On the Harmony Search Using Quaternions

João Papa[1(✉)], Danillo Pereira[1], Alexandro Baldassin[1], and Xin-She Yang[2]

[1] Department of Computing, São Paulo State University, São Paulo, Brazil
papa@fc.unesp.br, danilopereira@unoeste.br
[2] School of Science and Technology, Middlesex University, London, UK
x.yang@mdx.ac.uk

Abstract. Euclidean-based search spaces have been extensively studied to drive optimization techniques to the search for better solutions. However, in high dimensional spaces, non-convex functions might become too tricky to be optimized, thus requiring different representations aiming at smoother fitness landscapes. In this paper, we present a variant of the Harmony Search algorithm based on quaternions, which extend complex numbers and have been shown to be suitable to handle optimization problems in high dimensional spaces. The experimental results in a number of benchmark functions against standard Harmony Search, Improved Harmony Search and Particle Swarm Optimization showed the robustness of the proposed approach. Additionally, we demonstrated the robustness of the proposed approach in the context of fine-tuning parameters in Restricted Boltzmann Machines.

Keywords: Harmony Search · Quaternions · Optimization

1 Introduction

Function optimization plays an important role in a number of applications, ranging from simulation in aerodynamics to fine-tuning machine learning algorithms. Although one can find several problems that can be modeled by convex functions, most applications out there pose a bigger challenge, since they are usually encoded by non-convex functions, which means they may contain both local and global optima. In light of that, one may handle such sort of functions by means of two approaches: (i) the first one concerns with trying different optimization techniques, and (ii) the second one aims at finding a different representation of the search space in order to deal with smoother landscape functions.

In the last decades, the scientific community has focused even more on optimization techniques based on the nature, the so-called meta-heuristics [23]. Such techniques are based on different mechanisms that address the problem of optimization, such as evolutionary processes, mimetism, and swarm intelligence, just to name a few. These techniques have been in the spotlight mainly due to their elegance and easiness of implementation, as well as solid results in a number of well-known problems in the literature [1].

F. Schwenker et al. (Eds.): ANNPR 2016, LNAI 9896, pp. 126–137, 2016.
DOI: 10.1007/978-3-319-46182-3_11

Recently, Malan and Engelbrecht [11] presented an interesting study that aimed at predicting possible situations in which the well-known Particle Swarm Optimization (PSO) algorithm would fail based on the fitness landscape. Basically, depending on the "smoothness degree" of the fitness function landscape, one can expect a probable performance of the algorithm. Once again, the fitness function plays an important role in the optimization problem, and finding suitable representations of that function in different search spaces is of great importance to obtain more accurate results. Years before their work, Merz and Freisleben [12] presented a study about fitness landscape analysis concerning memetic algorithms, and Humeau et al. [9] conducted a similar study, but in the context of local search algorithms.

Some years ago, Fister et al. [4] presented a modified version of the Firefly Algorithm based on quaternions, and later on Fister et al. [3] proposed a similar approach to the Bat Algorithm. Roughly speaking, the quaternion algebra extends the complex numbers by representing a real number using four variables [6], being widely used in areas when one needs to perform rotations with minimal computation, such as spacecraft controllers, for instance. Usually, standard representations of meta-heuristic techniques, i.e., the ones based on Euclidean spaces, tend to get trapped from local optima in higher dimensional spaces, since the fitness landscapes may become even more complicated. Therefore, that is the main motivation in using quaternion-based algebra.

One of the main drawbacks related to swarm-driven meta-heuristics concerns with their computational burden, since the fitness function needs to be evaluate whenever a possible solution changes its position. Since all decisions of a movement are taken into account collectively, a single movement of a possible solution may affect all remaining ones, thus requiring their new positioning. Harmony Search (HS) is a technique that updates one solution at each iteration only, making it one of the fastest meta-heuristic optimization techniques [5]. The idea is to model each decision variable as an instrument, being the best combination of them the solution to the chosen. By best combination we mean the one which provides "the music with optimal harmony".

