpha}{x_n}\right), \quad \alpha \in (0, \infty), \quad x_n \in (-1, 1) \quad (11)$$

ICMIC map generates chaotic sequences in $(-1, 1)$. The value of α is set to 70 in the experiments.

(b) Logistic map: logistic map generates chaotic sequences in (0, 1) and it is given by:

$$x_{n+1} = \psi x_n(1 - x_n), \quad 0 < \psi \leq 4, \quad x_n \in (0, 1) \quad (12)$$

The value of ψ is set to 4 in the experiments.

4.3 Pitch adjustment with decreasing bandwidth and adaptive bandwidth

As bw plays an important role in the convergence rate and search ability (Mahdavi et al. 2007; Zou et al. 2010), this section focuses on it. In this paper, a decreasing bandwidth is combined with an adaptive bandwidth are designed for the purpose of enhancing the global and local search ability during the whole search process. On the one hand, the adaptive bandwidth can overcome the shortcoming when only a decreasing bandwidth is applied; what is more, the global search ability of adaptive bandwidth during entire search process and the local search ability of the decreasing bandwidth at latter iterations are obtained. On the other hand, the adaptive bandwidth is dynamically updated and it also can reduce the number of parameter settings. A probability P_{bw} is introduced into GMHS for the purpose of getting a good balance between exploration and exploitation. In the pitch adjustment, the first kind of decreasing bandwidth is used with the probability of P_{bw}, while with the probability of $1 - P_{bw}$, the second kind of adaptive bandwidth is selected. The detailed procedure of pitch adjustment works as Algorithm 2:

$$bw(t)_{1,j} = bw_{1,j,\min} + (bw_{1,j,\max} - bw_{1,j,\min}) \times \left(\frac{T-t}{T}\right)^{\varphi} \quad (13)$$

$$bw_{1,j,\max} = \frac{\mathrm{UB}_j - \mathrm{LB}_j}{2 \cdot \mathrm{HMS}} \quad (14)$$

In Algorithm 2, $bw(t)_{1,j}$ represents the first kind of bandwidth of jth variable in iteration t which is given by Eq. (13). In Eq. (13), $bw_{1,j,\max}$ is the maximum bandwidth values of the jth decision variable which is given by Eq. (14), φ is a positive value and T is the maximal iteration number, $bw_{1,j,\min}$ is the minimum bandwidth value of the jth decision variable. The $bw(t)_{2,j}$ represents the second kind of bandwidth of jth decision variable in iteration t which changes adaptively, $N(|x_{r1,j} - x_{r2,j}|, \; |x_{r1,j} - x_{r2,j}|/10)$ is a value that is normally distributed with mean $|x_{r1,j} - x_{r2,j}|$ and standard deviation $|x_{r1,j} - x_{r2,j}|/10$. The decision variables $x_{r1,j}$ and $x_{r2,j}$ are randomly chosen from different harmonies stored in HM. It should be noted that the PAR is updated by Logistic map before every pitch adjustment is performed.

Algorithm 2: The pseudo-code of pitch adjustment in the GMHS

```
Compute $PAR$ according to Logistic map
if $rand() \le PAR$ then
  if $rand() \le P_{bw}$ then
    Generate chaotic variable $\gamma_1$ according to ICMIC map
    $x_{new,j} = x_{new,j} + bw(t)_{1,j} \cdot \gamma_1$   // the first type of pitch adjustment
  else
    Generate chaotic variable $\gamma_2$ according to ICMIC map
    $r1$= randomly generated positive integers in the region of $[1, HMS]$
    $r2$= randomly generated positive integers in the region of $[1, HMS]$
    while $r1 == r2$ do
      $r2$= randomly generated positive integers in the region of $[1, HMS]$
    end while
    $bw(t)_{2,j} = N(|x_{r1,j} - x_{r2,j}|, \; |x_{r1,j} - x_{r2,j}|/10)$
    $x_{new,j} = x_{new,j} + bw(t)_{2,j} \cdot \gamma_2$   // the second type of pitch adjustment
  end if
  if $x_{new,j} > UB_j$ then
    $x_{new} = UB_j$
  end if
  if $x_{new,j} < LB_j$ then
    $x_{new,j} = LB_j$
  end if
end if
```

Though two parameters P_c and P_{bw} are introduced in the proposed algorithm, the P_{bw} is set to a fixed value. It means that the setting for the P_{bw} is unnecessary because the fixed value of P_{bw} is helpful for the GMHS to generate a good result. What is more, it is unnecessary to set PAR since it computed by a chaotic map. Therefore, the newly proposed GMHS has the same number of parameter settings with the basic HS.

4.4 Gaussian mutation operator

It is known that HS algorithm is able to converge to global optimum for single objective optimization, but this convergence can be disrupted when the algorithm is used to solve MOPs, e.g., the four variants of HS in Pavelski et al. (2012) has shown poor convergence for CEC 2009 problems. So, this shortcoming of HS algorithm has yet to be overcome. Mutation operator is really helpful for many optimization algorithms' global optimum and gaussian mutation operator is one of them. What is more, the gaussian mutation has used a gaussian density function which is shown in Fig. 2. It can be seen from Fig. 2 that the bigger the distance between the newly mutated value and the mean value μ, the smaller the probability of getting the newly mutated value. It means that the disturbance has big probability near the mean value, and the disturbance has small probability far from the mean value, which is different from other disturbance with same probability. This characteristic is effective for greatly improving local search ability, and it also has global search ability to some extent. Therefore, a gaussian mutation operator is added into the GMHS for speeding up convergence rate and enhancing the ability of escaping from local optimum after the completion of every improvisation. The procedure of the gaussian mutation operator in the GMHS is described as Algorithm 3.

$$\sigma = \frac{\mathrm{UB}_j - \mathrm{LB}_j}{kx} \quad (15)$$

In Algorithm 3, $N(x_{new,j}, \; \sigma)$ is a value that is normally distributed with mean $x_{new,j}$ and standard deviation σ. σ usually depends on the length $\triangle_x = \mathrm{UB}_j - \mathrm{LB}_j$ and it is computed by Eq. (15). In Eq. (15), kx is a coefficient for controlling the value of σ and it is equal to 20. The gaussian mutation probability P_{gm} usually has a small value; it is equal to $1/n$ (n is the number of decision variables) when the number of decision variables is bigger than 1, and it is equal to 0.1 when the number of decision variables is equal to 1.

It can be seen from Algorithm 3 that the value $x_{new,j}$ of jth decision variable remains unchanged with the probability of $1 - P_{gm}$, while $x_{new,j}$ is updated by gaussian mutation

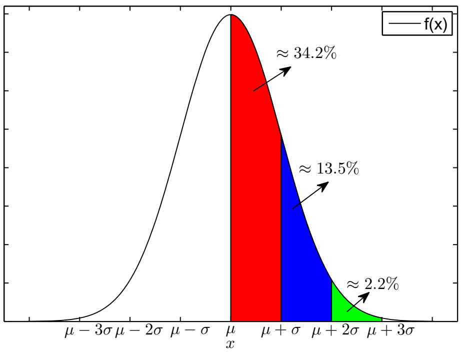


Fig. 2 Gaussian probability density function with mean μ and standard deviation σ

Algorithm 3: The pseudo-code of gaussian mutation operator in the GMHS

```
for $j = 1 : n$ do
  if $rand() < P_{gm}$ then
    $x_{new,j}^{mut} = N(x_{new,j}, \sigma)$
    if $x_{new,j}^{mut} < LB_j$ then
      $x_{new,j}^{mut} = LB_j$
    end if
    if $x_{new,j}^{mut} > UB_j$ then
      $x_{new,j}^{mut} = UB_j$
    end if
    $x_{new,j} = x_{new,j}^{mut}$
  else
    $x_{new,j} = x_{new,j}$
  end if
end for
```

step with the probability of P_{gm}. In the gaussian mutation step, a mutated value $x_{new,j}^{mut}$ is generated by a gaussian function $N(x_{new,j}, \sigma)$, and the value of $x_{new,j}^{mut}$ is confined in $[LB_j, UB_j]$. Then $x_{new,j}$ is displaced by the mutated value $x_{new,j}^{mut}$.

4.5 Update the HM

The HM updating in the proposed GMHS algorithm for MOPs is different from that of basic HS algorithm. A fast non-dominated sorting and crowding distance method proposed by Deb et al. (2002) has been used to update the HM in Sivasubramani and Swarup (2011). In this paper, the HM updating is the same as the method used in Sivasubramani and Swarup (2011), so the procedure of HM updating is briefly described as follows: First, generating HMS new harmonies by improvisation process and gaussian mutation operator, and these new harmonies form a new HM. Second, the new HM is combined with the existing HM, and the combined HM is of size $2 \cdot$ HMS. Then the combined HM is sorted based on a fast non-dominated sorting and crowding distance method. Ultimately, selecting HMS solutions from the combined HM in the sequence of their ranking to constitute the best HM for the next iteration.

