IJST, Transactions of Civil Engineering, Vol. 36, No. C2, pp 121-137
Printed in The Islamic Republic of Iran, 2012

SOCIAL HARMONY SEARCH ALGORITHM FOR CONTINUOUS OPTIMIZATION*

A. KAVEH** AND M. AHANGARAN

Centre of Excellence for Fundamental Studies in Structural Engineering, Iran University of Science and Technology, Narmak, Tehran-16, I. R. of Iran
Email: alikaveh@iust.ac.ir

Abstract– This paper presents a social harmony search algorithm to solve optimization problems with continuous design variables. Although the Harmony Search (HS) algorithm (HSA) has proven its ability in finding near global regions within a reasonable time, it is rather inefficient in performing local search. The proposed method applies the harmony search optimizer for global optimization and normal distribution is employed to update the position of each design variable of a new harmony found by the first rule of the HS (memory consideration) in every stage to rapidly attain the feasible solution space. Normal distribution works as a global search in early iterations and as a local search in final iterations to improve HS in order to quickly converge and find better solutions. Various benchmark optimization problems are used to illustrate the effectiveness and robustness of the proposed method. Finally, the experimental results reveal the superiority of the proposed method in quick convergence and finding better solutions compared to the classic HS, its recently developed variants, and some other optimization algorithms.

Keywords– Social harmony search, normal distribution, meta-heuristics, optimization, diversification, intensification

1. INTRODUCTION

In recent years, researchers have shown an increasing interest in the study of meta-heuristic algorithms, including evolutionary algorithms. Meta-heuristic literally means to find the solution using higher-level techniques [1]. There are two important components in meta-heuristic algorithms, named intensification and diversification. This is very important to make an efficient balance between these two seemingly opposing components [2]. In the diversification section one tries to search all solution space, and in the intensification section search is performed only in a fraction of solution space. Therefore, if the diversification is too intense, the algorithm converges very slowly and a desirable solution cannot be obtained with small number of iterations. Conversely, if the intensification is too intense, the algorithm may be trapped in a local optimum causing a premature convergence. In the early iterations one should have high diversification with low intensification, and in subsequent iterations diversification should be reduced gradually while increasing the intensification.

Recently Geem et al. [3] proposed a new evolutionary algorithm called Harmony Search (HS) algorithm that is used in many optimization problems, including function optimization, engineering optimization, NP-complete problems, vehicle routing, design of water distribution networks, groundwater modeling, and structural design [4-8].

This study presents a new version of HS, where concepts of normal distribution are used to improve the performance of the HS and its variants, and thus to achieve a better balance between diversification

*Received by the editors March 15, 2010; Accepted May 16, 2011.
**Corresponding author

and intensification. The new version of HS is named social harmony search (SHS). Additionally, with this method, we have simplified the second rule of the HS (pitch adjustment). We replace the bandwidth parameter with a new parameter that is adjusted automatically according to the previous experiments of all harmonies.

In order to show the effectiveness and robustness of this method, the algorithm is applied to various optimization problems. Numerical results demonstrate the capabilities of the proposed method, its quick convergence and ability to find a better solution compared to the classic HS.

The rest of this paper is organized as follows: section 2 provides a summarized overview of the classic HS algorithm and its recently developed variants. In section 3, we introduce the proposed method. In section 4, results of the experiments are presented and discussed. Finally, in section 5 some conclusions are presented.

2. OVERVIEW OF THE CLASSIC HARMONY SEARCH

In this section, a brief review of the harmony search algorithm and its recently developed variants are presented.

a) Classic harmony search algorithm

Recently, a new meta-heuristic optimization algorithm–harmony search (HS) with continuous design variables was developed by Geem et al. [3]. This meta-heuristic algorithm is conceptualized using the musical improvisation process of searching for a perfect state of harmony [3]. A musician normally seeks to find pleasing harmony as determined by an aesthetic standard, just as the optimization process seeks to find a global optimum as determined by an objective function. The pitch of each musical instrument determines the aesthetic quality, just as the set of values assigned to each decision variable determines the objective function value. The HS works as follows:

Step 1: Initialize the optimization problem and HS parameters.
Step 2: Initialize the harmony memory (HM).
Step 3: Improvise a new harmony.
Step 4: Update the HM.
Step 5: Repeat steps 3 and 4 until the termination criterion is satisfied.

(1) ***Step 1. Initialize the optimization problem and HS parameters.*** In step 1, the optimization problem and its decision variables bounds are defined. In addition, the parameters of HS are specified in this step. These parameters consist of the harmony memory size (HMS), harmony memory considering rate (HMCR), pitch adjusting rate (PAR) and the number of improvisation (NI). A large value for the HMS increases the search strength of the algorithm, and conversely a small value causes a quick convergence without performing a complete search. However, the running time increases when the number of HMS increases, and we cannot choose a high number for the HMS. Here, HMCR and PAR are parameters that are used to improve the solution vector. Both are defined in step 3.

(2) ***Step 2: Initialize the harmony memory (HM).*** In step 2, the harmony memory (HM) is filled as follows: $x_i^j = LB_i + r \times (UB_i - LB_i)$, $1 \le i \le N$, $j = 1,2,...,HMS$, where $r \sim U(0,1)$. LB_i and UB_i are the lower and upper bounds of each decision variable, respectively, and N is the number of decision variables.

(3) ***Step 3: Improvise a new harmony.*** Improvise a new harmony using tree rules: memory consideration, pitch adjustment and random selection. The three rules of HS can be summarized as follows:

while ($i \leq$ number of variables decision (N))
  if ($r_1 \leq HMCR$) **then**, memory consideration
    $x_i' = x_i^j$, where $j \sim$ int(U(0,*HMS*))+1
    if ($r_2 \leq PAR$) **then**, pitch adjustment
      $x_i' = x_i' + r_3 \times bw$, where $r_3 \sim U$ (−1, 1) and bw is an arbitrary distance bandwidth
    end if
  else random selection
    $x_i' = LB_i + r \times (UB_i - LB_i)$
  end if
end while

Memory consideration is the first rule of HS. Its importance is because it ensures that good harmonies are considered as elements of the new solution vectors. The effectiveness of this rule is shown by the value of *HMCR*. The parameter *HMCR*, which varies between 0 and 1, controls the balance between diversification and intensification [1]. If this rate is too low, it causes the algorithm to have a high diversification and low intensification, and only a few elite harmonies are selected, so it may converge too slowly. If this rate is extremely high, it leads the algorithm to a low diversification and high intensification, and the pitches in the harmony memory will be mostly used, while other ones will not be explored well, leading to a premature convergence.

The pitch adjustment is the second rule of HS which has parameters such as pitch bandwidth (bw) and *PAR*. As the pitch adjustment in music means changing the frequency, it means generating a slightly different value in the HS [1]. The pitch adjustment step is similar to the local search mechanism. Consequently, it clearly shows that *PAR* and *bw* have a great influence on the quality of the final solutions. A low *PAR* with a narrow bandwidth can slow down the convergence of HS because of the limitation in the diversification. Conversely, a very high *PAR* for a wide bandwidth causes the algorithm to not be able to achieve all potential optimums because it works like a random search algorithm.