One can refer to a number of different variants of HS, such as Improved Harmony Search [10], Global-best Harmony Search [13], and Self-adaptive Global-best Harmony Search [14], just to name a few. However, to the best of our knowledge, there is no Harmony Search implementation based on quaternions up to date. Therefore, the main contribution of this paper is to propose a quaternion-oriented Harmony Search, hereinafter called Quaternion Harmony Search (QHS). The proposed approach is compared against PSO, HS and Improved Harmony Search (IHS) for the task of benchmark function optimization. Also, we validate the proposed approach to fine-tune parameters of Restricted Boltzmann Machines to the task of binary image reconstruction, which turns out to be another contribution of this work.

The remainder of the paper ir organized as follows. Sections 2 and 3 present the theoretical background about quaternions and Harmony Search, respectively. Sections 4 and 5 present the methodology and experiments, respectively. Finally, Sect. 6 states conclusions and future works.

2 Quaternion Algebra

A quaternion q is composed of real and complex numbers, i.e., $q = x_0 + x_1 i + x_2 j + x_3 k$, where $x_0, x_1, x_2, x_3 \in \Re$ and i, j, k are imaginary numbers following the next set of equations:

$$ij = k, \tag{1}$$

$$jk = i, \tag{2}$$

$$ki = j, \tag{3}$$

$$ji = -k, \tag{4}$$

$$kj = -i, \tag{5}$$

$$ik = -j, \tag{6}$$

and

$$i^2 = j^2 = k^2 = -1. \tag{7}$$

Roughly speaking, a quaternion q is represented in a 4-dimensional space over the real numbers, i.e., $\Re^4$. Actually, we can consider the real numbers only, since most applications do not consider the imaginary part, as the one addressed in this work.

Given two quaternions $q_1 = x_0 + x_1 i + x_2 j + x_3 k$ and $q_2 = y_0 + y_1 i + y_2 j + y_3 k$, the quaternion algebra defines a set of main operations [2]. The addition, for instance, can be defined by:

$$\begin{aligned} q_1 + q_2 &= (x_0 + x_1 i + x_2 j + x_3 k) + (y_0 + y_1 i + y_2 j + y_3 k) \\ &= (x_0 + y_0) + (x_1 + y_1)i + (x_2 + y_2)j + (x_3 + y_3)k, \end{aligned} \tag{8}$$

while the subtraction is defined as follows:

$$\begin{aligned} q_1 - q_2 &= (x_0 + x_1 i + x_2 j + x_3 k) - (y_0 + y_1 i + y_2 j + y_3 k) \\ &= (x_0 - y_0) + (x_1 - y_1)i + (x_2 - y_2)j + (x_3 - y_3)k. \end{aligned} \tag{9}$$

Another important operation is the norm, which maps a given quaternion to a real-valued number, as follows:

$$\begin{aligned} N(q_1) &= N(x_0 + x_1 i + x_2 j + x_3 k) \\ &= \sqrt{x_0^2 + x_1^2 + x_2^2 + x_3^2}. \end{aligned} \tag{10}$$

Finally, Fister et al. [3,4] introduced two other operations, *qrand* and *qzero*. The former initializes a given quaternion with values drawn from a Gaussian distribution, and it can be defined as follows:

$$qrand() = \{x_i = \mathcal{N}(0, 1) | i \in \{0, 1, 2, 3\}\}. \tag{11}$$

The latter function initialized a quaternion with zero values, as follows:

$$qzero() = \{x_i = 0 | i \in \{0, 1, 2, 3\}\}. \tag{12}$$

Although there are other operations, we defined only the ones employed in this work.

3 Harmony Search

Harmony Search is a meta-heuristic algorithm inspired in the improvisation process of music players. Musicians often improvise the pitches of their instruments searching for a perfect state of harmony [5]. The main idea is to use the same process adopted by musicians to create new songs to obtain a near-optimal solution according to some fitness function. Each possible solution is modelled as a harmony, and each musician corresponds to one decision variable.

