Soft Computing
https://doi.org/10.1007/s00500-019-03887-7

METHODOLOGIES AND APPLICATION

Adaptive $\beta-$hill climbing for optimization

Mohammed Azmi Al-Betar[1] · Ibrahim Aljarah[2] · Mohammed A. Awadallah[3] · Hossam Faris[2] · Seyedali Mirjalili[4]

Abstract

In this paper, an adaptive version of $\beta-$hill climbing is proposed. In the original $\beta-$hill climbing, two control parameters are utilized to strike the right balance between a local-nearby exploitation and a global wide-range exploration during the search: $\mathcal{N}$ and β, respectively. Conventionally, these two parameters require an intensive study to find their suitable values. In order to yield an easy-to-use optimization method, this paper proposes an efficient adaptive strategy for these two parameters in a deterministic way. The proposed adaptive method is evaluated against 23 global optimization functions. The selectivity analysis to determine the optimal progressing values of $\mathcal{N}$ and β during the search is carried out. Furthermore, the behavior of the adaptive version is analyzed based on various problems with different complexity levels. For comparative evaluation, the adaptive version is initially compared with the original one as well as with other local search-based methods and other well-regarded methods using the same benchmark functions. Interestingly, the results produced are very competitive with the other methods. In a nutshell, the proposed adaptive $\beta-$hill climbing is able to achieve the best results on 10 out of 23 test functions. For more validation, the test functions established in IEEE-CEC2015 are used with various scaling values. The comparative results show the viability of the proposed adaptive method.

Keywords Metaheuristics · β-hill climbing · Global optimization · Control parameters

1 Introduction

The decision-making field is mostly concern with an optimization method to achieve efficient outcomes. Conventionally, optimization problems are tackled by searching for the "optimal" configuration of the problem decision variables as evaluated by well-regarded objective function to be minimized or maximized (Osman and Laporte 1996). In order to efficiently tackle an optimization problem, two main steps should be carefully defined: the solution representation and the objective function (BoussaïD et al. 2013). By means of these steps, the context of the problem will be successfully incorporated with the method framework. It is commonly known that the optimization methods iterate toward an optimal solution using different operators. Some of these operators are concerned with exploration while others are concerned with exploitation (Al-Betar et al. 2014). Exploration means the ability of the algorithm to navigate different search space regions during the search, while exploitation is the capability to making use of the historical search. The operators are often controlled by parameters which take a probability value that strike the suitable trade-off between exploration and exploitation search concepts (Sörensen 2015).

Communicated by V. Loia.

✉ Mohammed Azmi Al-Betar
mohbetar@bau.edu.jo

Ibrahim Aljarah
i.aljarah@ju.edu.jo

Mohammed A. Awadallah
ma.awadallah@alaqsa.edu.ps

Hossam Faris
hossam.faris@ju.edu.jo

Seyedali Mirjalili
seyedali.mirjalili@griffithuni.edu.au

1 Department of Information Technology, Al-Huson University College, Al-Balqa Applied University, Irbid, Jordan

2 King Abdullah II School for Information Technology, The University of Jordan, Amman, Jordan

3 Department of Computer Science, Al-Aqsa University, P.O. Box 4051, Gaza, Palestine

4 Institute for Integrated and Intelligent Systems, Griffith University, Nathan Campus, Brisbane, QLD 4111, Australia

Published online: 09 March 2019

Metaheuristics are the most recent optimization methods developed as a general framework that can be adapted for various types of optimization problems. They are categorized into three main classes (Blum and Roli 2003): population-based methods, local-search methods, and swarm-based methods. Local search methods, which is the focal point of this paper, start with a single initial solution. At each iteration, the current solution will be *moved* to its neighboring solution which is in the same region of the initial one. The neighboring solution will replace the current one, if it is better. This process will be repeated until the local optimal solution which is the peak point in the same region of the initial solution is reached. The most popular local search-based methods are hill climbing, β-hill climbing (Al-Betar 2017), simulated annealing (Kirkpatrick et al. 1983), Tabu search (Glover 1986), VNS (Hansen and Mladenovi'c 1999), GRASP (Feo and Resende 1995), and ILS (Lourenço et al. 2003).

β-hill climbing is a recent local search-based algorithm designed by Al-Betar (2017). It is simple, flexible, scalable, and adaptable local search that can be able to navigate the problem search space using two operators: $\mathcal{N}$-operator which is the source of exploitation and β operator which is the source of exploration. The $\mathcal{N}$-operator performs a neighboring search around the current solution based on parameter $\mathcal{N}$ probability. Note that this parameter determine the distance bandwidth between the current solution and its neighboring solution. In contrast, β operator performs a stochastic search controlled by the parameter β, in which, as the probability of β increases, as the exploration will increase. Due to its elegant characteristics, β-hill climbing algorithm is successfully tailored to several optimization problems such as suduko (Al-Betar et al. 2017), multiple-reservoir scheduling (Alsukni et al. 2017), text clustering (Abualigah et al. 2017a), ECG and EEG signal denoising (Alyasseri et al. 2018, 2017), and feature selection (Abualigah et al. 2017b).

The parameter settings is an important research track in the optimization field. This research topic is concerned with the effect of the parameter settings on the convergence behavior of the optimization algorithm. The parameter settings are categorized into two main classes (Eiben et al. 1999; Wessing et al. 2011): parameter tuning and parameter control. Parameter tuning initializes the values of the parameters in the initial stage and keep them constant during the course of run. This class of parameter setting, although conventional in the domain, requires an exhaustive sensitivity parameter analysis to find the optimal values used. For instance in β-hill climbing (Al-Betar 2017), the experimental results on the IEEE-CEC2005 benchmarks are obtained after an exhaustive parameter tuning. On the other hand, the control parameter determines the values of the method parameters in the initial search, and these parameters are adapted during the course of run to cope with the complexity of the search process (Eiben et al. 1999; Aleti and Moser 2016).

Normally, the control parameter class is divided into three subclasses: deterministic, adaptive, and self-adaptive. The deterministic control parameters update the value of the parameter during the search based on deterministic rule which usually considers the time-varying schedule (Mezura-Montes and Palomeque-Ortiz 2009). The adaptive parameter control updates the values of parameters adaptively during the search based on a feedback from the search which determines the magnitude of the change to the targeted parameter (Dragoi and Dafinescu 2016; Aleti and Moser 2016; Noorbin and Alfi 2018; Wang et al. 2017; Guo et al. 2018). Finally, the self-adaptive parameter control determines the adaptive parameter based on another heuristic or metaheuristic method that finds the optimal parameter values at each iteration (Angeline and Angeline 1995; Aleti and Moser 2016; Xue et al. 2018; Guo et al. 2017).

In this paper, a parameter control strategy based on the deterministic concept is proposed for β-hill climbing algorithm. As aforementioned, the two parameters ($\mathcal{N}$ and β) in the β-hill climbing are responsible for striking the balance between exploitation and exploration. The two parameters are updated during the run based on a deterministic rule that takes the iteration index or time search into account. The main motivation behind adapting deterministically the parameters of β-hill climbing algorithm is that the adaptive version is easy to use for beginner optimization users. Furthermore, it can be considered as a black-box optimizer where no intensive pre-experiments are required to determine the suitable parameters for any optimization problems. The new method is called adaptive β-hill climbing algorithm which is evaluated against 23 benchmark functions circulated in the literature. The results stress the efficiency of the proposed approach. For further evaluation, the test function established in IEEE-CEC2015 are used with various scaling values. The comparative results show the viability of the proposed adaptive method. In short, the new adaptive algorithm allows the users to implement a new efficient algorithm as a black box without prior knowledge about its parameter settings.

The paper is organized as follows: The presentation of the β-hill climbing algorithm is given in Sect. 2. The proposed adaptive β-hill climbing is explained in Sect. 3. Next, experiments and comparative results are presented in Sect. 4. Finally, the conclusion and some future directions are given in Sect. 5.

2 β-hill climbing algorithm

Recently, a β-hill climbing (βHC) algorithm, new version of the simple local search, was introduced by Al-Betar (2017) for continuous optimization benchmark functions. It has shown very successful results for IEEE-CEC2005 datasets (Suganthan et al. 2005). The main advantages of the β-hill climbing is its ability to navigating the wide search space regions with the hope of finding the global optima. The process of exploring the promising search space regions is performed by using β operator controlled by the β parameter. The β operator behavior is very tricky, where the decision variable values of the current solution are tested for whether or not to be replaced by random values from their feasible ranges.

Technically, β-hill climbing algorithm, as other local search methods, starts with a provisional solution, $\boldsymbol{x} = (x_1, x_2, \ldots, x_N)$. At each iteration, the current solution will be moved to a new neighboring solution $\boldsymbol{x}' = (x_1', x_2', \ldots, x_N')$ using two operators: $\mathcal{N}$-operator and β operator. $\mathcal{N}$-operator performs a neighborhood search of a random walk which is concerned with exploitation, while the work of β operator is similar to uniform mutation operator which is concerned with exploration. Iteration by iteration, the solution can be improved by $\mathcal{N}$-operator stage or β operator stage (or both) until the stop condition is met.

For more explanation, assume that the optimization problem is represented as below:

$$\min\{f(\boldsymbol{x}) \mid \boldsymbol{x} \in \mathbf{X}\},$$

where $f(\boldsymbol{x})$ is the objective function which evaluates the solution $\boldsymbol{x} = (x_1, x_2, \ldots, x_N)$ that involves a set of decision variables. Each decision variable $x_i \in \boldsymbol{X}_i$ where $\mathbf{X} = \{\boldsymbol{X}_i \mid i = 1, \ldots, N\}$ is the possible value range for each decision variable. Note that the $\boldsymbol{X}_i \in [LB_i, UB_i]$ and LB_i and UB_i are the lower and upper bounds for the decision variable x_i, respectively, and N is the total number of decision variables.

Initially, a random solution is generated and evaluated using the objective function $f(\boldsymbol{x})$. The solution is often iteratively improved using $\mathcal{N}$-operator where the function $improve(\mathcal{N}(\boldsymbol{x}))$ is used with the acceptance rule called 'random walk' (Al-Betar 2017) in which a random neighboring solution of the current solution $\boldsymbol{x}$ is visited as follows:

$$x_i' = x_i \pm U(0,1) \times \mathcal{N} \quad \exists i \in [1, N] \tag{1}$$

where i is randomly chosen from the feasible range, $i \in [1, 2, \ldots, N]$. The parameter $\mathcal{N}$ is the distance bandwidth between the current solution and the neighboring solution.

β operator Here, the current solution is the neighboring solution resulting from the $\mathcal{N}$-operator. New solution variables $\boldsymbol{x}''$ are assigned by values relayed on the existing values of the current solution or randomly from available range within the range of β parameter, where $\beta \in [0, 1]$ as follows:

$$x_i'' \leftarrow \begin{cases} x_k & ra \le \beta \\ x_i' & \text{otherwise.} \end{cases}$$

where ra generates a uniform random number between 0 and 1. $x_k \in \boldsymbol{X}_i$ is randomly selected from the possible range for the decision variable x_i'. Algorithm 1 shows the pseudocode of the β-hill climbing.

Algorithm 1 β-hill climbing pseudocode

```
x_i = LB_i + (UB_i − LB_i) × U(0,1), ∀i = 1, 2, ..., N      {The initial solution x}
Calculate(f(x))
t = 0
while (t ≤ Max_t) do
    x' = improve(N(x))      {N-operator}
    x'' = x'
    for i = 1, ··· , N do
        if (ra ≤ β) then
            x''_i = x_k
        end if    {ra ∈ [0, 1]}
    end for    {β operator}
    if (f(x'') ≤ f(x)) then
        x = x''
        f(x) = f(x'')
    end if
    t = t + 1
end while
```

3 Adaptive β-hill climbing (AβHC)

Inspired from the deterministic adaptive parameter concepts, in this paper, the two main parameters ($\mathcal{N}$ and β) of β-hill climbing algorithm are updated in the search based on Eqs. (2) and (4).

The $\mathcal{N}$ and β parameters introduced in $\mathcal{N}$-operator and β operator of β-hill climbing help the algorithm to control the rate of exploitation and exploration, respectively. The rules of these parameters are very important in fine-tuning the solution vector during the search process, and therefore, the convergence rate will be improved.

The original version of β-hill climbing use the aspects of parameter tuning in which the setting values of $\mathcal{N}$ and β parameters are initiated in advance and fixed during the search. The first drawback of parameter-tuning process is that an intensive sensitivity analysis is required to determine the suitable parameter values for each kind of any optimization problem. The second drawback is the convergence rate where the search requires a large number of iterations to find the optimal solution.

As aforementioned, the $\mathcal{N}$ parameter is used to determine the distance bandwidth between the current solution and the neighboring solution as formulated in Eq. (1). A large value of $\mathcal{N}$ means a wider distance is achieved. This means that the search will diversify the solution widely, and therefor, the process of exploiting the good element of the current solution will be not guaranteed. However, the diversification is conventionally required in the initial search and should be gradually minimized during the search. Therefor, in the proposed method, the $\mathcal{N}$ parameter is deterministically updated during the search which starts with a value closed to one

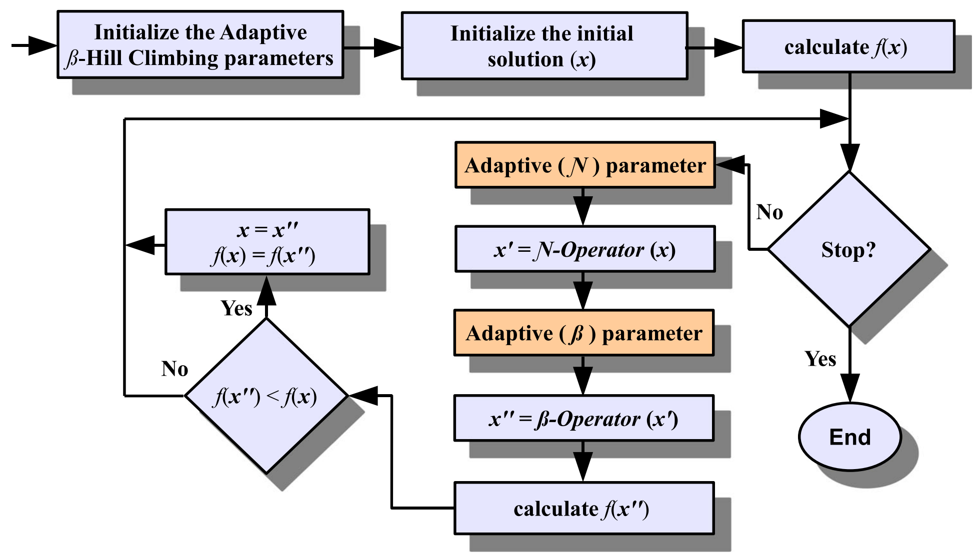

Fig. 1 Flowchart of adaptive β-hill climbing algorithm

and reduced during the search based on the varying time (or iteration number) as introduced in (Mirjalili et al. 2016) as follows:

$$\mathcal{N}_t = 1 - C_t \tag{2}$$

$$C_t = \frac{t^{\frac{1}{K}}}{\text{Max\_t}^{\frac{1}{K}}} \tag{3}$$

where $\mathcal{N}_t$ is the value of $\mathcal{N}$ at time t. K is a fixed number used to gradually degrade the value of $\mathcal{N}$ to a number close to 0 in the final course of run. Max_t is the maximum number of iterations.

Furthermore, in the proposed method, the value of β parameter is deterministically adapted within a specific range of $[\beta_{\min}, \beta_{\max}]$ as used in (Mahdavi et al. 2007) as follows:

$$\beta_t = \beta_{\min} + t \times \frac{\beta_{\max} - \beta_{\min}}{\text{Max\_t}} \tag{4}$$

where

β_t is the value of β rate at time t.
$\beta_{\min}$ is the minimum value of β rate.
$\beta_{\max}$ is the maximum value of β rate.
Max_t is the maximum number of iterations.
t is the current time.