The third rule is the random selection. In this section there is no balance between diversification and intensification. HS algorithm uses this rule to increase the diversity of the solution. The use of randomization can drive the system further to explore various diverse solutions so as to attain the global optimality [1].

(4) ***Step 4: Update the HM.*** If the new harmony has a better objective function value than the worst harmony in the HM, the new harmony $x' = (x_1', x_2', ..., x_N')$ replaces the worst harmony.

(5) ***Step 5: Repeat Steps 3 and 4 until the termination criterion is satisfied.*** The computation is terminated when the maximum number of improvisations is met. Otherwise, steps 3 and 4 are repeated.

b) Improved harmony search algorithm

The original HS algorithm uses fixed values for both *PAR* and *bw* and these parameters cannot be changed during the new improvisation. This drawback prevents the algorithm from finding an optimal solution in a reasonable number of iterations.

A low *PAR* with a wide bandwidth in early iterations can enforce the algorithm to have a good diversification, and a very high *PAR* with a narrow bandwidth in the final iterations can cause the algorithm to have a good intensification. Mahdavi et al. [9] proposed a variant of HS, called the improved harmony search (IHS), to dynamically increase *PAR* and decrease *bw*, respectively. The key difference between IHS and the traditional HS method is in the way of adjusting *PAR* and *bw*. The IHS dynamically updates *PAR* and *bw* according to the following equations:

$$\mathrm{PAR}(t) = \mathrm{PAR}_{\min} + \frac{(\mathrm{PAR}_{\max} - \mathrm{PAR}_{\min})}{(\mathrm{NI} - 1)} \times (t - 1) \tag{1}$$

where

$PAR(t)$	pitch adjusting rate for each generation
$PAR_{\min}$	minimum pitch adjusting rate
$PAR_{\max}$	maximum pitch adjusting rate
NI	number of solution vector generations
t	generation number

and

$$\mathrm{bw}(t) = \mathrm{bw}_{\max} \times e^{\left(\frac{\ln\left(\frac{\mathrm{bw}_{\min}}{\mathrm{bw}_{\max}}\right)}{(\mathrm{NI}-1)} \times (t-1) \right)} \tag{2}$$

where

$bw(t)$	bandwidth for each generation
$bw_{\min}$	minimum bandwidth
$bw_{\max}$	maximum bandwidth

c) Global-best harmony search algorithm

Another developed variant of HS, called the global-best harmony search (GHS), is developed by Omran & Mahdavi [10] and is inspired by the concept of swarm intelligence as proposed in Particle Swarm Optimization (PSO) [11,12]. This variation of HS simplifies the pitch adjustment step by eliminating the difficulties in selecting bw; however, it cannot make a good balance between diversification and intensification in pitch adjustment section.

The GHS has exactly the same steps as that of the IHS with the exception that step 3 is modified as follows:

while ($his \leq$ number of variables decision (N))
  if ($r_1 \leq HMCR$) **then** (memory consideration)
    $x_i' = x_i^j$, where $j \sim$ int(U(0,HMS))+1
    if ($r_2 \leq PAR(t)$) **then** (pitch adjustment)
      $x_i' = x_k^{best}$, where *best* is the index of the best harmony in the HM and $k \sim$ int(U(0,N))+1
    end if
  else (random selection)
    $x_i' = LB_i + r \times (UB_i - LB_i)$
  end if
end while

Selecting a new value for the corresponding decision variable from the current best harmony from the harmony memory causes a premature convergence and makes the algorithm unable to find the global optimum [6].

3. PROPOSED METHOD

As mentioned earlier, diversification and intensification are two important parameters in meta-heuristic algorithms. It is very important to make a good balance between these two important parameters. This insures that the final solutions are good results for corresponding optimization problems. One should gradually decrease diversification while at the same time, increasing intensification. In the HS algorithm two intelligent sections (memory consideration and pitch adjustment) were used to select a new value for decision variables. In the memory consideration section, HS makes a good balance between diversification and intensification. Since solutions often start with some randomly generated ones being inserted in harmony memory and gradually reducing their diversification while simultaneously increasing their intensification selecting a value from harmony memory in early iteration makes a good global search (diversification) and in the final iterations makes a good local search (intensification). The one serious drawback of the HS algorithm arising from the pitch adjustment section, is that it makes the algorithm incapable of making a good balance between these two important parameters. As mentioned before, some researchers have recently tried to eliminate this drawback, and some new derivations of the HS have been developed.

The key difference between social HS and earlier methods of the HS is in the way of adjusting a new harmony found by the first rule of harmony search (memory consideration). To improve the performance of the HS algorithm and its derivations, social HS algorithm uses the normal distribution for adjusting the new harmony found by the memory consideration rule, then one should check whether the current decision variable value violates the variable bounds or not. If it does, it should be reset to the previous position. When the harmonies search in the feasible space to find the solution, if any one of them searches into the infeasible region, it will be forced to come back to the previous position to guarantee a feasible solution. The harmony which comes back to the previous position may be closer to the boundary at the next iteration. This allows the harmonies to search to the global minimum with a higher probability. This method is called the fly-back mechanism [13]. Some experimental results have shown that it can find a better solution with fewer iterations than other techniques [13]. Additionally, the social HS uses all values of the *i*th decision variables in harmony memory to adjust the new harmony. This advantage enables the algorithm to find the new harmony with more social influence and uses experiments of all harmonies. The social HS algorithm adjusts the new harmony according to the following equations:

$$N(x_i', \sigma_i') \tag{3}$$

Where

x_i' the selected value of the *i*th variable in the HM

σ_i' the variance value of the *i*th variable in the HM

$N(x_i', \sigma_i')$ denotes a random number normally distributed with mean value x_i' and variance σ_i'.

$$\sigma_i' = \xi \times \sum_{j=1}^{HMS} \frac{\left| x_i' - x_i^j \right|}{HMS - 1} \tag{4}$$

Where

HMS harmony memory size

x_i^j all values of the *i*th decision variable in the HM. $1 \le j \le HMS$ and $1 \le i \le N$

ξ fixed value to adjusting the σ_i'. (This parameter is described in section 4)

In the proposed method, the pitch adjustment step was simplified and the mentioned drawback of HS and its derivations were eliminated. In this method σ_i' is similar to *bw*. As mentioned before, we need to have a wide and low bandwidth, respectively in the early and final iterations in order to attain diversification and intensification. Thus, because of the high variance between the selected *i*th decision variable and other *i*th decision variables (σ_i'), in the harmony memory in early iterations, normal distribution makes global search (exploration). Conversely, because of the low variance between the selected *i*th decision variable and other *i*th decision variables (σ_i'), in the harmony memory in the final iterations, normal distribution makes local search (exploitation). Therefore, (σ_i') is adjusted automatically according to the previous experiments of all harmonies. Consequently, the pitch adjustment step is used for suitable search in all iterations of the algorithm. This method ensures that HS algorithm achieves a good balance between diversification and intensification in the pitch adjustment step. In this method we also increase the sociality of HS algorithm, since the new selected variable is adjusted according to the previous experiments of all the harmonies.