Let $\phi = (\boldsymbol{\phi}_1, \boldsymbol{\phi}_2, \ldots, \boldsymbol{\phi}_N)$ be a set of harmonies that compose the so-called "Harmony Memory", such that $\boldsymbol{\phi}_i \in \Re^M$. The HS algorithm generates after each iteration a new harmony vector $\hat{\boldsymbol{\phi}}$ based on memory considerations, pitch adjustments, and randomization (music improvisation). Further, the new harmony vector $\hat{\boldsymbol{\phi}}$ is evaluated in order to be accepted in the harmony memory: if $\hat{\boldsymbol{\phi}}$ is better than the worst harmony, the latter is then replaced by the new harmony. Roughly speaking, HS algorithm basically rules the process of creating and evaluating new harmonies until some convergence criterion is met.

In regard to the memory consideration step, the idea is to model the process of creating songs, in which the musician can use his/her memories of good musical notes to create a new song. This process is modelled by the Harmony Memory Considering Rate ($HMCR$) parameter, which is the probability of choosing one value from the historic values stored in the harmony memory, being $(1-HMCR)$ the probability of randomly choosing one feasible value[1], as follows:

$$\hat{\phi}^j = \begin{cases} \phi_A^j & \text{with probability HMCR} \\ \theta \in \varPhi_j & \text{with probability (1-HMCR),} \end{cases} \tag{13}$$

where $A \sim \mathcal{U}(1, 2, \ldots, N)$, and $\boldsymbol{\varPhi} = \{\varPhi_1, \varPhi_2, \ldots, \varPhi_M\}$ stands for the set of feasible values for each decision variable[2].

Further, every component j of the new harmony vector $\hat{\boldsymbol{\phi}}$ is examined to determine whether it should be pitch-adjusted or not, which is controlled by the Pitch Adjusting Rate (PAR) variable, according to Eq. 14:

$$\hat{\phi}^j = \begin{cases} \hat{\phi}^j \pm \varphi_j \varrho & \text{with probability PAR} \\ \hat{\phi}^j & \text{with probability (1-PAR).} \end{cases} \tag{14}$$

The pitch adjustment is often used to improve solutions and to escape from local optima. This mechanism concerns shifting the neighbouring values of some decision variable in the harmony, where ϱ is an arbitrary distance bandwidth, and $\varphi_j \sim \mathcal{U}(0, 1)$. In the following, we briefly present the Improved Harmony Search technique, which has been used in the experimental section.

3.1 Improved Harmony Search

The Improved Harmony Search (IHS) [10] differs from traditional HS by updating the PAR and ϱ values dynamically. The PAR updating formulation at time

[1] The term "feasible value" means the value that falls in the range of a given decision variable.

[2] Variable A denotes a harmony index randomly chosen from the harmony memory.

step t is given by:

$$PAR^t = PAR_{min} + \frac{PAR_{max} - PAR_{min}}{T} t, \tag{15}$$

where T stands for the number of iterations, and PAR_{min} and PAR_{max} denote the minimum and maximum PAR values, respectively. In regard to the bandwidth value at time step t, it is computed as follows:

$$\varrho^t = \varrho_{max} \exp \frac{\ln(\varrho_{min}/\varrho_{max})}{T} t, \tag{16}$$

where ϱ_{min} and ϱ_{max} stand for the minimum and maximum values of ϱ, respectively.

3.2 Quaternion Harmony Search

The proposed approach aims at mapping the problem of optimizing variables in the Euclidean space to the quaternions space. As aforementioned, the idea is to obtain smoother representations of the fitness landscape, thus making the problem easier to handle.

In the standard Harmony Search, each harmony $\boldsymbol{\phi}_i \in \Re^M$, $i = 1, 2, \ldots, N$ is modeled as an array containing M variables to be optimized, such that $\phi_{ij} \in \Re$. In QHS, each decision variable j is now represented by a quaternion $q_j \in \Re^4$, such that each harmony can be seen as a matrix $\boldsymbol{\phi}'_i \in \Re^{4\times N}$. Therefore, each harmony is no longer an array of decision variables, but a matrix instead.

