Hindawi Publishing Corporation
Journal of Applied Mathematics
Volume 2012, Article ID 147950, 20 pages
doi:10.1155/2012/147950

Research Article

Theoretical and Empirical Analyses of an Improved Harmony Search Algorithm Based on Differential Mutation Operator

Longquan Yong,[1, 2] Sanyang Liu,[1] Jianke Zhang,[1, 3] and Quanxi Feng[1, 4]

[1] *Department of Applied Mathematics, Xidian University, Xi'an 710071, China*
[2] *School of Mathematics and Computer Science, Shaanxi University of Technology, Hanzhong 723001, China*
[3] *School of Science, Xi'an University of Posts and Telecommunications, Xi'an 710121, China*
[4] *School of Science, Guilin University of Technology, Guilin 541004, China*

Correspondence should be addressed to Longquan Yong, yonglongquan@sohu.com

Received 13 February 2012; Revised 24 April 2012; Accepted 18 May 2012

Academic Editor: Yuri Sotskov

Harmony search (HS) method is an emerging metaheuristic optimization algorithm. In this paper, an improved harmony search method based on differential mutation operator (IHSDE) is proposed to deal with the optimization problems. Since the population diversity plays an important role in the behavior of evolution algorithm, the aim of this paper is to calculate the expected population mean and variance of IHSDE from theoretical viewpoint. Numerical results, compared with the HSDE, NGHS, show that the IHSDE method has good convergence property over a test-suite of well-known benchmark functions.

1. Introduction

Most optimization algorithms are based on numerical linear and nonlinear programming methods that require substantial gradient information and usually seek to improve the solution in the neighborhood of an initial point. These algorithms, however, reveal a limited approach to complicated real-world optimization problems because gradient is often difficult to find out or does not exist. What is more, if there is more than one local optimum in the problem, the result may depend on the selection of the starting point, and the obtained optimal solution may not necessarily be the global optimum.

Recently, a new class of metaheuristics, named harmony search (HS), has been developed. The HS algorithm proposed in [1] has been developed in an analogy with music improvisation process where musicians in an ensemble continue to polish their pitches in order to obtain better harmony. Jazz improvisation seeks to find musically pleasing harmony similar to the optimum design process which seeks to find optimum solution. The pitch of each musical instrument determines the aesthetic quality, just as the objective function value is determined by the set of values assigned to each decision variable [2]. In addition, HS uses a stochastic random search instead of a gradient search so that derivative information is unnecessary.

HS may be viewed as a simple real-coded genetic algorithm (GA), since it incorporates many important features of GA like mutation, recombination, and selection. HS has been successfully applied to a wide variety of practical optimization problems like designing controller [3], economic dispatch problem [4, 5], optimal power flow problem [6], neural networks [7], medical diagnosis [8], broadcast scheduling in packet radio networks [9], university course timetabling [10], and other engineering optimization field [11].

Similar to the GA and particle swarm algorithms, the HS method is a random search technique. It does not require any prior domain knowledge, such as the gradient information of the objective functions. Unfortunately, empirical study has shown that the original HS method sometimes suffers from a slow search speed, and it is not suitable for handling the multimodal problems [2].

Recently, Omran and Mahdavi tried to improve the performance of HS by incorporating some techniques from swarm intelligence. The new variant called by them as global best harmony search (GHS) [12] reportedly outperformed three other HS variants over the benchmark problems. Chakraborty et al. proposed an improved harmony search algorithm with differential mutation operator [13]. Gao et al. proposed modified harmony search methods for unimodal and multimodal optimization [14]. Wang et al. proposed self-adaptive harmony search algorithm for optimization [15]. Pan et al. proposed a self-adaptive global best harmony search algorithm for continuous optimization problems [16]. Zou et al. proposed a novel global harmony search algorithm (NGHS, [17]). More latest HS algorithm can be found in [18, 19].

To overcome the shortcoming of premature convergence and stagnation, in this paper, we replace the pitch adjustment operation in classical HS (CHS) with a mutation strategy borrowed from the realm of the differential evolution (DE) algorithms, and we use

$$x_i^{\mathrm{new}} = x_i^j + F \times \left(x_i^{r_1} - x_i^{r_2}\right), \tag{1.1}$$

where F is chosen to be a uniformly distributed random variable between 0.6 and 1.0, $F \sim U[0.6, 1]$ instead of $F \sim U[0, 1]$ in [13], and j, r_1, r_2 are randomly selected with uniform distribution from the set $\{1, 2, \ldots, \mathrm{HMS}\}$, $j \neq r_1 \neq r_2$. The new mutation strategy is inspired by Chakraborty's [13], and S. Das's [19] work, especially in Chakraborty's work [13], where the author theoretically showed that the harmony search based on differential mutation scheme (HSDE) has greater explorative power than the classical HS with pitch adjustment operation.

The new algorithm proposed in this paper, called IHSDE (improved harmony search methods based on differential mutation operator), has been extensively compared with the HSDE, and the classical HS. Mathematical analysis will show that the IHSDE, under certain conditions, possesses an increasing population variance (with generation) as compared to

HSDE. The numerical experiments show that the proposed algorithm is effective in dealing with a test suite of well-known benchmark functions.

The rest of the paper is organized in the following way. Section 2 briefly outlines the classical HS. Section 3 presents IHSDE method based on differential mutation operator and analyzes the expected population mean and variance of IHSDE in terms of theory. Effectiveness of IHSDE method is demonstrated in Section 4 by solving well-known benchmarks. Section 5 concludes the paper.

2. Classical Harmony Search Algorithm

2.1. Harmony Search Algorithm Principles

Current metaheuristic algorithms imitate natural phenomena, and evolution in evolutionary algorithms. HS algorithm was conceptualized using the musical process of searching for a perfect state of harmony. In music improvisation, each player sounds any pitch within the possible range, together making one harmony vector. If all the pitches make a good harmony, that experience is stored in each player's memory, and the possibility to make a good harmony is increased next time. Similarly, in engineering optimization, each decision variable initially chooses any value within the possible range, together making one solution vector. If all the values of decision variables make a good solution, that experience is stored in each variable's memory, and the possibility to make a good solution is also increased next time. Figure 1 shows the details of the analogy between music improvisation and engineering optimization.

The HS algorithm does not require initial values for the decision variables. Furthermore, instead of a gradient search, the HS algorithm uses a stochastic random search that is based on the harmony memory considering rate and the pitch-adjusting rate so that derivative information is unnecessary. Compared to earlier metaheuristic optimization algorithms, the HS algorithm imposes fewer mathematical requirements and can be easily adopted for various types of engineering optimization problems.

2.2. Steps for Classical Harmony Search Algorithm

The steps in the procedure of standard harmony search algorithm are as follows:
Step 1 (Initialize the problem and algorithm parameters). The optimization problem is specified as follows.

$$\text{Minimize } f(x) \quad \text{subject to } x_i \in X_i,\ i = 1, 2, \ldots, N, \tag{2.1}$$

where $f(x)$ is an objective function; x is the set of each decision variable x_i; N is the number of decision variables; X_i is the set of the possible range of values for each decision variable, $X_i : {x_i}^L \leq X_i \leq {x_i}^U$. The HS algorithm parameters are also specified in this step. These are the harmony memory size (HMS), or the number of solution vectors in the harmony memory; the harmony memory (HM); harmony memory considering rate (HMCR); pitch-adjusting rate (PAR); the number of improvisations ($T_{\max}$).

The harmony memory (HM) is a memory location where all the solution vectors (sets of decision variables) are stored. HMCR and PAR are parameters that are used to improve the solution vector.

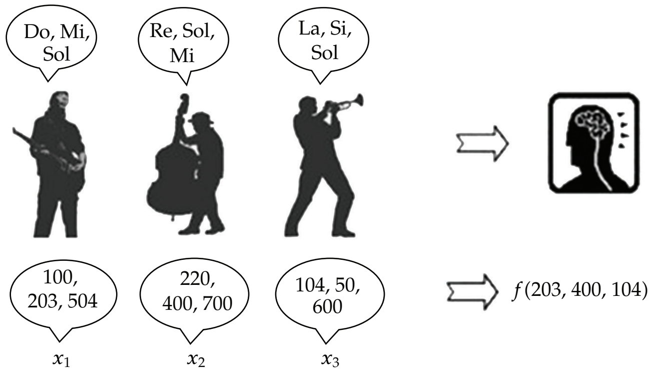


Figure 1: Analogy between music improvisation and engineering optimization.