In this method, fixed high values are used for *HMCR* and *PAR*, because of a good balance between diversification and intensification in the first two rules of HS. In the proposed method, the value of *HMCR*=0.99 is chosen for all the examples to decrease the random selection probability to 1%, as a musician does not play any pitch out of the perfect pitches in his/her memory with a high probability. Also, in all the examples, the value of *PAR* is fixed on 1, the reason can easily be realized from the above discussion. Social HS has exactly the same steps as that of the HS, with the exception of step 3 which is modified as follows:

while ($i \leq$ number of decision variables (N))
  if (rand $\leq$ *HMCR*) **then** (memory consideration)
    $x_i' = x_i^j$, where $j \sim$ U(1, …, *HMS*)
    if (rand $\leq$ *PAR*) **then** (pitch adjustment)
      $x_i' = N(x_i', \sigma_i')$ (pitch adjustment)
    end if
  else (Random selection)
    $x_i' = LB_i + r \times (UB_i - LB_i)$
  end if
end while

Figure 1 shows the flowchart of the new harmony improvisation. As illustrated, the key difference between the social HS and the classic HS methods is in the pitch adjustment section.

4. NUMERICAL EXAMPLES

In this study, several standard benchmark optimization examples are utilized to show the superiority of the proposed method in quick convergence and finding better solutions compared to the original HS, its recently developed variants, and other optimization algorithms. This method is implemented in a FORTRAN computer program.

a) Unconstrained function minimization examples

The following uni-modal and multi-modal functions are used as unconstrained function minimization examples. All functions are implemented in 30 and 100 dimensions except for the two-dimensional Camel-Back function. There is a balance between uni-modal and multi-modal functions among our tests.

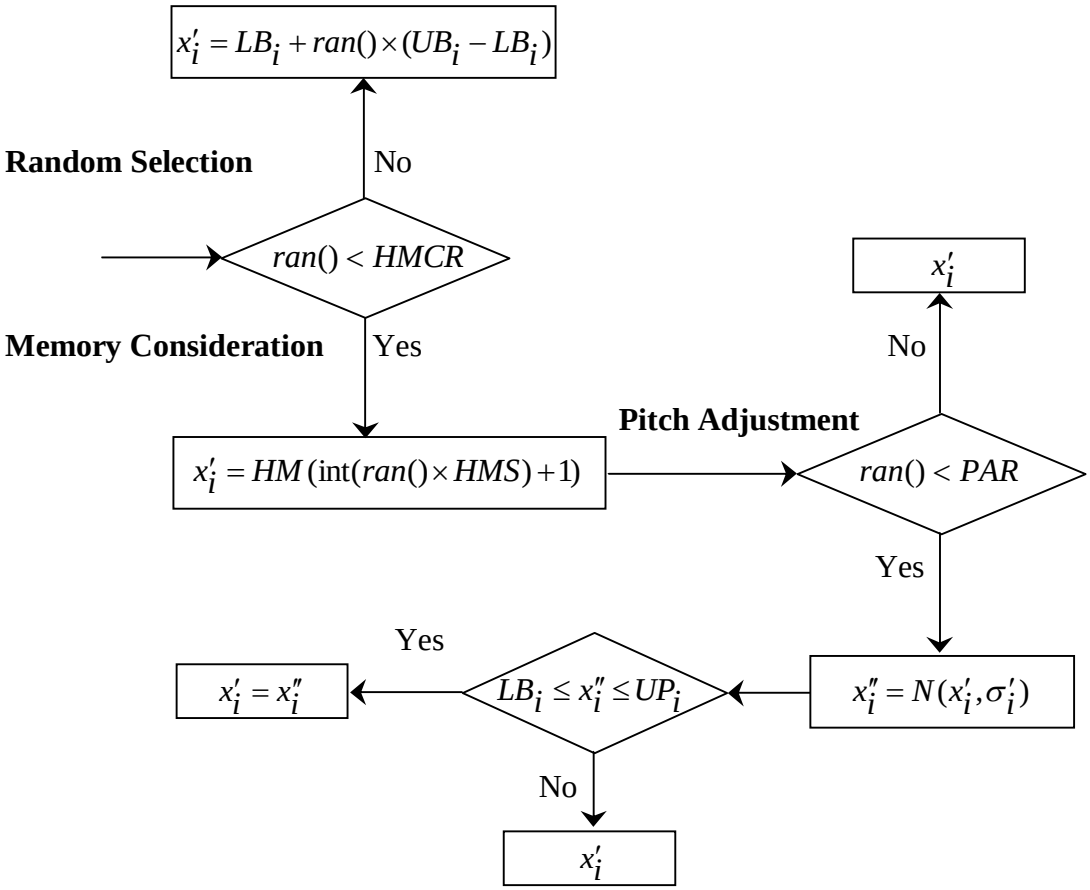


Fig. 1. The flowchart of the social harmony search algorithm (i=1 to N; N= number of decision variables)

The Camel-Back function is a low-dimensional one with only a few local optimal, while Sphere, Rosenbrock, Schwefel's Problem 2.22 and Rotated Hyper-ellipsoid are uni-modal functions. And finally, Ackley, Griewank and Schwefel's Problem 2.26 are difficult multi-modal functions.

Table 1 shows parameters for all methods. These values are suggested by Lee and Geem [4], Mahdavi et al. [6], and Omran and Mahdavi [10]. Table 2 summarizes the information of test functions. The initial harmony search is filled from a uniform distribution random in the feasible ranges specified in Table 2. Also, Table 3 shows the values of ξ that are used for these tests. For camel back function, $\xi = 1.2$ is adopted.

1. Numerical results: The maximum number of evaluations of the objective function is limited to 50000 iterations. The results obtained by all the algorithms in 30 and 100 dimensions are summarized in Tables 4 and 5, respectively. The numbers show the average value of objective functions and standard deviations from 30 runs. The results for HS, IHS and GHS algorithms were reported by Omran and Mahdavi [10]. The best solutions among all algorithms are shown in bold. The results show that the proposed method outperformed HS and its derivation in all test examples.