However, we can map each quaternion to a real-valued number in order to use standard HS. Basically, one has to compute $\phi_{ij} = N(\phi'_{ij})$, $i = 1, 2 \ldots, N$ and $j = 1, 2, \ldots, M$. Further, the standard HS procedure can be executed as usual. But note the optimization process is conducted at quaternion level, which means QHS aims finding the quaternions for each decision variable such that their norm values minimizes the fitness function. An ordinary HS aims at learning values for each decision variable that minimizes the fitness function.

4 Methodology

In this section, we present the methodology employed in this work to validate the proposed approach. The next sections describe the techniques used for comparison purposes, benchmark functions and the statistical analysis.

4.1 Functions

The benchmark functions used in the experiments are the following:

- **Sphere Function:**
  $f(x, y) = x^2 + y^2$
  The minimum is $f(0, 0) = 0$, and the domain is
  $-\infty < x, y < \infty$.

- **Ackley's Function:**
  $f(x,y) = -20e^{(-0.2\sqrt{0.5(x^2+y^2)})} - e^{(0.5(cos(2\pi x)+cos(2\pi y)))} + e + 20$
  The minimum is $f(0,0) = 0$, and the domain is $-5 \leq x, y \leq 5$.
- **Rosenbrock's Function:**
  $f(x,y) = 100\left(y - x^2\right)^2 + (x-1)^2$
  The minimum is $f(1,1) = 0$, and the domain is
  $-\infty < x, y < \infty$.
- **Beale's Function:**
  $f(x,y) = (1.5 - x + xy)^2 + \left(2.25 - x + xy^2\right)^2 + \left(2.625 - x + xy^3\right)^2$
  The minimum is $f(3, 0.5) = 0$, and the domain is $-4.5 < x, y < 4.5$.
- **Matyas Function:**
  $f(x,y) = 0.26\left(x^2 + y^2\right) - 0.48xy$
  The minimum is $f(0,0) = 0$, and the domain is
  $-10 \leq x, y \leq 10$.
- **Levi Function N.13:**
  $f(x,y) = \sin^2(3\pi x) + (x-1)^2\left(1 + \sin^2(3\pi y)\right) + (y-1)^2\left(1 + \sin^2(2\pi y)\right)$
  The minimum is $f(1,1) = 0$, and the domain is
  $-10 < x, y < 10$.
- **Three-Hump Camel Function:**
  $f(x,y) = 2x^2 - 1.05x^4 + \frac{x^6}{6} + xy + y^2$
  The minimum is $f(0,0) = 0$, and the domain is
  $-5 \leq x, y \leq 5$.
- **Easom Function:**
  $f(x,y) = -\cos(x)\cos(y)\, e^{\left(-\left((x-\pi)^2+(y-\pi)^2\right)\right)}$
  The minimum is $f(\pi,\pi) = -1$, and the domain is
  $-100 < x, y < 100$.
- **S. Tang:**
  $f(x,y) = 0.5\left((x^4 - 16x^2 + 5x) + (y^4 - 16y^2 + 5y)0\right.$
  The minimum is $f(-2.9035, -2.9035) = -78.3254$, and the domain is $-5 < x, y < 5$.
- **Schaffer Function N2:**
  $f(x,y) = 0.5 + (\sin^2(x^2 - y^2) - 0.5)/(1 + 0.001 * (x^x + y^y)^2)$
  The minimum is $f(0,0) = 0$, and the domain is $-100 < x, y < 100$.

4.2 Meta-Heuristic Techniques

We compare the efficiency and effectiveness of QHS against three other methods, say that: naïve HS, IHS and PSO. For each optimization method and function, we conducted a cross-validation with 20 runs. Table 1 displays the parameters setup, being their values empirically chosen [18]. Notice we employed 20 agents and 15,000 iterations for all meta-heuristic techniques, and both x and y decision variables of the aforementioned functions were initialized within the range $[-10, 10]$.

Table 1. Parameter values used for all optimization techniques.