4.6 Update the external archive

Many multi-objective algorithms (Zitzler and Thiele 1999; Wang et al. 2010) have shown that it is necessary to apply an external archive to retain non-dominated solutions. In our proposed algorithm, an external archive is used to keep the non-dominated solutions found so far during evolution, so the external archive needs to be updated after generating new solutions on each iteration. When the number of non-dominated solutions found in the optimization process exceeds the maximum size of external archive, a good measure method plays an important role in maintaining the diversity of Pareto optimality. A crowding distance method proposed by Deb et al. (2002), which is an easy but useful method to measure the crowding degree of solutions surrounding a solution, has been introduced into the GMHS for bi-objective problems. However, Shang et al. (2014) have pointed out that this method is not feasible for problems of over two objectives. In view of this, a truncation method in SPEA2 (Zitzler et al. 2001), which has the ability to maintain uniformity and diversity of the distribution of solutions, has been introduced into the GMHS for problems of over two objectives. The procedure of updating the external archive is described as Algorithm 4.

In Algorithm 4, A_t is the external archive in iteration t, HM$_{t+1}$ denotes the best HM for improvisation in iteration $t + 1$. N is the maximum size of the external archive. Firstly, find all non-dominated solutions in $A_t \cup HM_{t+1}$, and copy all these non-dominated solutions to A_{t+1}. Secondly, compute the size of A_{t+1}. If the size of A_{t+1} is exceeds the maximum size N, for bi-objective problems, a crowding distance method will be utilized to cut down the size of A_{t+1} until the size of A_{t+1} is equal to N, for problems of over two objec-

Algorithm 4: The pseudo-code of updating the external archive in the GMHS

```
$A_{t+1}$ = all non-dominated solutions found in $A_t \bigcup HM_{t+1}$
$M$ = the size of $A_{t+1}$
while $M > N$ do
  if The problem is of two objectives then
    All solutions in $A_{t+1}$ are assigned a crowding distance
    value using crowding distance metric (Deb et al. 2002).
    The solution with the smallest crowding distance value is
    deleted.
  else
    An archive truncation procedure in Zitzler et al. (2001) is
    adopted, the solution which has the minimum distance to
    another solution is deleted.
  end if
end while
```

tives, a truncation method is utilized. Otherwise, all these non-dominated solutions will be retained in the archive A_{t+1}.

4.7 Implementation of GMHS

The procedure of GMHS for MOPs is described as Algorithm 5.

5 Experimental results and the analysis

In this section, we first present the test problems and quality indicators used to compare the search ability of GMHS with other MOEAs. Then, the comparisons of GMHS with other MOEAs are described and discussed. Second, the performance of GMHS has been further investigated using more complicated test problems and other quality indicator. In addition, the comparisons between GMHS with other multi-objective HS algorithms have been conducted. Finally, the sensitivities to probability P_c and HMS in GMHS have been studied.

5.1 Comparisons of GMHS with other MOEAs

For the purpose of investigating the performance of the proposed algorithm, here GMHS is compared with three classical MOEAs: MOSADE (Wang et al. 2010), NSGA-II (Deb et al. 2002), SPEA2 (Zitzler et al. 2001).

Algorithm 5: THE PSEUDO- CODE OF GMHS FOR MOPS

```
Input: $HMS$, $N$ (archive size), $T$ (maximal iteration number),
       $HMCR$, $P_c$, $P_{bw}$, $bw_{1,j,min}$
Output: $A$ (non-dominated set)
1 Step 1: Initialize the algorithm parameters. Set the iteration
  number $t = 0$ and generate an initial harmony memory $HM_t$
  based on the problem range and constraints if any, evaluate all
  harmonies in $HM_t$ and create the empty external archive $A_t = \emptyset$.
2 Step 2: Improvise a new HM. Perform improvisation and
  gaussian mutation operator in the GMHS for $HMS$ times to
  generate a new harmony memory $HM_t^{new}$ and evaluate
  objectives functions of each harmony in $HM_t^{new}$.
3 Step 3: Update HM. Select the best harmony memory $HM_{t+1}$
  from the combined harmony memory $HM_t \bigcup HM_t^{new}$ as given
  in Section 4.5 for the next iteration.
4 Step 4: Update external archive. Get an updated external archive
  $A_{t+1}$ by executing Algorithm 4 in Section 4.6.
5 Step 5: Check termination criterion. If $t \geq T$ is reached, set $A$ to
  the set of solutions expressed by the non-dominated solutions in
  $A_{t+1}$, output the non-dominated set $A$ and terminate the
  algorithm. Otherwise, increase iteration counter ($t = t + 1$) and
  go to Step 2.
6 return $A$
```

5.1.1 Test problems

Fourteen classical test problems, which are frequently used in the MOEAs literature, are selected to test the performance of the GMHS. They are all unconstrained problems, including eight bi-objective problems: Schaffer, Fonseca and Kursawe (Deb et al. 2002), as well as ZDT1, ZDT2, ZDT3, ZDT4 and ZDT6 (Zitzler et al. 2000), and six tri-objective problems: DTLZ1, DTZL2, DTLZ4, DTLZ5, DTLZ6 and DTLZ7 (Deb et al. 2001).

5.1.2 Performance metrics

In this section, two quality indicators are used to measure the performance of the four algorithms. They are GD (Van Veldhuizen and Lamont 2000) and Spread ($\triangle$) which are frequently utilized in many MOEAs.

1. Generational distance (GD): the GD was introduced in Van Veldhuizen and Lamont (2000), it is used to measure how far the elements are in the set of non-dominated vectors found so far from those in the Pareto optimal set. This convergence indicator is calculated as

$$\mathrm{GD} = \frac{\sqrt{\sum_{i=1}^{N} d_i^2}}{N}, \tag{16}$$

where N is the number of solutions in the approximated PF, and d_i is the Euclidean distance (measured in objective space) between each solution in the approximated PF and the nearest solution in the real PF.

2. Spread ($\triangle$): this indicator $\triangle$ is to measure the spread in the obtained non-dominated solutions. This indicator $\triangle$ in Deb et al. (2002) is defined as Eq. (17) for evaluating bi-objectives problems; as for problems of over two objectives, the revised indictor $\triangle$ in Zhou et al. (2006) is defined as Eq. (18).

$$\triangle = \frac{d_f + d_l + \sum_{i=1}^{N-1} |d_i - \bar{d}|}{d_f + d_l + (N-1)\bar{d}}, \tag{17}$$

where N is the size of the obtained non-dominated set. The parameters d_f and d_l are the Euclidean distances (measured in objective space) between the extreme solutions and the boundary solutions of the obtained non-dominated set. The parameter d_i denotes the Euclidean distance between the adjacent solutions in the obtained non-dominated set of solutions and $\bar{d}$ is the average of all d_i.

$$\triangle = \frac{\sum_{i=1}^{m} d(E_i, \Omega) + \sum_{X \in \Omega} |d(X, \Omega) - \bar{d}|}{\sum_{i=1}^{m} d(E_i, \Omega) + (|\Omega| - m)\bar{d}}, \tag{18}$$