Table 1. Parameters for all the methods used for unconstrained problems

Methods	HMS	HMCR	PAR	PAR_{min}	PAR_{max}	bw	bw_{min}	bw_{max}
HS	5	0.9	0.3	–	–	0.01	–	–
IHS	5	0.9	–	0.01	0.99	–	0.0001	1/(20 × (UB–LB))
GHS	5	0.9	–	0.01	0.99	–	–	–
Proposed method	15	0.99	1	–	–	–	–	–

Table 2. Unconstrained functions

Functions	Function ange	x*	f(x*)
$Sphere = \sum_{i=1}^{n} x_i^2$	$-100 \le x_i \le 100$	$[0,...,0]$	0
$Rosenbrock = \sum_{i=1}^{n-1} 100(x_i - x_{i-1}^2)^2 + (x_{i-1} - 1)^2$	$-30 \le x_i \le 30$	$[1,...,1]$	0
$Ackley = -20\exp(-0.2\sqrt{(\frac{1}{30}\sum_{i=1}^{n} x_i^2)}) - \exp(\frac{1}{30}\sum_{i=1}^{n}\cos(2\pi x_i)) + 20 + \exp(1)$	$-32 \le x_i \le 32$	$[0,...,0]$	0
$Griewank = \sum_{i=1}^{n} \frac{(x_i)^2}{4000} - \prod_{i=1}^{n}\cos(\frac{x_i}{\sqrt{i}}) + 1$	$-600 \le x_i \le 600$	$[0,...,0]$	0
$Schwefel's\ Problem 2.22 = \sum_{i=1}^{n}\lvert x_i\rvert + \prod_{i=1}^{n}\lvert x_i\rvert$	$-10 \le x_i \le 10$	$[0,...,0]$	0
$Rotated\ hyper\_ellipsoid = \sum_{i=1}^{n}\left(\sum_{j=1}^{i} x_j\right)^2$	$-100 \le x_i \le 100$	$[0,...,0]$	0
$Schwefel's\ Problem 2.26 = -\sum_{i=1}^{n}\left(x_i \sin\left(\sqrt{\lvert x_i\rvert}\right)\right)$	$-500 \le x_i \le 500$	$[420.9687,...,420.9687]$	-12569.5
$Six\ Hump\ Camel\ Back = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	$-5 \le x_i \le 5$	$[-0.08983, 0.7126]$	-1.0316285

Table 3. The value of ξ

		Function						
	ξ	Sphere	Rosenbrock	Ackley	Griewank	Schwefel 2.22	Rotated hyper-ellipsoid	Schwefel 2.26
Dimension	30	1.2	1.2	1.2	1.2	1.2	1.2	1.2
	100	1.0	1.0	0.05	1.0	1.0	0.8	1.8

Table 4. Mean and standard deviation of the benchmark functions optimization results in 30 dimensions

	HS	IHS	GHS	Proposed method
Sphere	1.87E–04 (3.20E–05)	7.12E–04 (6.44E–04)	1.00E–05 (2.20E–05)	**1.40E–45** **(1.68E–45)**
Rosenbrock	3.40E+02 (2.67E+2)	6.24E+02 (5.60E+02)	4.97E+01 (5.91E+01)	**3.66E+01** **(2.09E+01)**
Ackley	1.13E+00 (4.07E–01)	1.89E+00 (3.15E–01)	2.10E–02 (2.17E–02)	**3.92E–07** **(1.00E–07)**
Griewank	1.12E+00 (4.12E–02)	1.12E+00 (4.09E–02)	1.02E–01 (1.76E–01)	**7.68E–3** **(1.02E–2)**
Schwefel 2.22	1.71E–01 (7.28E–02)	1.10E+00 (1.81E–01)	7.28E–02 (1.14E–01)	**3.38E–42** **(3.40E–42)**
Rotated hyper-ellipsoid	4.30E+03 (1.36E+03)	4.31E+03 (1.06E+03)	5.15E+03 (6.35E+03)	**1.79E+01** **(9.34E+00)**
Schwefel 2.26	–12539.237786 (1.2E+01)	–12534.968625 (1.04E+01)	–12569.458343 (5.04E–02)	**–12569.48** **(0.00E+00)**
Camel Back	**–1.031628** **(0.00E+00)**	**–1.031628** **(0.00E+00)**	–1.031600 (1.80E–05)	**–1.031628** **(0.00E+00)**

Table 5. Mean and standard deviation of the benchmark functions optimization results in 100 dimensions

	HS	IHS	GHS	Proposed method
Sphere	8.68E+00 (7.75E–01)	8.84E+00 (7.62E–01)	2.23E+00 (5.65E–01)	**3.65E–03** **(3.95E–03)**
Rosenbrock	1.67E+07 (3.18E+06)	1.73E+07 (2.94E+06)	2.60E+06 (9.16E+05)	**2.55E+02** **(5.53E+01)**
Ackley	1.39E+01 (2.85E–01)	1.38E+01 (5.30E–01)	8.77E+00 (8.80E–01)	**3.85E+00** **(1.66E+00)**
Griewank	1.96E+02 (2.48E+01)	2.04E+02 (1.92E+01)	5.42E+01 (1.86E+01)	**5.02E–02** **(5.16E–02)**
Schwefel 2.22	8.30E+01 (6.72E+00)	8.25E+01 (6.34E+00)	1.90E+01 (5.09E+00)	**1.24E–05** **(7.33E–06)**
Rotated hyper-ellipsoid	2.15E+05 (2.83E+04)	2.138E+05 (2.83E+04)	3.22E+05 (3.96E+04)	**3.48E+04** **(4.34E+03)**
Schwefel 2.26	–33937.364505 (5.72E+02)	–33596.899217 (7.31E+02)	–40627.345524 (3.95E+02)	**–14857.535333** **(7.54E+02)**

As mentioned, our test functions had a good balance between uni-modal and multi-modal functions. When the number of decision variables increases, the performance of all the methods decrease. However, the performance of the proposed method still remains the best.

To compare the convergence rate of the proposed method with HS and its earlier derivation (IHS), we have chosen two functions from Table 2. These were the Griewank and Rotated hyper-ellipsoid. In this section the maximum number of iteration is limited to 20000 and also the dimension number is taken as 100. The parameters of all methods are as in Table 1 and the values of ξ are similar to those in Table 3.

For these functions the minimum solution is $x^* = [0,0,...,0]$ with a function value equal to $f(x^*) = 0.0$. As shown in Fig. 2, the proposed method has a high convergence rate compared to the harmony search and improved harmony search algorithms. Additionally, the values of the final solutions in our method are closer to minimum solutions than the other two algorithms. These advantages are due to a good balance between diversification and intensification in the memory consideration and pitch adjustment sections. As shown, the present algorithm creates a high difference between its convergence trajectory with other algorithms even in earlier iterations. Additionally, as shown in Fig. 2, in high dimension problems IHS cannot beat the HS algorithm.

b) Constrained engineering optimization examples

In order to show the robustness and effectiveness of the proposed method, several constrained engineering examples are selected from the optimization literature. These test examples have been solved previously using a variety of other optimization algorithms.

