# Full factorial experimental design for parameters selection of Harmony Search Algorithm

**Aphirak Khadwilard[1,*], Pongchanun Luangpaiboon[2] and Pupong Pongcharoen[3]**

## Abstract

Metaheuristic may be defined as an iterative search process that intelligently performs the exploration and exploitation in the solution space aiming to efficiently find near optimal solutions. Various natural intelligences and inspirations have been artificially embedded into the iterative process. In this work, Harmony Search Algorithm (HSA), which is based on the melody fine tuning conducted by musicians for optimising the synchronisation of the music, was adopted to find optimal solutions of nine benchmarking non-linear continuous mathematical models including two-, three- and four-dimensions. Considering the solution space in a specified region, some models contained a global optimum and multi local optima. A series of computational experiments was used to systematically identify the best parameters of HSA and to compare its performance with other metaheuristics including the Shuffled Frog Leaping (SFL) and the Memetic Algorithm (MA) in terms of the mean and variance of the solutions obtained.

**Keywords:** Harmony Search algorithm, Shuffled Frog Leaping, Memetic Algorithm, Metaheuristics, Optimisation

(selected from 1st Symposium on Hands-on Research and Development, Chiang Mai

[1] Department of Mechanical Engineering, Faculty of Engineering, Rajamangala University of Technology Lanna Tak.

[2] Department of Industrial Engineering, Faculty of Engineering, Thammasat University.

[3] Department of Industrial Engineering, Faculty of Engineering, Naresuan University.

[*] Corresponding author, E-mail: aphirak@rmutl.ac.th, k_aphirak@hotmail.com Received 1 August 2011; Accepted 7 December 2011

## 1. Introduction

Optimisation algorithms can be categorised as being either conventional or approximation optimisation algorithms [1]. Conventional optimisation algorithms are usually based upon mathematical models such as Linear Programming, Branch and Bound or Dynamic Programming. These approaches were relatively well developed and contributed to the military services early in World War II. Based on the full enumerative search within these approaches, the optimal solutions are always guaranteed. However, the application of these methods might need exponential computational time. This becomes an impractical approach especially for solving a very large size problem. Alternative approaches that can guide the search process to find near optimal solutions in acceptable computational time are therefore more practical and desirable.

Approximation optimisation algorithms (called metaheuristics) have therefore received more attention in the last few decades. Metaheuristics iteratively conduct stochastic search process inspired by natural inspiration. There are many metaheuristic algorithms such as Simulated Annealing, Tabu Search, Ant Colony Optimisation, Genetic Algorithm, Particle Swarm Optimisation, Shuffled Frog Leaping and Harmony Search Algorithm [2-8]. These alternative approaches have been widely used to solve large scale combinatorial optimisation problems [9-12].

Among the metaheuristics, Harmony Search Algorithm (HSA) has been recently developed and applied to solve many real world engineering problems such as structural optimisation, continuous engineering optimisation, vehicle routing, scheduling of multiple dam system, groundwater management, reliability and function optimisation [13-19]. These problems can be often transformed into non-linear n-dimension mathematical models. These models were therefore used to study the performance of metaheuristic algorithms such as Shuffled Frog Leaping and Memetic Algorithm [23]. Harmony Search Algorithm (HSA) is another recently developed metaheuristic and its parameter setting has not been academically reported to solve multiple-dimension nonlinear continuous mathematical functions. The objectives of this paper were to investigate the impact of the setting of HSA parameters on the algorithm performance and to study the performance of HSA in terms of the best of the solutions obtained by comparing it with other metaheuristics, Shuffled Frog Leaping and Memetic Algorithm.

## 2. Harmony search algorithm (HSA)

Harmony Search Algorithm was developed by Geem et al. in 2001 [20]. The name of Harmony Search Algorithm is inspired by the observation that the aim of composing music is to search for a perfect state of harmony [8]. There are three ideas in HSA that are used to generate a new harmony (solution) including usage of harmony memory, pitch adjusting, and randomisation. The usage of harmony memory is to randomly select a new harmony from harmony memory. This process will be controlled by Harmony Memory Considering Rate (HMCR). The second concept is the pitch adjustment which consists of two parameters that are bandwidth (BW) and Pitch Adjusting Rate (PAR). Finally, randomisation will be executed if random value is declined from HMCR. Those three concepts are illustrated in Fig. 1 that demonstrates the analogous relationship between the musical improvisation and optimisation. Harmony search algorithm can be explained in the improvisation process of a musician. When it is improvised, there are three possible choices 1) to play any famous piece of music exactly from a musician memory 2) to play something similar to a known piece by adjusting the pitch slightly 3) to create new or random notes. If three options for optimisation are formalise, there are three corresponding components including usage of harmony memory, pitch adjusting, and randomisation.