Step 2 (Initialize the harmony memory). In Step 2, the HM matrix is filled with as many randomly generated solution vectors as the HMS:

$$\mathrm{HM} = \left[\begin{array}{c|c} x^1 & f(x^1) \\ x^2 & f(x^2) \\ \vdots & \vdots \\ x^{\mathrm{HMS}} & f(x^{\mathrm{HMS}}) \end{array}\right] = \left[\begin{array}{cccc|c} x_1^1 & x_2^1 & \cdots & x_N^1 & f(x^1) \\ x_1^2 & x_2^2 & \cdots & x_N^2 & f(x^2) \\ \vdots & \vdots & \cdots & & \vdots \\ x_1^{\mathrm{HMS}} & x_2^{\mathrm{HMS}} & \cdots & x_N^{\mathrm{HMS}} & f(x^{\mathrm{HMS}}) \end{array}\right]. \tag{2.2}$$

Step 3 (Improvise a new harmony). A new harmony vector, $x' = (x'_1, x'_2, \ldots, x'_N)$, is generated based on three rules: (a) memory consideration, (b) pitch adjustment, and (c) random selection. Generating a new harmony is called "improvisation." In the memory consideration, the value of the first decision variable x'_1 for the new vector is chosen from any of the values in the specified HM with a probability HMCR. The HMCR is the rate of choosing one value from the historical values stored in the HM, while (1-HMCR) is the rate of randomly selecting one value from the possible range of values:

$$x'_i = \begin{cases} x'_i \in \left(x_i^1, x_i^2, \ldots, x_i^{\mathrm{HMS}}\right), & \text{if rand} < \mathrm{HMCR}, \\ x'_i \in X_i, & \text{otherwise}, \end{cases} \tag{2.3}$$

where rand is a uniformly distributed random variable between 0 and 1.

Every component obtained by the memory consideration is examined to determine whether it should be pitch adjusted. This operation uses the PAR parameter, which is the rate of pitch adjustment as follows.

Pitch adjusting decision for x'_i:

$$x'_i = \begin{cases} x'_i \pm \text{rand} \times bw, & \text{if rand} < \mathrm{PAR}, \\ x'_i, & \text{otherwise}, \end{cases} \tag{2.4}$$

where bw is an arbitrary distance bandwidth.

In Step 3, HM consideration, pitch adjustment, or random selection is applied to each variable of the new harmony vector in turn.

Step 4 (Update harmony memory). If the new harmony vector $x' = (x'_1, x'_2, \dots, x'_N)$ is better than the worst harmony in the HM, the new harmony is included in the HM and the existing worst harmony is excluded from the HM.

Step 5 (Check stopping criterion). If the stopping criterion ($T_{\max}$) is satisfied, computation is terminated. Otherwise, Steps 3 and 4 are repeated.

In the next section, we employ the differential mutation operator to improve the fitness of all the members in the HS memory so that the overall convergence speed of the original HS method can be accelerated.

3. The Improved HS Based on Differential Mutation Operator

Experiments with the CHS algorithm over the standard numerical benchmarks show that the algorithm suffers from the problem of premature and/or false convergence, slow convergence especially over multimodal fitness landscape.

To circumvent these problems of premature, Chakraborty et al. proposed harmony search algorithm with differential mutation operator (HSDE). They replaced the pitch adjustment operation (2.4) in classical HS with a mutation strategy borrowed from the realm of the DE algorithms [20]. The mutation strategy has been presented as

$$x_i^{\text{new}} = x_i^j + F \times \left(x_i^{r_1} - x_i^{r_2}\right), \tag{3.1}$$

where $j, r_1, r_2 \in U\{1, 2, \dots, \text{HMS}\}$, $j \neq r_1 \neq r_2$, and the scale factor F is chosen to be a uniformly distributed random variable between 0 and 1, that is, $F \sim U[0,1]$.

In what follows, we reset F is chosen to be a uniformly distributed random variable between 0.6 and 1.0, $F \sim U[0.6, 1]$. This improved harmony search algorithm with differential mutation operator will be referred to as IHSDE. In the following, we will show that the IHSDE gives better performance than the HSDE algorithm in theory and experiment.

3.1. *IHSDE Algorithm*

The pseudocode of IHSDE is described in Algorithm 1. Every variable in the HM needs to go through the above DE mutation refinement procedure. Thus, the resulting HM members are expected to have better fitness than that of the original ones. This strategy can also overcome the premature shortcoming of the regular HS method.

3.2. *Theoretical Analyses of the Expected Population Variance*

Theoretical analyses of the properties of HS algorithms are very important to understand their search behaviors and to develop more efficient algorithms [13, 19]. Compared to the plethora of works concerning the empirical study of parameter selection and tuning process in HS [14–18], not much research has so far been devoted to theoretically analyze the

```
Procedure IHSDE algorithm
  Initiate_parameters
  Initialize_HM
  While (not_termination)
    For $i = 1$ to $N$ do            //$N$ denotes the number of decision variables
      if ( rand < HMCR)              //memory consideration
        Select one harmony from HM randomly: $x^{\text{new}} = x^j, j \in U\{1,2,\ldots,\text{HMS}\}$;
        Execute difference variation operation for the selected harmony:
        $x_i^{\text{new}} = x_i^j + F \times (x_i^{r_1} - x_i^{r_2})$, where $F \sim U[0.6,1], r_1, r_2 \in U\{1,2,\ldots,\text{HMS}\}, j \neq r_1 \neq r_2$.
      else
        $x_i^{\text{new}} = x_i^L + \text{rand} \times (x_i^U - x_i^L)$        //random selection
      end if
    End for
    Update harmony memory HM             // if applicable
  End while
End procedure
```

Algorithm 1: Pseudocode of the IHSDE Algorithm.

search mechanism and convergence properties of HS, and this area remains largely open to prospective future research.

The evolution of the expected population variance over generations provides a measure of the explorative power of the algorithm. In the following, we will estimate the expected mean and variance of the population obtained by applying mutation operator.

Our ideas are as follows, firstly we find an analytical expression for the population expected variance, and then we compare the expected population variance of IHSDE with HSDE to show that the IHSDE algorithm possesses greater explorative power.

In HS type algorithms, since each dimension is perturbed independently, without loss of generality, we can make our analysis for single-dimensional population members.

Let us consider a population of scalars $x = \{x_1, x_2, \ldots, x_m\}$ with elements $x_k \in R$, $k = 1, 2, \ldots, m$. The variance of this population is given by

$$\text{Var}(x) = \frac{1}{m}\sum_{i=1}^{m}(x_i - \overline{x})^2 = \overline{x^2} - \overline{x}^2, \tag{3.2}$$

where $\overline{x} = (1/m)\sum_{i=1}^{m} x_i$ is population mean and $\overline{x^2} = (1/m)\sum_{i=1}^{m} x_i^2$ is quadratic population mean.

If the elements of the population are perturbed with some random numbers or variables, $\text{Var}(x)$ will be a random variable, and $E[\text{Var}(x)]$ will be a measure of the explorative power. In the following discussion, we always suppose $\text{Var}(x) > 0$.

Lemma 3.1. *Let $x = \{x_1, x_2, \ldots, x_m\}$ be the current population of HS, y be an intermediate vector obtained after random selection with harmony memory consideration, and z the vector obtained after*

y by replacing the pitch adjustment operation with a mutation operator (3.1) *borrowed from DE algorithms in classical HS. Let $w = \{w_1, w_2, \ldots, w_m\}$ be the final population after selection. If we let*

$$p = \text{ harmonic memory consideration probability } (HMCR)\,, \tag{3.3}$$

the allowable range for the new values of x is $\{x_{\min}, x_{\max}\}$ where $x_{\min} = -a$, $x_{\max} = a$, and the required random numbers are continuously uniformly distributed random variable between 0 and 1 except for random variables F, then

$$\begin{aligned} E[\operatorname{Var}(w)] &= \frac{2}{m}\overline{F^2}\operatorname{Var}(x) + \frac{m-1}{3m^2}a^2(1-p) \\ &\quad + \frac{1}{m^2}\left[(m-1)p + (m-1)^2\right]\overline{x^2} - \frac{m-1}{m}\left(\frac{1}{m}p\overline{x} + \frac{m-1}{m}\overline{x}\right)^2, \end{aligned} \tag{3.4}$$

where $\overline{F^2} = E(F^2)$.

Proof. Since $x = \{x_1, x_2, \ldots, x_m\}$= HM is the current population of HS and y is an intermediate vector obtained after random selection with harmony memory consideration, that is,

$$y = \begin{cases} x_l, & \text{with probability } p \text{ from current population } \{x_1, x_2, \ldots, x_m\}, \\ x_r, & \text{with probability } (1-p) \text{ allowable range}\{x_{\min}, x_{\max}\}, \end{cases} \tag{3.5}$$

where $l \in \{1, 2, \ldots, m\}$ and x_r is a new random variable in allowable range $\{x_{\min}, x_{\max}\}$.