Example 1: *A welded beam design*

The welded beam structure shown in Fig. 3 is a common practical design problem that is widely used for testing different optimization methods [14–18]. This structure consists of a beam A that is welded to member B. The goal is to find a feasible set of decision variables $h(=x_1)$, $l(=x_2)$, $t(=x_3)$, and $b(=x_4)$ to sustain a certain load P, while minimizing the fabrication cost of the structure. The structure is subjected to constraints on shear stress (τ), bending stress (σ), buckling load (P_c), end deflection (δ),

and side constraint. The mathematical formulation of the objective function $f(\vec{x})$, and its constraints are defined as follows:

Minimize $f(\vec{x}) = 1.10471x_1^2 x_2 + 0.04811x_3x_4(14.0 + x_2)$

Subject to
$$g_1(\vec{x}) = \tau(\vec{x}) - \tau_{\max} \le 0,$$
$$g_2(\vec{x}) = \sigma(\vec{x}) - \sigma_{\max} \le 0,$$
$$g_3(\vec{x}) = x_1 - x_4 \le 0,$$
$$g_4(\vec{x}) = 0.10471x_1^2 + 0.04811x_3x_4(14 + x_2) - 5 \le 0,$$
$$g_5(\vec{x}) = 0.125 - x_1 \le 0,$$
$$g_6(\vec{x}) = \delta(\vec{x}) - \delta_{\max} \le 0,$$
$$g_7(\vec{x}) = p - p_c(\vec{x}) \le 0,$$
$$0.125 \le x_1 \le 5, \quad 0.1 \le x_2, x_3 \le 10, \quad 0.1 \le x_4 \le 5,$$

where

$$\tau(\vec{x}) = \sqrt{(\tau')^2 + 2\tau'\tau''\frac{x_2}{2R} + (\tau'')^2}, \quad \tau' = \frac{P}{\sqrt{2}x_1x_2}, \quad \tau'' = \frac{MR}{J},$$

$$M = P(L + \frac{x_2}{2}), \quad R = \sqrt{\frac{x_2^2}{4} + (\frac{x_1 + x_3}{2})^2},$$

$$J = 2\left\{\sqrt{2}x_1x_2\left[\frac{x_2^2}{4} + (\frac{x_1 + x_3}{2})^2\right]\right\},$$

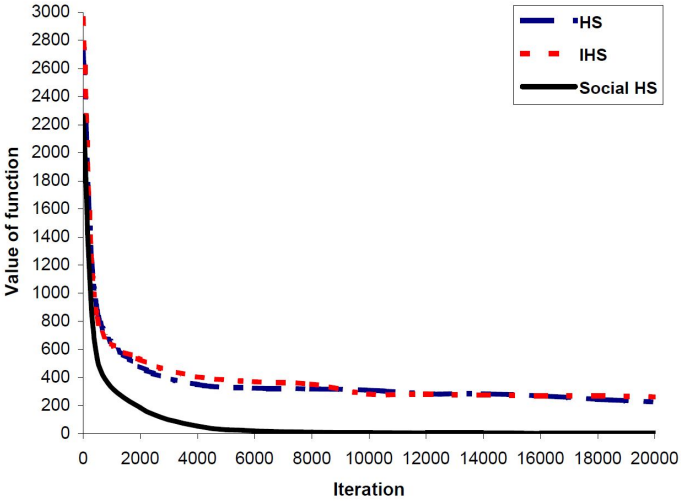
$$\sigma(\vec{x}) = \frac{6PL}{x_4x_3^2}, \quad \delta(\vec{x}) = \frac{6PL^3}{Ex_3^2x_4},$$

$$P_c(\vec{x}) = \frac{4.013E\sqrt{\frac{x_3^2x_4^6}{36}}}{L^2}(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}),$$

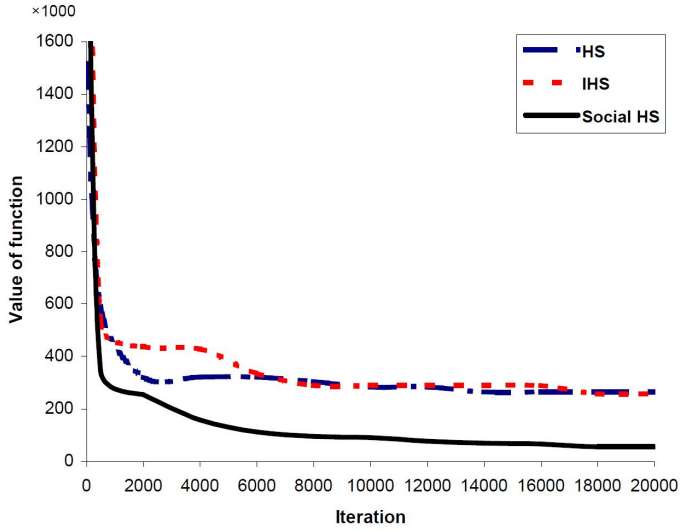
$P = 6000lb, \quad L = 14in., \quad \delta_{\max} = 0.25in., \quad E = 30 \times 10^6\ psi, \quad G = 12 \times 10^6\ psi,$
$\tau_{\max} 13{,}600\, psi, \quad \sigma_{\max} = 30{,}000\, psi.$

Deb [14, 18] and Coello [19, 20] solved this engineering problem using GA-based methods. Radgsdell and Phillips [17] compared the optimal results of different optimization methods that were mainly based on mathematical optimization algorithms such as: APPROX, DAVID, GP, SIMPLEX, and RANDOM algorithms. Lee and Geem [4] solved this problem using the HSA, Mahdavi et al. [6] solved it using IHS method, and Fesanghary et al. [21] solved it using HHS method. Also, Kaveh and Talatahari [22] using ACO solved this problem. Table 6 shows the comparison of results.

As shown in Table 6, the result of the proposed method, which is obtained after approximately 20,000 iterations (0.0625 s) without violating any constraint, is the same as that reported by Mahdavi et al. [6] and Fesanghary et al. [21]. Notice that IHS, HS and HHS algorithms, respectively need 200,000, 110,000 and 90,000 iterations to achieve this result. Additionally, HHS achieved this result in 4.138 seconds. Fig. 4 provides a comparison of the convergence rates for the three algorithms (HS, IHS and the proposed method).



(a) Griewank



(b) Rotated hyper-ellipsoid

Fig. 2. Comparison of the convergence rates between the three algorithms for Griewank and Rotated- hyper-ellipsoid functions with 100 dimensions

Table 6. Optimal results for a welded beam design (N/A means not available)

Method	Optimal design variables (x)				Cost
	h	l	t	b	
Deb [14]	N/A	N/A	N/A	N/A	2.38
Deb [18]	0.2489	6.1730	8.1789	0.2533	2.4328
Coello [19]	N/A	N/A	N/A	N/A	1.8245
Coello [20]	0.2088	3.4205	8.9975	0.2100	1.7483
Ragsdell and Phillips [17]					
APPROX	0.2444	6.2189	8.2915	0.2444	2.3815
DAVID	0.2434	6.2552	8.2915	0.2444	2.3841
GP	0.2455	6.1960	8.2730	0.2455	2.39
SIMPLEX	0.2792	5.6256	7.7512	0.2796	2.53
RANDOM	0.4575	4.7313	5.0853	0.6600	4.12
Lee and Geem [4]	0.2442	6.2231	8.2915	0.2443	2.38
Kaveh and Talatahari [22]	0.20570	3.47113	9.03668	0.20573	1.7249
Mahdavi et al. [6]	0.20573	3.47049	9.03662	0.20573	1.7248
Fesanghary et al. [21]	0.20572	3.47060	9.03682	0.20572	1.7248
Proposed method	0.20573	3.47049	9.03662	0.20573	1.7248