Technique	Parameters
HS	HMCR = 0.7, PAR = 0.7
IHS	HMCR = 0.7, PAR$_{min}$ = 0.1, PAR$_{max}$ = 0.9
PSO	$c1 = c2 = 2.0$, $w = 0.9$
QHS	HMCR = 0.7, PAR = 0.7

5 Experimental Evaluation

In this section, we present the experimental results considering the aforementioned optimization techniques in the context of benchmark functions (Sect. 5.1), as well as to fine-tune Restricted Boltzmann Machines (Sect. 5.2).

5.1 Benchmark Functions

As aforementioned in Sect. 4.1, we considered 10 benchmark functions to evaluate the robustness of QHS. Table 2 displays the mean values obtained by all techniques. Since we are dealing with minimization problems, the smaller the values, the better the techniques. The values in bold stand for the best techniques according to the Wilcoxon signed-rank test [22] with significance level of 0.05.

Clearly, we can observe QHS obtained the best results in 9 out 10 functions, and it has been the sole technique in 4 functions that achieved the best values. In fact, if one take into account the convergence of the techniques, one can realize HS took longer to converge, even on a convex function like Sphere, for instance. Figure 1a depicts the convergence process with respect to Sphere function. We constrained the convergence up to 600 iterations to have a better look at the beginning of the process. One can observe QHS converges at the very first iterations, while HS and IHS need a few hundred of them.

Table 2. Mean values obtained through cross-validation.

Function	PSO	IHS	HS	QHS
Sphere	**0.00 ± 0.00**	**0.00 ± 0.00**	**0.00 ± 0.00**	**0.00 ± 0.00**
Ackley's	0.24 ± 0.77	**0.00 ± 0.00**	**0.00 ± 0.00**	**0.00 ± 0.00**
Rosebrock's	0.06 ± 1.96	0.01 ± 0.00	0.13 ± 0.24	**0.00 ± 0.00**
Beale's	0.07 ± 0.19	0.10 ± 0.06	0.09 ± 0.30	**0.04 ± 0.05**
Matyas	0.42 ± 1.07	0.19 ± 0.07	0.18 ± 0.00	**0.00 ± 0.00**
Levi	0.02 ± 0.07	0.09 ± 0.01	0.08 ± 0.04	**0.01 ± 0.01**
Three Hump	0.01 ± 0.03	**0.00 ± 0.00**	**0.00 ± 0.00**	**0.00 ± 0.00**
Easom	−0.99 ± 0.36	−0.95 ± 0.03	−0.93 ± 0.03	**−0.99 ± 0.01**
S. Tang	**−57.76 ± 2.78**	−51.06 ± 21.34	−51.81 ± 13.45	−50.00 ± 0.01
Schaffer	0.004 ± 0.001	**0.00 ± 0.00**	**0.00 ± 0.00**	**0.00 ± 0.00**

