A Harmony Search Algorithm Comparison with Genetic Algorithms

Cinthia Peraza, Fevrier Valdez and Oscar Castillo

Abstract We describe in this paper a Harmony Search (HS) Algorithm and their areas of application, variants and comparison with other existing algorithms. HS is a metaheuristic music inspired algorithm used to solve a wide range of optimization problems applied to different areas, which has been very successful as indicated by the literature. A comparison with genetic algorithms was performed to evaluate the advantages of HS.

Keywords Harmony search · Optimization problems · Mathematical functions · Genetic algorithms

1 Introduction

We describe in this paper a harmony search algorithm which is metaheuristic algorithm inspired by music. In particular we refer to the improvisation of jazz version of Hs and its comparison with the genetic algorithm. These algorithms were applied to benchmark mathematical functions and comparative tables were made showing the optimization of results between Genetic Algorithm and Harmony Search algorithm.

The comparative study of the two algorithms is performed in order to show the effectiveness of harmony search algorithm versus optimization problem, in the same manner proving that it is more effective than the genetic algorithm.

The paper is organized as follows: in this Sect. 2 a description about Harmony Search Algorithm is presented, in this Sect. 3 a description of Genetic Algorithm is shown, in Sect. 4 description the mathematical functions is presented, in Sect. 5 a description about the optimization problems is shown, in Sect. 6 the simulations results are described and we can appreciate a comparison between Harmony Search

C. Peraza · F. Valdez (✉) · O. Castillo
Tijuana Institute of Technology, Tijuana, BC, Mexico
e-mail: fevrier@tectijuana.mx

O. Castillo and P. Melin (eds.), *Fuzzy Logic Augmentation of Nature-Inspired Optimization Metaheuristics*, Studies in Computational Intelligence 574,
DOI 10.1007/978-3-319-10960-2_7

algorithm and genetic algorithms, and in Sect. 7 the conclusions obtained after the study of the two algorithms versus mathematical functions is presented.

In the literature there are works where the Harmony Search has been used. In [4] a new Meta heuristic algorithm for continuous engineering optimization Theory and practice is presented. In this paper the authors propose a new harmony search(HS) meta heuristic algorithm based approach for engineering optimization problems with continuous design variables it uses a stochastic random search instead of a gradient search so that derivative information is unnecessary various engineering optimization problems, including mathematical function minimization and structural engineering optimization problems, are presented to demonstrate the effectiveness and robustness of the HS algorithm. The results indicate that the proposed approach is a powerful search and optimization technique that may yield better solutions to engineering problems than those obtained using current algorithms. In [5] the Parameter setting free harmony search algorithm is presented, the authors proposed this study a novel technique to eliminate tedious and experience requiring parameter assigning efforts. The new parameter setting free (PSF) technique which this study suggests contains one additional matrix which contains an operation type (random selection, memory consideration, or pitch adjustment) for every variable in harmony memory. In [15] the Harmony Search Benchmarking of heuristic optimization methods is presented, the authors propose it is short history when many heuristic optimization methods appear. As example Particle swarm optimization method (PSO) or Repulsive particle swarm optimization method (RPSO), Gravitational search algorithm (GSA), Central force optimization (CFO), Harmony search algorithm (HAS) etc. Those methods are working differently but all of them can optimize same problems. There is general question: Exists any standard benchmark which can be used for individual methods comparing. It is a bit hard to answer this question because it is possible to find some optimization problems which are widely used along some papers but in fact there does not exists summary which can be uses for standard evaluation of optimization process. In [9] the Global Best Harmony Search is presented, the authors propose a new variant of HS concepts from swarm intelligence are borrowed to enhance the performance of HS. The performance of the GHS is investigated and compared with HS and a recently developed variation of HS. The experiments performed show that the GHS generally outperformed the other approach when applied to ten benchmark problems. The effect of noise on the performance of the three HS variants is investigated and a scalability study is conducted. The effect of the GHS parameters is analyzed. Finally, the three HS variants are compared on several Integer Programming test problem. The results show that the three approaches seem to be an efficient alternative for solving Integer Programming problem. In [16] the Self adaptive: harmony search algorithm for optimization is presented, the authors proposed a new metaheuristic optimization algorithm harmony search (HS) with continuous design variables was developed. This algorithm is conceptualized using the musical improvisation process of searching for a perfect state of harmony. Although several variants and an increasing number of applications have appeared, one of its main difficulties is how to select suitable parameter values. In [10] the An improved harmony search algorithm for solving optimization problems is presented, in this paper the authors propose develops an improved harmony search (IHS) algorithm for solving optimization problems. IHS employs a novel method for generating new solution vectors that enhances accuracy and convergence rate of harmony search (HS) algorithm, in [11] the a survey on applications of the harmony search algorithm, in this paper they propose thoroughly reviews and analyses the main characteristics and application portfolio of the so called Harmony Search algorithm a meta heuristic approach that has been shown to achieve excellent results in a wide range of optimization problems. In [8] the A Tabu Harmony Search Based Approach to Fuzzy Linear Regression is presented, the authors propose an unconstrained global continuous optimization method based on tabu search and harmony search to support the design of fuzzy linear regression (FLR) models. Tabu and harmony search strategies are used for diversification and intensification of FLR, respectively. The authors propose approach offers the flexibility to use any kind of an objective function based on client's requirements or requests and the nature of the data set and then attains its minimum error. In [14] the A new gravitational search algorithm using fuzzy logic to parameter adaptation the authors propose a new method using fuzzy logic to change alpha parameter and give a different gravitation and acceleration to each agent in order to improve its performance, we use this new approach for mathematical functions and present a comparison with original approach. In [3] the Fuzzy Control of Parameters to Dynamically Adapt the PSO and GA Algorithms the authors propose a new hybrid approach for mathematical function optimization combining Particle Swarm Optimization (PSO) and Genetic Algorithms (GAs) using Fuzzy Logic for parameter adaptation and integrate the results. In [7] the Music Inspired Harmony Search Algorithm Theory and practice, the authors propose we show the performance of the algorithm and the areas in which it can be applied. In [6] the Harmony Search Algorithms for structural design optimization the authors propose a show us the type of problems you can solve the harmony search algorithm and some methods that have been proposed to improve in certain areas of application. In [12] the Differential evolution with dynamic adaptation of parameters for the optimization of fuzzy controllers is presented, the authors propose a new algorithm using fuzzy logic with dynamic adaptation of parameters.