Algorithm 2 shows the pseudocode of the proposed adaptive β-hill climbing algorithm. Also, the flowchart of the proposed method is drawn in Fig. 1

Algorithm 2 Adaptive β-hill climbing pseudocode

Initialize β_{min}, $\beta_{\max}$, and K
$x_i = LB_i + (UB_i - LB_i) \times U(0,1), \forall i = 1, 2, \ldots, N$ {The initial solution $\boldsymbol{x}$}
Calculate($f(\boldsymbol{x})$)
$t = 0$
while $(t \leq \text{Max\_t})$ **do**
  $\boldsymbol{x}' = \boldsymbol{x}$
  $C_t = \frac{t^{\frac{1}{K}}}{\text{Max\_t}^{\frac{1}{K}}}$
  $\mathcal{N}_t = 1 - C_t$ {Adaptive $\mathcal{N}$}
  $RndIndex \in (1, N)$
  $x'_{RndIndex} = x'_{RndIndex} \pm \mathcal{N}_t$
  $\boldsymbol{x}'' = \boldsymbol{x}'$
  $\beta_t = \beta_{min} + t \times \frac{\beta_{\max} - \beta_{min}}{\text{Max\_t}}$ {Adaptive β}
  for $i = 1, \cdots, N$ **do**
    if $(ra \leq \beta_t)$ **then**
      $x''_i = x_k$
    end if $\{ra \in [0,1]\}$
  end for
  if $(f(\boldsymbol{x}'') \leq f(\boldsymbol{x}))$ **then**
    $\boldsymbol{x} = \boldsymbol{x}''$
    $f(\boldsymbol{x}) = f(\boldsymbol{x}'')$
  end if
  $t = t + 1$
end while

4 Experiments and results

This section is expressed to evaluate the performance of the proposed adaptive method and to compare its results with the original β-hill climbing as well as the other comparative methods from the literature using the same benchmark datasets discussed in Sect. 4.1.

The conducted experiments are executed on a desktop computer with 2.66 Intel Core i7 with 16GB RAM. The operating system used is Microsoft windows 10. The program is coded using MATLAB Version 7.6.0.324(R2008a).

The sensitivity analysis for the proposed method is conducted. The experiments include studying the effect of the $\mathcal{N}$

Table 1 Description of unimodal benchmark functions

Function	Dimensions	Range	$f_{\min}$
$f_1(x) = \sum_{i=1}^{n} x_i^2$	10,30,50,100	[−100,100]	0
$f_2(x) = \sum_{i=1}^{n} \lvert x_i\rvert + \prod_{i=1}^{n} \lvert x_i\rvert$	10,30,50,100	[−10,10]	0
$f_3(x) = \sum_{i=1}^{n} \left(\sum_{j-1}^{i} x_j\right)^2$	10,30,50,100	[−100,100]	0
$f_4(x) = \max_i \{\lvert x_i\rvert, 1 \le i \le n\}$	10,30,50,100	[−100,100]	0
$f_5(x) = \sum_{i=1}^{n-1} \left[100\left(x_{i+1} - x_i^2\right)^2 + (x_i - 1)^2\right]$	10,30,50,100	[−30,30]	0
$f_6(x) = \sum_{i=1}^{n} ([x_i + 0.5])^2$	10,30,50,100	[−100,100]	0
$f_7(x) = \sum_{i=1}^{n} i x_i^4 + \text{random}[0, 1)$	10,30,50,100	[−128,128]	0

Table 2 Description of multimodal benchmark functions

Function	Dimensions	Range	$f_{\min}$
$f_8(x) = \sum_{i=1}^{n} -x_i \sin\left(\sqrt{\lvert x_i\rvert}\right)$	10,30,50,100	[−500,500]	−418.9829 × 5
$f_9(x)=\sum_{i=1}^{n}\left[x_i^2 - 10\cos(2\pi x_i) + 10\right]$	10,30,50,100	[−5.12,5.12]	0
$f_{10}(x) = -20\exp(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}) - \exp\left(\frac{1}{n}\sum_{i=1}^{n} cos\,(2\pi x_i)\right) + 20 + e$	10,30,50,100	[−32,32]	0
$f_{11}(x) = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n}\cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	10,30,50,100	[−600,600]	0
$f_{12}(x) = \frac{\pi}{n}\left\{10\sin(\pi y_1) + \sum_{i=1}^{n-1}(y_i - 1)^2\left[1 + 10\sin^2(\pi y_{i+1})\right] + (y_n - 1)^2\right\} + \sum_{i=1}^{n} u(x_i, 10, 100, 4)$ $y_i = 1 + \frac{x_i+1}{4} u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & x_i > a \\ 0 - a & < x_i < a \\ k(-x_i - a)^m & x_i < -a \end{cases}$	10,30,50,100	[−50,50]	0
$f_{13}(x) = 0.1\left\{\sin^2(3\pi x_1) + \sum_{i=1}^{n}(x_i - 1)^2\left[1 + \sin^2(3\pi x_i + 1)\right] + (x_n - 1)^2\left[1 + \sin^2(2\pi x_n)\right]\right\} + \sum_{i=1}^{n} u(x_i, 5, 100, 4)$	10,30,50,100	[−50,50]	0

Table 3 Description of fixed-dimension multimodal benchmark functions

Function	Dimensions	Range	$f_{\min}$
$f_{14}(x) = \left(\frac{1}{500} + \sum_{j=1}^{25}\frac{1}{j + \sum_{i=1}^{2}(x_i - a_{ij})^6}\right)^{-1}$	2	[−65, 65]	1
$f_{15}(x) = \sum_{i=1}^{11}\left[a_i - \frac{x_1(b_i^2 + b_i x_2)}{b_i^2 + b_i x_3 + x_4}\right]^2$	4	[−5, 5]	0.00030
$f_{16}(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	2	[−5, 5]	−1.0316
$f_{17}(x) = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos x_1 + 10$	2	[−5, 5]	0.398
$f_{18}(x) = \left[1 + (x_1 + x_2 + 1)^2\left(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2\right)\right] \times \left[30 + (2x_1 - 3x_2)^2 \times \left(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2\right)\right]$	2	[−2, 2]	3
$f_{19}(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{3} a_{ij}\left(x_j - p_{ij}\right)^2\right)$	3	[1, 3]	−3.86
$f_{20}(x) = -\sum_{i=1}^{4} c_i \exp\left(-\sum_{j=1}^{6} a_{ij}\left(x_j - p_{ij}\right)^2\right)$	6	[0, 1]	−3.32
$f_{21}(x) = -\sum_{i=1}^{5}\left[(X - a_i)(X - a_i)^T + c_i\right]^{-1}$	4	[0, 10]	−10.1532
$f_{22}(x) = -\sum_{i=1}^{7}\left[(X - a_i)(X - a_i)^T + c_i\right]^{-1}$	4	[0, 10]	−10.4028
$f_{23}(x) = -\sum_{i=1}^{10}\left[(X - a_i)(X - a_i)^T + c_i\right]^{-1}$	4	[0.10]	−10.5363

Table 4 Results of AβHC with different K values

Benchmark	$K=2$ AVE (SD)	$K=4$ AVE (SD)	$K=6$ AVE (SD)	$K=8$ AVE (SD)	$K=10$ AVE (SD)	$K=12$ AVE (SD)	$K=14$ AVE (SD)	$K=16$ AVE (SD)
f_1	3.86E−05	1.61E−05	1.06E−05	6.32E−06	4.81E−06	2.97E−06	2.30E−06	1.93E−06
	(1.83E−05)	(6.75E−06)	(4.48E−06)	(2.59E−06)	(2.46E−06)	(9.44E−07)	(8.39E−07)	(9.25E−07)
f_2	2.18E−02	1.46E−02	1.09E−02	8.98E−03	6.94E−03	6.19E−03	5.48E−03	4.73E−03
	(4.09E−03)	(2.20E−03)	(1.99E−03)	(1.72E−03)	(1.33E−03)	(1.06E−03)	(9.43E−04)	(7.16E−04)
f_3	3.69E+01	**2.78E+01**	4.88E+01	1.66E+02	3.87E+02	6.47E+02	8.91E+02	1.02E+03
	(1.28E+01)	(1.87E+01)	(5.70E+01)	(1.71E+02)	(2.42E+02)	(3.67E+02)	(4.17E+02)	(5.61E+02)
f_4	**8.41E−01**	1.60E+00	1.89E+00	2.17E+00	2.19E+00	2.23E+00	2.17E+00	2.31E+00
	(3.91E−01)	(4.00E−01)	(4.62E−01)	(4.47E−01)	(3.86E−01)	(4.77E−01)	(3.39E−01)	(4.50E−01)
f_5	5.70E+01	1.23E+02	5.90E+01	4.85E+01	5.94E+01	5.10E+01	**4.09E+01**	4.84E+01
	(6.01E+01)	(1.35E+02)	(6.28E+01)	(5.29E+01)	(4.83E+01)	(4.31E+01)	(4.69E+01)	(4.60E+01)
f_6	3.63E−05	1.80E−05	9.89E−06	6.34E−06	4.79E−06	3.66E−06	2.24E−06	2.52E−06
	(1.68E−05)	(8.34E−06)	(4.19E−06)	(3.93E−06)	(2.16E−06)	(2.13E−06)	(8.02E−07)	(1.22E−06)
f_7	2.43E−02	1.76E−02	1.36E−02	1.29E−02	1.16E−02	**1.00E−02**	1.07E−02	1.02E−02
	(8.13E−03)	(5.60E−03)	(4.71E−03)	(3.90E−03)	(3.93E−03)	(3.67E−03)	(4.39E−03)	(3.91E−03)
f_8	**−1.26E+04**	**−1.26E+04**	**−1.26E+04**	**−1.26E+04**	**−1.26E+04**	**−1.26E+04**	**−1.26E+04**	**−1.26E+04**
	(1.19E+02)	(9.32E+01)	(8.59E+01)	(4.78E+01)	(2.52E+01)	(2.58E−07)	(2.52E+01)	(1.59E−07)
f_9	2.56E+00	1.35E+00	6.76E−01	2.55E−01	4.73E−02	7.10E−02	2.78E−02	1.29E−03
	(1.41E+00)	(1.20E+00)	(6.88E−01)	(4.41E−01)	(1.83E−01)	(2.52E−01)	(1.26E−01)	(1.11E−03)
f_{10}	5.05E−03	7.76E−03	4.75E−03	3.22E−03	2.53E−03	2.01E−03	1.76E−03	1.50E−03
	(8.12E−04)	(4.11E−03)	(1.89E−03)	(7.89E−04)	(5.92E−04)	(4.75E−04)	(3.33E−04)	(2.76E−04)
f_{11}	2.18E−01	1.99E−01	1.77E−01	1.50E−01	1.40E−01	1.49E−01	1.27E−01	1.30E−01
	(8.58E−02)	(1.29E−01)	(9.30E−02)	(9.21E−02)	(7.23E−02)	(1.01E−01)	(1.08E−01)	(1.00E−01)
f_{12}	5.49E−06	1.58E−06	1.12E−06	8.45E−07	4.23E−07	5.51E−07	2.88E−07	1.58E−07
	(8.42E−06)	(2.71E−06)	(1.52E−06)	(1.76E−06)	(6.56E−07)	(7.12E−07)	(6.58E−07)	(1.79E−07)
f_{13}	3.83E−05	1.80E−05	**7.44E−06**	7.90E−06	1.12E−03	1.98E−03	3.07E−03	4.12E−03
	(4.08E−05)	(2.32E−05)	(1.10E−05)	(1.40E−05)	(3.35E−03)	(4.45E−03)	(8.59E−03)	(7.59E−03)
f_{14}	**9.98E−01**	**9.98E−01**	**9.98E−01**	**9.98E−01**	**9.98E−01**	**9.98E−01**	**9.98E−01**	**9.98E−01**
	(3.98E−16)	(3.14E−16)	(3.09E−16)	(3.49E−16)	(3.63E−16)	(3.29E−16)	(3.18E−16)	(3.23E−16)
f_{15}	1.12E−03	1.04E−03	1.26E−03	9.95E−04	1.16E−03	1.19E−03	1.26E−03	1.12E−03
	(4.81E−04)	(5.44E−04)	(5.03E−04)	(5.81E−04)	(5.63E−04)	(4.80E−04)	(5.34E−04)	(4.92E−04)
f_{16}	**−1.03E+00**	**−1.03E+00**	**−1.03E+00**	**−1.03E+00**	**−1.03E+00**	**−1.03E+00**	**−1.03E+00**	**−1.03E+00**
	(1.72E−11)	(2.15E−12)	(4.60E−13)	(1.07E−12)	(2.60E−13)	(3.69E−13)	(2.30E−13)	(1.51E−13)
f_{17}	**3.98E−01**	**3.98E−01**	**3.98E−01**	**3.98E−01**	**3.98E−01**	**3.98E−01**	**3.98E−01**	**3.98E−01**
	(1.19E−11)	(3.48E−12)	(4.56E−12)	(4.50E−13)	(4.04E−13)	(5.26E−13)	(8.94E−13)	(3.73E−13)
f_{18}	**3.00E+00**	**3.00E+00**	**3.00E+00**	3.00E+00	**3.00E+00**	**3.00E+00**	**3.00E+00**	**3.00E+00**
	(8.66E−10)	(1.41E−10)	(1.21E−10)	(5.44E−11)	(1.83E−11)	(4.64E−11)	(2.36E−11)	(7.03E−12)
f_{19}	**−3.86E+00**	**−3.86E+00**	**−3.86E+00**	**−3.86E+00**	**−3.86E+00**	**−3.86E+00**	**−3.86E+00**	**−3.86E+00**
	(7.71E−09)	(7.34E−10)	(1.52E−10)	(2.56E−10)	(7.43E−11)	(7.63E−11)	(1.82E−11)	(2.05E−11)
f_{20}	**−3.30E+00**	−3.28E+00	−3.27E+00	−3.28E+00	−3.28E+00	−3.29E+00	−3.28E+00	−3.27E+00
	(4.51E−02)	(5.70E−02)	(5.92E−02)	(5.70E−02)	(5.70E−02)	(5.54E−02)	(5.83E−02)	(6.03E−02)
f_{21}	−7.81E+00	−6.54E+00	−6.56E+00	−6.48E+00	−6.31E+00	−5.97E+00	**−8.90E+00**	−6.06E+00
	(3.22E+00)	(3.34E+00)	(3.51E+00)	(3.58E+00)	(3.33E+00)	(3.17E+00)	(2.61E+00)	(3.31E+00)
f_{22}	−5.66E+00	−6.68E+00	−6.44E+00	−6.96E+00	−7.34E+00	−6.90E+00	−6.46E+00	−7.34E+00
	(3.04E+00)	(3.24E+00)	(3.37E+00)	(3.54E+00)	(3.39E+00)	(3.39E+00)	(3.34E+00)	(3.39E+00)
f_{23}	−5.80E+00	−6.02E+00	**−7.81E+00**	−7.76E+00	−6.90E+00	−6.26E+00	−7.38E+00	−6.40E+00
	(3.23E+00)	(3.32E+00)	(3.24E+00)	(3.29E+00)	(3.33E+00)	(3.12E+00)	(3.48E+00)	(3.05E+00)