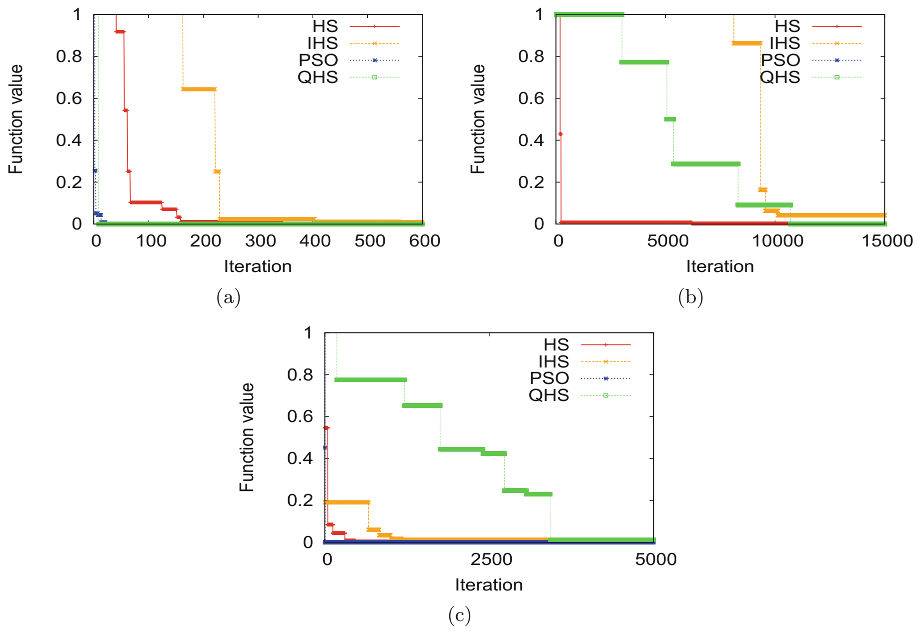

Fig. 1. Convergence analysis considering (a) Sphere, (b) Rosenbrock's and (c) Beale's benchmark functions.

Figure 1b displays the convergence study over Rosenbrock's function. Almost all techniques, except PSO, achieved the global minimum, being HS the fastest to converge, followed by QHS and IHS. Finally, in regard to Beale's function (Fig. 1c), QHS took longer than all techniques to converge, but it obtained the best results so far. In fact, HS-based techniques usually suffer from slow convergence when compared to swarm-based ones, since Harmony Search produces only one new solution at each iteration. We can also observe the convergence behaviour of QHS considering Rosenbrock's and Beale's functions, which follows a "stair pattern", which means QHS can get trapped from local optima (e.g., plateaus in the convergence curve), but also has the ability to escape from them.

Table 3 presents the mean computational load in seconds considering all techniques and functions. Clearly, QHS is the slowest one, since we need an extra loop to access the content of each decision variable (each variable is now represented as a 4-dimensional vector). In fact, for some datasets, QHS can be faster if we employ a different convergence criterion, since we fixed the number of iterations for all techniques in this work.

5.2 Fine-Tuning Restricted Boltzmann Machines

Restricted Boltzmann Machines (RBMs) are stochastic neural networks that aim at learning input representation by means of latent variables placed in a hidden layer [19]. They have become extremely used in the last years due to the "deep learning phenomenon", where we can stack RBMs on top of each other and obtain the so-called Deep Belief Networks.

Table 3. Mean computational load in seconds.

Function	PSO	IHS	HS	Q-HS
Sphere	2.76 ± 1.58	0.44 ± 0.10	0.39 ± 0.06	9.26 ± 1.06
Ackley's	2.12 ± 0.15	0.45 ± 0.11	0.39 ± 0.04	9.49 ± 1.17
Rosebrock's	2.14 ± 0.16	0.45 ± 0.06	0.41 ± 0.05	9.27 ± 1.20
Beale's	2.18 ± 0.14	0.47 ± 0.08	0.41 ± 0.05	9.41 ± 1.15
Matyas	2.16 ± 0.06	0.44 ± 0.07	0.40 ± 0.04	9.48 ± 1.32
Levi	2.51 ± 0.18	0.46 ± 0.076	0.45 ± 0.06	9.51 ± 1.26
Three Hump	2.18 ± 0.23	0.46 ± 0.09	0.40 ± 0.05	9.39 ± 1.16
Easom	2.20 ± 0.23	0.46 ± 0.08	0.40 ± 0.04	9.44 ± 0.91
S. Tang	2.16 ± 0.18	0.47 ± 0.08	0.41 ± 0.04	9.5 ± 0.78
Schaffer	2.21 ± 0.21	0.44 ± 0.09	0.40 ± 0.05	9.09 ± 0.70

Although RBMs have been extensively used in the last years for so many tasks, such as image classification and speech recognition, they are parameter-dependent, which may affect the final results. As far as we are concerned, only a few works have considered optimizing RBMs by means of meta-heuristic techniques [15–17]. In addition, we have not observed any work that attempted at using quaternion-based optimization in the context of RBMs fine-tuning.