2 Harmony Search Algorithm

Harmony search is a relatively new heuristic optimization algorithm inspired music and was first developed by ZW Gemm et al. in 2001 [7].

This algorithm can be explained more in detail with the process of improvisation that takes a musician, which consists of three options:

1. Play any song you have in your memory
2. Play a similar composition to an existing
3. Play a new song or randomly

If we formalize these three options for optimization, we have three corresponding components: memory usage of harmony, pitch adjustment and randomization [17].

2.1 Memory in Harmony Search Algorithm

The use of harmony memory is important because it is similar to choosing the best people in genetic algorithms. This will ensure the best harmonies will be transferred to the new memory harmony. In order to use this memory more effectively, we can assign a parameter r_{accept} € [0, 1] call acceptance rate memory. If this rate is too low, just select the best harmonies and may converge very slowly [17].

$$r_{accept} \in [0, 1] \tag{1}$$

2.2 Pitch Adjustment

To adjust the pitch slightly in the second component, we have to use such a method can adjust the frequency efficiently. In theory, the tone can be adjusted linearly or nonlinearly, but in practice the linear is used. If the current solution is X_{old} (or pitch), then the new solution (tone) is generated X_{new}.

$$x_{new} = x_{old} + b_p(2rand - 1) \tag{2}$$

where "*rand*" is a random number drawn from a uniform distribution [0, 1]. Here is it bandwidth, which controls the local range of tone adjustment in fact, we can see that the pitch adjustment (2) is a random step.

Pitch setting is similar to the mutation operator in genetic algorithms. We can assign a pitch adjustment rate to control the degree of adjustment. If too low, there is usually no change. If too high, then the algorithm may not converge at all [17].

2.3 Randomization

The third component is a randomization component (3) that is used to increase the diversity of the solutions. Although the tone setting has a similar role, but it is limited to certain local tone adjustment and therefore correspond to a local search. The use of randomization can further push the system to explore various regions with high diversity solution in order to find the global optimum [17]. So we have:

$$\mathrm{P_a} = P_{lower\,limit} + P_{range} * \ rand \tag{3}$$

where rand is a generator of random numbers in the range of 0 and 1. (Search space) $P_{range} = P_{upper\,limit} - P_{lower\,limit}$

The three components in harmony search can be summarized in the pseudo code shown in Sect. 2.4, where you can find that the probability of a true randomization (4) is

$$P_{random} = 1 - r_{accept} \tag{4}$$

And the actual probability of tone adjustment (5) is

$$P_{tono} = r_{accept} * r_{pa} \tag{5}$$

2.4 Pseudo Code for Harmony Search Algorithm

The pseudo code for HS is presented below:

```
Objective function $f(x), x = (x_1, ....,x_n)^T$
Initial generate harmonics (matrices of real numbers)
Define pitch adjustment rate (rpa) and limits of tone
Define acceptance rate of the harmony memory (r accept)
while (t <Maximum number of iterations)
Generate a new harmony and accept the best harmonies
Setting the tone for new harmonies (solutions)
if (rand>raccept)
Choose an existing harmony randomly
else if (rand>rpa)
Setting the tone at random within a bandwidth (2)
else
Generate a new harmony through a randomization (3)
End if
Accepting new harmonies (solutions) best
End while
To find the best solutions.
```