Table 1 Comparison results of GMHS and other MOEAs based on the GD

Algorithms	GMHS		MOSADE		NSGA-II		SPEA2	
Test problems	Mean	SD	Mean	SD	Mean	SD	Mean	SD
Schaffer	**3.6357e−4**	**1.7199e−5**	2.0034e−3	1.1321e−4	2.1572e−3	2.0999e−4	2.1232e−3	2.1130e−4
Kursawe	**1.5143e−3**	**1.7174e−4**	2.4435e−2	2.6424e−3	2.8974e−3	2.3637e−4	7.1576e−1	1.2523e−2
Fonseca	**1.1885e−3**	**1.9888e−5**	1.2402e−3	7.0682e−5	2.5656e−3	2.0082e−4	1.8573e−3	1.0731e−4
ZDT1	**2.3978e−4**	**2.6997e−5**	1.2485e−3	9.7574e−5	1.3437e−3	1.4078e−4	8.6104e−3	2.5973e−3
ZDT2	**1.2801e−4**	**1.1824e−5**	9.8051e−4	4.9107e−5	9.8112e−4	6.4138e−4	2.4766e−2	1.6083e−2
ZDT3	**6.2747e−4**	**2.5995e−5**	2.1620e−3	1.9962e−4	2.4783e−3	1.2746e−4	9.7165e−3	5.2305e−3
ZDT4	**4.4755e−4**	**5.9666e−5**	1.2014e−3	8.3745e−5	5.1635e−2	1.3281e−3	9.2512e−1	4.2821e−1
ZDT6	**1.2143e−4**	**9.7372e−6**	2.6156e−3	1.0967e−4	7.5818e−2	6.0797e−3	1.9309e−2	1.3994e−3
DTLZ1	**1.6446e−3**	**1.6994e−4**	5.2512e−3	1.5650e−4	2.7102e−2	6.6459e−2	9.0259e−1	7.6880e−1
DTLZ2	**2.7946e−3**	**4.9773e−4**	5.3344e−3	1.7429e−4	1.0189e−1	1.0559e−1	2.8136e−2	1.2532e−2
DTLZ4	**3.4710e−3**	**3.9053e−4**	4.7736e−3	2.7427e−4	1.0623e−2	7.9836e−2	1.7284e−2	9.9114e−2
DTLZ5	**4.8468e−4**	**3.2014e−5**	3.8930e−3	2.0137e−4	4.7763e−3	4.0435e−4	4.2982e−3	4.4141e−4
DTLZ6	**4.5499e−4**	**2.5584e−5**	3.7701e−3	3.1470e−4	4.5411e−1	2.0164e−1	4.7564e−1	1.2177e−1
DTLZ7	3.4598e−3	2.7990e−4	**2.3737e−3**	**0.001303**	2.8147e−2	1.8194e−2	1.4919e−1	6.9278e−2

The best value is highlighted bold

Table 2 Comparison results of GMHS and other MOEAs based on the Spread indicator ($\triangle$)

Algorithms	GMHS		MOSADE		NSGA-II		SPEA2	
Test problems	Mean	SD	Mean	SD	Mean	SD	Mean	SD
Schaffer	0.19916	1.4498e−2	**0.13505**	**1.8842e−2**	0.29228	2.1339e−2	0.27503	2.5711e−2
Kursawe	0.33525	9.7980e−3	**0.28220**	**2.1026e−2**	0.43404	1.9818e−2	0.28611	1.1474e−2
Fonseca	0.18771	1.0057e−2	**0.11581**	**6.5733e−3**	0.37672	2.5222e−2	0.17661	1.1100e−1
ZDT1	0.18893	1.2579e−2	**0.13195**	**5.6921e−3**	0.50429	3.9251e−2	0.29644	1.0850e−1
ZDT2	0.18895	9.0021e−3	**0.12099**	**7.9444e−3**	0.48775	2.7686e−2	0.50517	1.8356e−1
ZDT3	0.58591	8.5933e−3	**0.43783**	**8.0801e−3**	0.59025	3.0439e−2	0.50310	9.7283e−2
ZDT4	0.21870	2.0433e−2	**0.11827**	**5.8619e−3**	0.37524	2.4448e−2	0.72766	5.1517e−1
ZDT6	0.17753	1.0306e−2	**0.13319**	**9.8303e−3**	0.48611	3.6054e−2	0.24861	4.9667e−2
DTLZ1	**0.10069**	**9.4112e−3**	0.63755	3.0464e−2	0.91867	6.6108e−2	0.30236	2.4919e−1
DTLZ2	**0.10879**	**1.2310e−2**	0.61323	3.6732e−2	0.83092	6.8427e−2	0.24673	3.5181e−2
DTLZ4	**0.11077**	**8.4005e−3**	0.83230	1.0897e−1	0.77864	7.8099e−2	0.25568	7.5616e−2
DTLZ5	**0.14858**	**1.6162e−2**	0.35529	2.2054e−2	0.71080	7.8009e−2	0.29840	1.8370e−2
DTLZ6	**0.14821**	**9.0988e−3**	0.30720	4.7503e−2	1.08940	1.8259e−1	0.50815	9.2214e−2
DTLZ7	**0.13490**	**1.4293e−2**	0.85454	3.8634e−2	1.00960	5.0572e−2	0.39790	7.0304e−2

The best value is highlighted bold

where Ω is a set of known Pareto optimal solutions, $(E_i, \ldots, E_m)$ are m extreme solutions in Ω, m is the number of objectives and

$$d(X, \Omega) = \min_{Y \in \Omega, Y \neq X} \|F(X) - F(Y)\| \tag{19}$$

$$\overline{d} = \frac{1}{|\Omega|} \sum_{X \in \Omega} d(X, \Omega) \tag{20}$$

5.1.3 Algorithms parameter settings

In the experiments, the real-coded version of NSGA-II uses the following parameter values: a cross probability of 0.9, a mutation probability of $1/n$ (n is the number of decision variables), SBX and polynomial mutation with distribution indexes of $\eta_c = 20$ and $\eta_m = 20$ respectively, and a population size of 100. The parameter values used in the SPEA2 are the same as NSGA-II, including: cross probability, muta-

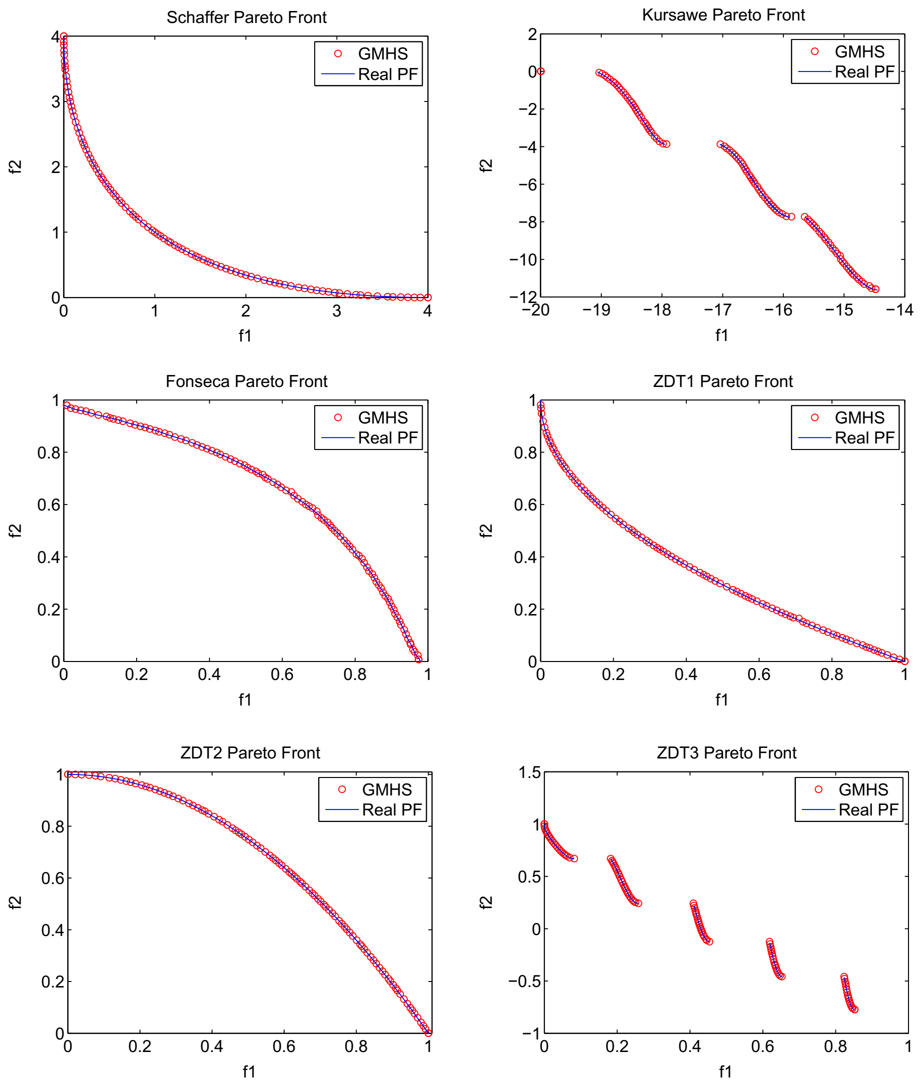


Fig. 3 Plots of the non-dominated solutions in the objective space of the Schaffer, Kursawe, Fonseca, ZDT1, ZDT2 and ZDT3

tion probability, distribution indexes, and population size. The MOSADE uses the following parameter values: a population size of 100, an external archive size of 100, the lower and upper limits of mutant constant and crossover probability $F_l = 0.1$, $F_u = 0.9$, $CR_l = 0$ and $CR_u = 0.5$, respectively. Parameters used in the GMHS are as follows: HMCR $= 0.99$, $P_c = 0.9$, HMS $= 100$, $bw_{1,j,\min} =$ 1e$-$4 and an external archive size of 100. As for parameter φ, it is easy to know that too small of φ can lead to low convergence rate but big probability of escaping from local optimum, too big of φ has high local search capability while it is easily trapped into the local optimum. To make a good trade-off in the performance, φ is set to 2 for all experiments in this paper. In addition, the GMHS can get a better balance between exploration and exploitation when the parameter P_{bw} is equal to 0.6, so it is set to this value for all experiments in this paper. To make a fair comparison, the four algorithms are executed under the same conditions, including a population size of 100; the maximum function evaluation is set to 25,000 for bi-objective problems and 50,000 for tri-objective problems.