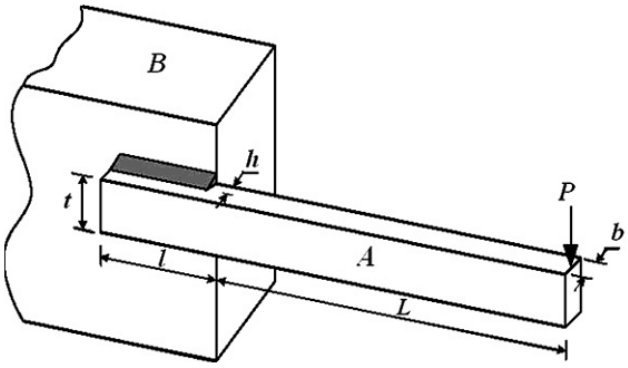


Fig. 3. A welded beam structure

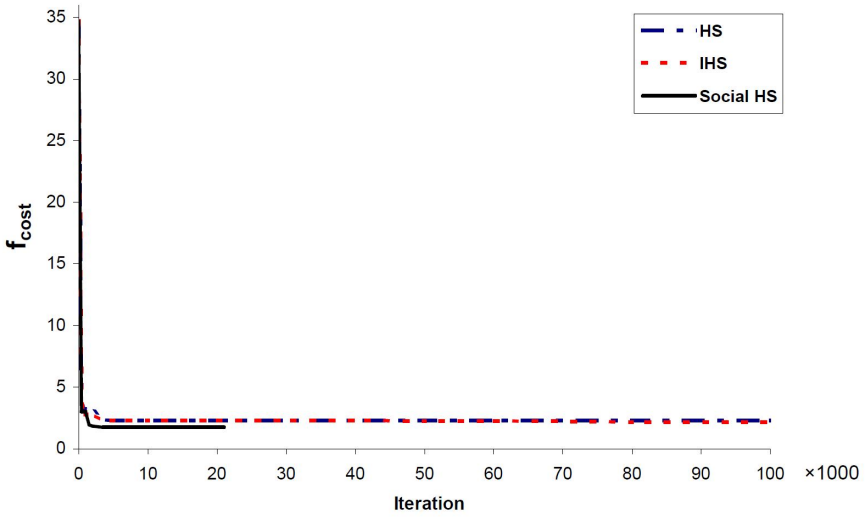


Fig. 4. Comparison of the convergence rates between the three algorithms for welded beam design

Example 2: *A pressure vessel design*

Figure 5 shows a pressure vessel problem. This engineering problem was previously analyzed by Wu and Chow [23] using GA-based, by Lee and Geem [4] using HIS, by Sandgren [24] employing branch and bound method and by Kannan and Kramer [25] using an augmented Lagrangian multiplier approach. The objective is to find a feasible set of decision variables T_s (shell thickness, x_1), T_h (head thickness, x_2), R (inner radius, x_3), and L (shell length) as shown in Fig. 5, while minimizing the total cost of the material, forming and welding. T_s and T_h are integer multiples of 0.0625 inch, the available thickness of the rolled steel plates, and R and L are continuous. The mathematical formulation of the objective function $f(\vec{x})$ and its constraints are expressed as follows:

Minimize $$f(\vec{x}) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3$$

Subject to
$$g_1(\vec{x}) = -x_1 + 0.0193x_3 \le 0,$$
$$g_2(\vec{x}) = -x_2 + 0.00954x_3 \le 0,$$
$$g_3(\vec{x}) = -\pi x_3^2 x_4 - \frac{4}{3}\pi x_3^3 + 1{,}296{,}000 \le 0,$$
$$g_4(\vec{x}) = x_4 - 240 \le 0,$$
$$g_5(\vec{x}) = 1.1 - x_1 \le 0,$$
$$g_6(\vec{x}) = 0.6 - x_2 \le 0,$$

Table 7 compares the proposed optimum result to earlier solutions reported by other optimization algorithms. The proposed method achieves \$7198.006 without violating any constraint. This result is better than all the earlier solutions. Figure 6 compares the convergence rate of the two algorithms (HS and proposed method).

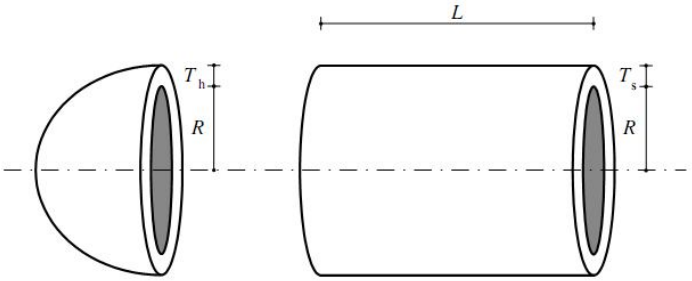


Fig. 5. Schematic presentation of the pressure vessel

Table 7. Optimal results for pressure vessel design

Method	Optimal design variables (x)				Cost
	$x_1(T_s)$	$x_2(T_h)$	$x_3(R)$	$x_4(L)$	
Wu and Chow [23]	1.125	0.625	58.1978	44.2930	7207.494
Sandgren [24]	1.125	0.625	48.97	106.72	7980.894
Kannan and Kramer [25]	1.125	0.625	58.291	43.6900	7198.043
Lee And Geem [4]	1.125	0.625	58.2789	43.7549	7198.433
Proposed method	1.125	0.625	58.29015	43.69269	7198.006

As shown in Fig. 6, the proposed method obtains a better result than HS algorithm after about 20000 iterations.

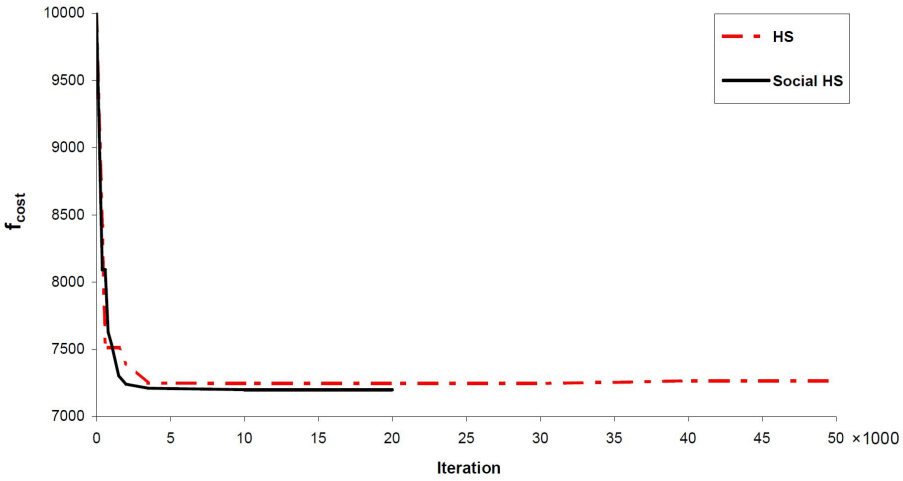


Fig. 6. Comparison of the convergence rates between the three algorithms for pressure vessel design