2.5 Variants

There are three variants of the algorithm that have been applied to achieve better results briefly explain each of them:

The improved harmony search algorithm (IHS)
To address the shortcomings of the HS, Mahdavi et al. [10] proposed a new variant of the HS, called the improved harmony search (IHS). The IHS dynamically updates (r_{pa}) according to the following equation,

$$Rpa(t) = Rpa_{\min} + \frac{(Rpa_{\max} - Rpa_{\min})}{NI} + t \tag{6}$$

where *Rpa(t)* is the pitch adjusting rate for generation *t*, $PAR_{\min}$ is the minimum adjusting rate, $PAR_{\max}$ is the maximum adjusting rate and t is the generation number.

In addition, *bp* is dynamically updated as follows:

$$bp(t) = bp_{\max^{e}}\left(\frac{\ln(\frac{bp_{\min)}}{bp_{\max}})}{NI}\right) * t \tag{7}$$

where *bp(t)* is the bandwidth for generation *t*, $bp_{\min}$ is the minimum bandwidth and $bp_{\max}$ is the maximum bandwidth.

A major drawback of the IHS is that the user needs to specify the values for $bw_{\min}$ and $bp_{\max}$ which are difficult to guess and problem dependent.

Best overall harmony search (GHS)
Inspired by the concept of swarm intelligence as proposed in Particle Swarm Optimization (PSO) [2], a new variation of HS is proposed in this paper. In a global best PSO system, a swarm of individuals (called particles) fly through the search space. Each particle represents a candidate solution to the optimization problem.

The position of a particle is influenced by the best position visited by itself (i.e. its own experience) and the position of the best particle in the swarm (i.e. the experience of swarm).

The new approach, called global-best harmony search (GHS), modifies the pitch-adjustment step of the HS such that the new harmony can mimic the best harmony in the HM. Thus, replacing the bp parameter altogether and adding a social dimension to the HS. Intuitively, this modification allows the GHS to work efficiently on both continuous and discrete problems.

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

```
for each i € [1,N] do
If U(0,1) ≤ $R_{accept}$ then /*memory consideration */
Begin
$X_i^j = x_i^j$, where j ~ U (1,…,HMS)
If U(0,1) ≤$Rpa$(t) then /* pitch adjustment */
Begin
$x_i^j = x_k^{best}$, where best is the index of the best harmony in the HM and k ~ U (1,N)
Endif
Else /* random selection */
$x_i^j = LB_i + r * (UB_i - LB_i)$
Endif
done
```

New global harmony search (NGHS)

The NGHS algorithm is an improved version of harmony search algorithm (HS), and it includes two important operations: position updating and genetic mutation with a low probability. The former can enhance the convergence of the NGHS, and the latter can effectively prevent the NGHS from trapping into the local optimum. Based on a large number of experiments, the NGHS has demonstrated high efficiency on solving chemical equation balancing. The results show that the NGHS can be an efficient alternative for solving chemical equation balancing [1].

2.6 Application Areas

The harmony search algorithm has been applied so far to various optimization problems. Moreover, the structure of the algorithm has been customized by case adjust basic structure. To overcome this situation, the algorithm of harmony search (HS) used a new stochastic derivative, using the experiences of musicians in jazz improvisation and may be applicable to discrete variables. Instead of tilting the information of an objective function, the stochastic derivative HS gives a probability of being selected for each value of a decision variable.

The HS algorithm has been applied to various problems in science and engineering optimization including:

Optimization function, the distribution of water, groundwater modeling, energy saving clearance, structural design, vehicle routing, and others. The possibility of combining harmony search with other algorithms such as particle swarm optimization and genetic algorithms has also been investigated [17].

3 Genetic Algorithms

Genetic algorithms (GA) emulate genetic evolution. The characteristics of individuals are therefore expressed using genotypes. The original form of the GA, as illustrated by John Holland in 1975, had the distinct features: (1) a bit string representation, (2) proportional selection, and (3) cross-over as the primary method to produce new individuals. Since then, several variations to the original Holland GA have been developed, using different representation schemes, selection, crossover, mutation and elitism operators [2].

3.1 Representation

The classical representation scheme for GAs is of binary vectors of fixed length. In the case of an n_x-dimensional search space, each individual consists on n_x variables with each variable encoded as a bit string. If variables have binary values, the length of each chromosome is n_x bits. In the case of nominal-valued variables, each nominal value can be encoded as an n_d-dimensional bit vectors where 2nd is the total numbers of discrete nominal values for that variable. Each n_d-bit string represents a different nominal value. In the case of continuous-valued variables, each variable should be mapped to an n_d-dimensional bit vector,

$$\varphi : R \rightarrow (0,1)^{n_d} \tag{8}$$

The range of continuous space needs to be restricted to a finite range, $[\boldsymbol{x}_{\mathbf{min}}, \boldsymbol{x}_{\mathbf{max}}]$. Using the standard binary decoding, each continuous variable $\boldsymbol{x}_{ij}$ of chromosome $\boldsymbol{x}_i$ is encoded using a fixed length bit string.