In this section, we considered optimizing four parameters concerning the task of binary image reconstruction: $n \in [5, 100]$, $\eta \in [0.1, 0.9]$, $\lambda \in [0.1, 0.9]$ and $\varphi \in [0.00001, 0.01]$. In this case, n stands for the number of hidden neurons, η is the learning rate, λ denotes the weight decay parameter, and φ stands for the momentum [8]. The learning rate is used to control the convergence, and λ and φ are used to avoid oscillations and to keep the values of weights small enough for the learning process.

Therefore, we have a four-dimensional search space with three real-valued variables, as well as the integer-valued number of hidden units. Roughly speaking, the proposed approach aims at selecting the set of RBM parameters that minimizes the minimum squared error (MSE) of the reconstructed images from the training set. After that, the selected set of parameters is thus applied to reconstruct the images of the test set. We employed the well-known MNIST dataset[3], which is composed of images of handwritten digits. The original version contains a training set with 60, 000 images from digits '0'–'9', as well as a test set with 10, 000 images. Due to the computational burden for RBM model selection, we decided to employ the original test set together with a reduced version of the training set (we used 2 % of the original training set).

We conducted a cross-validation with 20 runnings, 10 iterations for the learning procedure of each RBM, and mini-batches of size 20. In addition, we also considered three learning algorithms: Contrastive Divergence (CD) [7],

[3] http://yann.lecun.com/exdb/mnist/.

Persistent Contrastive Divergence (PCD) [20] and Fast Persistent Contrastive Divergence (FPCD) [21]. Finally, the Wilcoxon signed-rank test [22] with significance of 0.05 was used for statistical validation purposes. Table 4 presents the results concerning RBM fine-tuning, where PSO, HS, IHS and a random search (RS) are compared against QHS.

Table 4. Average MSE values considering MNIST dataset.

	CD	PCD	FPCD
PSO	0.1057	0.1058	0.1057
HS	0.1059	0.1325	0.1324
IHS	0.0903	0.0879	**0.0882**
RS	0.1105	0.1101	0.1102
QHS	**0.0876**	**0.0876**	0.0899

One can observe QHS obtained the best results using CD and PCD as the training algorithms, meanwhile IHS obtained the lowest errors using FPCD. As a matter of fact, FPCD is used to control possible deviations of the probability distribution of the input data, and it may require more iterations for convergence. Probably, QHS would require more iterations than IHS for some applications, as displayed in Fig. 1. However, the best results were obtained by means of CD and PCD using the proposed QHS technique.

6 Conclusions

In this paper, we proposed a Harmony Search approach based on quaternions, being inspired by previous works that proposed modifications of the well-known Firefly Algorithm and Bat Algorithm in the same context. The Quaternion Harmony Search was compared against naïve Harmony Search, Improved Harmony Search and Particle Swarm Optimization in 10 benchmark functions, and it has demonstrated to be able to escape from local optima as well as to convergence faster (for some datasets). However, QHS pays the price of a higher computational burden, since it maps each decision variable onto a 4-dimensional space. Therefore, given an optimization problem with N variables, each possible solution can be seen as a tensor of dimensions $4 \times N$ in this space. Also, we demonstrated the validity of quaternion-based optimization in the context of RBM parameter fine-tuning.

In regard to future works, we intend to implement other variants of the Harmony Search using quaternions, as well as other swarm-based meta-heuristics, such as Particle Swarm Optimization and Cuckoo Search. Additionally, we aim at studying the behavior of QHS in high dimensional spaces, and also parallel implementations of quaternion-based optimization techniques in both CPU and GPU devices.

Acknowledgment. The authors would like to thank Capes PROCAD 2966/2014, CNPq grants #306166/2014-3 and #470571/2013-6, and FAPESP grants #2014/16 250-9 and #2015/00801-9.