Example 3: *Minimization of the weight of a spring*

This problem is described by Arora [26], Coello [27] and Belegundu [28]. It consists of minimizing the weight of a tension/compression spring subjected to constraints on shear stress, surge frequency and minimum deflection as shown in Fig. 7. The design variables are the mean coil diameter D(=x_1), the wire diameter d(=x_2) and the number of active coils N(=x_3). The problem can be defined as

Minimize $f(\vec{x}) = (x_3 + 2)x_2x_1^2$

Subject to $g_1(\vec{x}) = 1 - \dfrac{x_2^3 x_3}{71785 x_1^4} \le 0,$

$$g_2(\vec{x}) = \frac{4x_2^2 - x_1 x_2}{12566(x_2 x_1^3 - x_1^4)} + \frac{1}{5108 x_1^2} - 1 \le 0,$$

$$g_3(\vec{x}) = 1 - \frac{140.45 x_1}{x_2^2 x_3} \le 0,$$

$$g_4(\vec{x}) = \frac{x_1 + x_2}{1.5} - 1 \le 0,$$

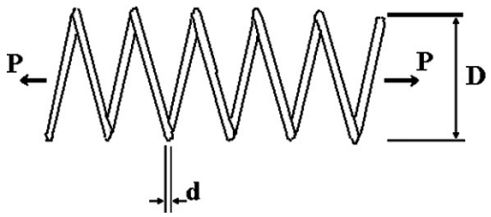


Fig. 7. A tension/compression spring

This problem has been solved by Belegundu [28] using eight different mathematical optimization techniques (only the best results are shown). Arora [26] has also solved this problem using a numerical optimization technique called constraint correction at the constant cost. Coello [20], as well as Coello and Montes [29] solved this problem using a GA-based method. Additionally, He and Wang [30] solved this problem using a Co-evolution strategy Particle Swarm Optimization (CPSO). Recently, Montes and Coello [31], Kaveh and Talatahari [22] and Mahdavi et al. [6] solved this problem, respectively using evolution strategies, an improved ant colony optimization and IHS method.

The result obtained using the proposed method is better than those reported previously in the literature. Table 8 shows a comparison of the results.

Table 8. Optimal results for minimization of the weight of the spring

Method	Optimal design variables (x)			Cost
	x_1(d)	x_2(D)	x_3(N)	
Belegundu [28]	0.050000	0.315900	14.250000	0.0128334
Arora [26]	0.053396	0.399180	9.185400	0.0127303
Coello [20]	0.051480	0.351661	11.632201	0.0127048
Coello and Montes [29]	0.051989	0.363965	10.890522	0.0126810
He and Wang [30]	0.051728	0.357644	11.244543	0.0126747
Montes and Coello [31]	0.051643	0.355360	11.397926	0.012698
Mahdavi et al. [8]	0.051154	0.349871	12.076432	0.0126706
Kaveh and Talatahari [22]	0.051865	0.361500	11.000000	0.0126432
Proposed method	0.051750	0.358689	11.156588	0.0126382

It is obvious from Table 8 that the proposed method achieves a better result than those reported previously in the literature. This result is obtained after 3000 iterations. Fig. 8 compares the convergence rate of the two algorithms (IHS and proposed method). The convergence rate of the proposed method is better than IHS algorithm.

Table 9 shows the parameters of the proposed method that are used for all engineering examples.

Recently, adaptive HS has also been developed, and does not require the selection of HMCR and PAR by the user [32, 33].

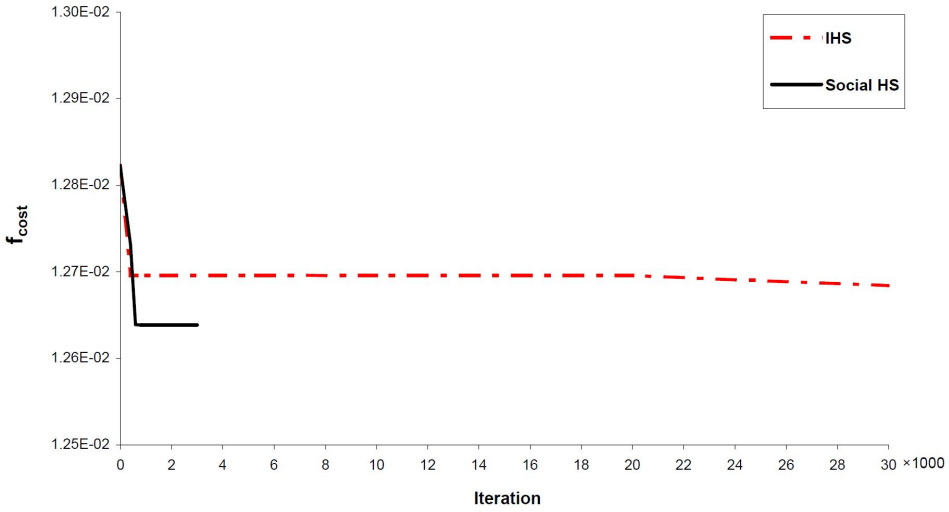


Fig. 8. Comparison of the convergence rates between the three algorithms for minimization of the weight of spring

Table 9. Parameters of the proposed method used for the engineering examples

Method	HMS	HMCR	PAR	PAR_{min}	PAR_{max}	bw	bw_{min}	bw_{max}	ξ
Social HS	15	0.99	1	–	–	–	–	–	3.0

Applications of other meta-heuristics can be found in the work of refs. [34-36].

c) Effect of the parameter ξ

Here we used ξ to adjust σ. This is an important parameter in the pitch adjustment section. If we use a big value for ξ, normal distribution will perform a similar global search in each step of the search and a desirable solution will not be obtained in a small number of iterations. On the contrary, if ξ is selected too small, the algorithm searches a fraction of feasible space, causing a premature convergence and preventing the algorithm from finding the global optimum. In order to make a good balance between diversification and intensification in pitch adjustment section, a fixed value of ξ between "0.5 to 3" is used.

5. CONCLUDING REMARKS

In the proposed method a procedure is introduced to obtain a good balance between diversification and intensification in the pitch adjustment section of HS. To achieve this goal a normal distribution rule is employed. Since variance of normal distribution for each decision variable is related to diversity of their previous value sorted in the harmony memory, their diversities are high in early iterations and gradually decrease when the algorithm reaches its final iterations. Hence, it makes global and local search in early and final iterations, respectively.

This is because of diversity and the uniformity of values of decision variables in the harmony memory, respectively. Also, in this method the sociality of the HS algorithm is increased by selecting variable form harmony memory adjusted according to the previous experiments of all the harmonies. Additionally, a fly-back mechanism is utilized to ensure that the optimum results are in the feasible space. Finally, the numerical examples reveal the superiority of the proposed method in quick convergence and find better solutions compared to the classic HS and its variants.

Acknowledgement - The first author is grateful to Iran National Science Foundation for the support.