GAs have also been developed that use integer or real valued representations and order based representations where the order of variables in a chromosome plays an important role. Also, it is not necessary that chromosomes be of fixed length [2].

3.2 Crossover Operations

Several crossover operators have been developed for GA's depending on the format in which individuals are represented. For binary representations, uniform crossover, one-point crossover and two-point crossover are the most popular:

- **Uniform Crossover**, where corresponding bit positions are randomly exchanged between the two parents to produce two offspring.
- **One-Point Crossover**, where a random bit position is selected, and the bit substrings after the selected bit are swapped between the two parents to produce two offspring.

- **Two-Point Crossover**, where two bit positions are randomly selected and the bit substrings between the selected bit positions are swapped between the two parents.

For continuous valued genes, arithmetic crossover can be used:

$$x_{ij} = r_j x_{1j} + (1.0 - r_j) x_{2j} \tag{9}$$

where $r_j \sim \mathrm{U}(0, 1)$ and x_i is the offspring produced from parents x_1 and x_2 [2].

3.3 Mutation

The mutation scheme used in a GA depends on the representation scheme. In the case of bit string representations, we here:

- **Random Mutation**, randomly negates bits, while
- **In-Order Mutation**, performs random mutation between two randomly selected bit positions.

For discrete genes with more than two possible values that a gene can.

Assume, random mutation selects a random value from the finite domain of the gene. In the case of continuous valued genes, a random value sampled from a Gaussian distribution with zero mean and small deviation is usually added to the current gene value. As an alternative, random noise can be sampled from a Cauchy distribution [2].

4 Benchmark Mathematical Functions

This section list a number of the classical benchmark functions used to validate optimization algorithms.

In the area of optimization using mathematical functions have been considered in the works mentioned below: A new gravitational search algorithm using fuzzy logic to parameter adaptation [14], Differential evolution with dynamic adaptation of parameters for the optimization of fuzzy controllers [12], Bat algorithm comparison with genetic algorithm using benchmark functions [13].

To validate our method we used a set of 6 benchmark mathematical functions, called Spherical, Rosenbrock, Rastrigin, Ackley, Zakharov, Sum Square; all functions were evaluated with 4, 5, 10, 20, 30 and 40 Harmonies.

Figure 1 shows the plot corresponding to the Spherical function and Eq. 10 represents the Spherical function. Figure 2 shows the plot corresponding to Rosenbrock function and Eq. 11 represents the Rosenbrock function.

Fig. 1 Spherical function

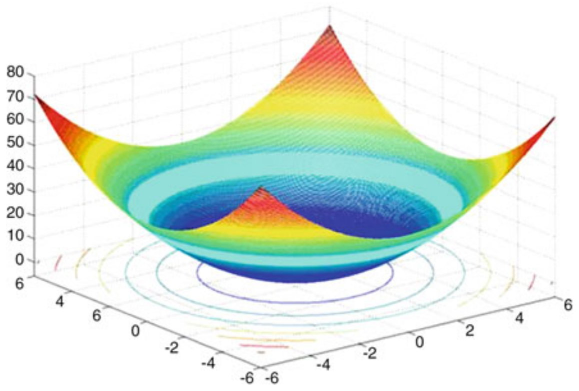
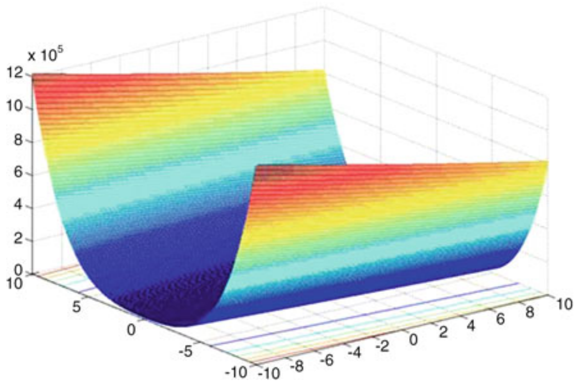


Fig. 2 Rosenbrock function



The mathematical functions are shown below:

$$f(x) = \sum_{j=1}^{n_x} x_j^2 \tag{10}$$

Witch $x_j \in [-100, 100]$ and $f^*(x) = 0.0$

$$f(x) = \sum_{j=1}^{n_z/2} [100(x_{2j} - x_{2j-1}^2)^2 + (1 - x_{2j-1})^2] \tag{11}$$