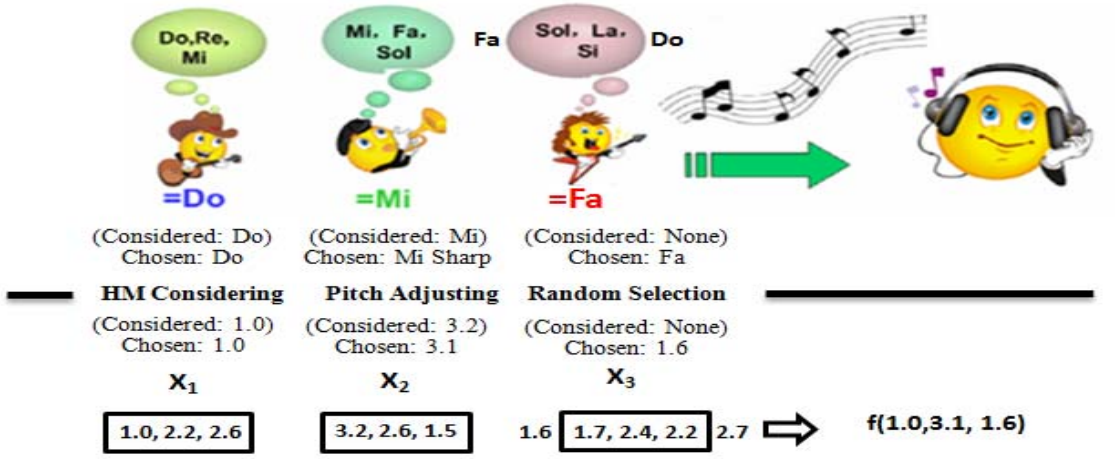


**Fig. 1** Analogy between musical improvisation and optimisation (Modified from [24])

In order to use effectively memories, A HMCR parameter can be assigned from 0 - 1. If HMCR is too low, only few best harmonies are selected and it may converge too slowly. If this rate is extremely high, almost all the harmonies are used in the harmony memory, and then other harmonies are not explored well, leading to potentially wrong solutions. Therefore HMCR should be set between 0.7 ～ 0.95.[8] To adjust the pitch in the second component, the new solution (pitch) $x_{new}$ is generated by Eq (1) when $x_{old}$ is the current solution (or pitch).

$$x_{new} = x_{old} + bw\,(2\,rand - 1) \qquad (1)$$

where *rand* is a random number which is drawn from a uniform distribution [0,1]. *bw* is the bandwidth, that controls the local search of pitch adjustment.

PAR can be assigned to control the rate of the adjustment. If PAR is too low, there is rarely any change. If it is too high, the algorithm may not converge at all. Therefore, PAR should be set between 0.7～ 0.95. [8]

The randomisation which is to increase the diversity of the solutions in the third component generated the new solution $x_{new}$ by Eq (2). The randomisation can explore various regions with high solution diversity to find the global optimality.

$$x_{new} = x_{lowerlimit} + (x_{upperlimit} - x_{lowerlimit})*rand \qquad (2)$$

where $x_{lowerlimit}$ and $x_{upperlimit}$ are lower and upper bounds of variable $x$ ,respectively.

The pseudo code of the Harmony Search Algorithm is provided in Fig. 2.

**Pseudo code of the Harmony Search algorithm (HSA)**
**Begin;**
Define objective function *f(x)*, $x=(x_1,x_2,\ldots,x_d)^T$
Define Harmony Memory Considering rate (HMCR)
Define Pitch adjusting rate (PAR) and other parameters
Generate Harmony Memory with random harmonies
**while** (t<max number of iterations)
**while** (i<=number of variables)
**if** *(rand*<HMCR*),*
*Choose a value from HM for the variable i*
**if** *(rand*<PAR*),*
*Adjust the value by adding certain amount*
**end if**
**else**
*Choose a random value*
**end if**
**end while**
Accept the New Harmony (solution) if better
**end while**
Find the current best solution
**end**

**Fig. 2** Pseudo code of the Harmony Search Algorithm

## 3. Memetic Algorithm (MA) and Shuffled Frog Leaping (SFL) algorithm

### 3.1 Memetic Algorithm

The name of the Memetic Algorithm (MA) is inspired by Dawkins' notation of a meme. MA is similar to Genetic Algorithm (GA) but the elements that form a chromosome are called memes, not genes. The main concept of the MA is that each individual and offspring is allowed to gain some experience through a local search before being involved in the evolutionary process [25].