So,

$$E(x_l) = \overline{x}, \qquad E\left(x_l^{\,2}\right) = \overline{x^2}. \tag{3.6}$$

According to [13, 19], we have

$$E(x_r) = 0, \qquad E\left(x_r^{\,2}\right) = \frac{a^2}{3}. \tag{3.7}$$

Using (3.6), we can obtain the following relations:

$$\begin{aligned} E(y) &= E(x_l)p + E(x_r)(1-p) = p\overline{x}; \\ E\left(y^2\right) &= E\left(x_l^{\,2}\right)p + E\left(x_r^{\,2}\right)(1-p) = p\overline{x^2} + \frac{a^2}{3}(1-p). \end{aligned} \tag{3.8}$$

Now, z is the vector obtained after mutating y. It has the following structure:

$$z = y + F \times (x_{r_1} - x_{r_2}). \tag{3.9}$$

Let $\overline{F} = E(F), \overline{F^2} = E(F^2)$, x_{r_1} and x_{r_2}, are two randomly chosen members from $x = \{x_1, x_2, \ldots, x_m\}$ such that $r_1, r_2 \in \{1, 2, \ldots, m\}, r_1 \neq r_2$.

Thus,

$$E(x_{r_1}) = E(x_{r_2}) = \overline{x}; \qquad E\left(x_{r_1}^2\right) = E\left(x_{r_2}^2\right) = \overline{x^2}. \tag{3.10}$$

According to [13], we have

$$E(x_{r_1}x_{r_2}) = \frac{1}{m(m-1)}\left[(m\overline{x})^2 - m\overline{x^2}\right]. \tag{3.11}$$

Therefore,

$$\begin{aligned}
E(z) &= E(y) + E(F) \times E(x_{r_1} - x_{r_2}) = E(y) + \overline{F} \times [E(x_{r_1}) - E(x_{r_2})] = E(y) = p\overline{x}, \\
E\left(z^2\right) &= E(y + F \times (x_{r_1} - x_{r_2}))^2 \\
&= E\left[y^2 + F^2 \times \left(x_{r_1}^2 + x_{r_2}^2 - 2x_{r_1}x_{r_2}\right) + 2F \times y(x_{r_1} - x_{r_2})\right] \\
&= E\left[y^2\right] + \overline{F^2} \times \left(E\left[x_{r_1}^2\right] + E\left[x_{r_2}^2\right] - 2E[x_{r_1}x_{r_2}]\right) + 2\overline{F} \times E[y](E[x_{r_1}] - E[x_{r_2}]) \\
&= E\left[y^2\right] + 2\overline{F^2}\frac{m}{m-1}\left(\overline{x^2} - \overline{x}^2\right).
\end{aligned} \tag{3.12}$$

Now, $w = \{w_1, w_2, \ldots, w_m\}$ is the final population. Each element w_k of the final population may be represented by the following:

$$w_k = \begin{cases} z, & \text{with probability } \dfrac{1}{m}, \\ x_k, & \text{with probability } \left(1 - \dfrac{1}{m}\right). \end{cases} \tag{3.13}$$

Thus,

$$\begin{gathered}
E(w_k) = E(z)\frac{1}{m} + E(x_k)\left(1 - \frac{1}{m}\right) = \frac{1}{m}p\overline{x} + \frac{m-1}{m}\overline{x}, \\
E\left(w_k{}^2\right) = E\left(z^2\right)\frac{1}{m} + E\left(x_k{}^2\right)\left(1 - \frac{1}{m}\right).
\end{gathered} \tag{3.14}$$

Let $\overline{w} = 1/m\sum_{k=1}^{m} w_k$ and $\overline{w^2} = 1/m\sum_{k=1}^{m} {w_k}^2$ represent the population mean and quadratic population mean of the final target population, respectively. Then,

$$E\left(\overline{w}^2\right) = E\left(\frac{1}{m}\sum_{k=1}^{m} w_k\right)^2 = \frac{1}{m^2}\left[mE\left({w_k}^2\right) + m(m-1)E^2(w_k)\right];$$
$$E\left(\overline{w^2}\right) = E\left(\frac{1}{m}\sum_{k=1}^{m} {w_k}^2\right) = \frac{1}{m}\left(\sum_{k=1}^{m} E\left({w_k}^2\right)\right) = \frac{1}{m}\left(mE\left({w_k}^2\right)\right) = E\left({w_k}^2\right). \tag{3.15}$$

Therefore, the variance of final population $w = \{w_1, w_2, \ldots, w_m\}$ is given as

$$\operatorname{Var}(w) = \overline{w^2} - \overline{w}^2, \tag{3.16}$$

and its expected population variance

$$\begin{aligned}
&E[\operatorname{Var}(w)]\\
&\quad = E\left[\overline{w^2} - \overline{w}^2\right] = E\left[\overline{w^2}\right] - E\left(\overline{w}^2\right)\\
&\quad = E\left({w_k}^2\right) - \frac{1}{m^2}\left[mE\left({w_k}^2\right) + m(m-1)E^2(w_k)\right]\\
&\quad = \frac{m-1}{m}\left[E\left({w_k}^2\right) - E^2(w_k)\right] = \frac{m-1}{m}E\left({w_k}^2\right) - \frac{m-1}{m}E^2(w_k)\\
&\quad = \frac{m-1}{m}\left[E\left(z^2\right)\frac{1}{m} + E\left({x_k}^2\right)\left(1-\frac{1}{m}\right)\right] - \frac{m-1}{m}\left(\frac{1}{m}p\overline{x} + \frac{m-1}{m}\overline{x}\right)^2\\
&\quad = \frac{m-1}{m}\left[E\left[y^2\right]\frac{1}{m} + 2\overline{F^2}\frac{m}{m-1}\left(\overline{x^2} - \overline{x}^2\right)\frac{1}{m} + \overline{x^2}\left(1-\frac{1}{m}\right)\right] - \frac{m-1}{m}\left(\frac{1}{m}p\overline{x} + \frac{m-1}{m}\overline{x}\right)^2\\
&\quad = \frac{m-1}{m}\left[\frac{1}{m}p\overline{x^2} + \frac{a^2}{3}(1-p)\frac{1}{m} + 2\overline{F^2}\frac{m}{m-1}\left(\overline{x^2} - \overline{x}^2\right)\frac{1}{m} + \overline{x^2}\left(1-\frac{1}{m}\right)\right]\\
&\qquad - \frac{m-1}{m}\left(\frac{1}{m}p\overline{x} + \frac{m-1}{m}\overline{x}\right)^2\\
&\quad = \frac{2}{m}\overline{F^2}\left(\overline{x^2} - \overline{x}^2\right) + \frac{a^2}{3}(1-p)\frac{m-1}{m^2} + \frac{1}{m^2}\left[(m-1)p + (m-1)^2\right]\overline{x^2}\\
&\qquad - \frac{m-1}{m}\left(\frac{1}{m}p\overline{x} + \frac{m-1}{m}\overline{x}\right)^2\\
&\quad = \frac{2}{m}\overline{F^2}\operatorname{Var}(x) + \frac{m-1}{3m^2}a^2(1-p) + \frac{1}{m^2}\left[(m-1)p + (m-1)^2\right]\overline{x^2} - \frac{m-1}{m}\left(\frac{1}{m}p\overline{x} + \frac{m-1}{m}\overline{x}\right)^2.
\end{aligned} \tag{3.17}$$

□

Remark 3.2. The conclusion in [13] (Theorem 1) is incorrect. In this paper, we give the correct expression for the population expected variance.

The main analytical result is expressed in the form of the following theorem.

Theorem 3.3. *Let $x = \{x_1, x_2, \ldots, x_m\}$, be the current population of HS. Giving the same values m, p, here m and p represent the harmony memory size (HMS) and harmony memory considering rate (HMCR), respectively. The intergeneration population expected variance of IHSDE algorithm is greater than HSDE.*

Proof. In HSDE, F is chosen to be a uniformly distributed random variable between 0 and 1, and in IHSDE, F is chosen to be a uniformly distributed random variable between 0.6 and 1. For convenience, let $F_1 \sim U[0,1]$, and the intergeneration expected variance of HSDE is $E_{\mathrm{HSDE}}[\mathrm{Var}(w)]$. Let $F_2 \sim U[0.6,1]$, and the intergeneration expected variance of IHSDE is $E_{\mathrm{IHSDE}}[\mathrm{Var}(w)]$.