REFERENCES

1. Geem, Z. W. (2009). Theory and applications: Music-inspired harmony search algorithm. Springer, Berlin, Germany.
2. Blum, C. & Roli, A. (2003). Metaheuristics in combinatorial optimization: Overview and conceptual comparison. *ACM Computing Surveys*, Vol. 35, pp. 268–308.
3. Geem, Z. W., Kim, J. H. & Loganathan, G. V. (2001). A new heuristic optimization algorithm: Harmony search. *Simulation*, Vol. 76, No. 2, pp. 60–68.
4. Lee, K. S. & Geem, Z. W. (2005). A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice. *Computer Methods in Applied Mechanics and Engineering*, Vol. 194, pp. 3902–3933.
5. Harmony Search Algorithm (2007). (accessed December 7, 2008), http://www.hydroteq.com
6. Mahdavi, M., Fesanghary, M. & Damangir, E. (2007). An improved harmony search algorithm for solving optimization problems. *Applied Mathematics and Computation*, Vol. 188, pp. 1567–1579.
7. Geem, Z. W. (2008). Novel derivative of harmony search algorithm for discrete design variables. *Applied Mathematics and Computation*, Vol. 199, No. 1, pp. 223-230.
8. Das, S., Mukhopadhyay, A., Roy, A., Abraham, A. & Panigrahi, B. K. (2010). Exploratory power of the harmony search algorithm: Analysis and Improvements for Global Numerical Optimization. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, Vol. 99, pp. 1–18.
9. Wang, C-M. & Huang, Y-F. (2010). Self-adaptive harmony search algorithm for optimization. *Expert Systems with Applications*, Vol. 37, No. 4, pp. 2826-2837.
10. Omran, M. G. & Mahdavi, M. (2008). Global-best harmony search. *Applied Mathematics and Computation*, Vol. 198, No. 2, pp. 643–656.
11. Eberhart, R. & Kennedy, J. (1995). A new optimizer using particle swarm theory. *Proceedings of the Sixth International Symposium on Micromachine and Human Science*, pp. 39–43.
12. Kennedy, J. & Eberhart, R. C. (1995). Particle swarm optimization. *Proceedings of the IEEE International Joint Conference on Neural Networks*, IEEE Press, pp. 1942–1948.
13. He, S., Prempain, E. & Wu, Q. H. (2004). An improved particle swarm optimizer for mechanical design optimization problem. *Engineering Optimization*, Vol. 36 No. 5, pp. 585-605.
14. Deb, K. (2000). An efficient constraint handling method for genetic algorithms. *Computer Methods in Applied Mechanics and Engineering*, Vol. 186, pp. 311–338.
15. Reklaitis, G. V., Ravindran, A. & Ragsdell, K. M. (1983). *Engineering optimization methods and applications*. Wiley, New York, USA.
16. Siddall, J. N. (1972). *Analytical decision-making in engineering design*. Prentice-Hall, Engle Wood Cliffs, NJ.
17. Ragsdell, K. M. & Phillips, D. T. (1976). Optimal design of a class of welded structures using geometric programming. *Journal of Engineering for Industry ASME, Series B*, Vol. 98, No. 3, pp. 1021–1025.
18. Deb, K. (1991). Optimal design of a welded beam via genetic algorithms. *Journal of AIAA*, Vol. 29, No. 11, pp. 2013-2015.
19. Coello, C. A. C. (2000). Constraint-handling using an evolutionary multiobjective optimization technique, *Civil Engineering Environmental Systems*, Vol. 17, pp. 319–346.
20. Coello, C. A. C. (2000). Use of a self-adaptive penalty approach for engineering optimization problems. *Industrial Computations*, Vol. 41, No. 2, pp. 113–127.
21. Fesanghary, M., Mahdavi, M., Minary-Jolandan, M. & Alizadeh, Y. (2008) Hybridizing harmony search algorithm with sequential quadratic programming for engineering optimization problems. *Computer Methods in Applied Mechanics and Engineering*, Vol. 197, pp. 3080–3091.

22. Kaveh, A. & Talatahari, S. (2010). An improved ant colony optimization for constrained engineering design problems, *Engineering Computations*, Vol. 27, No. 1, pp. 155–182.
23. Wu, S. J. & Chow, P. T. (1995). Genetic algorithms for nonlinear mixed discrete integer optimization problems via meta-genetic parameter optimization. *Engineering Optimization*, Vol. 24, pp. 137–159.
24. Sandgren, E. (1994). Nonlinear integrand discrete programming in mechanical design optimization. *Journal of Mechanical Design ASME*, Vol. 112, pp. 223–229.
25. Kannan, B. K. & Kramer, S. N. (1994). An augmented Lagrange multiplier based method for mixed integer discrete continuous optimization and its applications to mechanical design. *Journal of Mechanical Design ASME*, Vol. 116, pp. 318–320.
26. Arora, J. S. (1989). *Introduction to optimum design*. McGraw-Hill, New York.
27. Coello, C. A. C. (2002). Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art. *Computer Methods in Applied Mechanics and Engineering*, Vol. 191, pp. 1245–1287.
28. Belegundu, A. D. (1982). A study of mathematical programming methods for structural optimization. Ph.D. thesis, Department of Civil and Environmental Engineering, University of Iowa, USA.
29. Coello, C. A. C. & Montes, E. M. (2002). Constraint-handling in genetic algorithms through the use of dominance-based tournament selection. *Advanced Engineering Informatics*, Vol. 16, pp. 193–203.
30. He, Q. & Wang, L. (2007). An effective co-evolutionary particle swarm optimization for constrained engineering design problems. *Engineering Applications of Artificial Intelligence*, Vol. 20, pp. 89–99.
31. Montes, E. M. & Coello, C. A. C. (2008). An empirical study about the usefulness of evolution strategies to solve constrained optimization problems. *International Journal of General Systems*, Vol. 37, No. 4, pp. 443–473.
32. Geem, Z. W. & Sim, K. B. (2010). Parameter-Setting-Free Harmony Search Algorithm, *Applied Mathematics and Computation*, Vol. 217, No. 8, pp. 3881-3889.
33. Hasancebi, O., Erdal, F. & Saka, M. P. (2010). Adaptive harmony search method for structural optimization. *Journal of Structural Engineering ASCE*, Vol. 136, No. 4, pp. 419-431.
34. Kaveh, A. & Shakouri, A. (2010). Harmony search algorithm for optimum design of slab formwork. *Iranian Journal of Science and Technology, Transaction B: Engineering*, Vol. **34**, No. B4, pp. 335-351.
35. Kaveh, A. & Malakouti Rad, S. (2010). Hybrid genetic algorithm and particle swarm optimization for the force method-based simultaneous analysis and design. *Iranian Journal of Science and Technology, Transaction B: Engineering*, Vol. 34, pp. 15-34.
36. Kaveh, A. & Masoudi, M. S. (2011). Cost optimization of a composite floor system using ant colony system, *Iranian Journal of Science and Technology, Transaction B: Engineering*, to appear.