### 3.2 Shuffled Frog Leaping algorithm

Shuffled Frog Leaping (SFL) algorithm is one of the biologically-based inspirations. In the SFL algorithm, a group of frogs (candidate solutions) is divided into subgroups (memeplexes), each of which has different cultures by performing a local search. Each frog has their own idea and can be influenced by the ideas of other frogs during the iterative shuffling process of memetic evolution [26].

## 4. Benchmarking functions

In this paper, nine non-linear continuous mathematical functions were used to test the performance of the proposed method for searching the optimal solutions. The functions including the equations and its contour plot showed in Fig. 3-10 are illustrated in the following subsections.

### 4.1 Todd function

$$f(x) = 4 - \frac{(3-\frac{x}{1.6})(0.5-\frac{x}{1.6})(6.2-\frac{x}{1.6})(2-\frac{x}{1.6})}{6} \qquad (3)$$

Range between: $0 < x < 10$

$f(x^*) = 9.515504816$, where $x^* = 8.29584$

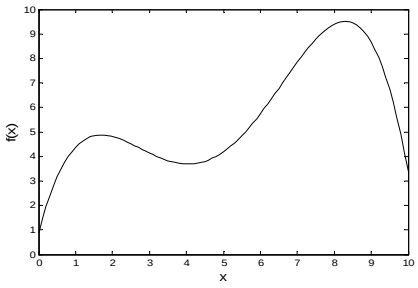


**Fig. 3** Todd Function

### 4.2 Camelback function

$$f(x,y) = 10 - \log_{10}[x^2(4-2.1x^2+\frac{1}{3}x^4) + xy + 4y^2(y^2+1)] \qquad (4)$$

Range between: $-20 < x < 20$ ; $-20 < y < 20$

$f(x^*,y^*) = \infty$, where $x^* = 0$ and $y^* = 0$

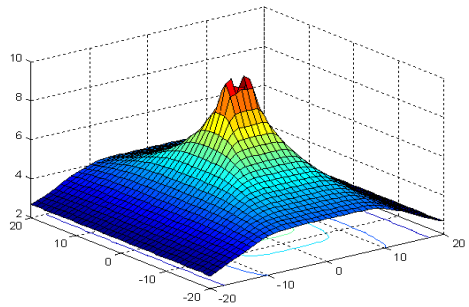

**Fig. 4** Camelback Function

### 4.3 Goldstein-price function

$$f(x,y) = 10 - \log_{10}[\{1+(1+x+y)^2(19-14x+3x^2-14y+6xy+3y^2)\} * \{30+(2x-3y)^2(18-32x+12x^2+48y-36xy+27y^2)\}] \qquad (5)$$

Range between: $-20 < x < 20$ ; $-20 < y < 20$

$f(x^*,y^*) = 9.522878745$, where $x^* = 0$ and $y^* = -1$

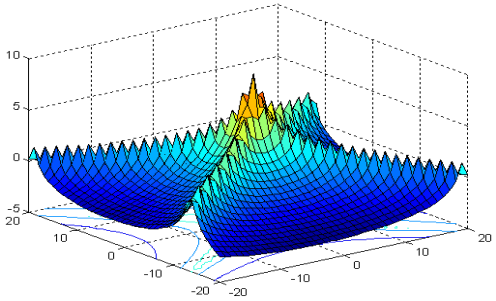

**Fig. 5** Goldstein-price Function

### 4.4 Montgomery's function

$$f(x,y) = 1217.3 - 31.256x + 86.017y + 0.12917x^2 - 2.8733y^2 + 0.02875xy \quad (6)$$