Witch $x_j \in [-2.048, 2.048]$ and $f^*(x) = 0.0$

Figure 3 shows the plot corresponding to Rastrigin function and Eq. 12 shows the description the Rastrigin function.

$$f(x) = \sum_{j=1}^{n_x} (x_j^2 - 10\cos(2\pi x_j) + 10) \tag{12}$$

With $x_j \in [-5.12, 5.12]$ and $f^*(x) = 0.0$

Fig. 3 Rastrigin function

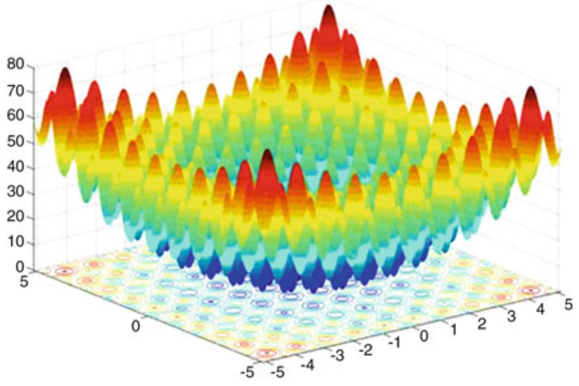


Fig. 4 Ackley function

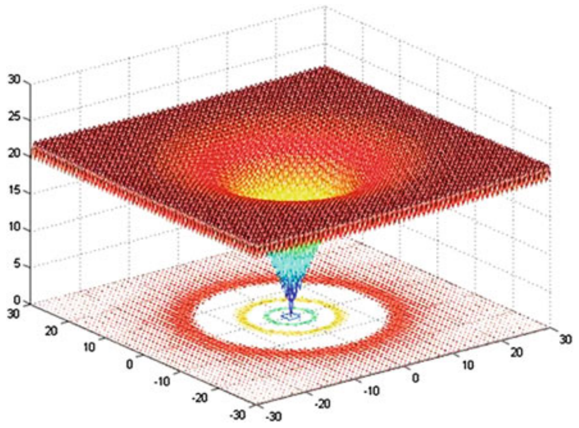


Figure 4 shows the plot corresponding to Ackley function and Eq. 13 shows the description the Ackley function.

$$f(x) = -20e^{-0.2\sqrt{\frac{1}{n_x}\sum_{j=1}^{n_x} x_j^2 - \frac{1}{e^{n_x}}\sum_{j=1}^{n_x}\cos(2\pi x_j)}} + 20 + e \tag{13}$$

With $x_j \in [-30, 30]$ and $f^*(x) = 0.0$

Figure 5 shows the plot corresponding to Zakharov function and Eq. 14 shows the description the Zakharov function.

$$f(x) = \sum_{i=1}^{n} x_i^2 + (\sum_{i=1}^{n} 0.5ix_i)^2 + (\sum_{i=1}^{n} 0.5ix_i)^4 \tag{14}$$

Witch $x_i \in [-5, 10]$ and $f^*(x) = 0.0$

Fig. 5 Zakharov function

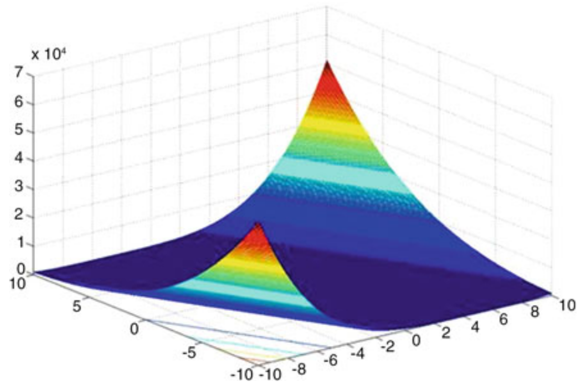


Fig. 6 Sum square function

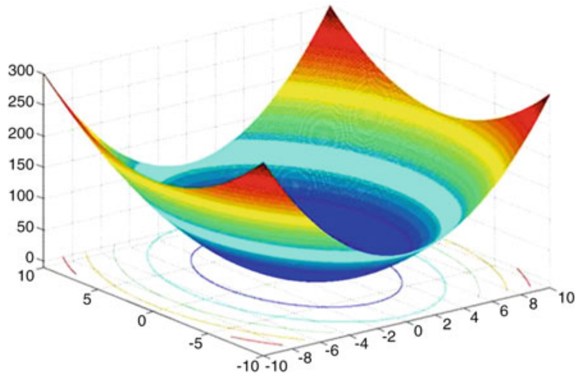


Figure 6 shows the plot corresponding to Sum Square function and Eq. 15 shows the description the Sum Square function.

$$f(x) = \sum_{i=1}^{n} i x_i^2 \tag{15}$$

Witch $x_i \in [-2, 2]$ and $f^*(x) = 0.0$

5 Optimization Problems

The optimization problem can be defined as:

$$\text{Min } y = f(x) \tag{16}$$

Basically the problems can be divided into unimodal and multimodal.