5.1.4 Experimental results and comparisons

The previously presented performance metrics using the algorithms GMHS, MOSADE, NSGA-II and SPEA2 are reported in Tables 1 and 2. The experimental results of the algorithms MOSADE, NSGA-II and SPEA2 are available in the literature (Wang et al. 2010). Each problem is run 30 independent replications and the following statistical parameters are used: the mean value (Mean) and the standard deviation of the mean value (SD).

It can be seen from Table 1 that the GMHS generates 13 best results out of 14 test problems for the GD indicator. More specifically, comparing with MOSADE, the GMHS produces much better results for almost all test problems except DTLZ7. While comparing with algorithms NSGA-II

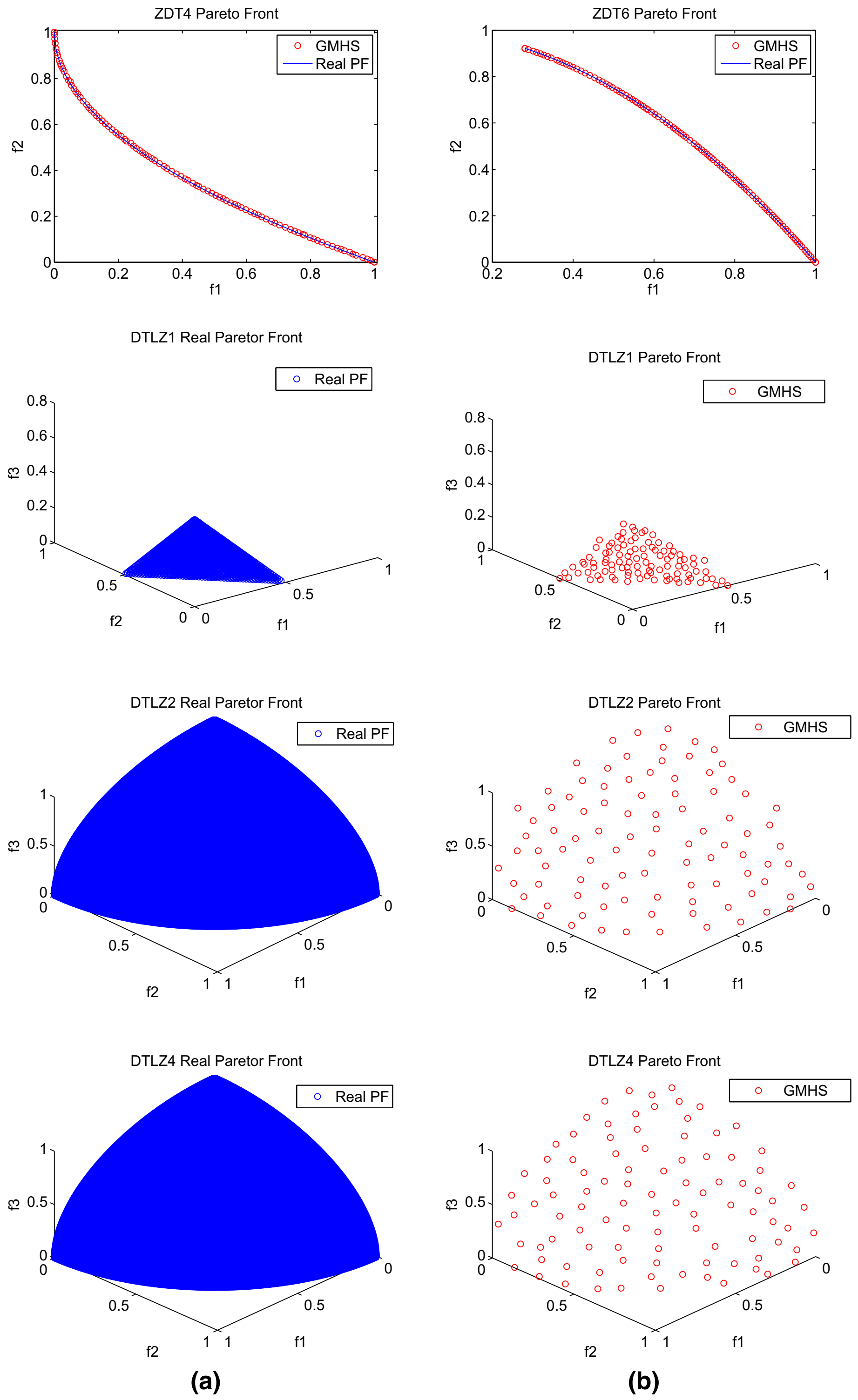


Fig. 4 Plots of the non-dominated solutions in the objective space of the ZDT4, ZDT6, DTLZ1, DTZL2 and DTLZ4

and SPEA2, the GMHS obtains much better results for all test problems. This means that the approximated Pareto fronts obtained by GMHS are closer to the optimal Pareto fronts than those obtained by MOSADE, NSGA-II and SPEA2.

Table 2 shows that the GMHS produces six best results out of 14 test problems for the Spread indicator. More specifically, comparing with NSGA-II, GMHS generates better results for all bi-objective problems. Here GMHS is compared with NSGA-II for bi-objective problems, because GMHS and NSGA-II both apply crowding distance as density estimation of solutions for bi-objective problems. It is apparent that GMHS outperforms NSGA-II concerning

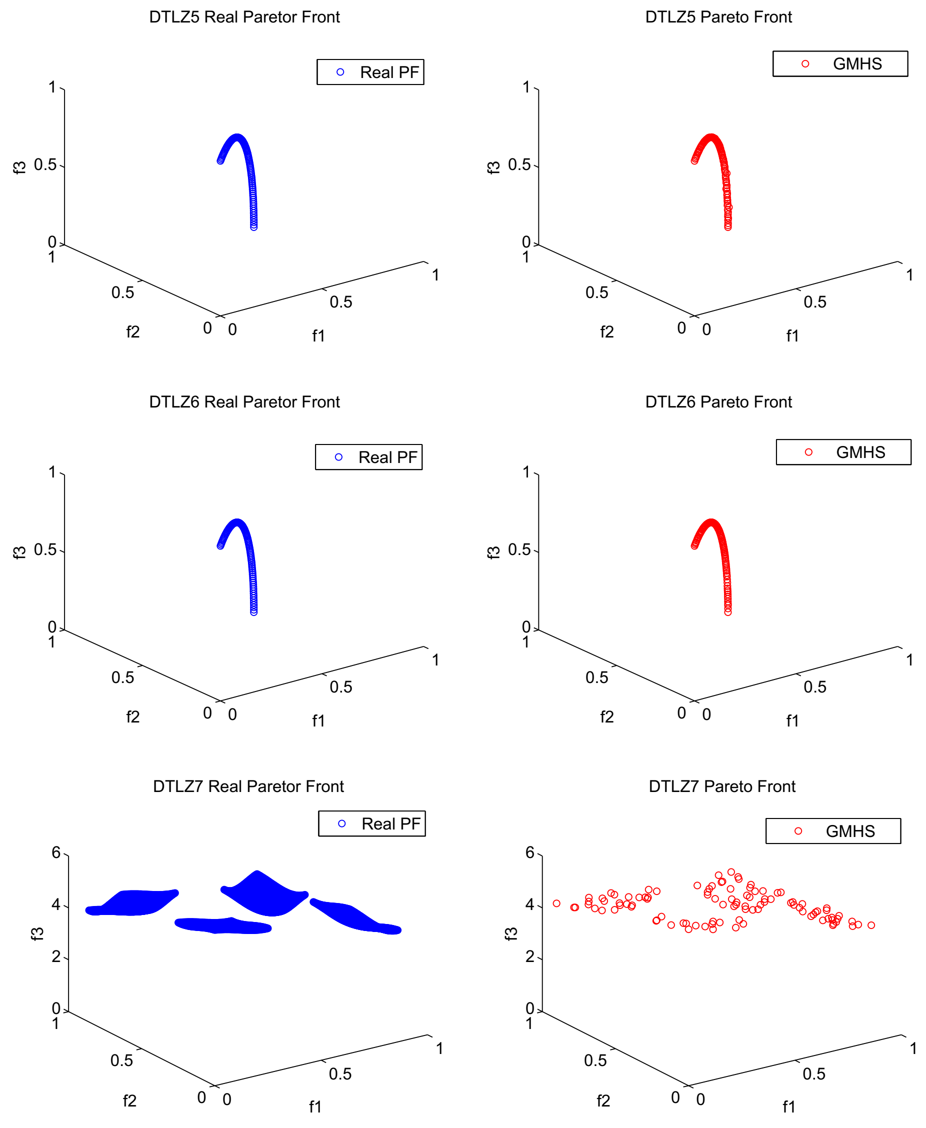


Fig. 5 Plots of the non-dominated solutions in the objective space of the DTLZ5, DTLZ6 and DTLZ7

the diversity of the approximated Pareto fronts for bi-objective problems. While comparing with SPEA2, GMHS yields much better results for all tri-objective problems. Here GMHS are compared with SPEA2 for tri-objective problems, because GMHS and SPEA2 both use the nearest neighbor density estimation method. Obviously, GMHS outperforms NSGA-II concerning the diversity of the approximated Pareto fronts for tri-objective problems.