Range between: $100 < x < 140$ ; $10 < y < 20$

$f(x^*, y^*) = -66.03402$ , where $x^* = 119.87494$ and $y^* = 10$

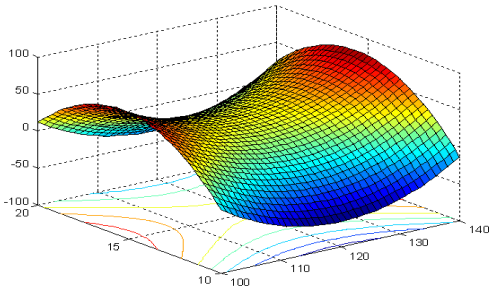

**Fig. 6** Montgomery's Function

### 4.5 Parabolic function

$$f(x,y) = 12 - \frac{x^2 + y^2}{100} \quad (7)$$

Range between: $-20 < x < 20$ ; $-20 < y < 20$

$f(x^*, y^*) = 12$ , where $x^* = 0$ and $y^* = 0$

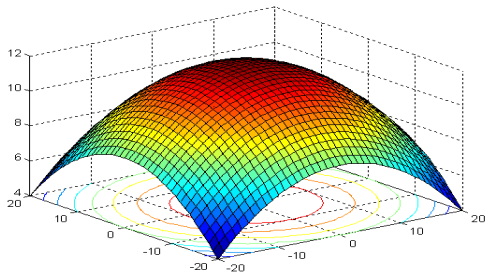

**Fig. 7** Parabolic Function

### 4.6 Rastrigin function

$$f(x,y) = 80 - [20 + x^2 + y^2 - 10(\cos(2\pi x) + \cos(2\pi y))] \quad (8)$$

Range between: $-5 < x < 5$ ; $-5 < y < 5$

$f(x^*, y^*) = 80$ , where $x^* = 0$ and $y^* = 0$

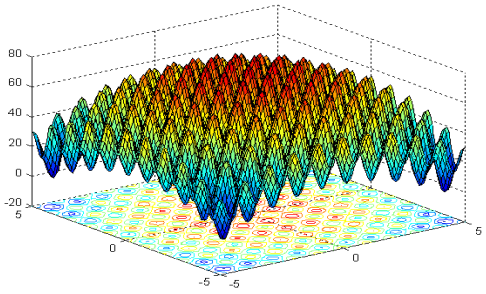

**Fig. 8** Rastrigin Function

### 4.7 Rosenbrock function

$$f(x,y) = 70\left[\frac{[20 - \{(\frac{-x}{-7})^2 + (\frac{y}{6} + (\frac{x}{-7})^2)^2\}] + 150}{170}\right] + 10 \quad (9)$$

Range between: $-20 < x < 20$ ; $-20 < y < 20$

$f(x^*, y^*) = 80$ , where $x^* = 0$ and $y^* = 0$

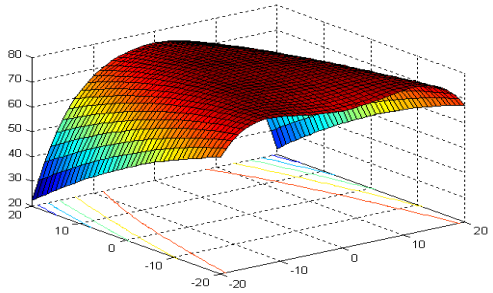

**Fig. 9** Rosenbrock Function

### 4.8 Shekel function

$$f(x,y) = 100\left[\frac{1}{10 + (x+4)^2 + (y+4)^2} + \frac{1}{10 + (x+4)^2 + (y-4)^2} + \frac{1}{5 + (x-4)^2 + (y+4)^2} + \frac{1}{10 + (x-4)^2 + (y-4)^2}\right] \quad (10)$$