Another aspect are constrains. Some problems have not constrains. The best optimization method has to find optimal value in all cases.

There exist many algorithms for problem solving. Some of them are special, some are more general. Many problems cannot be solved by deterministic algorithm, so heuristic algorithm are used. In the set of metaheuristic algorithms we can find PSO [3], GSA [14], DE [12] and others.

Unimodal functions

Have only one local optimum. Those functions are relatively easy to analyze for optimums. They are used for checking the speed of optimization and convergence. There are used commonly two functions. First one is Schefel's and second one is Rastrigin's.

Multimodal functions

Multimodal functions have multiple local optimums. Some methods will stuck in local optimum. Main goal of those problems is to test solvers how they are able to avoid local optimum. Some problems have no single global optimum. Some of them have one global optimum and many local which are very close in the term of fitness function. First one is Sphere, second one is Sum and Product and third one is Griewank's.

6 Simulation Results

In this section the comparison of the Harmony Search algorithm is made against genetic algorithms [13]. In each of the algorithms 6 mathematical functions Benchmark were considered separately, a dimension of 10 variables was used with 30 runs for each function varying the parameters of the algorithms.

The parameters used in the HS were:

- Size solution harmonies: 4–40 Harmonies.
- Harmony memory accepting: 0.75–0.95.
- Pitch adjustment: 0.1–0.5.
- Pitch range: 200–400.

The parameters for the genetic algorithm are shown below [13]:

- Number of Individuals: 4–40.
- Selection: Stochastic, Remainder, Uniform, Roulette.
- Crossover: Scattered, Single Point, Two Point, Heuristic and Arithmetic.
- Mutation: Gaussian, Uniform.

6.1 Simulation Results with Harmony Search Algorithm

In this section we show the experimental results obtained by the Harmony Search algorithm in separate tables of the mathematical functions. Table 1 shows the simulation results for the Spherical function.

From Table 1 it can be appreciated that after executing the HS Algorithm 30 runs, with different parameters, we can find the best, average and worst results for the Spherical function. Table 2 shows the simulation results for the Rosenbrock function.

From Table 2 it can be appreciated that after executing the HS Algorithm 30 runs, with different parameters, we can find the best, average and worst results for the Rosenbrock function. Table 3 shows the simulation results for the Rastrigin function.

Table 1 Simulation results for the spherical function

Number of harmonies	Best	Worst	Mean
4	0.000038975	0.00014863	0.0000762868
5	0.000039501	0.00014258	0.0000792732
10	0.000053865	0.00012135	0.0000883132
20	0.000029966	0.000079598	0.0000476648
30	0.000024495	0.000080963	0.0000499808
40	0.000023567	0.000077813	0.0000523875

Table 2 Simulation results for the Rosenbrock function

Number of harmonies	Best	Worst	Mean
4	0.0000000010631	0.00000039836	0.0000000578908
5	0.00000000073035	0.00000032337	0.0000000521
10	0.000000000079568	0.00000023364	0.0000000475
20	0.000000000014716	0.001	0.0000387
30	0.0000000010014	0.001	0.00112
40	0.00000000010979	0.0079	0.00176

Table 3 Simulation results for the Rastrigin function

Number of harmonies	Best	Worst	Mean
4	0.000000000011045	0.00000007236	0.0000000139
5	0.0000000000093081	0.000000011282	0.0000000126
10	0.00000000061454	0.00000022195	0.0000000324
20	0.000000000032507	0.000000092489	0.0000000172
30	0.00000000006964	0.00000022386	0.0000000321
40	0.00000000037211	0.00000017014	0.0000000275

Table 4 Simulation results for the Ackley function

Number of harmonies	Best	Worst	Mean
4	0.0000098012	0.00031775	0.000122722
5	0.0000087238	0.00017561	0.0000969749
10	0.000022005	0.00035406	0.000139821
20	0.0000093788	0.00028147	0.0000940209
30	0.000016206	0.00040138	0.000136185
40	0.000031467	0.00045383	0.000116993

Table 5 Simulation results for the Zakharov function

Number of Harmonies	Best	Worst	Mean
4	0.000095054	0.00032929	0.00019524
5	0.00007544	0.00033906	0.00022226
10	0.000095173	0.00040378	0.00027224
20	0.000073328	0.00037254	0.00019199
30	0.000077163	0.00036816	0.00022529
40	0.00019508	0.0018	0.00052135

From Table 3 it can be appreciated that after executing the HS Algorithm 30 runs, with different parameters, we can find the best, average and worst results for the Rastrigin function. Table 4 shows the simulation results for the Ackley function.