In order to graphically illustrate the approximated Pareto fronts for all test problems, Figs. 3, 4 and 5 demonstrate the location of the approximated set of non-dominated solutions in the objective space obtained by GMHS. It can be seen from these figures that these approximated Pareto fronts generated by GMHS for all test problems are distributed evenly along the real Pareto fronts of corresponding test problems. These figures clearly show that GMHS is able to converge to the real Pareto fronts for all test problems. In addition, the convergence and uniformity of approximated Pareto fronts illustrated in Figs. 3, 4 and 5 confirm that the experimental results in Tables 1 and 2 are convincing.

In order to support above conclusions, analysis of mean values of the two performance indicators using Wilcoxon test (Garca et al. 2009) is implemented. The Wilcoxon test is a pair test and its goal is to detect significant differences between the performance of the two compared algorithms. The computations in relation to this test are given by

$$R^{+} = \sum_{d_i>0} rank(d_i) + \frac{1}{2}\sum_{d_i=0} rank(d_i) \tag{21}$$

$$R^{-} = \sum_{d_i<0} rank(d_i) + \frac{1}{2}\sum_{d_i=0} rank(d_i), \tag{22}$$

where d_i is the difference between the performance scores of the two algorithms on ith out of N' benchmark test functions.

Table 3 Comparison between GMHS and MOEAs with Wilcoxon test

Algorithms comparison	GD			△		
	R^+	R^-	P value	R^+	R^-	P value
GMHS vs. MOSADE	100.00	5.00	0.003	69.00	36.00	0.300
GMHS vs. NSGA-II	105.00	0.00	0.001	105.00	0.00	0.001
GMHS vs. SPEA2	105.00	0.00	0.001	97.00	8.00	0.005

Table 4 Experiment parameters values for NSHS, NSIHS, NSGHS and GMHS

	HMCR	PAR	bw	HMS	Maximum function evaluation
NSHS	0.95	0.10	$0.01 \times \triangle_x$	200	150,000
NSIHS	0.95	$\text{PAR}_{\min} = 0.01$	$bw_{\min} = 0.0001$	200	150,000
		$\text{PAR}_{\min} = 0.20$	$bw_{\max} = 0.05 \times \triangle_x$		
NSGHS	0.95	$\text{PAR}_{\min} = 0.01$		200	150,000
		$PAR_{\min} = 0.40$			
GMHS	0.99		$bw_{1,j,\max}$ is computed by Eq. (14)	200	150,000
			$bw_{1,j,\min} = 1\text{e}{-4}$		

Table 5 Comparison results of GMHS and other multi-objective harmony search algorithms based on the IGD

Algorithms	GMHS		NSHS		NSIHS		NSGHS	
Test problems	Mean	SD	Mean	SD	Mean	SD	Mean	SD
UF1	**0.063518**	**4.7839e−3**	0.106106	5.2735e−3	0.106848	7.6893e−3	0.109347	1.0606e−2
UF2	**0.027386**	**2.3172e−3**	0.089148	4.7562e−3	0.087283	4.8692e−3	0.083667	4.6124e−3
UF3	0.430354	1.3222e−2	0.267328	2.1836e−2	**0.264460**	**2.1703e−2**	0.293428	2.4063e−2
UF4	**0.042272**	**4.9415e−4**	0.083904	2.3508e−3	0.084573	2.5793e−3	0.078500	2.3308e−3
UF5	**0.205940**	**9.0791e−2**	0.227786	9.4543e−2	0.226528	8.7330e−2	0.223122	7.4646e−2
UF6	**0.207535**	**1.6516e−1**	0.229601	4.3604e−2	0.255578	6.8620e−2	0.282839	7.2029e−2
UF7	**0.031491**	**2.3243e−3**	0.171527	1.2021e−1	0.174940	1.1207e−1	0.229019	1.1151e−1

The best value is highlighted bold

R^+ is the sum of ranks for the benchmark test function on which the second algorithm is superior to the first, and R^- is the sum of ranks for the opposite. So a bigger value R^+ means that the first algorithm is superior to the second algorithm. The Family Wise Error Rate (FWER) (Garca et al. 2009) is defined as the probability of making one or more false detections among all the hypotheses when implementing multiple pairwise tests. The results obtained by the four algorithms are analyzed independently for all test functions by computing the P value in a pairwise comparison. All the computations of the P value for this test are implemented by the well-known statistical software package SPSS.

The results of the Wilcoxon test for GMHS against MOSADE, NSGA-II and SPEA2 algorithms are reported in Table 3. It can be observed that the proposed GMHS is much better than MOSADE, NSGA-II and SPEA2 with a level of significance $\alpha = 0.01$ (P value of FWER is 0.005) for GD. As for Spread indicator (△), GMHS outperforms NSGA-II and SPEA2 with a level of significance $\alpha = 0.01$ (P value of FWER is 0.006); however, the difference between GMHS and MOSADE is not evident. According to above experimental results and analysis, it can be concluded that GMHS algorithm outperforms the MOSADE, NSGA-II and SPEA2 algorithms in convergence performance and that GMHS obtains a better diversity performance than NSGA-II and SPEA2.

5.2 Comparison between GMHS and other multi-objective harmony search algorithms

For the purpose of studying the performance of the proposed algorithm because of the improved harmony search algorithm, the GMHS is compared with NSHS, NSIHS and NSGHS proposed by Pavelski et al. (2012). Since NSHS, NSIHS, NSGHS and GMHS all use fast non-dominated sorting and crowding distance method to update the HM, it is fair to demonstrate the performance of the GMHS.