Table 4 continued

Benchmark	$K = 18$ AVE (SD)	$K = 20$ AVE (SD)	$K = 22$ AVE (SD)	$K = 24$ AVE (SD)	$K = 26$ AVE (SD)	$K = 28$ AVE (SD)	$K = 30$ AVE (SD)
f_1	1.57E−06	1.44E−06	1.16E−06	9.91E−07	8.67E−07	8.27E−07	**6.91E−07**
	(7.82E−07)	(5.17E−07)	(4.81E−07)	(4.37E−07)	(4.59E−07)	(3.90E−07)	(3.56E−07)
f_2	4.43E−03	4.04E−03	3.66E−03	3.31E−03	3.26E−03	2.89E−03	**2.74E−03**
	(8.55E−04)	(8.35E−04)	(7.78E−04)	(5.49E−04)	(5.22E−04)	(6.03E−04)	(5.08E−04)
f_3	1.11E+03	1.29E+03	1.26E+03	1.49E+03	1.55E+03	1.47E+03	1.56E+03
	(5.23E+02)	(5.75E+02)	(4.82E+02)	(6.22E+02)	(5.04E+02)	(6.28E+02)	(6.57E+02)
f_4	2.37E+00	2.39E+00	2.33E+00	2.45E+00	2.38E+00	2.23E+00	2.35E+00
	(3.77E−01)	(4.41E−01)	(3.53E−01)	(3.34E−01)	(4.33E−01)	(2.96E−01)	(4.59E−01)
f_5	6.52E+01	9.11E+01	7.58E+01	9.01E+01	7.22E+01	5.52E+01	4.13E+01
	(9.71E+01)	(1.95E+02)	(1.32E+02)	(1.21E+02)	(1.16E+02)	(7.28E+01)	(4.17E+01)
f_6	1.62E−06	1.28E−06	1.15E−06	1.03E−06	9.29E−07	8.98E−07	**8.23E−07**
	(8.14E−07)	(5.22E−07)	(4.40E−07)	(4.28E−07)	(5.04E−07)	(4.30E−07)	(4.29E−07)
f_7	1.07E−02	1.09E−02	1.12E−02	1.03E−02	1.27E−02	1.25E−02	1.52E−02
	(3.72E−03)	(3.96E−03)	(3.54E−03)	(4.18E−03)	(4.52E−03)	(3.93E−03)	(4.41E−03)
f_8	**−1.26E+04**	**−1.26E+04**	**−1.26E+04**	**−1.26E+04**	**−1.26E+04**	**−1.26E+04**	**−1.26E+04**
	(1.22E−07)	(3.56E−07)	(1.18E−07)	(1.96E−07)	(5.29E−08)	(1.30E−07)	(7.93E−08)
f_9	1.72E−03	8.00E−04	4.81E−04	5.28E−04	3.26E−04	**2.84E−04**	3.11E−04
	(3.73E−03)	(7.67E−04)	(2.92E−04)	(3.48E−04)	(1.88E−04)	(1.41E−04)	(3.37E−04)
f_{10}	1.42E−03	1.25E−03	1.13E−03	1.03E−03	9.23E−04	8.81E−04	**7.98E−04**
	(3.63E−04)	(4.23E−04)	(2.44E−04)	(2.42E−04)	(3.25E−04)	(2.24E−04)	(1.63E−04)
f_{11}	1.26E−01	1.31E−01	8.97E−02	1.42E−01	9.77E−02	**8.61E−P02**	1.05E−01
	(9.74E−02)	(8.28E−02)	(4.73E−02)	(9.64E−02)	(7.58E−02)	(7.14E−02)	(7.28E−02)
f_{12}	1.07E−07	1.70E−07	1.70E−07	1.08E−07	1.14E−07	7.66E−04	**4.93E−08**
	(1.95E−07)	(2.19E−07)	(2.27E−07)	(1.52E−07)	(3.73E−07)	(4.19E−03)	(9.71E−08)
f_{13}	5.11E−03	3.33E−03	4.74E−03	4.00E−03	2.99E−03	5.13E−03	4.60E−03
	(9.39E−03)	(5.11E−03)	(6.15E−03)	(6.01E−03)	(4.91E−03)	(9.00E−03)	(5.40E−03)
f_{14}	**9.98E−01**	**9.98E−01**	**9.98E−01**	**9.98E−01**	**9.98E−01**	**9.98E−01**	**9.98E−01**
	(3.18E−16)	(2.16E−16)	(2.84E−16)	(3.13E−16)	(2.78E−16)	(2.78E−16)	(2.66E−16)
f_{15}	1.11E−03	1.11E−03	1.16E−03	1.12E−03	1.28E−03	**9.94E−04**	1.20E−03
	(5.13E−04)	(5.70E−04)	(4.91E−04)	(6.13E−04)	(5.59E−04)	(5.39E−04)	(5.06E−04)
f_{16}	**−1.03E+00**	**−1.03E+00**	**−1.03E+00**	**−1.03E+00**	**−1.03E+00**	**−1.03E+00**	**−1.03E+00**
	(7.93E−14)	(1.95E−13)	(1.48E−13)	(1.34E−13)	(1.27E−13)	(7.47E−14)	(6.02E−14)
f_{17}	**3.98E−01**	**3.98E−01**	**3.98E−01**	**3.98E−01**	**3.98E−01**	**3.98E−01**	**3.98E−01**
	(3.88E−13)	(7.21E−14)	(2.78E−13)	(2.60E−14)	(2.35E−13)	(1.01E−13)	(1.61E−13)
f_{18}	**3.00E+00**	**3.00E+00**	**3.00E+00**	**3.00E+00**	**3.00E+00**	**3.00E+00**	**3.00E+00**
	(6.69E−12)	(5.20E−12)	(5.61E−12)	(4.09E−12)	(2.89E−12)	(2.72E−12)	(3.25E−12)
f_{19}	**−3.86E+00**	**−3.86E+00**	**−3.86E+00**	**−3.86E+00**	**−3.86E+00**	**−3.86E+00**	**−3.86E+00**
	(4.86E−11)	(2.91E−11)	(1.10E−11)	(1.17E−11)	(1.00E−11)	(1.35E−11)	(8.84E−12)
f_{20}	−3.29E+00	**−3.30E+00**	−3.28E+00	−3.29E+00	−3.29E+00	−3.27E+00	−3.27E+00
	(5.35E−02)	(4.51E−02)	(5.83E−02)	(5.11E−02)	(5.35E−02)	(5.92E−02)	(5.99E−02)
f_{21}	−6.72E+00	−5.55E+00	−6.98E+00	−6.40E+00	−5.97E+00	−7.54E+00	−6.47E+00
	(3.57E+00)	(3.22E+00)	(3.53E+00)	(3.64E+00)	(3.17E+00)	(3.14E+00)	(3.39E+00)
f_{22}	−6.58E+00	−6.79E+00	−5.83E+00	−5.75E+00	−5.45E+00	−6.92E+00	**−7.60E+00**
	(3.25E+00)	(3.50E+00)	(3.16E+00)	(3.17E+00)	(3.10E+00)	(3.36E+00)	(3.31E+00)
f_{23}	−6.96E+00	−7.00E+00	−6.54E+00	−5.79E+00	−7.56E+00	−6.87E+00	−7.32E+00
	(3.25E+00)	(3.41E+00)	(3.39E+00)	(3.21E+00)	(3.50E+00)	(3.54E+00)	(3.57E+00)

Table 5 Six experiment cases to evaluate the sensitivity of AβHC to its β value ranges

Experiment cases	β_{min}	β_{max}
Case 1	0.0001	0.4
Case 2	0.001	0.6
Case 3	0.01	0.8
Case 4	0.1	1.0
Case 5	0.0005	0.5
Case 6	0.05	0.9

parameter (see Sect. 4.2) and the optimal range of β parameter (i.e., β_{min} and β_{max}) (see Sect. 4.3). The performance of the proposed method over different problem dimensions is also studied in Sect. 4.4. The comparative study is a very important process. Therefore, the adaptive version of β-hill climbing is compared with the original version of β-hill climbing in Sect. 4.5.1. Thereafter, the AβHC is compared by other local-search-based methods in Sect. 4.5.2. Furthermore, five methods using the classical benchmark functions are also conducted in Sect. 4.5.3. For further validation using the recent benchmark functions, the IEEE-CEC2015 is used and 21 comparative methods using the same functions are compared with using three different function dimensionality (i.e., $D = 10$, $D = 30$, and $D = 50$) in Sect. 4.6.

4.1 Classical benchmark functions

In the field of optimization, benchmark functions with known global optima are popular tools for testing the performance of an algorithm. The current benchmark problems are divided into two main classes: unimodal and multimodal. The former class is suitable for testing exploitation, while the latter class tests exploration. This work employs the most well-regarded test suite including seven unimodal and 16 multimodal test functions (Mirjalili and Lewis 2013; Molga and Smutnicki 2005) which are discussed in Tables 1, 2, and 3, respectively.

4.2 The effect of $\mathcal{N}$ parameter

The effect of $\mathcal{N}$ parameter on the behavior of the proposed AβHC algorithm is studied in this section. It should be noted that the value of $\mathcal{N}$ parameter is affected by the value of K as shown in Eqs. (2) and (3). As mentioned earlier, the bigger the value of K, the higher rate of exploitation capability. The K parameter is studied using 15 different values (i.e., K=2, K=4, ..., K=30) as summarized in Table 4. The numbers in this table refers to the averages (AVE) and standard deviations (Stdev) of 30 independent runs. The best results were highlighted in **bold** font (lowest is best).

As observed from the results, the best outcomes are obtained when the value of K is 30. This is due to the fact that the bigger value of K, the bigger value of $\mathcal{N}$, thus the higher exploitation is used. It should be noted that when the value of K is bigger than 30, then the proposed method achieves worst than or the same results as in K=30. It is conventionally known that the exploration and exploitation concepts should be balanced during the search to achieve fruitful results. The tendency of overusing one concept without the second one is not necessarily useful which might negatively affect the search.

4.3 The effect of optimal range of β parameter

In this section, the effect of β parameter on the behavior of the proposed AβHC algorithm is studied. As aforementioned, the value of the β parameter at each iteration is affected by the value of β_{min} and β_{max} using Eq. (4). Table 5 provides six experimental cases with various range values of the β_{min} and β_{max} which are used to measure their effect on the convergence of the proposed method. The value range of β_{min} and β_{max} lies in the uniform range between 0 and 1. It should be noted that, the bigger the value of β, the higher rate of exploration capability will be.

Table 6 summarizes the average (AVE) and standard deviation (Stdev) of 30 runs for each experimental case. The best results were highlighted in **bold** font (lowest is best). For the unimodal test functions $f_1 - f_7$, the second case obtained the best results on f_2, f_3, and f_5, whereas, Case 1, Case 3, and Case 6 obtained the best results on one unimodal function. Unlike unimodal functions, the third case obtained the best results on f_9, f_{11}, and f_{13}, where these functions are classified as multimodal functions (i.e., f_8–f_{13}). Furthermore, Case 2, Case 4, and Case 6 achieved the best results for only one multimodal test function. For the last group of test functions (i.e., fixed-dimension multimodal functions), the fourth case (i.e., Case 4) obtained the best solutions on six functions (f_{16}, and f_{19}–f_{23}). The first case obtained the best results on thee test functions, and Case 6 achieved the best solutions on two test functions. Finally, the fourth case achieved the best results on seven test functions in the all function groups, and this the highest number between the six cases. In general, adapting the value of β parameter from β_{max} to β_{to} is able to provide a proper exploration in the initial search and the percentage of employing exploration is reduced gradually until the search process is completed. This empowers the search to make use of multi-regions in the search space avoiding the local optima dilemma.

4.4 Scalability study

In order to study the effect of the proposed method on different problem dimensions, the test functions are experimented with using various values of D (i.e., $D = 10$, $D = 30$, $D = 50$, $D = 100$). The parameter values used are: $K = 30$,

Table 6 Results of AβHC of different experiment cases of β value ranges

Benchmark	Case 1 AVE (Stdev)	Case 2 AVE (Stdev)	Case 3 AVE (Stdev)	Case 4 AVE (Stdev)	Case 5 AVE (Stdev)	Case 6 AVE (Stdev)
f_1	8.72E−07	8.01E−07	7.01E−07	2.03E−05	8.58E−07	**6.69E−07**
	(3.63E−07)	(4.32E−07)	(2.38E−07)	(2.71E−07)	(7.04E−07)	(2.39E−07)
f_2	2.87E−03	**2.69E−03**	3.06E−03	8.51E−02	3.05E−03	3.02E−03
	(4.77E−04)	(5.84E−04)	(5.47E−04)	(5.62E−04)	(5.11E−04)	(4.69E−04)
f_3	1.49E+03	**1.37E+03**	1.54E+03	4.49E+04	1.46E+03	1.47E+03
	(5.07E+02)	(5.47E+02)	(4.81E+02)	(4.94E+02)	(6.03E+02)	(5.26E+02)
f_4	**2.29E+00**	2.35E+00	2.45E+00	6.90E+01	2.36E+00	2.40E+00
	(4.79E−01)	(5.05E−01)	(4.54E−01)	(3.25E−01)	(3.26E−01)	(3.88E−01)
f_5	7.12E+01	**4.58E+01**	5.25E+01	1.94E+03	5.75E+01	6.10E+01
	(9.77E+01)	(3.70E+01)	(4.20E+01)	(6.28E+01)	(7.19E+01)	(7.04E+01)
f_6	8.34E−07	7.29E−07	**7.20E−07**	2.53E−05	7.56E−07	8.44E−07
	(4.69E−07)	(3.35E−07)	(2.91E−07)	(3.70E−07)	(3.27E−07)	(3.78E−07)
f_7	1.43E−02	1.33E−02	1.29E−02	3.73E−01	1.34E−02	**1.22E−02**
	(4.71E−03)	(4.17E−03)	(5.12E−03)	(4.28E−03)	(4.83E−03)	(4.24E−03)
f_8	− 1.26E+04	− 1.26E+04	− 1.26E+04	**− 3.77E+05**	− 1.26E+04	− 1.26E+04
	(5.02E−07)	(8.14E−03)	(5.37E−08)	(1.33E−07)	(1.25E−07)	(1.41E−07)
f_9	3.58E−04	3.72E−04	**2.65E−04**	8.30E−03	3.04E−04	3.25E−04
	(2.74E−04)	(2.72E−04)	(1.71E−04)	(1.81E−04)	(2.91E−04)	(1.60E−04)
f_{10}	9.54E−04	8.98E−04	9.10E−04	2.67E−02	9.09E−04	**8.77E−04**
	(2.54E−04)	(2.58E−04)	(2.67E−04)	(2.61E−04)	(2.44E−04)	(1.99E−04)
f_{11}	9.33E−02	8.39E−02	**7.74E−02**	3.12E+00	1.07E−01	8.93E−02
	(6.60E−02)	(6.45E−02)	(7.49E−02)	(8.30E−02)	(7.18E−02)	(8.96E−02)
f_{12}	7.35E−08	**5.59E−08**	1.14E−07	2.64E−06	7.05E−08	7.88E−08
	(1.50E−07)	(1.04E−07)	(2.13E−07)	(2.15E−07)	(1.16E−07)	(1.33E−07)
f_{13}	4.40E−03	4.44E−03	**3.70E−03**	1.08E−01	4.23E−03	5.48E−03
	(5.47E−03)	(6.05E−03)	(5.24E−03)	(6.49E−03)	(5.27E−03)	(6.80E−03)
f_{14}	**9.98E−01**	**9.98E−01**	**9.98E−01**	2.99E+01	**9.98E−01**	**9.98E−01**
	(2.72E−16)	(3.02E−16)	(2.72E−16)	(2.90E−16)	(3.23E−16)	(2.84E−16)
f_{15}	**1.08E−03**	1.09E−03	1.08E−03	2.95E−02	1.17E−03	1.16E−03
	(5.71E−04)	(5.90E−04)	(5.39E−04)	(5.66E−04)	(6.52E−04)	(5.21E−04)
f_{16}	− 1.03E+00	− 1.03E+00	− 1.03E+00	**− 3.09E+01**	− 1.03E+00	− 1.03E+00
	(2.54E−13)	(6.52E−14)	(3.76E−14)	(7.43E−14)	(5.76E−14)	(5.98E−14)
f_{17}	3.98E−01	3.98E−01	3.98E−01	1.19E+01	3.98E−01	**3.98E−01**
	(5.02E−14)	(4.68E−14)	(6.01E−14)	(9.15E−14)	(1.94E−13)	(2.09E−14)
f_{18}	**3.00E+00**	3.00E+00	3.00E+00	9.00E+01	3.00E+00	3.00E+00
	(2.32E−12)	(8.34E−12)	(5.90E−12)	(7.35E−12)	(8.31E−12)	(3.51E−12)
f_{19}	− 3.86E+00	− 3.86E+00	− 3.86E+00	**− 1.16E+02**	− 3.86E+00	− 3.86E+00
	(1.09E−11)	(6.03E−12)	(4.88E−12)	(7.43E−12)	(4.37E−12)	(9.81E−12)
f_{20}	− 3.30E+00	− 3.29E+00	− 3.28E+00	**− 9.88E+01**	− 3.29E+00	− 3.28E+00
	(4.84E−02)	(5.35E−02)	(5.83E−02)	(5.11E−02)	(5.35E−02)	(5.70E−02)
f_{21}	− 6.48E+00	− 7.56E+00	− 6.14E+00	**− 1.94E+02**	− 7.39E+00	− 6.21E+00
	(3.58E+00)	(3.32E+00)	(3.44E+00)	(3.39E+00)	(3.31E+00)	(3.20E+00)
f_{22}	− 6.31E+00	− 6.87E+00	− 6.27E+00	**− 1.77E+02**	− 6.74E+00	− 5.70E+00
	(3.46E+00)	(3.43E+00)	(3.26E+00)	(3.46E+00)	(3.53E+00)	(3.19E+00)
f_{23}	− 6.64E+00	− 7.69E+00	− 6.60E+00	**− 1.96E+02**	− 8.31E+00	− 7.19E+00
	(3.52E+00)	(3.14E+00)	(3.32E+00)	(3.39E+00)	(3.25E+00)	(3.27E+00)

Table 7 Results of adaptive β-hill climbing of $f_1 - f_{13}$ functions with different dimensions