From Table 4 it can be appreciated that after executing the HS Algorithm 30 runs, with different parameters, we can find the best, average and worst results for the Ackley function. Table 5 shows the simulation results for the Zakharov function.

From Table 5 it can be appreciated that after executing the HS Algorithm 30 runs, with different parameters, we can find the best, average and worst results for the Zakharov function. Table 6 shows the simulation results for the Sum Square function.

Table 6 Simulation results for the sum square function

Number of harmonies	Best	Worst	Mean
4	0.0000086056	0.000037648	0.00002513
5	0.0000065348	0.000041995	0.0000238824
10	0.000021245	0.00010459	0.000064354
20	0.0000048182	0.000046886	0.000029147
30	0.000011206	0.000054988	0.0000351173
40	0.000017024	0.000056469	0.0000379489

From Table 6 it can be appreciated that after executing the HS Algorithm 30 runs, with different parameters, we can find the best, average and worst results for the Sum Square function.

6.2 Simulation Results with the Genetic Algorithm

In this section we show the experimental obtained by the genetic algorithm in separate tables of the mathematical functions [13]. Table 7 shows the simulation results for the Sum Sphere function using genetic algorithm.

From Table 7 it can be appreciated that after executing the Genetic Algorithm 30 runs, with different parameters, we can find the best, average and worst results for the Sphere function. Table 8 shows the simulation results for the Rosenbrock function using genetic algorithm.

From Table 8 it can be appreciated that after executing the Genetic Algorithm 30 runs, with different parameters, we can find the best, average and worst results for the Rosenbrock function. Table 9 shows the simulation results for the Rastrigin function using genetic algorithm.

From Table 9 it can be appreciated that after executing the Genetic Algorithm 30 runs, with different parameters, we can find the best, average and worst results

Table 7 Simulation results for the sphere function

Population	Best	Worst	Mean
4	0.000655746	0.867154805	0.501445118
5	0.016158419	0.735175081	0.297672568
10	0.029477858	0.985900891	0.616489628
20	0.125851757	1.018067545	0.250640697
30	0.050431819	0.928690136	0.521845011
40	0.004944109	1.847289399	0.558953430

Table 8 Simulation results for the Rosenbrock function

Population	Best	Worst	Mean
4	0.089847334	0.802024183	0.561886727
5	0.045156568	0.878087097	0.476680695
10	0.026476082	0.788665597	0.212878969
20	0.008755795	0.654965394	0.151633433
30	0.001220403	0.292128413	0.050677876
40	0.000245092	0.843183891	0.242982579

Table 9 Simulation results for the Rastrigin function

Population	Best	Worst	Mean
4	0.014939893	0.997649164	0.532383503
5	0.025389969	1.023785562	0.484650675
10	0.000983143	1.991943912	0.591455815
20	0.005860446	0.973098541	0.416048173
30	0.001461684	1.030270667	0.402938563
40	0.0017108	1.011176844	0.304818043

for the Rastrigin function. Table 10 shows the simulation results for the Ackley function using genetic algorithm.

From Table 10 it can be appreciated that after executing the Genetic Algorithm 30 runs, with different parameters, we can find the best, average and worst results for the Ackley function. Table 11 shows the simulation results for the Zakharov function using genetic algorithm.

From Table 11 it can be appreciated that after executing the Genetic Algorithm 30 runs, with different parameters, we can find the best, average and worst results for the Zakharov function. Table 12 shows the simulation results for the Sum Square function using genetic algorithm.

Table 10 Simulation results for the Ackley function

Population	Best	Worst	Mean
4	0.003764969	0.974296237	0.346336685
5	0.034816548	1.008082845	0.555913584
10	0.00068874	1.006617458	0.309683405
20	0.00000943	0.999864848	0.217317909
30	0.0024992	0.045123886	0.01309672
40	0.00110442	0.963070253	0.229423519

Table 11 Simulation results for the Zakharov function

Population	Best	Worst	Mean
4	0.00291835	0.974030463	0.442892629
5	0.003929128	2.494950163	0.368494675
10	0.000621031	0.999709009	0.475767117
20	0.00754580	0.787983798	0.444793802
30	0.00074412	4.568706165	1.518534814
40	0.01262987	2.859991576	1.30142878

Table 12 Simulation results for the sum square function

Population	Best	Worst	Mean
4	0.029476905	0.834760272	0.379557704
5	0.009588642	0.441979109	0.172141341
10	0.000848578	0.045665911	−0.002597231
20	0.01018758	0.668265573	0.194550999
30	0.00259567	0.226922795	0.103921866
40	0.00638302	0.406954403	0.149111035

From Table 12 it can be appreciated that after executing the Genetic Algorithm 30 runs, with different parameters, we can find the best, average and worst results for the Sum Square function.