Since

$$\begin{aligned} \overline{F_1{}^2} &= E\left(F_1{}^2\right) = \overline{F}_1{}^2 + \mathrm{Var}(F_1) = 0.5^2 + \frac{(1-0)^2}{12} = \frac{1}{3} = 0.3333, \\ \overline{F_2{}^2} &= E\left(F_2{}^2\right) = \overline{F}_2{}^2 + \mathrm{Var}(F_2) = 0.8^2 + \frac{(1-0.6)^2}{12} = 0.6533. \end{aligned} \tag{3.18}$$

Thus,

$$E_{\mathrm{IHSDE}}[\mathrm{Var}(w)] - E_{\mathrm{HSDE}}[\mathrm{Var}(w)] = \frac{2}{m}\left(\overline{F_2{}^2} - \overline{F_1{}^2}\right)\mathrm{Var}(x) > 0. \tag{3.19}$$

□

Remark 3.4. Improper parameters F can lead to problem of premature and/or false convergence. Usually, $F \in [0,2]$. If F is chosen too small, it gets more difficult to escape local optima. A larger F (or $\overline{F^2}$) increases the probability (or the population expected variance of IHSDE) for escaping a local optimum. However, for $F > 1$, the convergence speed decreases. It is more difficult to converge for a population when the perturbation is larger than the distance between two members. Thus, though standard Gaussian distribution $N(0,1)$ has much larger $\overline{F^2} = E(F^2) = 1$ than that of $F_2 \sim \mathrm{U}[0.6,1]$, it cannot get better convergence. In [21], Mperle et al. suggest that a good initial choice for the amplification factor should be $F = 0.6$. If one suspects that with this setting only a local optimum is found, then F should be increased. This fully shows that our choosing $F_2 \sim \mathrm{U}[0.6,1]$ is appropriate.

The above mathematical analysis show that the IHSDE possesses an increasing population variance as compared to HSDE. This ensures that the explorative power of IHSDE is on average greater than that of HSDE, which in turn results into better accuracy of theIHSDE algorithm.

In the following section, we give some numerical experiments over standard test functions.

4. Computational Results

The effectiveness of the IHSDE algorithm has been evaluated on a test suite of well-known benchmark functions (Table 1) [14–16, 22, 23]. In Table 1, n represents the number of dimensions. Here, $n = 5$, except for function $F_{12} \sim F_{16}$, where $n = 2$. The global minima of all the above functions are at $F(x^*) = 0$, except for F_4, F_{10}, F_{13} and F_{14}. Table 1 summarizes the initialization and search ranges used for these functions.

All the experiments were performed on Windows XP 64 System running on an Hp desktop with Intel(R) Xeon(R) 4×2.4 GHz and 6 GB RAM, and the codes were written in MATLAB R2008a. Simulations were carried out to compare the optimization (minimization) capabilities of the proposed method (IHSDE) with respect to (a) HSDE [13], (b) classical HS (CHS) [1, 2], (c) NGHS [17]. To make the comparison fair, the populations for all the competitor algorithms (for all problems tested) were initialized using the same random seeds. The HS-variant algorithm parameters were set the same parameters: harmony memory size HMS = 10, harmony memory consideration rate HMCR = 0.8, and the number of improvisations $T_{\max} = 10000$. In HSDE, F is chosen to be a uniformly distributed random variable between 0 and 1, and in IHSDE F is chosen to be a uniformly distributed random variable between 0.6 and 1, $F = 0.6 + 0.4 * \mathrm{rand}$, here rand is a uniformly distributed random variable between 0 and 1. In classical HS, we set pitch-adjusting rate PAR = 0.4.

To judge the accuracy of different algorithms, 50 independent runs of each of the four algorithms were carried out and the best, the mean, the worst fitness values, and the standard deviation (Std) were recorded. Table 2 compares the algorithms on the quality of the optimum solution.

Figure 2 shows the convergence and its boxplot of the best fitness in the population for the different algorithms (CHS, HSDE, IHSDE, NGHS) for all benchmark functions. The values plotted for every generation are averaged over 50 independent runs. The boxplot is the best fitness in the final population for the different algorithms (CHS, HSDE, IHSDE, NGHS) over 50 independent runs. We can see that the behavior of the two former algorithms (CHS and HSDE) is similar for all benchmark functions. From the graphical point of view, the classical HS algorithm is clearly the worst algorithm for most benchmark problems, while the IHSDE outperforms HSDE in most cases except for F_2 and F_7. Compared with the two former algorithms (CHS, HSDE), for most test functions, IHSDE can achieve much better optimization results within the same iterations, and IHSDE can reach or exceed the advanced level of the NGHS in most cases.

In order to accurately give search process of each algorithm, we set the same parameters, and run CHS, HSDE, IHSDE method, respectively, for multimodal function F_{16}. Figure 3 shows iterations for three methods.

Remark 4.1. We omitted plots for all the other functions (F_{12}-F_{15}) to save space and also in consideration of the fact that they display more or less the same trend.

Experimental results on benchmark functions show that the IHSDE method can outperform the other methods. From Figure 3, we confirm that the IHSDE method has better ability of global search, and not easily fall in local optimum.

Table 1: Benchmark test functions.

Function name	Benchmark functions expression	Search range	Optimum value
F_1 Ackley function	$F_1(x) = -20e^{(1/5)\sqrt{(1/n)\sum_{i=1}^{n} x_i^2}} - e^{(1/n)\sum_{i=1}^{n}\cos(2\pi x_i)} + 20 - e$	$[-15, 30]^n$	$x^* = (0, 0, \ldots, 0)$, $F_1(x^*) = 0$
F_2 Dixon and Price function	$F_2(x) = (x_1 + 1)^2 + \sum_{i=2}^{n} i(2x_i^2 - x_{i-1})^2$	$[-10, 10]^n$	$F_2(x^*) = 0$
F_3 Levy function	$F_3(x) = \sin^2(\pi y_1) + \sum_{i=1}^{n-1}\left[(y_i - 1)^2\left(1 + 10\sin^2(\pi y_i + 1)\right)\right] + (y_n - 1)^2\left(1 + 10\sin^2(2\pi y_n)\right)$ $y_i = 1 + \frac{x_i - 1}{4}, i = 1, 2, \ldots, n$	$[-10, 10]^n$	$x^* = (1, 1, \ldots, 1)$, $F_3(x^*) = 0$
F_4 Michalewics function	$F_4(x_1, x_2) = -\sum_{j=1}^{2}\sin(x_j)\left(\sin(jx_j^2/\pi)\right)^{2m}; m = 10$	$[0, \pi]^n$	$n = 5; F_4(x^*) = -4.687658$
F_5 Perm function	$F_5(x) = \sum_{k=1}^{n}\left[\sum_{i=1}^{n}(i^k + \beta)\left((x_i/k)^k - 1\right)\right]^2$	$[-n, n]$	$x^* = (1, 2, \ldots, n)$, $F_5(x^*) = 0$
F_6 Powell function	$F_6(x) = \sum_{i=1}^{n/4}[(x_{4i-3} + 10x_{4i-2})^2 + 5(x_{4i-1} - x_{4i})^2 + (x_{4i-2} - x_{4i-1})^2 + 10(x_{4i-3} - x_{4i})^4]$	$[-4, 5]^n$	$x^* = (3, -1, 0, 1, \ldots, 3, -1, 0, 1)$, $F_6(x^*) = 0$
F_7 Rastrigin function	$F_7(x) = 10n + \sum_{i=1}^{n}(x_i^2 - 10\cos(2\pi x_i))$	$[-5.12, 5.12]^n$	$x^* = (0, 0, \ldots, 0)$, $F_7(x^*) = 0$
F_8 Rosenbrock function	$F_8(x) = \sum_{i=1}^{n-1}\left[100(x_i^2 - x_{i-1})^2 + (x_i - 1)^2\right]$	$[-5, 10]^n$	$x^* = (1, 1, \ldots, 1)$, $F_8(x^*) = 0$
F_9 Sphere function	$F_9(x) = \sum_{i=1}^{n} x_i^2$	$[-5.12, 5.12]^n$	$x^* = (0, 0, \ldots, 0)$, $F_9(x^*) = 0$
F_{10} Trid function	$F_{10}(x) = \sum_{i=1}^{n}(x_i - 1)^2 - \sum_{i=2}^{n} x_i x_{i-1}$	$[-n^2, n^2]^n$	$n = 5; F_{10}(x^*) = -30$
F_{11} Zakharov function	$F_{11}(x) = \sum_{i=1}^{n} x_i^2 + (\sum_{i=1}^{n} 0.5ix_i)^2 + (\sum_{i=1}^{n} 0.5ix_i)^4$	$[-5, 10]^n$	$x^* = (0, 0, \ldots, 0)$, $F_{11}(x^*) = 0$
F_{12} Beale function	$F_{12}(x) = (1.5 - x_1 + x_1x_2)^2 + (2.25 - x_1 + x_1x_2^2)^2 + (2.625 - x_1 + x_1x_2^3)^2$	$[-4.5, 4.5]^2$	$x^* = (3, 0.5)$, $F_{12}(x^*) = 0$
F_{13} Easom function	$F_{13}(x) = -\cos(x_1)\cos(x_2)\exp\left(-(x_1 - \pi)^2 - (x_2 - \pi)^2\right)$	$[-100, 100]^2$	$x^* = (\pi, \pi)$, $F_{13}(x^*) = -1$
F_{14} Goldstein and Price function	$F_{14}(x) = \left[1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)\right] \times\left[30 + (2x_1 - 3x_2)^2(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)\right]$	$[-2, 2]^2$	$x^* = (0, -1)$, $F_{14}(x^*) = 3$
F_{15} Hump function	$F_{15}(x) = 4x_1^2 - 2.1x_1^4 + (1/3)x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	$[-5, 5]^2$	$x^* = (0.0898, -0.7126)$ or $x^* = (-0.0898, 0.7126)$, $F_{15}(x^*) = 0$
F_{16} My function	$F_{16}(x) = (x_1^2 + x_2 - 11)^2 + (x_1 + x_2^2 - 7)^2 + 0.1\left((x_1 - 3)^2 + (x_2 - 2)^2\right)$	$[-6, 6]^2$	$x^* = (3, 2)$, $F_{16}(x^*) = 0$