Benchmark	$D = 10$ AVE (SD)	$D = 30$ AVE (SD)	$D = 50$ AVE (SD)	$D = 100$ AVE (SD)
f_1	9.62E−10	7.01E−07	2.55E−03	8.45E+01
	(1.27E−09)	(2.38E−07)	(1.32E−02)	(2.49E+01)
f_2	6.22E−05	3.06E−03	1.60E−02	2.60E−01
	(2.21E−05)	(5.47E−04)	(2.51E−03)	(6.54E−02)
f_3	1.36E−05	1.54E+03	1.19E+04	9.05E+04
	(5.38E−06)	(4.81E+02)	(2.67E+03)	(9.98E+03)
f_4	7.99E−05	2.45E+00	8.12E+00	2.79E+01
	(3.09E−05)	(4.54E−01)	(1.03E+00)	(1.64E+00)
f_5	6.87E+00	5.25E+01	3.36E+02	1.21E+03
	(7.75E+00)	(4.20E+01)	(6.19E+02)	(5.00E+02)
f_6	1.03E−09	7.20E−07	2.13E−04	7.68E+01
	(7.21E−10)	(2.91E−07)	(1.03E−03)	(2.26E+01)
f_7	5.53E−04	1.29E−02	5.67E−02	4.50E−01
	(2.95E−04)	(5.12E−03)	(1.62E−02)	(7.44E−02)
f_8	−4.20E+03	−1.26E+04	−2.09E+04	−4.15E+04
	(3.51E+01)	(5.37E−08)	(1.57E+00)	(8.59E+01)
f_9	2.13E−07	2.65E−04	1.02E+00	3.40E+01
	(2.11E−07)	(1.71E−04)	(8.78E−01)	(5.08E+00)
f_{10}	3.64E−05	9.10E−04	2.48E−01	2.71E+00
	(1.35E−05)	(2.67E−04)	(2.84E−01)	(1.97E−01)
f_{11}	6.08E−02	7.74E−02	5.82E−01	2.49E+00
	(2.46E−02)	(7.49E−02)	(2.29E−01)	(2.96E−01)
f_{12}	3.37E−10	1.14E−07	4.15E−03	1.88E−01
	(8.47E−10)	(2.13E−07)	(2.27E−02)	(1.15E−01)
f_{13}	3.10E−09	3.70E−03	1.86E−02	5.24E+00
	(5.38E−09)	(5.24E−03)	(1.77E−02)	(1.31E+00)

$\beta_{\min} = 0.1$, and $\beta_{\max} = 1.0$. These parameters are selected from the previous sections because they produced the best results for the test functions used. Table 7 summarizes the results of the test functions in terms of average (AVE) and standard deviation (Stdev) over 30 replication runs. The best results obtained by any dimension for any test function are highlighted in **bold**. Notably, only the first 13 test functions have been used in this study because the other test functions have predetermined dimensionality.

Apparently, the most best results for all test functions are achieved when the value of $D = 10$ is used. Then the second-best results are achieved when the value of $D = 30$ is used, etc. In general, as the value of dimensionality D is reduced, the better results will be achieved. This conclusion is in-line with the theory of optimization where the dimensionality of the problem reflect its complexity and normally the optimization method worked well when the complexity of the problem is reduced.

4.5 Comparative analysis

In this section, the comparison is carried out by means of threefold: (i) comparison between AβHC and βHC is done in Sect. 4.5.1, (ii) comparison with other local search algorithms which discussed in Sect. 4.5.2, and (iii) comparison with other methods as discussed in Sect. 4.5.3.

4.5.1 A comparison between AβHC and βHC

This subsection first compares the proposed AβHC algorithm with the original βHC algorithm. The results of both algorithms on unimodal test cases are given in Table 8. This table shows the experimental results on f_1 to f_6 with 10, 30, 50, and 100 variables. The most interesting pattern is the significant superiority of AβHC when solving high-dimensional unimodal test functions with 50 and 100 parameters. It is evident in Table 8 that AβHC outperforms βHC on the majority of high-dimensional unimodal test functions.

Table 8 Comparison between adaptive β-hill climbing (AβHC) and standard β-hill climbing (βHC) using $f_1 - f_{13}$ functions with dimensions: 10, 30, 50, and 100

Benchmark	D=10		D=30		D=50		D=100	
	AβHC AVE (SD)	βHC AVE (SD)	AβHC AVE (SD)	βHC AVE (SD)	AβHC AVE (SD)	βHC AVE (SD)	AβHC AVE (SD)	βHC AVE (SD)
f_1	9.62E−10	**1.21E−13**	**7.01E−07**	1.02E−02	**2.55E−03**	9.39E+00	**8.45E+01**	7.84E+03
	(1.27E−09)	(1.05E−13)	(2.38E−07)	(1.06E−02)	(1.32E−02)	(2.97E+00)	(2.49E+01)	(9.32E+02)
f_2	6.22E−05	**6.19E**−07	3.06E−03	**5.62E−05**	**1.60E−02**	1.31E+00	**2.60E−01**	5.95E+01
	(2.21E−05)	(2.38E−07)	(5.47E−04)	(1.16E−05)	(2.51E−03)	(2.47E−01)	(6.54E−02)	(3.50E+00)
f_3	**1.36E−05**	1.12E−01	1.54E+03	**4.94E+02**	1.19E+04	**6.73E+03**	**9.05E+04**	1.09E+05
	(5.38E−06)	(2.64E−01)	(4.81E+02)	(1.71E+02)	(2.67E+03)	(1.46E+03)	(9.98E+03)	(1.25E+04)
f_4	7.99E−05	**1.26E−06**	2.45E+00	**1.13E+00**	8.12E+00	**6.81E+00**	**2.79E+01**	3.37E+01
	(3.09E−05)	(4.35E−07)	(4.54E−01)	(1.84E−01)	(1.03E+00)	(7.49E−01)	(1.64E+00)	(1.57E+00)
f_5	**6.87E+00**	1.57E+06	**5.25E+01**	8.87E+01	**3.36E+02**	4.96E+02	**1.21E+03**	1.68E+06
	(7.75E+00)	(3.18E+05)	(4.20E+01)	(3.48E+01)	(6.19E+02)	(2.61E+02)	(5.00E+02)	(3.18E+05)
f_6	1.03E−09	**8.68E−14**	**7.20E−07**	1.36E−02	**2.13E−04**	1.08E+01	**7.68E+01**	8.09E+03
	(7.21E−10)	(6.15E−14)	(2.91E−07)	(2.37E−02)	(1.03E−03)	(3.02E+00)	(2.26E+01)	(9.08E+02)
f_7	**5.53E−04**	1.08E−03	**1.29E−02**	1.73E−02	**5.67E−02**	1.14E−01	**4.50E−01**	4.08E+00
	(2.95E−04)	(6.36E−04)	(5.12E−03)	(5.29E−03)	(1.62E−02)	(3.36E−02)	(7.44E−02)	(7.58E−01)
f_8	− **4.20E+03**	− 4.19E+03	− **1.26E+04**	− **1.26E+04**	− **2.09E+04**	− **2.09E+04**	− **4.15E+04**	− 3.56E+04
	(3.51E+01)	(1.46E−12)	(5.37E−08)	(9.17E−02)	(1.57E+00)	(1.11E+01)	(8.59E+01)	(5.45E+02)
f_9	2.13E−07	**1.56E−11**	2.65E−04	**3.63E−08**	**1.02E+00**	2.77E+00	**3.40E+01**	2.56E+02
	(2.11E−07)	(1.27E−11)	(1.71E−04)	(1.66E−08)	(8.78E−01)	(8.40E−01)	(5.08E+00)	(1.39E+01)
f_{10}	3.64E−05	**3.49E−07**	9.10E−04	**2.20E−05**	**2.48E−01**	1.20E+00	**2.71E+00**	1.03E+01
	(1.35E−05)	(1.46E−07)	(2.67E−04)	(1.50E−05)	(2.84E−01)	(2.04E−01)	(1.97E−01)	(3.93E−01)
f_{11}	6.08E−02	**4.46E−02**	**7.74E−02**	1.40E−01	**5.82E−01**	1.09E+00	**2.49E+00**	7.12E+01
	(2.46E−02)	(2.64E−02)	(7.49E−02)	(4.68E−02)	(2.29E−01)	(2.88E−02)	(2.96E−01)	(7.56E+00)
f_{12}	3.37E−10	**3.66E−14**	**1.14E−07**	4.37E−05	**4.15E−03**	1.95E−02	**1.88E−01**	1.88E+03
	(8.47E−10)	(4.55E−14)	(2.13E−07)	(1.68E−04)	(2.27E−02)	(1.48E−02)	(1.15E−01)	(2.20E+03)
f_{13}	3.10E−09	**1.17E−13**	3.70E−03	**2.58E−03**	**1.86E−02**	4.06E−01	**5.24E+00**	6.31E+05
	(5.38E−09)	(1.90E−13)	(5.24E−03)	(4.59E−03)	(1.77E−02)	(1.17E−01)	(1.31E+00)	(3.48E+05)

On the other hand, βHC provides very comparative results when solving low-dimensional unimodal test functions. βHC outperforms AβHC on the majority of unimodal test functions when there are only 10 variables. This superiority degrades when the number of variables changes to 30, showing that the convergence speed of βHC becomes slower proportional to the number of variables. The unimodal test functions have no local optimum, so an algorithm needs to constantly improve the quality of the best solution obtained so far to coverage toward the global optimum. These results prove that AβHC improves this drawback of βHC and is able to deliver fast convergence speed. The adaptive mechanism requires this algorithm to efficiently solve problems with a large number of variables as well.

To better observe the convergence of both algorithm, the convergence curves are illustrated in Fig. 2. This figure shows that AβHC benefits from a much faster convergence speed than βHC on f_1, f_5, and f_6. In the rest of unimodal case studies, βHC is better. The convergence curves are for test functions with 30 variables only, so this is why βHC is competitive. The discrepancy of results, and consequently superiority of AβHC, is more significant for test functions with more parameters.

The results of AβHC and βHC when solving multimodal test functions are provided in Table 8 (f_8 to f_{13}) and Table 9. It may be seen that AβHC is significantly better than βHC on all test functions. In multimodal test functions, there is a large number of local solutions. This allows us to challenge an algorithm from local optima avoidance perspective. Once again, the superiority of AβHC becomes more evident proportional to the number of variables. These results prove that the probability of trapping in local optima in AβHC is lower that that of βHC.

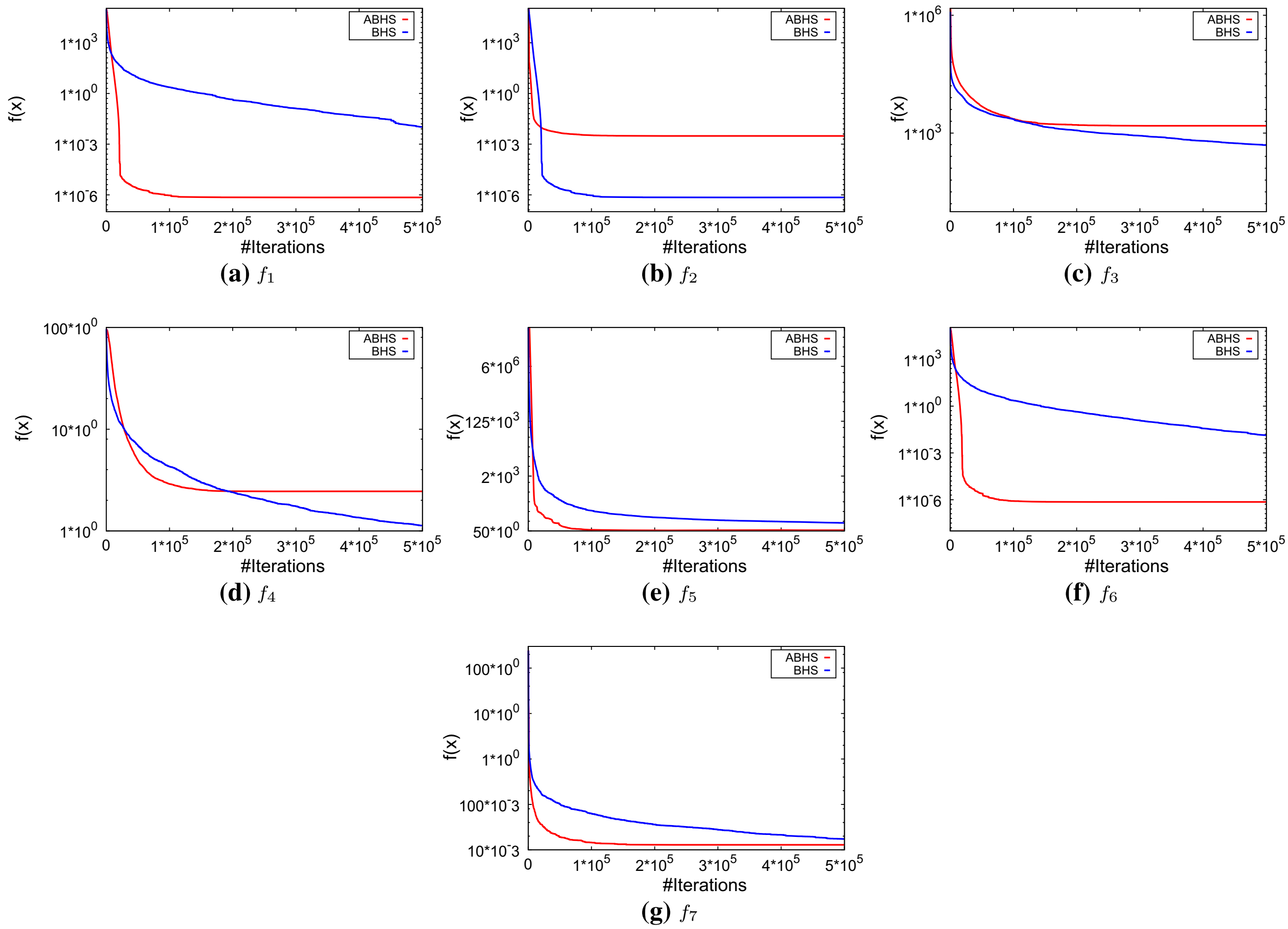

Fig. 2 Plots showing the convergence behavior tendency of both standard β-hill climbing and adaptive β-hill climbing for unimodal benchmark functions $f_1 - f_7$. with Dimension=30

To see whether high local optima avoidance impacts on the convergence speed, Figs. 3 and 4 are provided. It may be seen that the convergence of the AβHC algorithm is fast and better than that of βHC on the multimodal test functions. An interesting pattern is the accelerated convergence, in which the proposed algorithm shows faster convergence proportional to the number of iterations. This originates from the adaptive mechanism that assists AβHC to first avoid local solutions and then converges toward the global optimum. The convergence of βHC is gradual and monotonic since there is no adaptive mechanism in this algorithm.

4.5.2 Comparison with other local search-based algorithms

As the AβHC is a local search-based method, in this section, its performance is compared against other local search-based techniques such as hill climbing (HC), β-hill climbing (Al-Betar 2017), simulated annealing (SA) (Kirkpatrick et al. 1983), tabu search (TS) (Glover 1986), and variable neighborhood search (VNS) (Hansen and Mladenovi'c 1999). Table 10 summarizes the results in terms of average (AVE) and standard deviation (Stdev) of 30 replication runs for all test functions. The best results obtained are highlighted in **bold**.

The parameter settings used for comparative methods (SA and TS) is selected from the original β-hill climbing (Al-Betar 2017) where for βHC, $\beta = 0.05$ and $\mathcal{N} = 0.5$ for SA, the initial value of temperature $T_0 = 100$ and temperature rate $\alpha = 85\%$ are used as suggested in (Corana et al. 1987). For TS, the tabu list size $TN = N/2$.

Apparently, the performance of AβHC and original βHC seems similar where both are able to produce the best recorded results for 10 test functions. SA in specific is able to produce the best results for only one test functions, while TS is able to yield the best recorded results for 6 test functions. VNS is also obtained the best recorded results for two test functions while the simple form of HC is not able to produce any best result. In general, the performance of AβHC and original βHC are almost better than others using the 23 test functions.