Range between: $-20 < x < 20$ ; $-20 < y < 20$

$f(x^*, y^*) = 23.44523$ , where $x^* = 3.9524$ and $y^* = -3.9524$

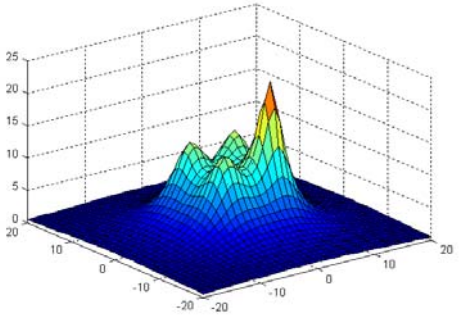

**Fig. 10** Shekel Function

**4.9 Dejong function (4 dimensions)**

$$f(x,y,z) = x^2 + y^2 + z^2 \quad (11)$$

Range between: $-5.12 < x < 5.12$ ; $-5.12 < y < 5.12$ ; $-5.12 < z < 5.12$

$f(x^*, y^*, z^*) = 0$, where $x^* = 0$, $y^* = 0$ and $z^* = 0$

## 5. Experimental design and analysis

In this work, a computer simulation program was developed using a two-step sequential experiment. The first experiment (Experiment A) was aimed to investigate the impact of parameters' setting on the HSA performance. A sequential experiment (Experiment B) was planned to study the performance comparison of the proposed methods with Shuffled Frog Leaping (SFL) and Memetic Algorithm (MA) in terms of the mean and standard deviation of the solution obtained.

**5.1 Experiment A**

Full factorial experimental design [21] with five replications was carried out to solve Camelback function. There were four factors to be considered. The first factor was the combination of harmony memory size and number of iteration (HMS/NI), which determines the total harmony memory to be investigated. This factor had an influence on the exploration process of seeking (generated) results in the solution space and also delaying the execution time of the computational run. In this present work, a total amount of harmony memory generated was fixed at 10,000. The remaining factors including Harmony Memory Considering Rate (HMCR), Pitch Adjusting Rate (PAR) and Bandwidth (BW) were suggested between 0-1, 0-1 and 0.01 [20], respectively. However, Yang [8] suggested that HMCR and PAR should be set between 0.7-0.95 and 0.1- 0.5 respectively. Other research works have defined these values differently [19, 22]. To find the appropriate parameters for the test problems, $3^k$ fractional design were used in this work. The experimental factors and their values considered are shown in Table 1.

**Table 1** experimental factors and their levels

| Factors | Level (coded) | | |
|---|---|---|---|
| | -1 | 0 | +1 |
| Harmony Memory Size / Number of Iteration (HMS/NI) | 5/2000 | 100/100 | 2000/5 |
| Harmony Memory Considering Rate (HMCR) | 0.1 | 0.5 | 0.9 |
| Pitch Adjusting Rate (PAR) | 0.1 | 0.5 | 0.9 |
| Bandwidth (BW) | 0.1 | 0.3 | 0.5 |

The experimental results obtained from 405 (81x5) runs were analysed using a general linear form of analysis of variance (ANOVA), main effect plots and interaction plots. Table 2 shows an ANOVA table consisting of Source of Variation (Source), Degrees of Freedom (*DF*), Sum of Square (*SS*), Mean Square (*MS*), and F and *P* values. A factor with value of $P \leq 0.05$ was considered statistically significant with a 95% confidence interval. From Table 2, it can be seen that all HSA parameters including HMS/NI, HMCR, PAR and BW were statistically significant with 95% confidence interval. In case of maximisation, main effect plot in Fig. 11 suggested that HMS/NI, HMCR, PAR and BW factors should be set at 5/2000, 0.9, 0.5 and 0.1, respectively. These factors correspond to interaction plots in Fig. 12-15. For interaction plots in Fig. 12, the highest point is associated with the HMI/NI of 5/2000 and the HMCR of 0.9. Interaction plots in Fig. 13 suggest HMCR at 5/2000 and PAR at 0.5 while interaction plots in Fig. 14 introduce HMCR at 5/2000 and BW at 0.1. Fig. 15 shows that HMCR and PAR should be set at 0.9 and 0.5, respectively, by considering the highest point in the graphs.