Table 2: The statistical results for 50 runs tested on sixteen benchmark functions.

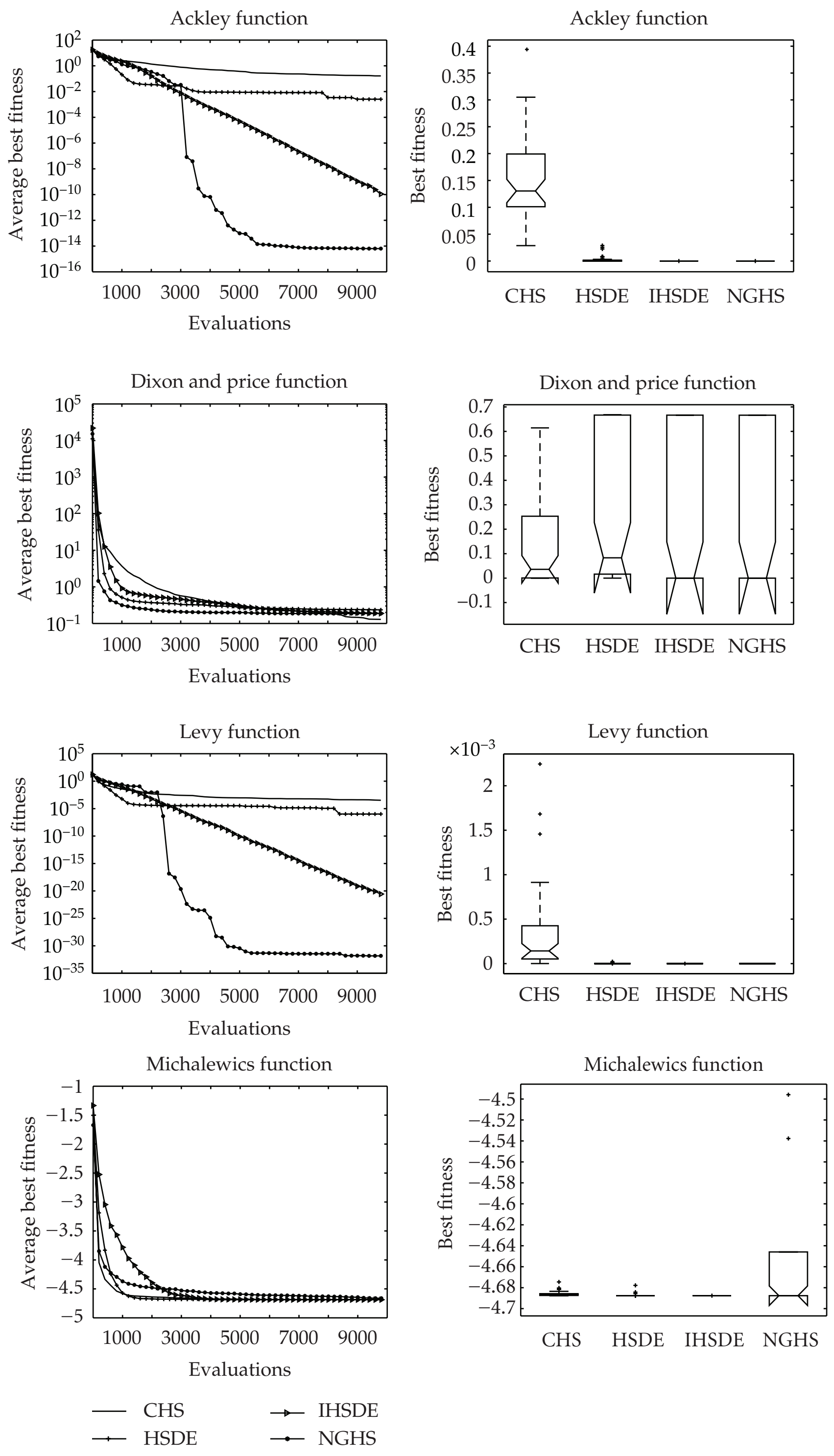
Function name	Algorithm	Best	Mean	Worst	Std
F_1 Ackley function	CHS	$2.8576E-02$	$1.5581E-01$	$3.9193E-01$	$7.6256E-02$
	HSDE	$8.8818E-16$	$2.5301E-03$	$2.8793E-02$	$6.2847E-03$
	IHSDE	$9.0683E-13$	$6.7733E-11$	$1.0303E-09$	$1.5937E-10$
	NGHS	$4.4409E-15$	$6.3594E-15$	$1.5099E-14$	$2.6087E-15$
F_2 Dixon and price function	CHS	$8.8618E-06$	$1.2597E-01$	$6.1434E-01$	$1.5982E-01$
	HSDE	$2.5471E-08$	$2.3591E-01$	$6.6820E-01$	$2.7753E-01$
	IHSDE	$1.6000E-13$	$1.8755E-01$	$6.6667E-01$	$3.0089E-01$
	NGHS	$5.0758E-18$	$1.8658E-01$	$6.6667E-01$	$3.0222E-01$
F_3 Levy function	CHS	$6.8353E-07$	$3.2798E-04$	$2.2443E-03$	$4.5606E-04$
	HSDE	$1.4998E-32$	$1.0481E-06$	$2.3110E-05$	$3.8915E-06$
	IHSDE	$2.2192E-27$	$8.1646E-22$	$1.2126E-20$	$2.2009E-21$
	NGHS	$1.4998E-32$	$1.4998E-32$	$1.4998E-32$	$1.3823E-47$
F_4 Michalewics function	CHS	$-4.6877E+00$	$-4.6859E+00$	$-4.6745E+00$	$2.4486E-03$
	HSDE	$-4.6877E+00$	$-4.6873E+00$	$-4.6778E+00$	$1.4840E-03$
	IHSDE	$-4.6877E+00$	$-4.6877E+00$	$-4.6876E+00$	$3.1708E-06$
	NGHS	$-4.6877E+00$	$-4.6615E+00$	$-4.4959E+00$	$4.4690E-02$
F_5 Perm function	CHS	$3.8305E-01$	$3.0003E+02$	$2.7696E+03$	$5.6432E+02$
	HSDE	$1.1911E-02$	$1.3767E+02$	$4.9112E+03$	$6.9678E+02$
	IHSDE	$7.7908E-02$	$1.1573E+02$	$1.1205E+03$	$1.8330E+02$
	NGHS	$2.5956E-02$	$3.0299E+01$	$3.0618E+02$	$6.7526E+01$
F_6 Powell function	CHS	$4.9848E-06$	$1.5997E-02$	$1.1282E-01$	$1.8387E-02$
	HSDE	$2.3029E-07$	$5.6853E-03$	$3.3194E-02$	$7.9504E-03$
	IHSDE	$2.7878E-07$	$9.6453E-06$	$5.0805E-05$	$1.2526E-05$
	NGHS	$3.4739E-08$	$1.3536E-06$	$3.0482E-06$	$6.9536E-07$
F_7 Rastrigin function	CHS	$1.2141E-03$	$3.8574E-02$	$3.1431E-01$	$5.0927E-02$
	HSDE	$0.0000E+00$	$4.0904E-03$	$8.3710E-02$	$1.3816E-02$
	IHSDE	$4.2633E-14$	$2.5944E+00$	$8.1800E+00$	$2.6713E+00$
	NGHS	$0.0000E+00$	$9.9496E-02$	$9.9496E-01$	$3.0152E-01$
F_8 Rosenbrock function	CHS	$4.3234E-03$	$1.7703E+00$	$7.2601E+00$	$1.9667E+00$
	HSDE	$2.2513E-04$	$1.9541E+00$	$3.3526E+00$	$1.0636E+00$
	IHSDE	$9.5471E-03$	$1.1625E+00$	$2.1110E+00$	$3.9876E-01$
	NGHS	$1.0784E-03$	$9.3133E-01$	$3.5246E+00$	$6.5434E-01$
F_9 Sphere function	CHS	$3.9500E-06$	$2.2513E-04$	$1.3357E-03$	$2.6271E-04$
	HSDE	$1.2298E-48$	$1.1425E-06$	$1.7339E-05$	$3.5860E-06$
	IHSDE	$4.9171E-28$	$2.5756E-23$	$5.8627E-22$	$9.2448E-23$
	NGHS	$8.0887E-67$	$1.2697E-54$	$6.1078E-53$	$8.6366E-54$
F_{10} Trid Function	CHS	$-3.0000E+01$	$-2.9875E+01$	$-2.9156E+01$	$2.3411E-01$
	HSDE	$-3.0000E+01$	$-2.9962E+01$	$-2.9826E+01$	$4.7768E-02$
	IHSDE	$-3.0000E+01$	$-3.0000E+01$	$-3.0000E+01$	$1.4565E-10$
	NGHS	$-3.0000E+01$	$-3.0000E+01$	$-3.0000E+01$	$9.1416E-08$