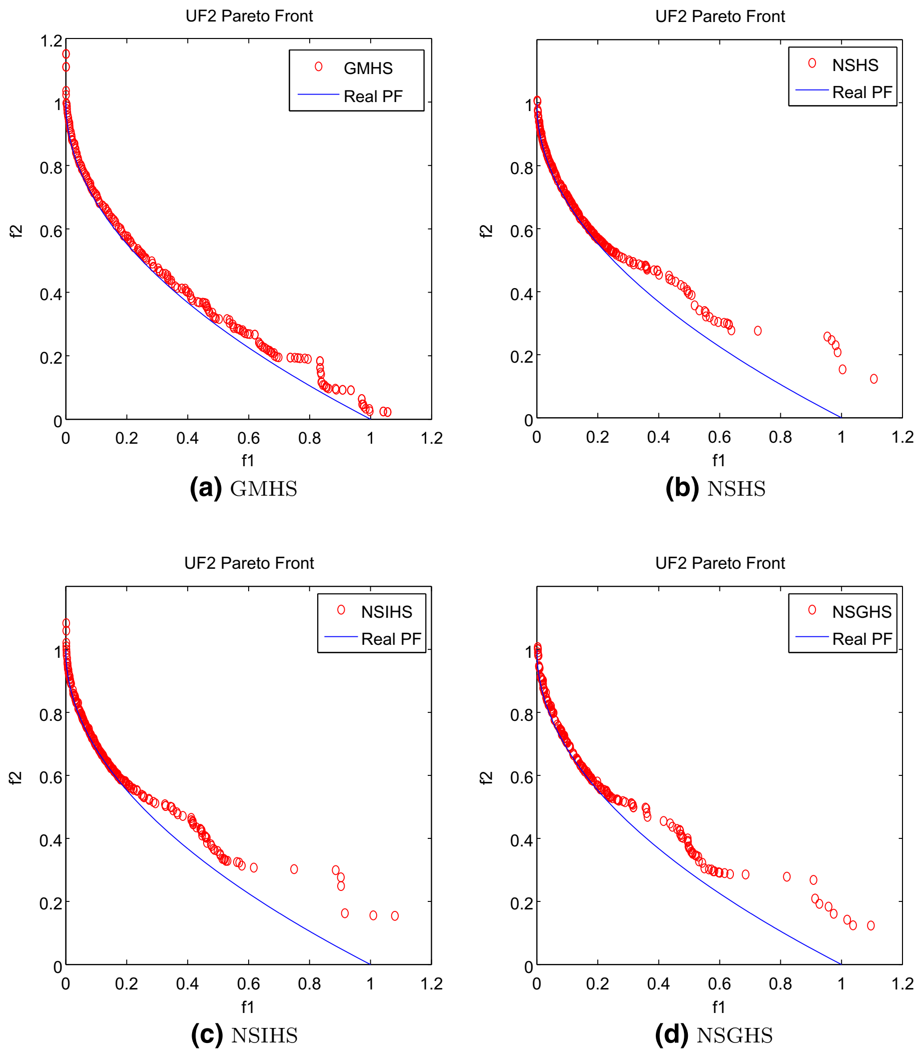


Fig. 6 Pareto fronts obtained with GMHS, NSHS, NSIHS and NSGHS on problem UF2

5.2.1 Test problems

To further study the performance of the proposed algorithm, seven instances from the CEC 2009 multi-objective benchmark (Zhang et al. 2009) are used. They are all unconstrained problems, with different characteristics, including seven bi-objective problems: UF1, UF2, UF3, UF4, UF5, UF6 and UF7.

5.2.2 Performance metric

In order to assess the convergence and diversity of the approximated set at the same time, inverted generational distance abbreviated as IGD (Zhang and Li 2007) has been used as a performance indicator in this section. The IGD value for the approximated set is defined as follows:

$$\mathrm{IGD}(A, P^*) = \frac{\sum_{v \in P^*} d(v, A)}{|P^*|}, \tag{23}$$

where P^* is the set of uniformly distributed Pareto optimal solutions in the objective space and A is an approximated set which is obtained by these algorithms. $d(v, A)$ is the minimum Euclidean distance between v and the points in A and $|P^*|$ is the number of points in P^*. The smaller IGD-metric value, the better the approximated set distributed.

5.2.3 Algorithms parameter settings

The parameters for the NSHS, NSIHS, NSGHS and GMHS are presented in Table 4, where $\triangle_x$ is the range of values of search space. It is unnecessary to set PAR since it is computed by chaotic map in the GMHS. It should be noted that bw used in the NSGHS is not presented in this table as this algorithm uses an adaptive step, and the second kind of bw used in GMHS is not presented in this table for this bw is also an adaptive step. Additionally, the P_c and external archive size in the GMHS are set to 0.1 and 200, respectively. Table 4 clearly

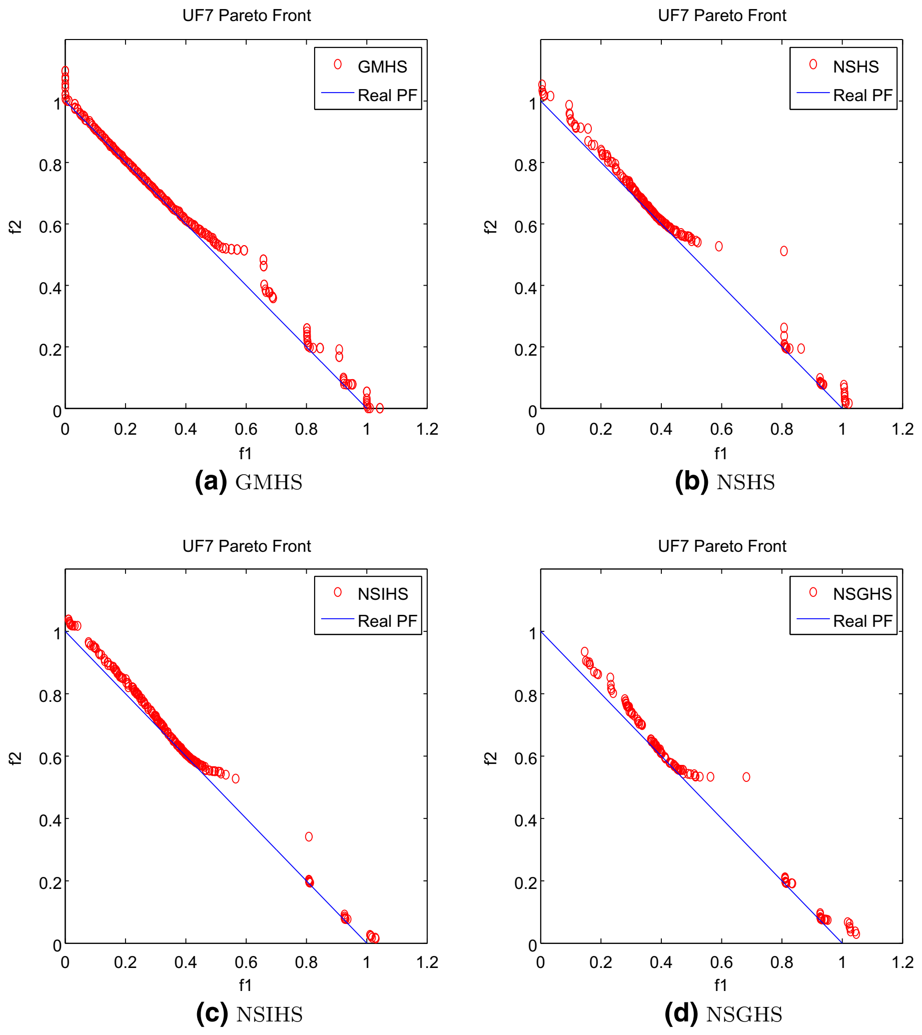


Fig. 7 Pareto fronts obtained with GMHS, NSHS, NSIHS and NSGHS on problem UF7

shows that the HMS and maximum function evaluation for these four algorithms are all set to 200 and 150,000, respectively. Therefore, the comparisons between the GMHS and other three algorithms are fair. In order to calculate the IGD-metric values, we have taken P^* of uniformly distributed points in the PF of size 1000 for CEC 2009 test problems.

5.2.4 Experimental results and comparisons

The IGD-metric values using four aforementioned algorithms after 150,000 function evaluations in 30 independent runs are presented in Table 5. It can be seen from Table 5 that GMHS algorithm generates six best results out of seven test problems for the IGD indicator. These four algorithms generate almost similar results for the problem UF3, which could be attributed to the fact that UF3 is a very difficult problem and the four algorithms all can not obtain satisfactory approximated Pareto fronts. Therefore, it can be concluded that the approximated Pareto fronts obtained by GMHS are much better than those computed by NSHS, NSIHS and NSGHS in terms of convergence and diversity.

To graphically demonstrate above claim, two typical simulation results for problems UF2 and UF7 are chosen here. Figs. 6 and 7 depict the approximated Pareto fronts of the best IGD values in the best run among 30 independent runs of the four algorithms on problems UF2 and UF7, respectively. It can be observed from Figs. 6 and 7 that the GMHS has produced better approximated Pareto fronts than other three algorithms in terms of convergence and diversity.

5.3 Different probability P_c for GMHS

To investigate the performance of GMHS to different probability P_c on ZDT problems. The GMHS parameters are $T = 250$, HMS $= 100$, HMCR $= 0.99$, $N = 100$, $bw_{1,j,\min} = 1$ e−4. To calculate the IGD-metric values, we have taken P^* of uniformly distributed points in the PF of size 500 for ZDT test problems. The simulation results for ZDT problems by GMHS using different P_c in 30 runs are

Table 6 The IGD values statistics by GMHS with different probability P_c

Parameters	$P_c = 0.1$		$P_c = 0.3$		$P_c = 0.6$		$P_c = 0.9$		$P_c = 1.0$	
Test problems	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD
ZDT1	4.5607e−3	1.0586e−4	4.1259e−3	6.4739e−5	3.9931e−3	3.7030e−5	3.9810e−3	4.2093e−5	**3.9801e−3**	**4.2765e−5**
ZDT2	4.8971e−3	1.7986e−4	4.3698e−3	9.9061e−5	4.2080e−3	7.2850e−5	4.1763e−3	5.8733e−5	**4.1700e−3**	**6.7201e−5**
ZDT3	4.9634e−3	8.5028e−5	4.6958e−3	8.4297e−5	4.6715e−3	5.7867e−5	4.6555e−3	6.7389e−5	**4.6447e−3**	**6.6353e−5**
ZDT4	1.4775e−2	8.0485e−3	5.6649e−3	5.1257e−4	4.6570e−3	3.1889e−4	**4.4991e−3**	**1.3622e−4**	4.5116e−3	1.3318e−4
ZDT6	3.2126e−3	5.9996e−5	3.1973e−3	6.3695e−5	3.2059e−3	5.9010e−5	3.1941e−3	4.9687e−5	**3.1856e−3**	**4.6995e−5**