Table 9 Comparison between adaptive β-hill climbing (AβHC) and standard β-hill climbing (βHC) using fixed-dimension multimodal functions ($f_{14} - f_{23}$)

Benchmark	AβHC AVE Stdev	βHC AVE Stdev
f_{14}	**9.98E−01**	**9.98E−01**
	(2.72E−16)	(1.98E−16)
f_{15}	1.08E−03	**5.21E−04**
	(5.39E−04)	(3.78E−04)
f_{16}	**−1.03E+00**	**−1.03E+00**
	(3.76E−14)	(1.24E−15)
f_{17}	**3.98E−01**	**3.98E−01**
	(6.01E−14)	(8.96E−16)
f_{18}	**3.00E+00**	**3.00E+00**
	(5.90E−12)	(7.63E−14)
f_{19}	**−3.86E+00**	**−3.86E+00**
	(4.88E−12)	(3.58E−14)
f_{20}	**−3.28E+00**	−3.30E+00
	(5.83E−02)	(4.51E−02)
f_{21}	**−6.14E+00**	−5.65E+00
	(3.44E+00)	(3.37E+00)
f_{22}	**−6.27E+00**	−5.21E+00
	(3.26E+00)	(3.28E+00)
f_{23}	**−6.60E+00**	−5.20E+00
	(3.32E+00)	(3.39E+00)

According to results of the statistical test shown in Table 11, we can summarize that AβHC is significantly better than βHC, HC, SA, TS and VNS in 7, 14, 18, 13, and 12 test functions, respectively.

4.5.3 Comparison with other methods

To confirm and verify the performance of this proposed algorithm, Table 12 compares its results with well-known (particle swarm optimization (PSO) and genetic algorithm (GA)) and recent (multi-verse optimizer (MVO), grey wolf optimizer (GWO), and gravitational search algorithm (GSA)) optimization algorithms. Note that all test functions have 50 parameters in this experiment. The parameter settings used for each method are recorded in Table 13 (Mirjalili and Lewis 2016). Inspecting the results of these two tables, it is observed that the proposed AβHC algorithm outperforms other algorithms significantly on the majority of unimodal and multimodal test functions. The superiority is substantial in most of the cases, proving the merits of the proposed technique.

According to results of the statistical test shown in Table 14, we can summarize that AβHC is significantly better than MVO, GWO, GSA, PSO, and GA in 11, 13, 6, 14, 18, and 20 test functions, respectively.

Taken together, the proposed AβHC algorithm improves both exploration and exploitation of βHC. The adaptive

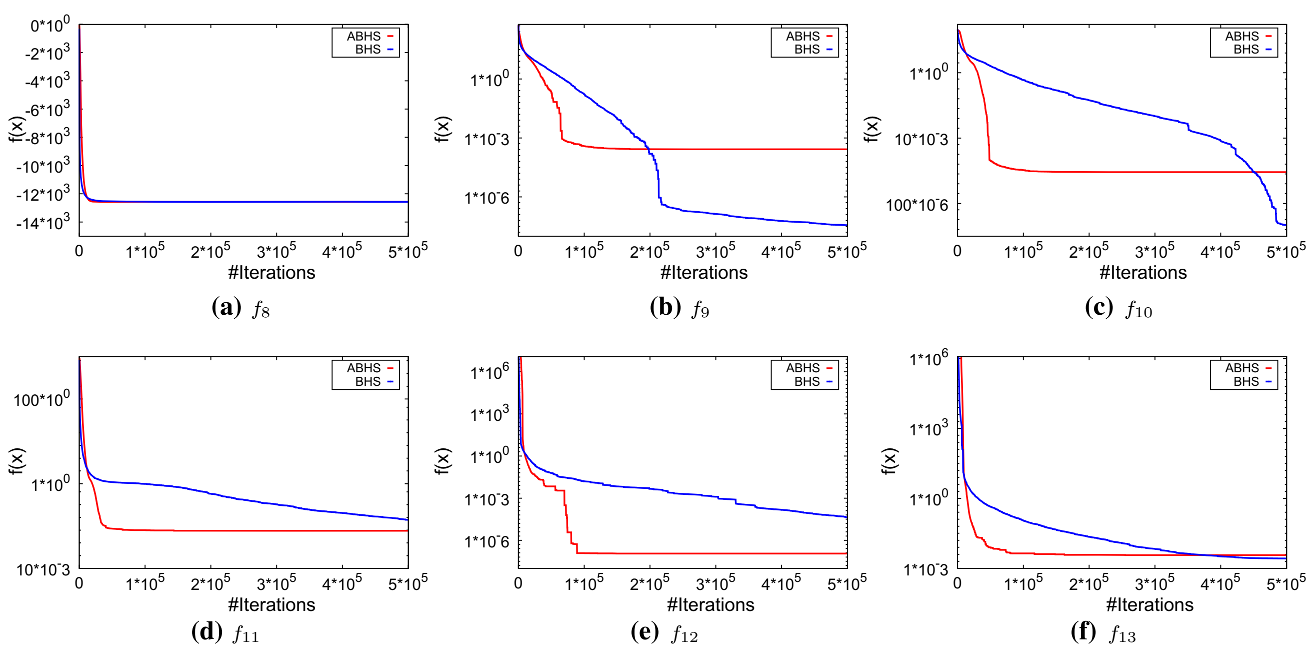

Fig. 3 Plots showing the convergence behavior tendency of both standard β-hill climbing and adaptive β-hill climbing for multimodal benchmark functions $f_8 - f_{13}$. with Dimension=30

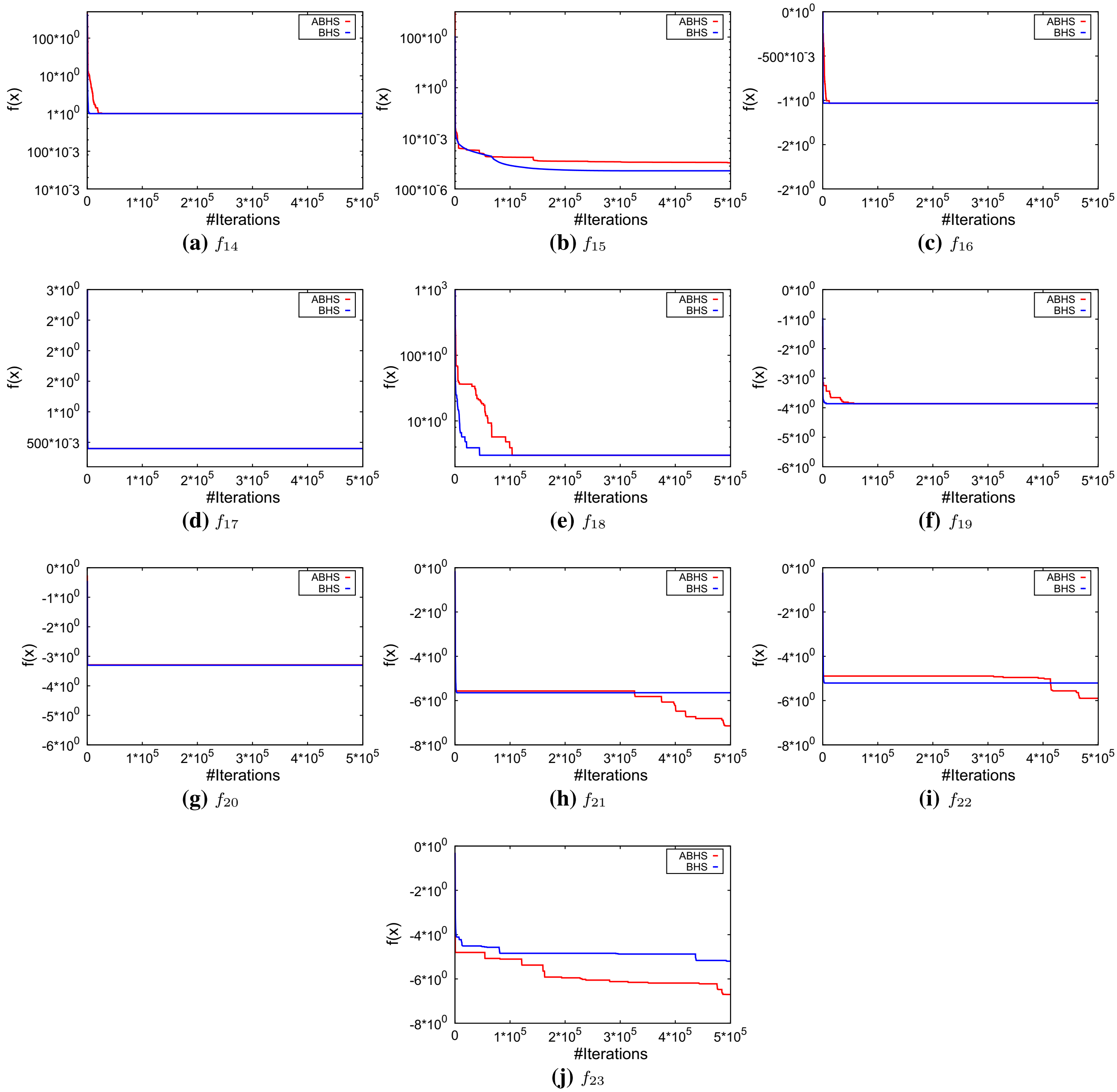

Fig. 4 Plots showing the convergence behavior tendency of both standard β-hill climbing and adaptive β-hill climbing for fixed-dimension multimodal benchmark functions $f_{14} - f_{23}$

mechanism allows this algorithm to emphasize random changes for the solutions in the initial steps of optimization. It then decreases the randomness and boosts improvement to solution(s) proportional to the number of iterations. The adaptive mechanism is problem independent and calculated based on the iteration number, so it changes the magnitude of exploration and exploitation with respect to the progress of optimization process. High exploration in the initial iterations promotes local optima avoidance of AβHC. Gradual changes in the last iterations require AβHC to improve the accuracy of the best solution obtained so-far and converge toward the global optimum.

4.6 Experimented with CEC2015 datasets

In order to validate the performance of the proposed adaptive method on the recent test functions established in IEEE-CEC2015 (Liang et al. 2014), this section provides a comparative evaluation between the proposed methods and other 15 methods including the original β-hill climb-

Table 10 Comparison between AβHC and other local search-based methods

Benchmark	AβHC AVE (SD)	βHC AVE (SD)	HC AVE (SD)	SA AVE (SD)	TS AVE (SD)	VNS AVE (SD)
f_1	7.01E−07	1.02E−02	3.26E−09	8.06E−06	**1.78E−10***	7.33E+01
	(2.38E−07)	(1.06E−02)	(1.35E−09)	(2.23E−06)	(8.11E−11)	(1.29E+02)
f_2	3.06E−03	5.62E−05	1.85E−04	5.75E−03	4.66E−05	**4.75E−06***
	(5.47E−04)	(1.16E−05)	(3.67E−05)	(1.22E−03)	(7.64E−06)	(1.02E−06)
f_3	1.54E+03	4.94E+02	4.64E−03	2.22E−03	**2.05E−04***	4.57E+01
	(4.81E+02)	(1.71E+02)	(1.84E−03)	(1.67E−03)	(5.74E−05)	(9.86E+01)
f_4	2.45E+00	1.13E+00	4.63E+01	**1.69E−03***	2.67E+01	6.75E+01
	(4.54E−01)	(1.84E−01)	(5.81E+00)	(1.31E−03)	(7.13E+00)	(4.42E+00)
f_5	5.25E+01	8.87E+01	4.31E+01	2.10E+01	**1.73E+01***	2.10E+01
	(4.20E+01)	(3.48E+01)	(6.00E+01)	(4.02E+01)	(3.92E+01)	(4.88E+01)
f_6	7.20E−07	1.36E−02	3.06E−09	8.15E−06	**1.67E−10***	5.21E+01
	(2.91E−07)	(2.37E−02)	(1.31E−09)	(1.75E−06)	(8.33E−11)	(1.07E+02)
f_7	1.29E−02	1.73E−02	9.85E−04	1.71E−02	**2.53E−04***	4.73E−03
	(5.12E−03)	(5.29E−03)	(3.56E−04)	(6.38E−03)	(9.11E−05)	(1.77E−03)
f_8	**−1.26E+04**	**−1.26E+04**	−7.54E+03	−7.83E+03	−7.70E+03	−7.52E+03
	(5.37E−08)	(9.17E−02)	(9.20E+02)	(7.98E+02)	(9.30E+02)	(9.22E+02)
f_9	2.65E−04	**3.63E−08***	2.61E+02	2.63E+02	2.63E+02	2.61E+02
	(1.71E−04)	(1.66E−08)	(5.04E+01)	(3.68E+01)	(4.78E+01)	(3.78E+01)
f_{10}	9.10E−04	**2.20E−05***	1.95E+01	1.95E+01	1.94E+01	1.95E+01
	(2.67E−04)	(1.50E−05)	(1.92E−01)	(1.14E−01)	(1.61E−01)	(1.36E−01)
f_{11}	**7.74E−02***	1.40E−01	3.75E+01	4.02E+01	2.47E+01	3.86E+02
	(7.49E−02)	(4.68E−02)	(1.42E+01)	(1.71E+01)	(1.23E+01)	(1.00E+02)
f_{12}	**1.14E−07**	4.37E−05	1.95E+01	1.76E+01	2.18E+01	2.52E+01
	(2.13E−07)	(1.68E−04)	(1.75E+01)	(1.71E+01)	(1.79E+01)	(1.39E+01)
f_{13}	3.70E−03	**2.58E−03**	9.45E+01	6.10E+01	8.79E+01	9.58E+01
	(5.24E−03)	(4.59E−03)	(2.59E+01)	(3.92E+01)	(2.76E+01)	(2.71E+01)
f_{14}	**9.98E−01**	**9.98E−01**	1.39E+01	1.61E+01	1.27E+01	1.28E+01
	(2.72E−16)	(1.98E−16)	7.00E+00	7.31E+00	7.77E+00	8.29E+00
f_{15}	1.08E−03	**5.21E−04***	2.45E−02	3.60E−03	4.24E−02	3.71E−02
	(5.39E−04)	(3.78E−04)	4.12E−02	8.01E−03	5.37E−02	4.70E−02
f_{16}	**−1.03E+00**	**−1.03E+00**	5.80E−02	−8.96E−01	−4.14E−01	−5.96E−01
	(3.76E−14)	(1.24E−15)	1.09E+00	3.09E−01	7.93E−01	4.14E−01
f_{17}	3.98E−01	3.98E−01	3.98E−01	3.98E−01	3.98E−01	**3.98E−01**
	(6.01E−14)	(8.96E−16)	2.38E−11	9.43E−07	7.60E−13	1.24E−14
f_{18}	**3.00E+00**	**3.00E+00**	1.88E+02	1.30E+02	1.07E+02	2.42E+02
	(5.90E−12)	(7.63E−14)	3.33E+02	2.84E+02	2.50E+02	3.67E+02
f_{19}	**−3.86E+00**	**−3.86E+00**	−2.61E+00	−3.10E+00	−2.94E+00	−2.61E+00
	(4.88E−12)	(3.58E−14)	1.28E+00	1.37E+00	1.23E+00	1.28E+00
f_{20}	−3.28E+00	**−3.30E+00**	−3.28E+00	−1.21E+00	−3.29E+00	−3.30E+00
	(5.83E−02)	(4.51E−02)	5.83E−02	1.61E+00	5.54E−02	4.84E−02
f_{21}	−6.14E+00	−5.65E+00	−5.41E+00	−6.56E+00	**−7.24E+00**	−5.90E+00
	(3.44E+00)	(3.37E+00)	3.49E+00	3.52E+00	3.65E+00	3.60E+00
f_{22}	**−6.27E+00**	−5.21E+00	−4.75E+00	−4.81E+00	−5.27E+00	−5.02E+00
	(3.26E+00)	(3.28E+00)	2.97E+00	2.98E+00	3.47E+00	3.13E+00
f_{23}	**−6.60E+00***	−5.20E+00	−4.92E+00	−4.80E+00	−3.79E+00	−4.08E+00
	(3.32E+00)	(3.39E+00)	3.30E+00	3.32E+00	2.17E+00	2.70E+00

* means that there are significant differences with all comparative results against the best

Table 11 P values of the Wilcoxon test of AβHC results vs other local search-based algorithms (p $\geq$ 0.05 are underlined)

	βHC	HC	SA	TS	VNS
f_1	5.57E−10	3.02E−11	3.02E−11	3.02E−11	6.63E−01
f_2	3.02E−11	3.02E−11	4.50E−11	3.02E−11	3.02E−11
f_3	3.34E−11	3.02E−11	3.02E−11	3.02E−11	3.02E−11
f_4	3.02E−11	3.02E−11	3.02E−11	3.02E−11	3.02E−11
f_5	7.70E−04	5.19E−02	1.17E−04	7.22E−06	1.75E−05
f_6	3.02E−11	3.02E−11	3.02E−11	3.02E−11	3.79E−01
f_7	1.30E−03	3.02E−11	3.85E−03	3.02E−11	2.61E−10
f_8	1.21E−12	1.21E−12	1.21E−12	1.21E−12	1.21E−12
f_9	3.02E−11	3.02E−11	3.02E−11	3.02E−11	3.02E−11
f_{10}	3.02E−11	3.02E−11	3.02E−11	3.02E−11	3.02E−11
f_{11}	7.20E−05	3.02E−11	3.02E−11	3.02E−11	3.02E−11
f_{12}	6.95E−01	8.48E−09	3.16E−10	8.48E−09	5.57E−10
f_{13}	2.58E−01	3.02E−11	3.02E−11	3.02E−11	3.02E−11
f_{14}	NaN	1.20E−12	4.46E−12	5.70E−11	4.44E−12
f_{15}	6.35E−07	2.27E−03	9.12E−01	1.91E−02	1.91E−02
f_{16}	NaN	3.36E−09	1.21E−12	1.65E−06	3.80E−06
f_{17}	NaN	NaN	1.21E−12	NaN	NaN
f_{18}	NaN	1.26E−07	1.21E−12	7.86E−07	1.16E−07
f_{19}	NaN	4.32E−08	1.65E−11	2.78E−05	2.92E−07
f_{20}	3.15E−07	9.48E−01	8.50E−07	5.57E−04	1.95E−06
f_{21}	6.50E−01	4.62E−01	1.94E−01	2.24E−01	7.86E−01
f_{22}	1.19E−01	4.19E−02	2.29E−03	5.68E−02	1.53E−01
f_{23}	2.13E−02	6.46E−03	8.84E−05	8.90E−05	1.31E−04

ing. The characteristics of the test functions are summarized in Table 15. These test functions are published for special session of learning-based real-parameter single-objective optimization at IEEE-CEC2015. The table includes the functions labels (key and name), the value range of the variables, the function category, and the predefined optimal value for each function. Note that the functions fall in the following categories: U: unimodal; M: multimodal; N: non-separable; H: hybrid (unimodal or multimodal); and C: composition. The detailed information about these test functions is given in (Liang et al. 2014).