Table 2: Continued.

Function name	Algorithm	Best	Mean	Worst	Std
F_{11} Zakharov Function	CHS	$2.3830E-05$	$1.9843E-02$	$1.4567E-01$	$3.0974E-02$
	HSDE	$2.2372E-22$	$4.1294E-03$	$1.2299E-01$	$1.8304E-02$
	IHSDE	$2.1141E-15$	$6.1236E-10$	$2.9812E-08$	$4.2138E-09$
	NGHS	$2.5239E-29$	$1.0790E-18$	$3.7173E-17$	$5.3803E-18$
F_{12} Beale function	CHS	$4.3647E-10$	$5.9406E-03$	$9.3633E-02$	$1.7594E-02$
	HSDE	$1.4369E-12$	$3.6930E-03$	$3.9410E-02$	$6.6767E-03$
	IHSDE	$0.0000E+00$	$9.5965E-05$	$3.5321E-03$	$5.0902E-04$
	NGHS	$0.0000E+00$	$4.3172E-01$	$9.3929E+00$	$1.3416E+00$
F_{13} Easom function	CHS	$-9.9999E-01$	$-6.3682E-01$	$-7.6526E-05$	$4.7875E-01$
	HSDE	$-1.0000E+00$	$-9.5984E-01$	$-7.9071E-05$	$1.9790E-01$
	IHSDE	$-1.0000E+00$	$-1.0000E+00$	$-1.0000E+00$	$0.0000E+00$
	NGHS	$-1.0000E+00$	$-7.8001E-01$	$0.0000E+00$	$4.1842E-01$
F_{14} Goldstein and Price function	CHS	$3.0000E+00$	$3.0016E+00$	$3.0277E+00$	$4.1190E-03$
	HSDE	$3.0000E+00$	$3.0003E+00$	$3.0063E+00$	$1.1281E-03$
	IHSDE	$3.0000E+00$	$3.0000E+00$	$3.0000E+00$	$1.6324E-07$
	NGHS	$3.0000E+00$	$1.5420E+01$	$8.4000E+01$	$2.5122E+01$
F_{15} Hump function	CHS	$4.6668E-08$	$2.8159E-05$	$3.5591E-04$	$5.9106E-05$
	HSDE	$4.6510E-08$	$1.7936E-06$	$4.2616E-05$	$8.1164E-06$
	IHSDE	$4.6510E-08$	$3.3050E-07$	$1.3125E-05$	$1.8532E-06$
	NGHS	$4.6510E-08$	$4.6510E-08$	$4.6510E-08$	$1.0318E-16$
F_{16} My function	CHS	$1.3297E-08$	$4.7659E-01$	$3.4872E+00$	$7.9292E-01$
	HSDE	$0.0000E+00$	$1.5107E-01$	$1.5166E+00$	$4.5650E-01$
	IHSDE	$0.0000E+00$	$1.4701E-07$	$7.1040E-06$	$1.0045E-06$
	NGHS	$0.0000E+00$	$2.1788E+00$	$7.3673E+00$	$2.5506E+00$

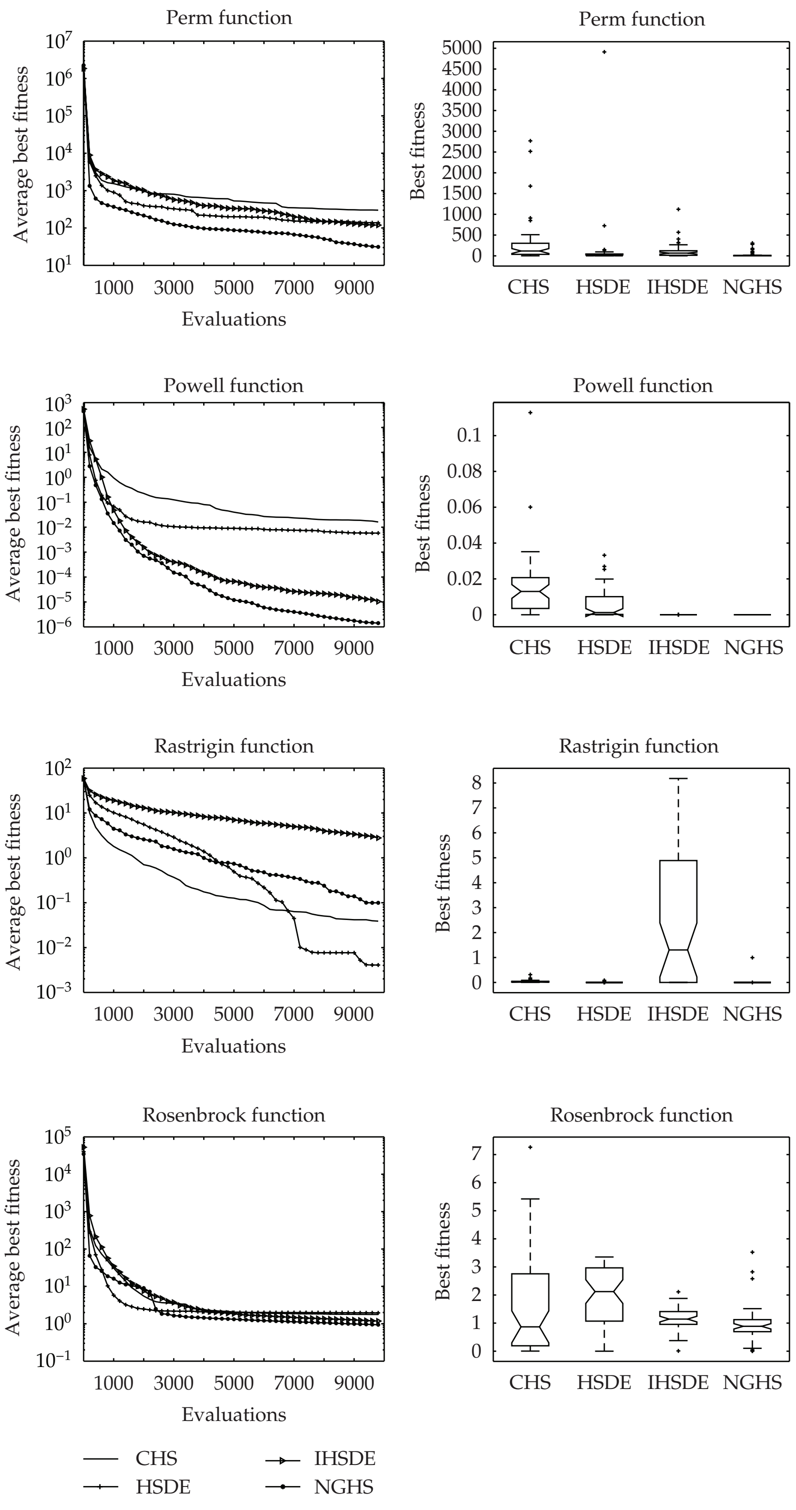
5. Conclusion

This paper has presented an improved harmony search algorithm by blending with it a different vector-based mutation operator borrowed from the DE algorithms. The HM members are fine tuned by the DE's mutation operator to improve their affinities so that enhanced optimization performances can be achieved. Mathematical analysis indicates that the IHSDE posses an increasing population variance as compared to HSDE. This ensures that the explorative power of IHSDE is on average greater than that of HSDE, which in turn results in better accuracy of the IHSDE algorithm. Several simulation examples of the unimodal and multimodal functions have been used to verify the effectiveness of the proposed methods. Compared with the HSDE and CHS, better optimization results are obtained using IHSDE approaches in most cases. Checking the effect of variation of the scale factor F of the differential mutation operator may be a worthy issue for preventing the premature convergence in future investigations. Future works will also focus on studying the applications of IHSDE on engineering optimization problems.