The best value is highlighted bold

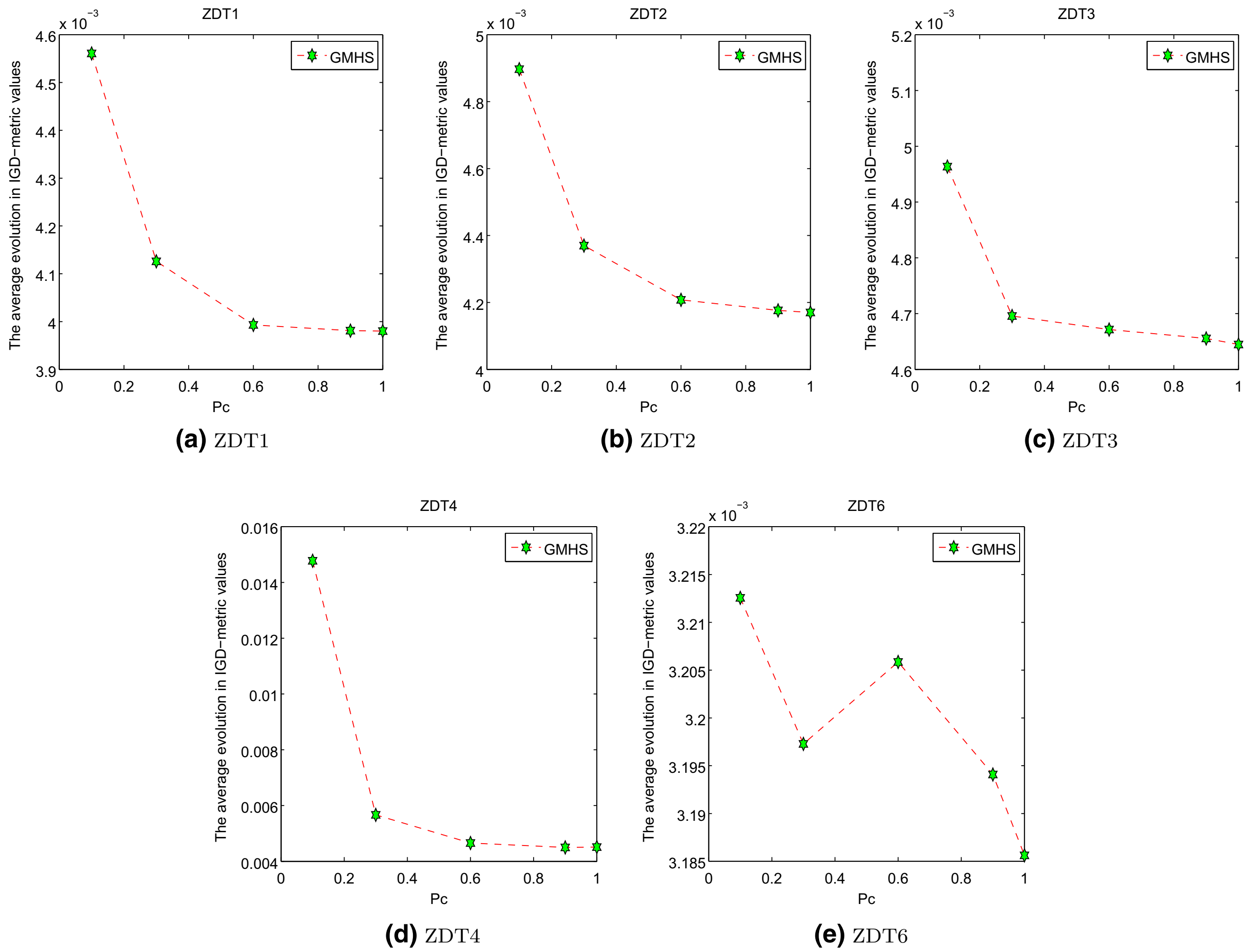


Fig. 8 Average evolution of the IGD-metric values versus the probability P_c in GMHS for ZDT1, ZDT2, ZDT3, ZDT4 and ZDT6

reported in Table 6, and the evolution of the average IGD-metric values versus the P_c with different values created by GMHS is also illustrated in Fig. 8. It can be seen from Table 6 and Fig. 8 that the GMHS has better performance when P_c is within 0.9–1.0.

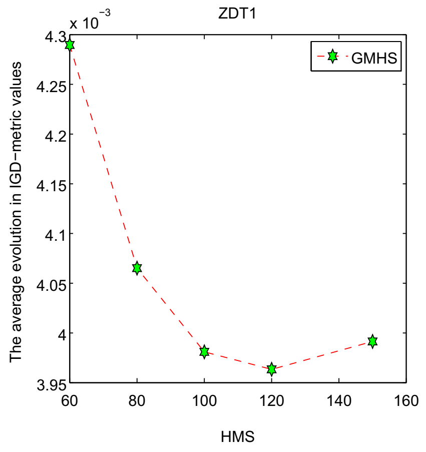
5.4 Different HMS for GMHS

The performance of GMHS using different HMS has been investigated here. The following values of HMS 60, 80, 100, 120 and 150 have been examined for ZDT problems. The

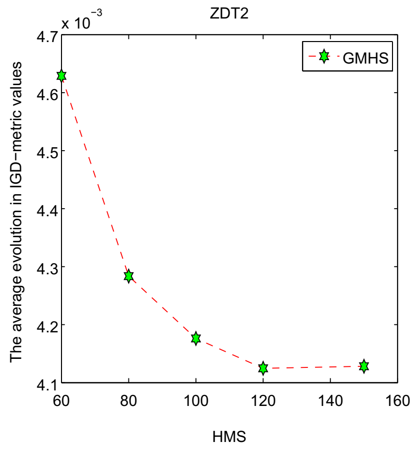
Table 7 The IGD values statistics by GMHS with different HMS

Parameters	HMS = 60		HMS = 80		HMS = 100		HMS = 120		HMS = 150	
Test problems	Mean	SD	Mean	SD	Mean	SD	Mean	SD	Mean	SD
ZDT1	4.2895e−3	8.4338e−5	4.0652e−3	5.7975e−5	3.9810e−3	4.2093e−5	**3.9634e−3**	**4.2428e−5**	3.9914e−3	4.6551e−5
ZDT2	4.6289e−3	1.0751e−4	4.2837e−3	8.1482e−5	4.1763e−3	5.8733e−5	**4.1248e−3**	**5.7178e−5**	4.1283e−3	5.3886e−5
ZDT3	4.8127e−3	9.8335e−5	4.6913e−3	7.3654e−5	4.6555e−3	6.7389e−5	**4.6391e−3**	**6.5469e−5**	4.6561e−3	5.5719e−5
ZDT4	8.5553e−3	4.1526e−3	5.2783e−3	6.6434e−4	4.4991e−3	1.3622e−4	4.2754e−3	9.2246e−5	**4.1484e−3**	**5.8428e−5**
ZDT6	**3.1718e−3**	**5.6501e−5**	3.1837e−3	6.1909e−5	3.1941e−3	4.9687e−5	3.1883e−3	4.2297e−5	3.1837e−3	4.3883e−5

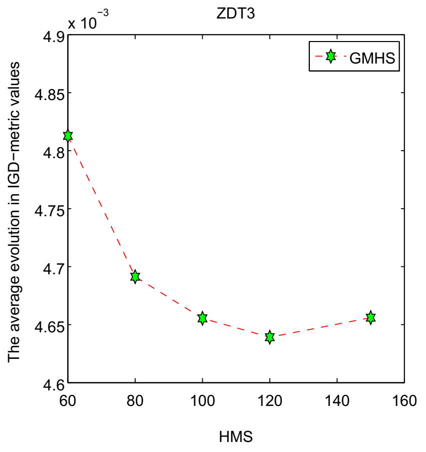
The best value is highlighted bold



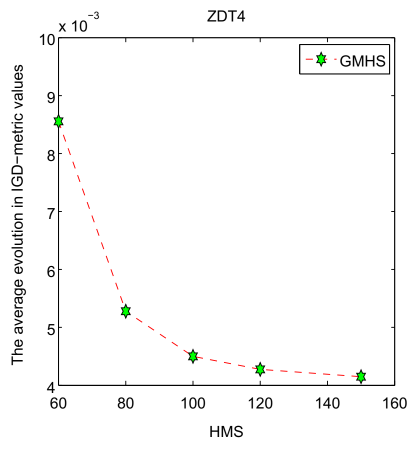
(a) ZDT1



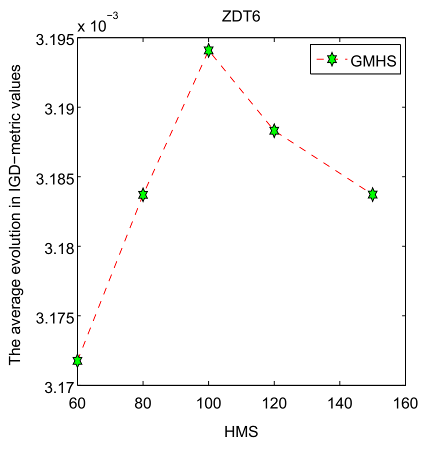
(b) ZDT2



(c) ZDT3



(d) ZDT4



(e) ZDT6

Fig. 9 Average evolution of the IGD-metric values versus HMS in GMHS for ZDT1, ZDT2, ZDT3, ZDT4 and ZDT6