7 Conclusions

The HS algorithm is a new method which can solve various types of problem very easily and effectively because it not requires many complex calculations. The HS can handle discrete, continuous variables and can be applied to linear and nonlinear functions.

In the analysis of results obtained with the genetic algorithm and harmony search, we conclude that the HS is better than the GA this is demonstrated with the tables mentioned in the previous section to obtain a minimum error in all the benchmark functions which was applied, the same number of dimensions were used to perform the comparison.

The analysis of simulation results between HS and GA method considered in this work, lead us to the conclusion that for the optimization of benchmark functions, the HS method is a good alternative because it is easier to optimize and achieve good results try that with GA.

As we can realize in each of the tables where the results of the experiments with HS is best values were obtained.

With this it has been that the algorithm HS is better than GA.

Acknowledgments We would like to express our gratitude to the CONACYT and Tijuana Institute of Technology for the facilities and resources granted for the development of this research.

References

1. Dexuan, Z., Yanfeng, G., Liqun, G., Peifeng, W.: A novel global harmony search algorithm for chemical equation balancing. International Conference on Computer Design and Appliations, pp. 1–3. IEEE (2010)

2. Eberhart, R., Kennedy, J.: A new optimizer using particle swarm theory, In: Proceedings of the 6th International Symposium on Micromachine and Human Science, pp. 39–43. IEEE (1995)
3. Fevrier, V., Melin, P., Oscar, C.: Fuzzy Control of Parameters to Dynamically Adapt the PSO and GA Algorithms, pp. 1–8. IEEE, Barcelona, Spain (2010)
4. Geem, Z., Lee, K.: A New Meta-Heuristic Algorithm for Continuous Engineering Optimization Harmony Search Theory and Practice, Department of Civil and Environmental Engineering, University of Maryland, College Park, pp. 3-20. Elsevier, Maryland, USA (2004)
5. Geem, Z., Sim, K.: Parameter Setting Free Harmony Search Algorithm, School of Electrical and Electronics Engineering, Chung Ang University, pp. 2–10. Elsevier, Chung Ang, China (2010)
6. Geem, Z.: Harmony Search Algorithms For Structural Design Optimization. Studies in Computational Intelligence, pp. 8–121. Springer, Heidelberg, Germany (2009)
7. Geem Z.: Music Inspired Harmony Search Algorithm Theory and Applications, Studies in Computational Intelligence, pp. 8–121, Springer, Heidelberg, Germany (2009)
8. Hadi, M., Mehmet, A., Mashinchi, M., Pedrycz, W.:A Tabu Harmony Search Based Approach to Fuzzy Linear Regression, Transactions on Fuzzy Systems, pp. 1–13. IEEE, New Jersey, USA (2011)
9. Mahamed, G., Mahdavi, M.: Global Best Harmony Search, Applied Mathematics and Computation, pp. 1–14. Elsevier, Amsterdam, Holland (2008)
10. Mahdavi, M., Fesanghary, M., Damangir, E.: An Improved Harmony Search Algorithm for Solving Optimization Problems, Applied Mathematics and Computation, pp. 1567–1579. Elsevier, Amsterdam, Holland (2007)
11. Manjarres, D., Torres, L., Lopez, S., DelSer J, Bilbao M., Salcedo S., Geem Z.: A Survey on Applications of the Harmony Search Algorithm, Engineering Applications of Artificial Intelligence, pp. 3–14, Elsevier, Amsterdam, Holland (2013)
12. Ochoa, P., Castillo, O., Soria, J., Differential Evolution with Dynamic Adaptation of Parameters for the Optimization of Fuzzy Controllers, Recent Advances on Hybrid Approaches for Designing Intelligent Systems, pp. 275–288. Springer, Heidelberg, Germany (2013)
13. Perez, J., Valdez, F., Castillo, O.: Bat Algorithm Comparison with Genetic Algorithm Using Benchmark Functions, Recent Advances on Hybrid Approaches for Designing Intelligent Systems, pp. 225–237, Springer (2013)
14. Sombra, A., Valdez, F., Melin, P., Castillo, O.: A new gravitational search algorithm using fuzzy logic to parameter adaptation. IEEE Congress on Evolutionary Computation, pp. 1068–1074 (2013)
15. Štefek, A.: Benchmarking of Heuristic Optimization Methods, University of Defence, pp 1-4, IEEE, New Jersey, USA (2011)
16. Wang, C., Huang, Y.: Self Adaptive Harmony Search Algorithm for Optimization, Department of Computer Science and Information Engineering, National Yunlin University of Science and Technology, pp. 1–12, Elsevier (2010)
17. Yang, X.: Nature Inspired Metaheuristic Algorithms, 2nd edn, pp 73–76. Luniver Press, London, UK (2010)