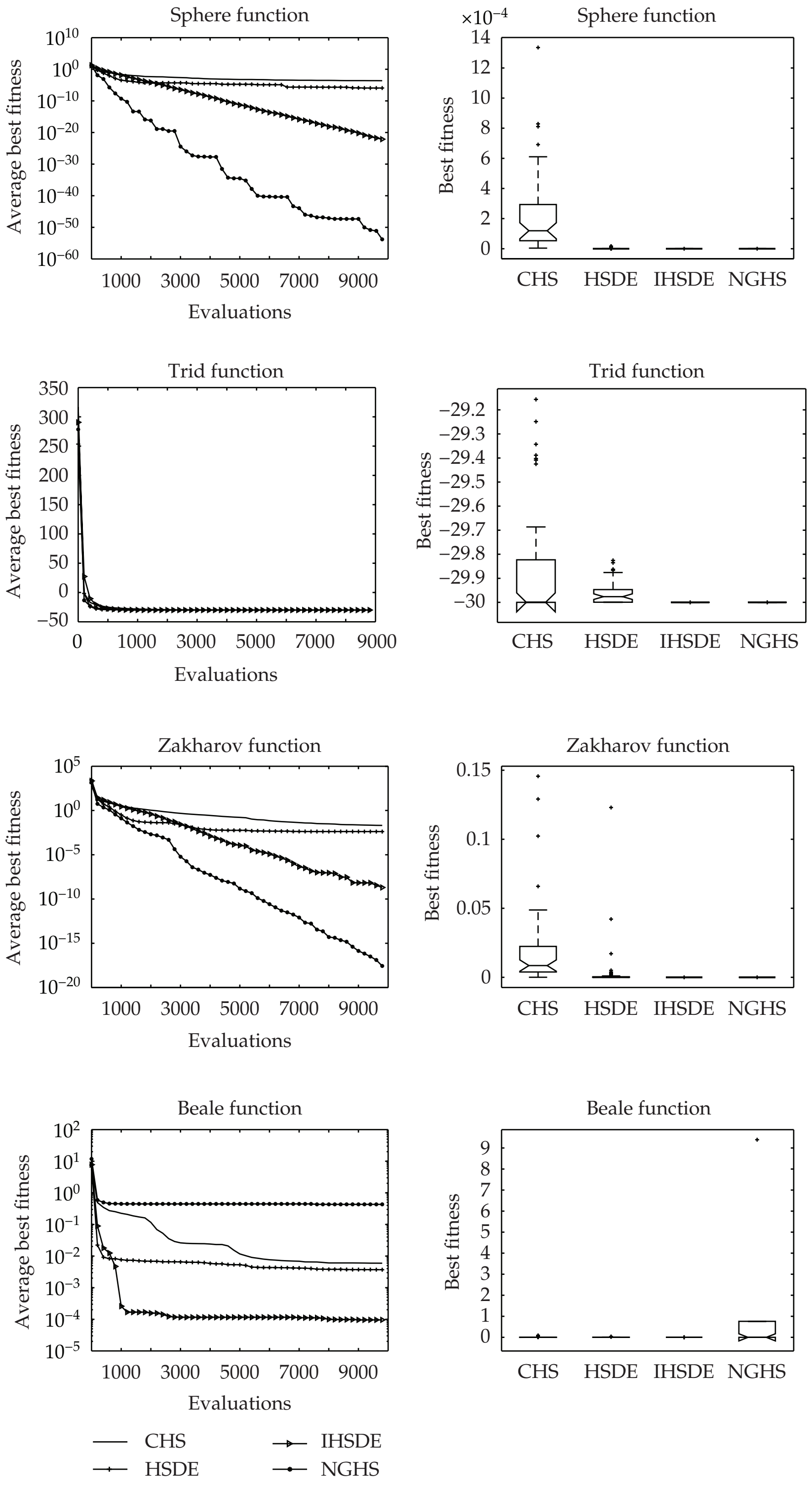
(a)

Figure 2: Continued.



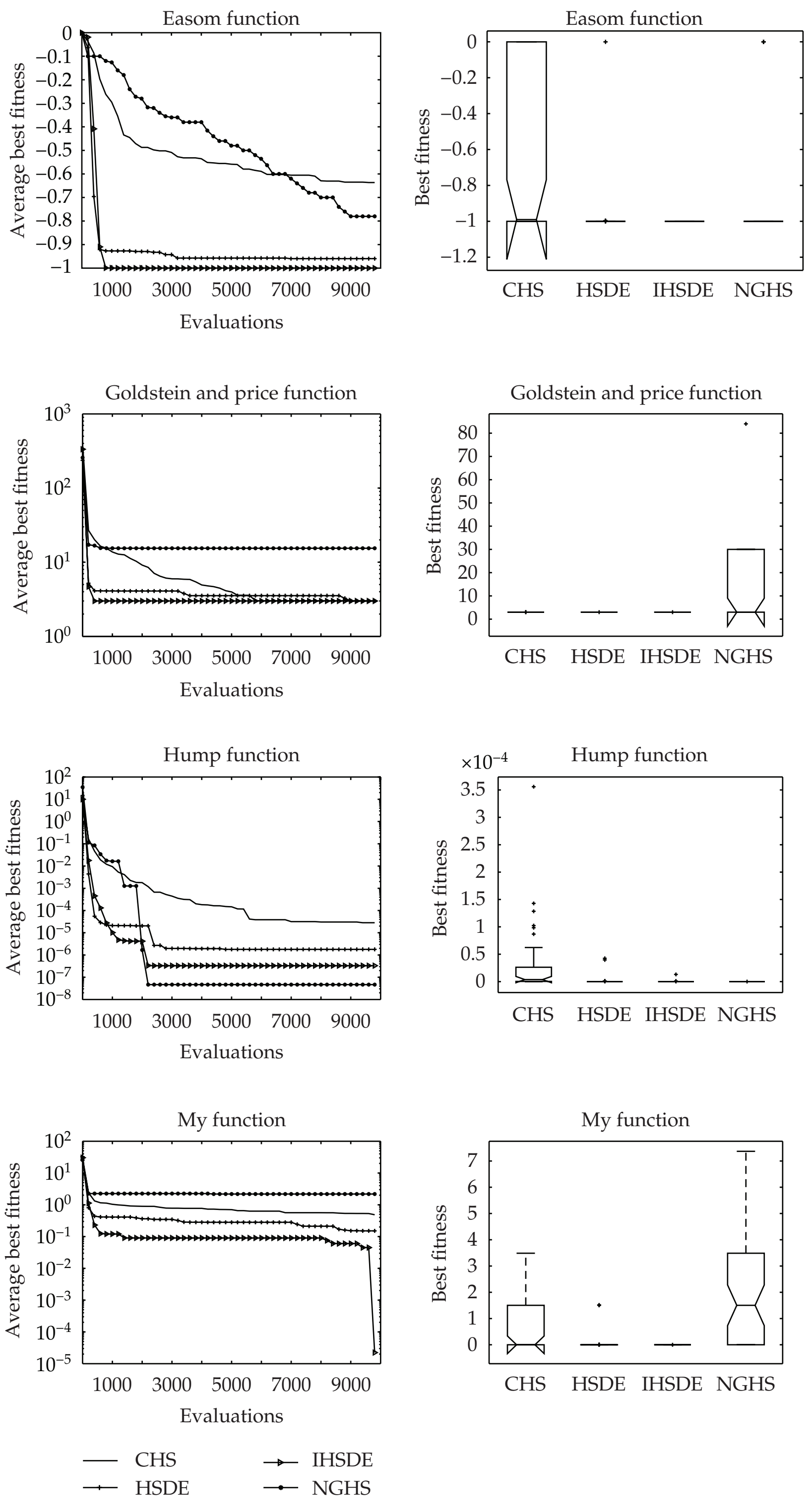
(b)

Figure 2: Continued.