GMHS parameters are $T = 250$, $P_c = 0.9$, HMCR $= 0.99$, $N = 100$ and $bw_{1,j,\min} = 1$ e−4. The experimental results for ZDT problems by GMHS using different HMS in 30 runs are presented in Table 7, and the evolution of the average IGD-metric values versus the different HMS created by GMHS is also illustrated in Fig. 9. Table 7 and Fig. 9 show that, as the increase of HMS, the IGD-metric values become slightly better in general, but the algorithm's execution time becomes longer. Therefore, considering the performance and execution time of the algorithm, it is good to set HMS within 80–120.

6 Conclusion and discussion

In this paper, a harmony search algorithm with gaussian mutation is proposed for solving MOPs. Benchmark test results have shown that the GMHS algorithm has better performance than other MOEAs. The impact of key parameter P_c and HMS on the performance of GMHS algorithm is also discussed. It is observed that considerable performance improvement is possible by applying reasonable parameter values in the proposed algorithm. From the presented investigation, it can be concluded that the proposed GMHS is able to generate true and well distributed Pareto optimal solutions. This paper presents results on unconstrained MOPs, and the future research considers constrained MOPs.

Considering aforementioned promising results, the following future course of action is proposed: (a) to investigate the performance of GMHS using different chaotic maps. (b) To apply GMHS algorithm to solve practical MOPs. (c) To investigate the performance of the proposed algorithm using different mutation operators, such as cauchy mutation, adaptive lévy mutation and combined mutation operators (Koenig 2002).

Compliance with ethical standards

Conflict of interest No conflict of interest exits in the publication of this manuscript, and manuscript is approved by all authors for publication. I would like to declare on behalf of my co-authors that the work described was original research that has not been published previously, and not under consideration for publication elsewhere, in whole or in part. All the authors listed have approved the manuscript that is enclosed.

References

Chen J, Pan QK, Li JQ (2012) Harmony search algorithm with dynamic control parameters. Appl Math Comput 219(2):592–604

Coello CA, Pulido GT, Lechuga MS (2004) Handling multiple objectives with particle swarm optimization. IEEE Trans Evol Comput 8(3):256–279

Corne DW, Jerram NR, Knowles JD et al. (2001) PESA-II: region-based selection in evolutionary multiobjective optimization. In: Proceedings of the genetic and evolutionary computation conference, pp 283–290

Deb K, Pratap A, Agarwal S et al (2002) A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans Evol Comput 6(2):182–197

Deb K, Jain H (2014) An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: solving problems with box constraints. IEEE Trans Evol Comput 18(4):577–601

Deb K, Thiele L, Laumanns M et al. (2001) Scalable test problems for evolutionary multi-objective optimization. In: Technical report, Comput Eng and Networks Lab (TIK), ETH Zurich, Switzerland

Diao R, Shen Q (2012) Feature selection with harmony search. IEEE Trans Syst Man Cybern Part B Cybern 42(6):1509–1523

Garca S, Molina D, Lozano M et al (2009) A study on the use of non-parametric tests for analyzing the evolutionary algorithm behaviour: a case study on the CEC'2005 special session on real parameter optimization. J Heuristics 15(6):617–644

Geem ZW, Kim JH, Loganathan GV (2001) A new heuristic optimization algorithm: harmony search. Simulation 76(2):60–68

Geem ZW (2010) Multiobjective optimization of time-cost trade-off using harmony search. Constr Eng Manage 136(6):711–716

He D, He C, Jiang LG et al (2001) Chaotic characteristics of a one-dimensional iterative map with infinite collapses. IEEE Trans Circuits Syst I Fundam Theor Appl 48(7):900–906

Knowles JD, Corne DW (2000) M-PAES: A memetic algorithm for multiobjective optimization. Proc IEEE Conf Evolut Comput 1:325–332

Koenig AC (2002) A study of mutation methods for evolutionary algorithms. Adv Top Artif Intell 1–8

Kulluk S, Ozbakir L, Baykasoglu A (2012) Training neural networks with harmony search algorithms for classification problems. Eng Appl Artif Intell 25(1):11–19

Landa I, Ortiz E, Salcedo S et al (2012) Evaluating the internationalization success of companies through a hybrid grouping harmony search-extreme learning machine approach. IEEE J Sel Top Signal Process 6(4):388–398

Lee KS, Geem ZW (2005) A new meta-heuristic algorithm for continues engineering optimization: harmony search theory and practice. Comput Methods Appl Mech Eng 194(36–38):3902–3933

Mahdavi M, Fesanghary M, Damangir E (2007) An improved harmony search algorithm for solving optimization problems. Appl Math Comput 188(2):1567–1579

Obayashi S, Takahashi S, Takeguchi Y (1998) Niching and elitist models for mogas. Lect Notes Comput Sci 1498:260–269

Parks GT, Miller I (1998) Selective breeding in a multiobjective genetic algorithm. Lect Notes Comput Sci 1498:250–259

Pavelski LM, Almeida CP, Gonçalves RA (2012) Harmony search for multi-objective optimization. In: Proceedings 2012 Brazilian conference on neural networks, pp 220–225

Ricart J, Hüttermann G, Lima J et al (2011) Multiobjective harmony search algorithm proposals. Electron Notes Theor Comput Sci 281:51–67

Shang RH, Zhou YJ, Hu CX et al. (2014) Distance computation based NSGA-II for multiobjective optimization. Bejing: sciencepaper online, pp 1–25. http://www.paper.edu.cn/releasepaper/content/201403-40. Accessed 25 Jan 2015

Sivasubramani S, Swarup KS (2011) Multi-objective harmony search algorithm for optimal power flow problem. Int J Electr Power Energy Syst 33(3):745–752

Van Veldhuizen DA, Lamont GB (2000) On measuring multiobjective evolutionary algorithm performance. In: Proceedings of the 2000 congress on evolutionary computation, vol 1, pp 204–211

Van Veldhuizen DA, Lamont GB (2000) Multiobjective evolutionary algorithms: analyzing the state-of-the-art. Evol Comput 8(2):125–147

Vasebi A, Fesanghary M, Bathaee SMT (2007) Combined heat and power economic dispatch by harmony search algorithm. Int J Electr Power Energy Syst 29(10):713–719

Wang YN, Wu LH, Yuan XF (2010) Multi-objective self-adaptive differential evolution with elitist archive and crowding entropy-based diversity measure. Soft Comput 14(3):193–209

Yuan XF, Zhao JY, Yang YM et al (2014) Hybrid parallel chaos optimization algorithm with harmony search algorithm. Appl Soft Comput J 17:12–22

Zhang Q, Li H (2007) MOEA/D: A multiobjective evolutionary algorithm based on decomposition. IEEE Trans Evol Comput 11(6):712–731

Zhang Q, Zhou A, Zhao S et al. (2009) Multiobjective optimization test instances for the CEC 2009 special session and competition. In: Technical Report CES-487, pp 1–30

Zhou A, Jin Y, Zhang Q et al. (2006) Combing model-based and generics-based offspring generation for multi-objective optimization using a convergence criterion. In: Proceedings of the IEEE congress on evolutionary computation, pp 3234–3241

Zitzler E, Deb K, Thiele L (2000) Comparison of multiobjective evolutionary algorithms: empirical results. Evol Comput 8(2):173–195

Zitzler E, Laumanns M, Thiele L (2001) SPEA2: Improving the strength pareto evolutionary algorithm. In: Technical report. Comput Eng and Networks Lab (TIK), ETH Zurich, Switzerland

Zitzler E, Thiele L (1999) Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. IEEE Trans Evol Comput 3(4):257–271

Zou DX, Gao LQ, Wu JH et al (2010) Novel global harmony search algorithm for unconstrained problems. Neurocomputing 73(16–18):3308–3318