**Table 2** Analysis of variance on Camelback function

| Source | DF | SS | MS | F | P |
|---|---|---|---|---|---|
| HS/NI | 2 | 454.23 | 227.12 | 578.47 | 0.000 |
| HMCR | 2 | 27.831 | 13.92 | 35.44 | 0.000 |
| PAR | 2 | 3.9099 | 1.955 | 4.98 | 0.007 |
| BW | 2 | 5.2857 | 2.64 | 6.73 | 0.001 |
| HS/NI*HMCR | 4 | 48.924 | 12.23 | 31.15 | 0.000 |
| HS/NI*PAR | 4 | 8.9941 | 2.248 | 5.73 | 0.000 |
| HS/NI*BW | 4 | 11.846 | 2.961 | 7.54 | 0.000 |
| HMCR*PAR | 4 | 5.2943 | 1.324 | 3.37 | 0.010 |
| HMCR*BW | 4 | 0.6027 | 0.151 | 0.38 | 0.820 |
| PAR*BW | 4 | 0.136 | 0.034 | 0.09 | 0.987 |
| HS/NI*HMCR*PAR | 8 | 10.951 | 1.369 | 3.49 | 0.001 |
| HS/NI*HMCR*BW | 8 | 2.6458 | 0.331 | 0.84 | 0.566 |
| HS/NI*PAR*BW | 8 | 1.5745 | 0.197 | 0.50 | 0.855 |
| HMCR*PAR*BW | 8 | 1.7796 | 0.222 | 0.57 | 0.805 |
| HS/NI*HMCR*PAR*BW | 16 | 1.9839 | 0.124 | 0.32 | 0.995 |
| Error | 324 | 127.21 | 0.393 | | |
| Total | 404 | 713.20 | | | |