References

1. Boussaïd, I., Lepagnot, J., Siarry, P.: A survey on optimization metaheuristics. Inf. Sci. **237**(10), 82–117 (2013)
2. Eberly, D.: Quaternion algebra and calculus. Technical report, Magic Software (2002)
3. Fister, I., Brest Jr., J., Fister, I., Yang, X.S.: Modified bat algorithm with quaternion representation. In: IEEE Congress on Evolutionary Computation, pp. 491–498 (2015)
4. Fister, I., Yang, X.S., Breast, J., Fister Jr., I.: Modified firefly algorithm using quaternion representation. Expert Syst. Appl. **40**(18), 7220–7230 (2013)
5. Geem, Z.W.: Music-Inspired Harmony Search Algorithm: Theory and Applications, 1st edn. Springer Publishing Company, Incorporated, Berlin (2009)
6. Hamilton, W.R.: Elements of Quaternions. 2nd edn (1899)
7. Hinton, G.E.: Training products of experts by minimizing contrastive divergence. Neural Comput. **14**(8), 1771–1800 (2002)
8. Hinton, G.E., Li, D., Dong, Y., Dahl, G.E., Mohamed, A., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T.N., Kingsbury, B.: Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups. IEEE Sig. Process. Mag. **29**(6), 82–97 (2012)
9. Humeau, J., Liefooghe, A., Talbi, E.G., Verel, S.: ParadisEO-MO: from fitness landscape analysis to efficient local search algorithms. J. Heuristics **19**(6), 881–915 (2013)
10. Mahdavi, M., Fesanghary, M., Damangir, E.: An improved harmony search algorithm for solving optimization problems. Appl. Math. Comput. **188**(2), 1567–1579 (2007)
11. Malan, K., Engelbrecht, A.: Particle swarm optimisation failure prediction based on fitness landscape characteristics. In: IEEE Symposium on Swarm Intelligence, pp. 1–9 (2014)
12. Merz, P., Freisleben, B.: Fitness landscape analysis and memetic algorithms for the quadratic assignment problem. IEEE Trans. Evol. Comput. **4**(4), 337–352 (2000)
13. Omran, M.G., Mahdavi, M.: Global-best harmony search. Appl. Math. Comput. **198**(2), 643–656 (2008)
14. Pan, Q.K., Suganthan, P., Tasgetiren, M.F., Liang, J.: A self-adaptive global best harmony search algorithm for continuous optimization problems. Appl. Math. Comput. **216**(3), 830–848 (2010)
15. Papa, J.P., Rosa, G.H., Costa, K.A.P., Marana, A.N., Scheirer, W., Cox, D.D.: On the model selection of Bernoulli restricted Boltzmann machines through harmony search. In: Proceedings of the Genetic and Evolutionary Computation Conference, GECCO 2015, pp. 1449–1450. ACM, New York (2015)
16. Papa, J.P., Rosa, G.H., Marana, A.N., Scheirer, W., Cox, D.D.: Model selection for discriminative restricted Boltzmann machines through meta-heuristic techniques. J. Comput. Sci. **9**, 14–18 (2015)
17. Papa, J.P., Scheirer, W., Cox, D.D.: Fine-tuning deep belief networks using harmony search. Appl. Soft Comput. **46**, 875–885 (2015)

18. Pereira, D.R., Delpiano, J., Papa, J.P.: On the optical flow model selection through metaheuristics. EURASIP J. Image Video Process. **2015**, 11 (2015). http://dx.doi.org/10.1186/s13640-015-0066-5
19. Smolensky, P.: Information processing in dynamical systems: foundations of harmony theory. In: Parallel Distributed Processing: Explorations in the Microstructure of Cognition, vol. 1, pp. 194–281. MIT Press, Cambridge (1986)
20. Tieleman, T.: Training restricted Boltzmann machines using approximations to the likelihood gradient. In: Proceedings of the 25th International Conference on Machine Learning, pp. 1064–1071. ACM, New York (2008)
21. Tieleman, T., Hinton, G.E.: Using fast weights to improve persistent contrastive divergence. In: Proceedings of the 26th Annual International Conference on Machine Learning, pp. 1033–1040. ACM, New York (2009)
22. Wilcoxon, F.: Individual comparisons by ranking methods. Biom. Bull. **1**(6), 80–83 (1945)
23. Yang, X.S.: Nature-Inspired Metaheuristic Algorithms, 2nd edn. Luniver Press, Beckington (2010)