The comparative methods are abbreviated in Table 16. These comparative methods are heuristically designed to compete in the IEEE-CEC2015 competition. Most of these methods are either a modification or hybridization versions. Therefore, the results produced by each of them seems very successful. In Table 16, the key, the method name, and the publication reference are provided. It should be noted that one of the comparative methods are local search-based method which is SA. Note that most of these methods participated in the IEEE-CEC2015 competition and achieved premium ranks.

In order to meet the requirements of the IEEE-CEC2015, the scalability study of three solution dimensions (D = 10, $D = 30$, $D = 50$) are experimented. The required competition rules are as follows: The number of experiment runs for each tested case is 51 replications; the number of iterations is $D \times 10,000$. The fitness of the solution is calculated in terms of average error (AE) which is $AE = |f(\boldsymbol{x}^*)$ - $f(\boldsymbol{x}^{best})|$. Three tables of different dimensions (Table 17 for $D = 10$, Table 18 for $D = 30$, and Table 19 for $D = 50$) are created to summarize the results obtained by the proposed adaptive method against the other comparative methods. The best results in these tables are highlighted in **bold** (lowest is best). Note that for adaptive β-hill climbing, [$\beta_{\max} = 1.0$, $\beta_{\min}$=0.1] and $K = 30$ are used in these experiments because they almost achieved the best recorded results in the previous sections.

Apparently, in Table 17 where the solution dimension equals to 10, the proposed method obtains the best recorded optimal solution for test function f_{36} and test function f_{38} and produces a comparable results for the other test functions. In comparison with the original βHC, the proposed AβHC performed better for almost all test functions. The competitive SA is also a local-search-based method. The AβHC shows better results than SA in 8 out of 15 tested cases. From a different perspective, the AβHC seems to work very well with the problem of composite category.

The results obtained when the solution dimensions equal to 30 and 50 are summarized in Tables 18, and 19. It is con-

Table 12 Comparison between the proposed AβHC and other algorithms in the literature using $f_1 - f_{13}$ (Dimension=50) functions, and other with fixed dimensions

Benchmark	AβHC AVE (SD)	βHC AVE (SD)	MVO AVE (SD)	GWO AVE (SD)	GSA AVE (SD)	PSO AVE (SD)	GA AVE (SD)
f_1	2.55E−03	9.39E+00	9.44E+00	**8.64E−20***	2.09E+02	3.89E+04	8.57E+03
	(1.32E−02)	(2.97E+00)	(1.74E+00)	(9.21E−20)	(2.37E+02)	(5.51E+03)	(3.46E+03)
f_2	1.60E−02	1.31E+00	1.36E+02	**2.62E−12***	1.41E+00	6.28E+07	8.19E+01
	(2.51E−03)	(2.47E−01)	(7.13E+01)	(1.15E−12)	(9.54E−01)	(2.37E+08)	(1.35E+01)
f_3	1.19E+04	6.73E+03	5.78E+03	**3.02E−01***	3.09E+03	1.07E+05	7.25E+04
	(2.67E+03)	(1.46E+03)	(1.61E+03)	(6.48E−01)	(7.03E+02)	(1.76E+04)	(1.62E+04)
f_4	8.12E+00	6.81E+00	1.42E+01	**3.04E−04***	1.26E+01	5.39E+01	6.53E+01
	(1.03E+00)	(7.49E−01)	(4.75E+00)	(2.38E−04)	(2.41E+00)	(2.84E+00)	(8.79E+00)
f_5	3.36E+02	4.96E+02	9.38E+02	**4.77E+01***	1.05E+03	6.43E+07	7.12E+05
	(6.19E+02)	(2.61E+02)	(8.89E+02)	(7.45E−01)	(8.43E+02)	(1.72E+07)	(7.59E+05)
f_6	**2.13E−04***	1.08E+01	9.16E+00	2.61E+00	2.61E+02	4.06E+04	7.89E+03
	(1.03E−03)	(3.02E+00)	(2.45E+00)	(4.84E−01)	(2.27E+02)	(5.68E+03)	(3.17E+03)
f_7	5.67E−02	1.14E−01	1.18E−01	**3.43E−03***	3.73E−01	4.99E+01	2.66E−01
	(1.62E−02)	(3.36E−02)	(3.87E−02)	(1.80E−03)	(2.26E−01)	(1.58E+01)	(1.68E−01)
f_8	**− 2.09E+04***	**− 2.09E+04**	− 1.24E+04	−9.18E+03	− 3.32E+03	−5.15E+03	− 2.03E+04
	(1.57E+00)	(1.11E+01)	(1.06E+03)	(1.31E+03)	(7.08E+02)	(4.03E+02)	(3.31E+03)
f_9	**1.02E+00***	2.77E+00	2.53E+02	3.76E+00	5.76E+01	5.37E+02	7.58E+01
	(8.78E−01)	(8.40E−01)	(4.97E+01)	(3.37E+00)	(1.08E+01)	(2.68E+01)	(2.17E+01)
f_{10}	2.48E−01	1.20E+00	3.55E+00	**3.56E−11***	1.51E+00	1.85E+01	1.66E+01
	(2.84E−01)	(2.04E−01)	(3.05E+00)	(1.79E−11)	(6.94E−01)	(3.33E−01)	(8.36E−01)
f_{11}	5.82E−01	1.09E+00	1.09E+00	**6.72E−03***	1.29E+02	3.64E+02	6.54E+01
	(2.29E−01)	(2.88E−02)	(1.90E−02)	(1.15E−02)	(1.66E+01)	(4.62E+01)	(2.25E+01)
f_{12}	**4.15E−03***	1.95E−02	5.18E+00	1.20E−01	4.31E+00	7.17E+07	4.96E+02
	(2.27E−02)	(1.48E−02)	(2.48E+00)	(4.39E−02)	(1.75E+00)	(3.23E+07)	(1.67E+03)
f_{13}	**1.86E−02***	4.06E−01	6.75E+00	2.17E+00	5.20E+01	2.15E+08	1.53E+05
	(1.77E−02)	(1.17E−01)	(1.03E+01)	(3.74E−01)	(1.37E+01)	(6.94E+07)	(2.62E+05)
f_{14}	**9.98E−01**	**9.98E−01**	**9.98E−01**	3.55E+00	6.29E+00	1.17E+00	**9.98E−01**
	(2.72E−16)	(1.98E−16)	(4.05E−11)	(3.61E+00)	(4.28E+00)	(3.22E−01)	(5.65E−16)
f_{15}	1.08E−03	**5.21E−04***	4.72E−03	5.05E−03	4.03E−03	1.55E−03	1.73E−02
	(5.39E−04)	(3.78E−04)	(7.96E−03)	(8.59E−03)	(2.95E−03)	(5.75E−04)	(1.78E−02)
f_{16}	**− 1.03E+00**	**− 1.03E+00**	**− 1.03E+00**	**− 1.03E+00**	**− 1.03E+00**	**− 1.03E+00**	− 6.99E−01
	(3.76E−14)	(1.24E−15)	(3.14E−07)	(1.71E−06)	(5.22E−16)	(2.94E−03)	(3.77E−01)
f_{17}	**3.98E−01**	**3.98E−01**	**3.98E−01**	**3.98E−01**	**3.98E−01**	4.00E−01	5.04E−01
	(6.01E−14)	(8.96E−16)	(1.13E−07)	(7.63E−07)	(0.00E+00)	(2.52E−03)	(6.73E−02)
f_{18}	**3.00E+00**	**3.00E+00**	**3.00E+00**	**3.00E+00**	**3.00E+00**	**3.03E+00**	**3.00E+00**
	(5.90E−12)	(7.63E−14)	(2.74E−06)	(6.41E−05)	(2.68E−15)	(2.55E−02)	(0.00E+00)
f_{19}	**− 3.86E+00**	**− 3.86E+00**	**− 3.86E+00**	**− 3.86E+00**	**− 3.86E+00**	**− 3.86E+00**	− 3.27E+00
	(4.88E−12)	(3.58E−14)	(2.23E−06)	(2.79E−03)	(2.32E−15)	(2.92E−03)	(4.46E−01)
f_{20}	− 3.28E+00	− 3.30E+00	− 3.24E+00	− 3.26E+00	**− 3.32E+00***	− 3.12E+00	− 1.96E+00
	(5.83E−02)	(4.51E−02)	(5.81E−02)	(7.07E−02)	(1.37E−15)	(7.20E−02)	(5.05E−01)
f_{21}	− 6.14E+00	− 5.65E+00	− 6.70E+00	**− 9.59E+00**	− 6.78E+00	− 3.97E+00	− 5.90E+00
	(3.44E+00)	(3.37E+00)	(2.76E+00)	(1.74E+00)	(3.68E+00)	(1.27E+00)	(3.42E+00)
f_{22}	− 6.27E+00	− 5.21E+00	− 8.50E+00	− 1.02E+01	**− 1.04E+01***	− 4.78E+00	− 3.48E+00
	(3.26E+00)	(3.28E+00)	(3.02E+00)	(9.63E−01)	(9.33E−16)	(1.76E+00)	(2.00E+00)
f_{23}	− 6.60E+00	− 5.20E+00	− 7.89E+00	− 9.90E+00	**− 1.03E+01***	− 5.10E+00	− 6.39E+00
	(3.32E+00)	(3.39E+00)	(3.39E+00)	(1.96E+00)	(1.47E+00)	(1.89E+00)	(3.72E+00)

* means that there are significant differences with all comparative results against the best

Table 13 Parameters settings of the metaheuristic algorithms

Algorithm	Parameter	Value
GA	Crossover probability	0.9
	Mutation probability	0.01
	Selection mechanism	Roulette wheel
PSO	Inertial constant	0.3
	Cognitive constant	1
	Social constant	1
BBO	Habitat modification probability	1.0
	Immigration probability	[0,1]
	Step size	1.0
	Maximum immigration	1.0
	Migration rates	1.0
	Mutation probability	0
MVO	Minimum wormhole existence probability	0.2
	Maximum wormhole existence probability	1.0
	P	6
GSA	α	20
	G_0	100
	K_0	Linearly decreases [N,0]
GWO	a	Linearly decreases over [2,0]

Table 14 P values of the Wilcoxon test of AβHC results vs other algorithms ($p \geq 0.05$ are underlined)

Benchmark	βHC	MVO	GWO	GSA	PSO	GA
f_1	3.02E−11	3.02E−11	3.02E−11	3.02E−11	3.02E−11	3.02E−11
f_2	3.02E−11	3.02E−11	3.02E−11	3.02E−11	3.02E−11	3.02E−11
f_3	5.07E−10	1.33E−10	3.02E−11	3.02E−11	3.02E−11	3.02E−11
f_4	1.02E−05	1.11E−06	3.02E−11	7.38E−10	3.02E−11	2.99E−11
f_5	5.60E−07	1.07E−07	3.99E−04	8.35E−08	3.02E−11	3.02E−11
f_6	3.02E−11	3.02E−11	3.02E−11	3.02E−11	3.02E−11	3.02E−11
f_7	8.10E−10	2.60E−08	3.02E−11	3.02E−11	3.02E−11	7.04E−07
f_8	3.02E−11	3.02E−11	3.02E−11	3.02E−11	3.02E−11	3.02E−11
f_9	3.35E−08	3.02E−11	2.42E−02	3.02E−11	3.02E−11	3.02E−11
f_{10}	3.69E−11	3.02E−11	3.02E−11	1.29E−09	3.02E−11	3.02E−11
f_{11}	3.02E−11	3.02E−11	1.44E−11	3.02E−11	3.02E−11	3.02E−11
f_{12}	5.57E−10	3.02E−11	1.61E−10	3.02E−11	3.02E−11	3.02E−11
f_{13}	3.02E−11	3.02E−11	3.02E−11	3.02E−11	3.02E−11	3.02E−11
f_{14}	NaN	NaN	2.81E−07	1.21E−12	4.57E−12	1.69E−14
f_{15}	6.35E−07	7.51E−01	9.63E−02	3.82E−10	7.29E−03	1.31E−08
f_{16}	NaN	1.21E−12	5.73E−11	NaN	1.21E−12	1.19E−12
f_{17}	NaN	1.21E−12	1.21E−12	NaN	1.21E−12	1.21E−12
f_{18}	NaN	1.21E−12	1.21E−12	NaN	1.21E−12	NaN
f_{19}	NaN	1.21E−12	1.21E−12	NaN	1.21E−12	1.21E−12
f_{20}	3.15E−07	3.97E−07	4.42E−05	2.56E−12	1.40E−09	1.91E−11
f_{21}	6.50E−01	8.06E−01	1.43E−01	2.74E−01	1.84E−01	7.20E−02
f_{22}	1.19E−01	3.77E−01	3.08E−02	1.23E−07	1.75E−01	4.24E−07
f_{23}	2.13E−02	6.29E−01	3.45E−01	4.16E−06	1.64E−01	1.26E−02

Table 15 The characteristics of the IEEE-CEC2015 benchmark functions

Key	Function	Range	Category	$f(x^*)$
f_{24}	Rotated high conditioned elliptic function	$x_i \in [100, -100]$	UN	100
f_{25}	Rotated cigar function	$x_i \in [100, -100]$	UN	200
f_{26}	Shifted and rotated ackleys function	$x_i \in [100, -100]$	MN	300
f_{27}	Shifted and rotated rastrigin function	$x_i \in [100, -100]$	MN	400
f_{28}	Shifted and rotated schwefel function	$x_i \in [100, -100]$	MN	500
f_{29}	Hybrid function 1 ($N = 3$)	$x_i \in [100, -100]$	H	600
f_{30}	Hybrid function 2 ($N = 4$)	$x_i \in [100, -100]$	H	700
f_{31}	Hybrid function 3($N = 5$)	$x_i \in [100, -100]$	H	800
f_{32}	Composition function 1 ($N = 3$)	$x_i \in [100, -100]$	C	900
f_{33}	Composition function 2 ($N = 3$)	$x_i \in [100, -100]$	C	1000
f_{34}	Composition function 3 ($N = 5$)	$x_i \in [100, -100]$	C	1100
f_{35}	Composition function 4 ($N = 5$)	$x_i \in [100, -100]$	C	1200
f_{36}	Composition function 5 ($N = 5$)	$x_i \in [100, -100]$	C	1300
f_{37}	Composition function 6 ($N = 7$)	$x_i \in [100, -100]$	C	1400
f_{38}	Composition function 7 ($N = 10$)	$x_i \in [100, -100]$	C	1500