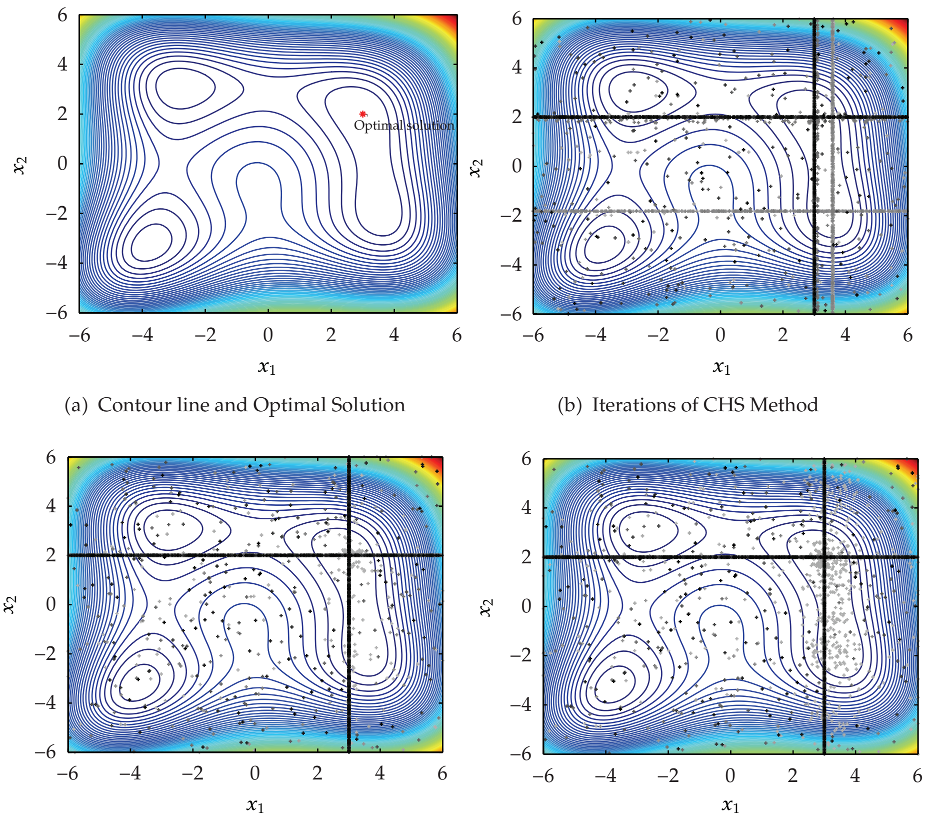
(c)

Figure 2: Continued.



(d)

Figure 2: The convergence and its boxplot.



(a) Contour line and Optimal Solution

(b) Iterations of CHS Method

(c) Iterations of HSDE Method

(d) Iterations of IHSDE Method

Figure 3: Show search processes of CHS, HSDE, and IHSDE methods for F_{16}.

Acknowledgments

This work is supported by National Natural Science Foundation of China under Grant no. 60974082, and Scientific Research Program Funded by Shaanxi Provincial Education Department under Grant no. 12JK0863.

References

[1] Z. W. Geem, J. H. Kim, and G. V. Loganathan, "A new heuristic optimization algorithm: harmony search," *Simulation*, vol. 76, no. 2, pp. 60–68, 2001.

[2] K. S. Lee and Z. W. Geem, "A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice," *Computer Methods in Applied Mechanics and Engineering*, vol. 194, no. 36-38, pp. 3902–3933, 2005.

[3] G. G. Roy, P. Chakraborty, and S. Das, "Designing fractional-order PIλDμ controller using differential harmony search algorithm," *International Journal of Bioinspired Computing*, vol. 2, no. 5, pp. 303–309, 2010.

[4] B. K. Panigrahi, V. R. Pandi, S. Das, and Z. Cui, "Dynamic economic load dispatch with wind energy using modified harmony search," *International Journal of Bio-Inspired Computing*, vol. 2, no. 3-4, pp. 282–289, 2010.

[5] V. R. Pandi, B. K. Panigrahi, R. C. Bansal, S. Das, and A. Mohapatra, “Economic load dispatch using hybrid swarm intelligence based harmony search algorithm,” *Electric Power Components and Systems*, vol. 39, no. 8, pp. 751–767, 2011.

[6] B. K. Panigrahi, V. R. Pandi, S. Das, and A. Abraham, “A bandwidth-adaptive harmony search algorithm to solve optimal power flow problems with non-smooth cost functions,” in *Recent Advances in Harmony Search Algorithm*, Z. W. Geem, Ed., Studies in Computational Intelligence, pp. 65–75, Springer, 2010.

[7] S. Kulluk, L. Ozbakir, and A. Baykasoglu, “Training neural networks with harmony search algorithms for classification problems,” *Engineering Applications of Artificial Intelligence*, vol. 25, no. 1, pp. 11–19, 2012.

[8] T. K. Gandhi, P. Chakraborty, G. G. Roy, and B. K. Panigrahi, “Discrete harmony search based expert model for epileptic seizure detection in electroencephalography,” *Expert Systems with Applications*, vol. 39, no. 4, pp. 4055–4062, 2012.

[9] I. Ahmad, M. G. Mohammad, A. A. Salman, and S. A. Hamdan, “Broadcast scheduling in packet radio networks using Harmony Search algorithm,” *Expert Systems with Applications*, vol. 39, no. 1, pp. 1526–1535, 2012.

[10] M. A. Al-Betar, A. T. Khader, and M. Zaman, “University course timetabling using a hybrid harmony search metaheuristic algorithm,” *IEEE Transactions on Systems, Man and Cybernetics Part C*, vol. 42, no. 5, pp. 664–681, 2012.

[11] A. Kaveh and M. Ahangaran, “Discrete cost optimization of composite floor system using social harmony search model,” *Applied Soft Computing*, vol. 12, no. 1, pp. 372–381, 2012.

[12] M. G. H. Omran and M. Mahdavi, “Global-best harmony search,” *Applied Mathematics and Computation*, vol. 198, no. 2, pp. 643–656, 2008.

[13] P. Chakraborty, G. G. Roy, S. Das, D. Jain, and A. Abraham, “An improved harmony search algorithm with differential mutation operator,” *Fundamenta Informaticae*, vol. 95, no. 4, pp. 401–426, 2009.

[14] X. Z. Gao, X. Wang, and S. J. Ovaska, “Uni-modal and multi-modal optimization using modified Harmony Search methods,” *International Journal of Innovative Computing, Information and Control*, vol. 5, no. 10, pp. 2985–2996, 2009.

[15] C. M. Wang and Y. F. Huang, “Self-adaptive harmony search algorithm for optimization,” *Expert Systems with Applications*, vol. 37, no. 4, pp. 2826–2837, 2010.

[16] Q. K. Pan, P. N. Suganthan, M. F. Tasgetiren, and J. J. Liang, “A self-adaptive global best harmony search algorithm for continuous optimization problems,” *Applied Mathematics and Computation*, vol. 216, no. 3, pp. 830–848, 2010.

[17] D. Zou, L. Gao, J. Wu, and S. Li, “Novel global harmony search algorithm for unconstrained problems,” *Neurocomputing*, vol. 73, no. 16–18, pp. 3308–3318, 2010.

[18] O. M. Alia and R. Mandava, “The variants of the harmony search algorithm: an overview,” *Artificial Intelligence Review*, vol. 36, no. 1, pp. 49–68, 2011.

[19] S. Das, A. Mukhopadhyay, A. Roy, A. Abraham, and B. K. Panigrahi, “Exploratory power of the harmony search algorithm: analysis and improvements for global numerical optimization,” *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, vol. 41, no. 1, pp. 89–106, 2011.

[20] S. Das and P. N. Suganthan, “Differential evolution: a survey of the state-of-the-art,” *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 1, pp. 4–31, 2011.

[21] R. Ga Mperle, S. D. Muller, and P. Koumoutsakos, “A parameter study for differential evolution,” in *Proceedings of the WSEAS International Conference on Advances in Intelligent Systems, Fuzzy Systems, Evolutionary Computation*, pp. 293–298, 2002.

[22] P. N. Suganthan, N. Hansen, J. Liang et al., “Problem definitions and evaluation criteria for the CEC2005 special session on real-parameter optimization,” Tech. Rep. KanGAL 2005005, Nanyang Technological University, Singapore, IITKanpur, India, 2005.

[23] http://www-optima.amp.i.kyoto-u.ac.jp/member/student/hedar/Hedar_files/TestGO_files/Page 364.htm.