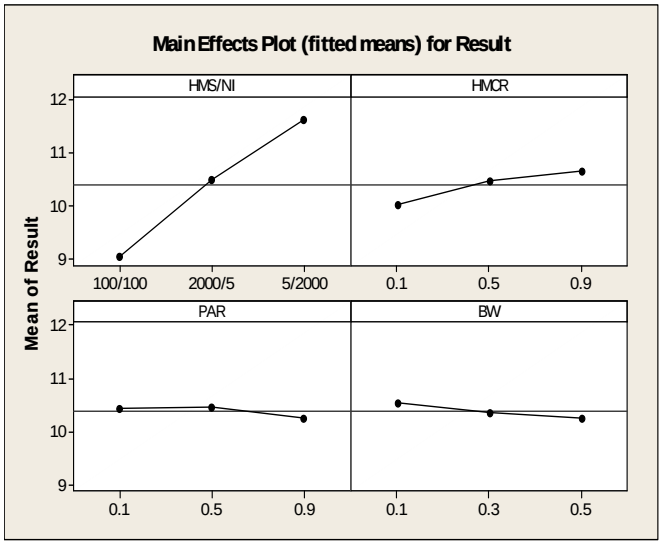


**Fig. 11** Main effect plot of HAS

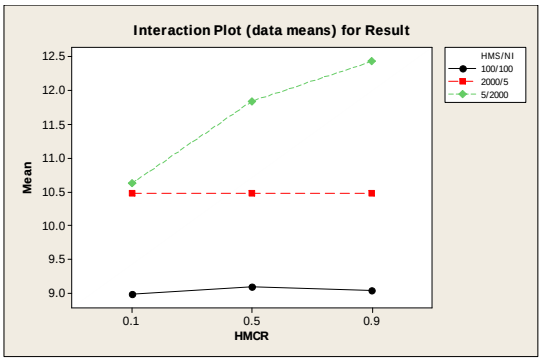


**Fig. 12** Interaction plot of HMS/NI and HMCR

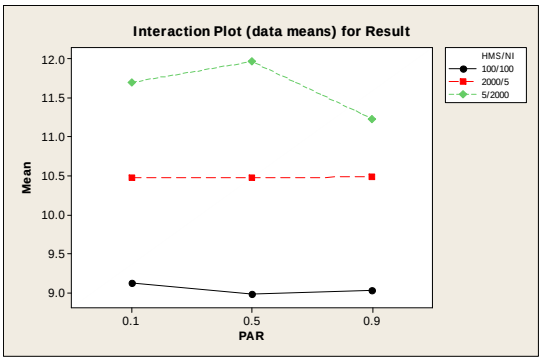


**Fig. 13** Interaction plot of HMS/NI and PAR

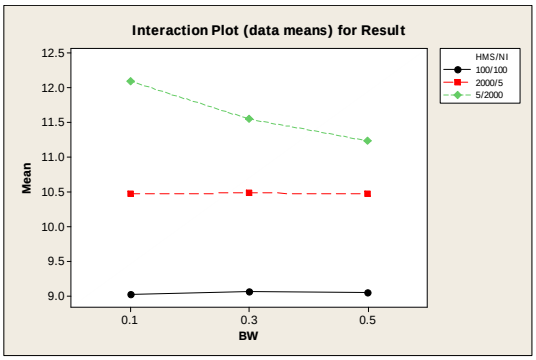


**Fig. 14** Interaction plot of HMS/NI and BW

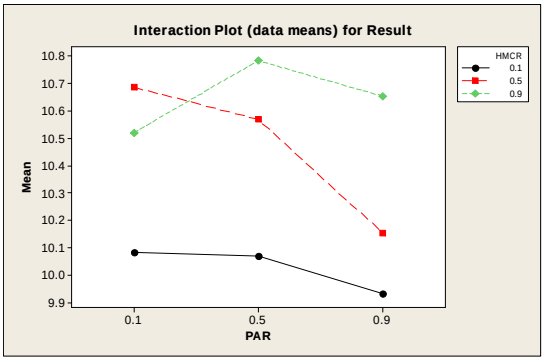


**Fig. 15** Interaction plot of HMC and PAR

**Table 3** Experimental results obtained from the proposed methods on each testing function

| F(x) | Function name | Bound | Optimal solutions | Best so far | | |
|---|---|---|---|---|---|---|
| | | | | Shuffled Frog Leaping | Memetic Algorithm | Harmony Search |
| 1 | Todd | Upper | 9.515504816 | 9.515504816 | 9.515504816 | 9.515504816 |
| 2 | Camelback | Upper | Infinity value | 11.379013687 | 14.433180315 | **14.97650836** |
| 3 | Goldstein-price | Upper | 9.522878745 | 9.328871327 | 9.517646070 | **9.522842492** |
| 4 | Montgomery | Lower | -66.03402000 | -65.911482081 | -66.033756829 | **-66.03401921** |
| 5 | Parabolic | Upper | 12.000000000 | 11.999993465 | 11.999998622 | **11.99999999** |
| 6 | Rastrigin | Upper | 80.000000000 | 79.773552099 | 79.996073710 | **79.99990626** |
| 7 | Rosenbrock | Upper | 80.000000000 | 79.999918610 | 79.999999088 | **79.99999999** |
| 8 | Shekel | Upper | 23.445230201 | 23.443375697 | **23.445202585** | 23.44519853 |
| 9 | Dejong | Lower | 0.000000000 | 0.001878667 | 0.000676320 | **0.000005215** |

It can be summerised that HSA at 5/2000, HMCR at 0.9, PAR at 0.5 and BW at 0.1 were the best setting for Camelback function. These findings of parameters' setting were used in the sequential experiment presented in the next section.

### 5.2 Experiment B

This experiment was aimed to compare the results obtained from the Harmony Search Algorithm (HSA) with the results obtained from the Shuffled Frog Leaping (SFL) and the Memetic Algorithm (MA) conducted in our previous research [23]. In Table 3, it can be seen that Shuffled Frog Leaping (SFL), Memetic Algorithm (MA) and Harmony Search Algorithm (HSA) found the optimal solutions (0.00 %) only the first (Todd) function with single factor. For the functions 2-8 (all with two factors), the best-so-far (BSF) solutions obtained from HSA were dramatically better than those results obtained from SFL. When HSA was compared with MA for the functions 2-7, it was found that the best-so-far (BSF) solutions obtained from HSA were better than those results obtained from MA but, for the function 8, MA was slightly better than HAS. For the last (Dejong) function with three factors, the performance of HSA was better than that of the SFL and MA. Obviously, HSA outperform both SFL and MA. Therefore, for the future research, HSA is interested to apply and modify for solving other combinatorial problems such as job shop scheduling, timetabling and bin packing problem.

## 6. Conclusion

In this work, Harmony Search Algorithm (HSA) was adopted to find optimal solutions of nine non-linear continuous mathematical models. Considering the solution space in a specified region, some models contained a global optimum and multi local optima. A series of computational experiments was used to systematically identify the best parameters of the HSA and to compare its performance with other metaheuristics including the Shuffled Frog Leaping (SFL) and the Memetic Algorithm (MA) in terms of the best of the solutions obtained. It was found that HSA outperformed both SFL and MA.

## 7. References

[1] C. Blum and A. Roli. “Metaheuristics in combinatorial optimisation”, ACM Computing surveys, 35, 2003,pp. 268-308.

[2] S. Kirkpatrick, C.D. Gelatt, and M.P. Vecchi. "Optimisation by simulated annealing". Science, 220,1983, pp.671-679.

[3] F. Glover. Tabu search - part I", ORSA Journal on Computing, 1, 1986, pp. 190-206.