Table 16 Key to comparative methods for CEC2015 benchmark functions

Key	Method name	References
HumanCog	HumanCog: a cognitive architecture	(Al-Dujaili et al. 2015)
SA	Simulated annealing	(Al-Dujaili et al. 2015)
ABC-X-LS	A configurable generalized artificial bee colony algorithm with local search strategies	(Aydın and Sfflτzle 2015)
SaDPSO	A self-adaptive dynamic particle swarm optimizer	(Liang et al. 2015)
SPS-L-SHADE-EIG	A self-optimization approach for L-SHADE incorporated with eigenvector-based crossover and successful-parent-selecting framework	(Guo et al. 2015)
COA	Cooperation of optimization algorithms: a simple hierarchical model	(Poláková et al. 2015)
DEsPA	A Differential evolution algorithm with success-based parameter adaptation	(Awad et al. 2015)
dynFWA	Dynamic search fireworks algorithm with covariance mutation	(Yu et al. 2015)
hCC	Hybrid cooperative co-evolution	(El-Abd 2015)
LSHADE-ND	Neurodynamic differential evolution algorithm	(Sallam et al. 2015)
MVMO	Mean-variance mapping optimization	(Rueda and Erlich 2015)
EBO	Ecogeography-based optimization	(Zheng and Wu 2015)
HOCO	Higher order cognitive optimization	(Tanweer et al. 2017)
PgAFWA	A best firework updating information guided adaptive fireworks algorithm	(Zhao et al. 2019)
NS-ABC	Natural selection methods for artificial bee colony	(Awadallah et al. 2018)
C-EGPSO	Combined-evolutionary game particle swarm optimization	(Leboucher et al. 2018)
HFPSO	A hybrid firefly and particle swarm optimization algorithm	(Aydilek 2018)
DPABC	Dual-population framework of artificial bee colony	(Cui et al. 2018)
DPGABC	Dual-population framework of global-best artificial bee colony	(Cui et al. 2018)
DPCABC	Dual-population framework of modified artificial bee colony	(Cui et al. 2018)
CABC	Modified artificial bee colony	(Cui et al. 2018)

Table 17 Average error rate obtained in CEC2015 benchmark functions for problem size of 10 variables

Algorithm	f_{24}	f_{25}	f_{26}	f_{27}	f_{28}	f_{29}	f_{30}	f_{31}
HumanCog	3.27E+09	7.80E+04	1.12E+01	2.09E+03	2.82E+00	3.63E+00	2.74E+01	7.77E+03
SA	2.14E+08	4.55E+04	1.34E+01	1.30E+03	**1.77E+00**	4.62E+00	3.40E+01	7.61E+02
ABC-X-LS	**0.00E+00**	**0.00E+00**	1.76E+01	**2.10E−02**	9.55E+00	9.59E+00	4.81E−01	1.59E+00
SaDPSO	3.61E+01	1.36E+02	2.00E+01	4.49E+00	1.41E+02	3.04E+02	6.73E−01	6.86E+01
SPS-L-SHADE-EIG	**0.00E+00**	**0.00E+00**	1.80E+01	1.17E+00	2.69E+01	**1.23E−01**	2.78E+02	8.81E+04
COA	**0.00E+00**	**0.00E+00**	1.90E+01	3.05E+00	1.03E+02	3.31E+00	2.30E−01	4.79E−01
DEsPA	**0.00E+00**	**0.00E+00**	1.67E+01	3.56E+00	5.19E+01	1.59E+00	3.42E−01	1.95E−01
dynFWA	1.11E+05	8.86E+03	2.00E+01	1.67E+01	5.18E+02	1.74E+03	1.45E+00	1.96E+03
hCC	**0.00E+00**	**0.00E+00**	1.92E+01	3.93E−01	7.16E+00	1.26E−01	4.94E−02	**3.19E−02**
LSHADE-ND	**0.00E+00**	**0.00E+00**	1.61E+01	2.77E+00	1.25E+01	6.14E−01	5.53E−02	3.04E−01
MVMO	**0.00E+00**	**0.00E+00**	1.09E+01	2.26E+00	2.20E+01	5.89E+00	5.80E−02	2.47E−01
TEBO	**0.00E+00**	**0.00E+00**	1.73E+01	5.07E+00	9.82E+01	1.88E+01	1.40E−01	5.68E+00
HOCO	3.19E+07	3.19E+04	5.13E+00	9.97E+02	2.87E+00	6.18E−01	6.80E−01	5.85E+00
PgAFWA	3.46E+05	3.30E+05	3.20E+02	4.13E+02	8.81E+02	3.19E+03	7.01E+02	3.31E+03
NS-ABC	1.21E+05	8.59E+00	**0.00E+00**	1.66E+00	5.85E+00	2.87E+02	**4.22E−02**	3.43E+02
C-EGPSO	1.27E+07	3.42E+04	6.53E+00	1.69E+03	2.71E+00	5.69E−01	6.99E−01	5.68E+00
HFPSO	1.38E+07	3.85E+04	3.07E+02	1.32E+03	5.03E+02	6.01E+02	7.01E+02	8.08E+02
DPABC	1.11E+06	3.61E+02	1.87E+01	8.49E+00	1.64E+02	5.30E+03	5.07E−01	3.34E+04
DPGABC	1.20E+06	1.29E+03	1.85E+01	4.95E+00	1.22E+02	6.10E+03	4.58E−01	3.28E+04
DPCABC	1.43E+06	3.10E+03	1.83E+01	4.53E+00	1.58E+02	7.20E+03	4.81E−01	3.33E+04
CABC	1.05E+06	4.73E+03	1.88E+01	5.85E+00	2.17E+02	7.40E+03	6.06E−01	8.04E+03
βHC	1.17E+06	1.31E+07	2.00E+01	1.66E+01	3.30E+02	1.85E+04	1.81E+00	3.92E+03
AβHC	1.77E+06	7.88E+03	2.00E+01	1.30E+01	3.39E+02	2.44E+04	1.44E+00	4.95E+03

Algorithm	f_{32}	f_{33}	f_{34}	f_{35}	f_{36}	f_{37}	f_{38}
HumanCog	4.16E+00	1.19E+06	2.16E+01	3.08E+02	4.33E+02	2.15E+02	4.74E+02
SA	4.37E+00	1.01E+06	2.65E+01	3.88E+02	3.88E+02	2.23E+02	5.42E+02
ABC-X-LS	1.00E+02	2.16E+02	2.08E+02	**1.01E+02**	2.92E+01	1.23E+03	**1.00E+02**
SaDPSO	1.00E+02	3.97E+02	1.60E+02	**1.01E+02**	2.90E+01	9.33E+02	**1.00E+02**
SPS-L-SHADE-EIG	1.00E+02	2.17E+02	4.72E+01	**1.01E+02**	3.04E+02	6.56E+02	**1.00E+02**
COA	1.00E+02	1.43E+02	1.49E+01	1.11E+02	9.28E−02	6.41E+03	**1.00E+02**
DEsPA	1.06E+02	**7.56E+00**	9.46E+01	**1.01E+02**	1.77E+01	3.09E+02	2.05E+02
dynFWA	1.00E+02	5.47E+02	1.85E+02	1.13E+02	1.21E−01	6.30E+03	**1.00E+02**
hCC	1.00E+02	1.43E+02	8.15E+00	1.02E+02	3.05E−02	3.75E+03	**1.00E+02**
LSHADE-ND	1.00E+02	2.17E+02	1.24E+02	**1.01E+02**	3.05E−02	4.65E+03	**1.00E+02**
MVMO	1.00E+02	2.17E+02	8.41E+01	**1.01E+02**	3.04E−02	1.00E+02	**1.00E+02**
TEBO	1.00E+02	1.61E+02	1.42E+02	**1.01E+02**	2.86E+01	2.92E+03	**1.00E+02**
HOCO	3.85E+00	4.46E+05	6.94E+00	2.36E+02	3.31E+02	2.04E+02	2.72E+02
PgAFWA	1.00E+03	2.45E+03	1.28E+03	1.30E+03	1.33E+03	4.15E+03	1.60E+03
NS-ABC	**2.16E−01**	4.60E+02	8.07E+00	**1.01E+02**	2.08E+01	5.35E+01	**1.00E+02**
C-EGPSO	3.83E+00	3.35E+05	**6.83E+00**	1.88E+02	3.32E+02	2.03E+02	3.88E+02
HFPSO	9.04E+02	3.31E+05	1.11E+03	1.40E+03	1.65E+03	1.60E+03	1.92E+03
DPABC	9.66E+01	4.71E+03	1.36E+02	1.03E+02	3.05E−02	4.63E+02	**1.00E+02**
DPGABC	1.00E+02	4.00E+03	1.23E+02	1.02E+02	3.05E−02	**5.40E−02**	**1.00E+02**
DPCABC	1.00E+02	8.66E+03	1.44E+02	1.03E+02	3.07E−02	1.12E+03	**1.00E+02**
CABC	1.00E+02	2.70E+03	8.87E+01	1.03E+02	3.07E−02	1.94E+03	**1.00E+02**
βHC	1.01E+02	3.51E+03	1.04E+02	1.05E+02	3.71E−02	2.97E+03	1.04E+02
AβHC	1.01E+02	3.26E+03	1.58E+02	1.05E+02	**3.02E−02**	4.29E+03	**1.00E+02**

Table 18 Average error rate obtained in CEC2015 benchmark functions for problem size of 30 variables

Algorithm	f_{24}	f_{25}	f_{26}	f_{27}	f_{28}	f_{29}	f_{30}	f_{31}
HumanCog	4.74E+10	1.13E+05	4.13E+01	7.99E+03	4.39E+00	5.03E+00	8.86E+01	5.24E+06
SA	3.81E+09	1.63E+05	4.69E+01	5.05E+03	**3.80E+00**	5.39E+00	1.10E+02	2.48E+05
ABC-X-LS	**0.00E+00**	**0.00E+00**	**2.00E+01**	**2.19E+00**	7.78E+01	4.82E+02	6.48E+00	1.69E+02
SaDPSO	1.93E−02	2.88E+02	**2.00E+01**	4.25E+01	2.52E+03	1.45E+03	9.53E+00	1.63E+03
SPS-L-SHADE-EIG	**0.00E+00**	**0.00E+00**	**2.00E+01**	1.03E+01	1.54E+03	9.91E+01	2.44E+00	**2.11E+01**
COA	**0.00E+00**	**0.00E+00**	**2.00E+01**	1.58E+01	1.08E+03	1.23E+03	5.86E+00	2.16E+02
DEsPA	**0.00E+00**	**0.00E+00**	2.01E+01	9.71E+00	1.85E+03	1.61E+02	3.09E+00	2.55E+01
dynFWA	6.17E+05	3.31E+03	**2.00E+01**	1.30E+02	3.38E+03	2.69E+04	1.46E+01	2.40E+04
hCC	**0.00E+00**	**0.00E+00**	2.02E+01	1.07E+01	6.06E+02	2.50E+02	3.26E+00	8.06E+01
LSHADE-ND	**0.00E+00**	**0.00E+00**	**2.00E+01**	1.08E+01	1.28E+03	1.74E+02	5.06E+00	3.24E+01
MVMO	**0.00E+00**	**0.00E+00**	**2.00E+01**	9.54E+00	1.14E+03	3.10E+02	3.41E+00	8.14E+01
TEBO	3.69E+02	4.545E+07	**2.00E+01**	4.41E+01	1.96E+03	6.98E+02	4.42E+00	1.21E+02
HOCO	7.74E+08	8.49E+04	2.46E+01	4.92E+03	4.36E+00	7.09E−01	6.60E−01	6.51E+02
PgAFWA	1.70E+06	3.98E+05	3.20E+02	5.09E+02	3.69E+03	1.99E+05	7.15E+02	1.12E+05
NS-ABC	6.25E+05	5.45E+00	**2.00E+01**	3.39E+01	1.17E+03	1.79E+05	4.20E+00	8.85E+04
C-EGPSO	1.51E+08	1.02E+05	2.75E+01	6.00E+03	3.38E+00	**5.70E−01**	**4.88E−01**	3.91E+02
HFPSO	1.18E+09	8.57E+04	3.26E+02	5.12E+03	5.04E+02	6.01E+02	7.01E+02	2.64E+03
DPABC	2.69E+06	4.11E+02	2.01E+01	7.35E+01	1.88E+03	9.77E+05	6.45E+00	3.02E+05
DPGABC	2.69E+06	1.32E+03	2.01E+01	4.91E+01	1.80E+03	1.28E+06	6.70E+00	3.74E+05
DPCABC	3.72E+06	2.51E+03	2.02E+01	4.42E+01	1.87E+03	2.34E+06	7.86E+00	3.46E+05
CABC	4.73E+06	4.23E+03	2.03E+01	5.03E+01	2.22E+03	1.81E+06	1.03E+01	3.15E+05
βHC	1.52E+08	1.66E+10	2.09E+01	2.82E+02	6.40E+03	4.14E+06	3.61E+01	7.28E+05
AβHC	1.86E+07	1.93E+08	**2.00E+01**	1.19E+02	3.18E+03	4.82E+06	2.85E+01	1.16E+06

Algorithm	f_{32}	f_{33}	f_{34}	f_{35}	f_{36}	f_{37}	f_{38}
HumanCog	1.39E+01	5.60E+07	2.76E+02	1.60E+03	8.35E+02	3.94E+02	1.49E+03
SA	1.41E+01	2.64E+07	1.84E+02	1.35E+03	5.64E+02	3.68E+02	1.51E+03
ABC-X-LS	1.03E+02	8.86E+02	3.06E+02	**1.03E+02**	9.43E+01	1.40E+04	**1.00E+02**
SaDPSO	1.03E+02	6.69E+03	3.20E+02	1.05E+02	1.01E+02	2.00E+04	**1.00E+02**
SPS-L-SHADE-EIG	1.02E+02	3.53E+02	3.01E+02	**1.03E+02**	2.59E+02	3.16E+04	**1.00E+02**
COA	1.06E+02	6.78E+02	3.98E+02	1.07E+02	**1.04E−02**	3.85E+04	**1.00E+02**
DEsPA	1.80E+02	**1.71E+02**	3.11E+02	1.08E+02	8.13E+01	2.81E+04	2.72E+02
dynFWA	1.08E+02	3.15E+04	6.72E+02	1.17E+02	2.62E−02	4.49E+04	**1.00E+02**
hCC	1.06E+02	6.75E+02	4.67E+02	1.06E+02	2.62E−02	3.23E+04	**1.00E+02**
LSHADE-ND	1.03E+02	5.95E+02	4.05E+02	1.04E+02	2.60E−02	3.22E+04	**1.00E+02**
MVMO	1.03E+02	6.51E+02	3.01E+02	1.04E+02	2.71E−02	3.16E+04	**1.00E+02**
TEBO	1.08E+02	6.21E+02	4.81E+02	1.06E+02	9.87E+01	3.45E+04	**1.00E+02**
HOCO	1.36E+01	6.22E+06	**3.45E+01**	7.85E+02	3.99E+02	**2.54E+02**	8.89E+02
PgAFWA	1.12E+03	1.73E+05	1.68E+03	1.31E+03	1.42E+03	2.22E+04	1.60E+03
NS-ABC	1.04E+02	1.10E+05	2.82E+02	1.05E+02	8.33E+01	2.16E+04	**1.00E+02**
C-EGPSO	**1.33E+01**	4.82E+06	3.51E+01	5.63E+02	4.15E+02	2.55E+02	1.01E+03
HFPSO	9.13E+02	5.47E+06	1.13E+03	1.78E+03	1.69E+03	1.65E+03	2.45E+03
DPABC	1.04E+02	6.14E+05	3.26E+02	1.06E+02	3.11E−02	3.07E+04	**1.00E+02**
DPGABC	1.04E+02	6.64E+05	3.26E+02	1.06E+02	2.83E−02	3.15E+04	**1.00E+02**
DPCABC	1.04E+02	1.23E+06	3.43E+02	1.06E+02	2.79E−02	3.16E+04	**1.00E+02**
CABC	1.04E+02	1.09E+06	3.39E+02	1.06E+02	2.80E−02	3.17E+04	**1.00E+02**
βHC	1.66E+02	2.62E+06	7.15E+02	1.39E+02	5.59E−01	4.41E+04	4.60E+02
AβHC	1.76E+02	2.80E+06	8.49E+02	1.23E+02	4.43E−02	3.37E+04	1.08E+02

Table 19 Average error rate obtained in CEC2015 benchmark functions for problem size of 50 variables