[4] M. Dorigo and T. Stutzle. "Ant colony optimization", Massachusetts, Bradford Book, 2004.

[5] D.E. Goldberg, "Genetic algorithms in search, optimisation and machine learning", Massachusetts, Addison-Wesley, 1989.

[6] J. Kennedy and R.C. Eberhart. Swarm intelligence", San Francisco, CA: Morgan Kaufmann Publishers, 2001.

[7] M. Eusuff, K. Lansey, and F. Pasha. "Shuffled frog leaping algorithm: A memetic meta-heuristic for discrete optimization", Engineering Optimisation, 38, 2006, pp. 129-154.

[8] Yang. Xin She. "Nature-inspired metaheuristic algorithm", Luniver Press, United Kingdom, 2008.

[9] H. Aytug, M. Knouja, and F.E. Vergara.0 "Use of genetic algorithms to solve production and operations management problems: A review", International Journal of Production Research, 41, 2003, pp. 3955–4009.

[10] S.S. Chaudhry and W. Luo. "Application of genetic algorithms in production and operations management: A review", International Journal of Production Research, 43, 2005, pp. 4083-4101.

[11] D. Dasgupta. "Artificial Immune Systems and their applications", Springer-Verlag, 1998.

[12] M. Dorigo and C. Blum. "Ant colony optimisation theory: A survey", Theoretical Computer Science, 344, 2005, pp. 243-278.

[13] K.S. Lee and Z.W. Geem. "A new structural optimization method based on the Harmony Search Algorithm", Computers and Structures, 82, 2004, pp. 781–798.

[14] K.S. Lee and Z.W. Geem. "A new meta-heuristic algorithm for continuous engineering optimisation: harmony search theory and practice", Computer Methods Applied Mechanics and Engineering, 194, 2005, pp. 3902–3933.

[15] Z.W. Geem, K.S. Lee, and Y.J. Park. "Application of Harmony Search to Vehicle Routing", American Journal of Applied Sciences, 2, 2005, pp. 1552–1557.

[16] Z.W. Geem. "Optimal scheduling of multiple dam system using Harmony Search Algorithm", Sandoval, F. Prieto, A.G. Cabestany, J. Graña, M. (eds.) IWANN 2007. LNCS, Heidelberg, Springer, 4507, 2007, pp. 316–323.

[17] M.T. Ayvaz. "Application of Harmony Search algorithm to the solution of groundwater management models", Advances in Water Resources, 32, 2009, pp. 916–924.

[18] D.X. Zou, L.G. Gao, J.H. Wu, S. Li, and Y. Li. "A novel global Harmony Search Algorithm for reliability problems", Computers & Industrial Engineering, 58, 2010, pp. 307–316.

[19] Q.K. Pan, P.N. Suganthan , M.F. Tasgetiren , and J.J. Liang. "A self-adaptive global best harmony search algorithm for continuous optimisation problems", Applied Mathematics and Computation, 216, 2010, pp. 830-848.

[20] Z.W. Geem, J.H. Kim, and G.V. Loganathan. 2001."A new heuristic optimisation algorithm: Harmony search", Simulation, 76, 60-68.

[21] D.C. Montgomery. 2001. "Design and analysis of experiments", NY, John Wiley and Sons.

[22] C.M. Wang and Y.F. Huang. 2010. "Self-adaptive harmony search algorithm for optimisation", Expert Systems with Applications, 37, 2826-2837.

[23] P. Ittipong, P. Luangpaiboon, and P. Pongcharoen. 2008."Solving Non-linear continuous mathematical models using Shuffled Frog Leaping and Memetic Algorithms, Proceedings of the 5th National Conference on Operations Research, 79-85, KMUTNB, Bangkok, Thailand, July 24-25.

[24] H. Xu, X.Z. Gao, T. Wang, and K. Xue. 2010. "Harmony Search Optimization Algorithm: Application to a Reconfigurable Mobile Robot Prototype", Recent Advances in Harmony Search Algorithm, Springer-Verlag Berlin Heidelberg.

[25] Merz, P., Freisleben, B., 1999. Fitness landscapes andmemetic algorithm design. London, UK: McGraw-Hill.

[26] Eusuff, M.M., Lansey, K.E., 2003. Optimization of water distribution network design using the shuffled frog leaping algorithm. Journal of Water Resources Planning and Management-Asce, 129 (3), 210-225.