Algorithm	f_{24}	f_{25}	f_{26}	f_{27}	f_{28}	f_{29}	f_{30}	f_{31}
ABC-X-LS	**0.00E+00**	**0.00E+00**	**2.00E+01**	6.09E+00	**2.90E+02**	1.88E+03	**1.51E+01**	1.27E+03
SaDPSO	2.78E+02	1.96E+02	**2.00E+01**	9.12E+01	4.82E+03	3.18E+03	3.48E+01	2.52E+03
SPS-L-SHADE-EIG	**0.00E+00**	**0.00E+00**	**2.00E+01**	**2.30E+00**	2.68E+03	**2.39E+02**	2.74E+01	**6.58E+01**
COA	1.53E−01	**0.00E+00**	2.01E+01	3.43E+01	2.03E+03	2.66E+03	3.56E+01	1.66E+03
DEsPA	9.50E+01	**0.00E+00**	2.01E+01	1.69E+01	4.25E+03	9.44E+02	4.06E+01	2.93E+02
dynFWA	1.80E+06	5.17E+03	**2.00E+01**	2.44E+02	5.78E+03	8.81E+04	5.70E+01	6.88E+04
hCC	**0.00E+00**	**0.00E+00**	2.04E+01	2.13E+01	1.04E+03	2.33E+04	4.17E+01	3.11E+03
LSHADE-ND	2.12E+03	**0.00E+00**	**2.00E+01**	2.15E+01	3.01E+03	1.67E+03	3.48E+01	2.69E+02
MVMO	**0.00E+00**	**0.00E+00**	**2.00E+01**	1.66E+02	9.93E+03	3.67E+03	4.58E+01	1.99E+03
TEBO	1.14E+06	9.49E+02	2.01E+01	4.39E+02	1.22E+04	3.50E+05	7.39E+01	1.99E+05
PgAFWA	4.57E+06	1.91E+06	3.21E+02	6.13E+02	7.05E+03	4.19E+05	7.37E+02	5.55E+05
βHC	9.93E+08	6.64E+10	2.11E+01	6.35E+02	1.28E+04	3.19E+07	2.13E+02	8.33E+06
AβHC	1.05E+08	3.48E+09	2.06E+01	3.36E+02	8.60E+03	1.14E+07	1.08E+02	7.91E+06

Algorithm	f_{32}	f_{33}	f_{34}	f_{35}	f_{36}	f_{37}	f_{38}
ABC-X-LS	1.04E+02	2.00E+03	**3.00E+02**	**1.04E+02**	1.77E+02	4.46E+04	**1.00E+02**
SaDPSO	1.05E+02	1.79E+04	3.88E+02	1.08E+02	1.93E+02	**2.78E+04**	**1.00E+02**
SPS-L-SHADE-EIG	1.04E+02	8.35E+02	**3.00E+02**	1.10E+02	7.62E−02	6.35E+04	**1.00E+02**
COA	**1.02E+02**	1.39E+03	4.59E+02	1.14E+02	**2.54E−02**	5.27E+04	**1.00E+02**
DEsPA	1.90E+02	**5.82E+02**	3.78E+02	1.08E+02	1.48E+02	3.13E+04	2.84E+02
dynFWA	1.11E+02	4.21E+04	1.08E+03	1.75E+02	1.17E−01	5.49E+04	**1.00E+02**
hCC	1.05E+02	1.59E+03	5.40E+02	1.61E+02	8.01E−02	6.46E+04	**1.00E+02**
LSHADE−ND	1.04E+02	1.02E+03	4.12E+02	1.18E+02	7.76E−02	6.37E+04	**1.00E+02**
MVMO	1.08E+02	3.91E+03	1.01E+03	1.26E+02	1.06E−01	1.09E+05	1.01E+02
TEBO	1.12E+02	4.32E+04	2.55E+03	1.21E+02	4.15E+02	1.24E+05	1.45E+02
PgAFWA	1.22E+03	4.48E+05	2.23E+03	1.31E+03	1.52E+03	5.42E+04	1.60E+03
βHC	3.15E+02	9.83E+06	2.14E+03	1.91E+02	7.88E+00	1.14E+05	1.66E+04
AβHC	2.49E+02	1.49E+07	1.51E+03	1.79E+02	3.14E−01	6.96E+04	1.30E+02

ventionally known that as the optimization problem size is large, as the complexity in achieving the optimal solution becomes harder. Although, the proposed AβHC is not able to record any new best results in comparison with other methods, the obtained results are either better than other methods or falls within the range. I believe that although AβHC is not designed for IEEE-CEC2015 competition like other comparative methods, it performs well in accordance to others.

5 Conclusion and future work

In this paper, an adaptive version of β-hill climbing algorithm (i.e., AβHC) is proposed for global optimization functions. In the original β-hill climbing (i.e., βHC), there are two control parameters responsible for achieving the right balance between exploration and exploitation: $\mathcal{N}$ and β. In the new version, these two parameters are deterministically changed during the search in which $\mathcal{N}$ probability is decreased from 1 to 0 according to the value of K which determines the slop of decreasing. The β parameter starts form an initial value determined by $\beta_{\max}$, and it deterministically decreases till $\beta_{\min}$. The adaptive version is evaluated using 23 global optimization test functions from the literature.

The sensitivity of the adaptive β-hill climbing algorithm to its parameters is analyzed using various convergence cases. In conclusion, as the value of K increases, the results gets better. Furthermore, the value range of $\beta_{\min}$ and $\beta_{\max}$ should be wide enough to ensure proper diversification. Finally, the parameter dimensions is another factor that affects the performance of adaptive β-hill climbing algorithm where as the value of dimensions increases, the performance degrades.

For comparative evaluation, three set of experiments are conducted: (i) comparing AβHC with βHC, (ii) comparing AβHC with other local search-based methods, and (iii) comparing AβHC with other advanced metaheuristic methods using the same test functions. In the set of experiments, AβHC algorithm was able to outperform the original βHC algorithm in almost all test functions. For second set of experiments, AβHC algorithm was compared with five local search-based methods using the same test functions. Remarkably, AβHC was able to outperform other comparative local search-based methods in almost all test functions. Finally, the AβHC was compared with five other metaheuristic algorithms using the same test functions. The AβHC achieved the best recorded results in 10 out 23 test functions, and it obtained very promising results for other functions.

The experiments were also expanded to tackle the recent benchmark functions established in IEEE-CEC2015. These benchmarks include 15 global optimization functions with different characteristics and complexity. The proposed AβHC was experimented based on these functions with various function dimensions ($D = 10$,$D = 30$, and $D = 50$). The results were directly compared with 21 well-established methods heuristically proposed to compete with them using these competition benchmarks. Although these methods very hard to compete with, AβHC was able to yield new results for two IEEE-CEC2015 benchmark functions when $D = 10$ and very competitive results for the others.

Based on these findings, the AβHC algorithm can be further studied using other combinatorial and discrete optimization problems. The way of changing the value of $\mathcal{N}$ and β parameters can be studied further using the self-adaptive context.

Compliance with ethical standards

Conflict of interest The authors declare that there is no conflict of interest regarding the publication of this paper.

Ethical approval This article does not contain any studies with animals performed by any of the authors.

References

Abualigah LM, Khader AT, Al-Betar MA(2017a) β-hill climbing technique for the text document clustering. In: New trends in information technology NTIT2017 conference, Amman, Jordan. IEEE, pp 1–6

Abualigah LM, Khadery AT, Al-Betar MA, Alyasseri ZAA, Alomari OA,Hanandehk ES(2017b) Feature selection with β-hill climbing search for text clustering application. In: Second palestinian international conference on information and communication technology (PICICT 2017), Gaza, Palestine. IEEE, pp 22–27

Al-Betar MA, Awadallah MA, Bolaji AL, Alijla BO (2017) β-hill climbing algorithm for sudoku game. In: Second Palestinian international conference on information and communication technology (PICICT 2017), Gaza, Palestine. IEEE, pp 84–88

Al-Betar MA, Khader AT, Doush IA (2014) Memetic techniques for examination timetabling. Ann Oper Res 218(1):23–50

Al-Betar MA (2017) β-hill climbing: an exploratory local search. Neural Comput Appl 28(1):153–168

Al-Dujaili A, Subramanian K, Suresh S (2015) Humancog: a cognitive architecture for solving optimization problems. In: 2015 IEEE congress on, evolutionary computation (CEC). IEEE, pp 3220–3227

Aleti A, Moser I (2016) A systematic literature review of adaptive parameter control methods for evolutionary algorithms. ACM Comput Surv (CSUR) 49(3):56

Alsukni E, Arabeyyat OS, Awadallah MA, Alsamarraie L, Abu-Doush I, Al-Betar MA (2017) Multiple-reservoir scheduling using B-hill climbing algorithm. J Intell Syst. https://doi.org/10.1515/jisys-2017-0159

Alyasseri ZAA, Khader AT, Al-Betar MA (2017) Optimal EEG signals denoising using hybrid β-hill climbing algorithm and wavelet transform. In: ICISPC ’17, Penang, Malaysia. ACM, pp 5–11

Alyasseri ZAA, Khader AT, Al-Betar MA, Awadallah MA (2018) Hybridizing β-hill climbing with wavelet transform for denoising ECG signals. Inf Sci 429:229–246

Angeline PJ, Angeline PJ (1995) Adaptive and self-adaptive evolutionary computations. In: Computational intelligence: a dynamic systems perspective. IEEE Press, pp 152–163

Awad N, Ali MZ, Reynolds RG (2015) A differential evolution algorithm with success-based parameter adaptation for cec2015 learning-based optimization. In: 2015 IEEE congress on, evolutionary computation (CEC). IEEE, pp 1098–1105

Awadallah MA, Al-Betar MA, Bolaji AL, Alsukhni EM, Al-Zoubi H (2018) Natural selection methods for artificial bee colony with new versions of onlooker bee. Soft Comput

Aydilek İB (2018) A hybrid firefly and particle swarm optimization algorithm for computationally expensive numerical problems. Appl Soft Comput 66:232–249

Aydın D, Sffltzle T (2015) A configurable generalized artificial bee colony algorithm with local search strategies. In: 2015 IEEE congress on, evolutionary computation (CEC). IEEE, pp 1067–1074

Blum C, Roli A (2003) Metaheuristics in combinatorial optimization: overview and conceptual comparison. ACM Comput Surv (CSUR) 35(3):268–308

BoussaïD I, Lepagnot J, Siarry P (2013) A survey on optimization metaheuristics. Inf Sci 237:82–117

Corana A, Marchesi M, Martini C, Ridella S (1987) Minimizing multimodal functions of continuous variables with the “simulated annealing” algorithm corrigenda for this article is available here. ACM Trans Math Softw (TOMS) 13(3):262–280

Cui L, Li G, Luo Y, Chen F, Ming Z, Lu N, Lu J (2018) An enhanced artificial bee colony algorithm with dual-population framework. Swarm Evolut Comput 43:184–206

Dragoi E-N, Dafinescu V (2016) Parameter control and hybridization techniques in differential evolution: a survey. Artif Intell Rev 45(4):447–470

Eiben ÁE, Hinterding R, Michalewicz Z (1999) Parameter control in evolutionary algorithms. IEEE Trans Evolut Comput 3(2):124–141

El-Abd M (2015) Hybrid cooperative co-evolution for the cec15 benchmarks. In: 2015 IEEE congress on, evolutionary computation (CEC). IEEE, pp 1053–1058

Feo TA, Resende MGC (1995) Greedy randomized adaptive search procedures. J Global Optim 6(2):109–133

Glover F (1986) Future paths for integer programming and links to artificial intelligence. Comput Oper Res 13(5):533–549

Guo S-M, Tsai JS-H, Yang C-C, Hsu P-H (2015) A self-optimization approach for l-shade incorporated with eigenvector-based crossover and successful-parent-selecting framework on cec 2015 benchmark set. In: 2015 IEEE congress on, evolutionary computation (CEC). IEEE, pp 1003–1010

Guo Z, Liu G, Li D, Wang S (2017) Self-adaptive differential evolution with global neighborhood search. Soft Comput 21(13):3759–3768

Guo Z, Yang H, Wang S, Zhou C, Liu X (2018) Adaptive harmony search with best-based search strategy. Soft Comput 22(4):1335–1349

Hansen P, Mladenovi'c N (1999) An introduction to variable neighborhood search. In: Voß S, Martello S, Osman IH, Roucairol C (eds) Metaheuristics: advances and trends in local search paradigms for optimization, chapter 30. Kluwer Academic Publishers, Dordrecht, pp 433–458

Kirkpatrick S, Gelatt CD, Vecchi MP (1983) Optimization by simulated annealing. Science 220(1):671–680

Leboucher C, Shin H-S, Chelouah R, Le Menec S, Siarry P, Formoso M, Tsourdos A, Kotenkoff A (2018) An enhanced particle swarm optimisation method integrated with evolutionary game theory. IEEE Transactions on Games, pp 1–11

Liang J, Qu B, Suganthan P, Chen Q (2014) Problem definitions and evaluation criteria for the cec 2015 competition on learning-based real-parameter single objective optimization. Technical Report201411A, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore

Liang J, Guo L, Liu R, Qu B (2015) A self-adaptive dynamic particle swarm optimizer. In: 2015 IEEE Congress on, Evolutionary Computation (CEC). IEEE, pp 3206–3213

Lourenço HR, Martin OC, Stützle T (2003) Iterated local search. In: Glover F, Kochenberger GA (eds) Handbook of Metaheuristics. International series in operations research & management science, vol 57. Springer, Boston, MA, pp 320–353

Mahdavi M, Fesanghary M, Damangir E (2007) An improved harmony search algorithm for solving optimization problems. Appl Math Comput 188(2):1567–1579

Mezura-Montes E, Palomeque-Ortiz AG (2009) Self-adaptive and deterministic parameter control in differential evolution for constrained optimization. Springer, Berlin, pp 95–120

Mirjalili S, Lewis A (2013) S-shaped versus v-shaped transfer functions for binary particle swarm optimization. Swarm Evolut Comput 9:1–14

Mirjalili S, Lewis A (2016) The whale optimization algorithm. Adv Eng Softw 95:51–67

Mirjalili S, Mirjalili SM, Hatamlou A (2016) Multi-verse optimizer: a nature-inspired algorithm for global optimization. Neural Comput Appl 27(2):495–513

Molga M, Smutnicki C (2005) Test functions for optimization needs. In: Test functions for optimization needs

Noorbin SFEH, Alfi A (2018) Adaptive parameter control of search group algorithm using fuzzy logic applied to networked control systems. Soft Computing 22(23):7939–7960

Osman IH, Laporte G (1996) Metaheuristics: A bibliography. Ann Oper Res 63(5):511–623

Poláková R, Tvrdík J, Bujok P (2015) Cooperation of optimization algorithms: a simple hierarchical model. In: 2015 IEEE congress on, evolutionary computation (CEC), IEEE, pp 1046–1052

Rueda JL, Erlich I (2015) Testing mvmo on learning-based real-parameter single objective benchmark optimization problems. In: 2015 IEEE congress on, evolutionary computation (CEC), IEEE, pp 1025–1032

Sallam KM, Sarker RA, Essam DL, Elsayed SM (2015) Neurodynamic differential evolution algorithm and solving cec2015 competition problems. In: 2015 IEEE congress on, evolutionary computation (CEC), IEEE, pp 1033–1040

Sörensen K (2015) Metaheuristics-the metaphor exposed. Int Trans Oper Res 22(1):3–18

Suganthan P, Hansen N, Liang J, Deb K, CY, Auger A, Tiwari S (2005) Problem definitions and evaluation criteria for the CEC 2005 special session on real parameter optimization. In: Technical report, Nanyang Technological University, Nanyang

Tanweer MR, Suresh S, Sundararajan N (2017) Development of a higher order cognitive optimization algorithm. In: 2017 IEEE congress on evolutionary computation (CEC), pp 2752–2758

Wang H, Zhou X, Sun H, Yu X, Zhao J, Zhang H, Cui L (2017) Firefly algorithm with adaptive control parameters. Soft Comput 21(17):5091–5102

Wessing S, Preuss M, Rudolph G (2011) When parameter tuning actually is parameter control. In: Proceedings of the 13th annual conference on Genetic and evolutionary computation, ACM, pp 821–828

Xue Y, Jiang J, Zhao B, Ma T (2018) A self-adaptive artificial bee colony algorithm based on global best for global optimization. Soft Comput 22(9):2935–2952

Yu C, Kelley LC, Tan Y (2015) Dynamic search fireworks algorithm with covariance mutation for solving the CEC 2015 learning based competition problems. In: 2015 IEEE congress on, evolutionary computation (CEC), IEEE, pp 1106–1112

Zhao H, Zhang C, Ning J (2019) A best firework updating information guided adaptive fireworks algorithm. Neural Comput Appl 31(1):79–99

Zheng Y-J, Wu X-B (2015) Tuning maturity model of ecogeography-based optimization on cec 2015 single-objective optimization test problems. In: 2015 IEEE congress on, evolutionary computation (CEC), IEEE, pp 1018–1024