Engineering Applications of Artificial Intelligence 74 (2018) 1–9

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

Fibonacci indicator algorithm: A novel tool for complex optimization problems

A. Etminaniesfahani, A. Ghanbarzadeh *, Z. Marashi

Department of Mechanical Engineering, Shahid Chamran University of Ahvaz, Ahvaz, Iran

ARTICLE INFO

Keywords:
Benchmark function
Mechanical benchmark problem
Fibonacci indicator
Stock market
Meta-heuristic algorithm
Number of function evaluation

ABSTRACT

In this paper a new meta-heuristic algorithm is introduced. This optimization algorithm is inspired by the very popular tool among the technical traders in the stock market called the Fibonacci Indicator. The Fibonacci Indicator uses to predict possible local maximum and minimum prices, and periods in which the price of a stock will experience a significant amount of movement. The proposed Fibonacci Indicator algorithm is validated on several Benchmark functions up to 100 dimensions to have a comparison to algorithms such as DE extensions, PSO extensions, ABC, ABC-PS, CS, MCS and GSA in the ability of convergence and finding the global optimum in different research areas. Finally two engineering design problems are used to show the performance of the algorithm. Application of the proposed Fibonacci Indicator Algorithm in a wide set of benchmark functions has asserted its capability to deal with difficult optimization problems.

1. Introduction

In the last few decades, the use of meta-heuristic algorithms has been much improved to approach the optimized solution of nonlinear functions. A heuristic algorithm is a method to find the solution to an optimization problem by "trial-and-error". How ever, these algorithms may not find the global best solution to the problem and might get trapped in the local optimum points. On the other hand, the meta-heuristic algorithms finds the optimum solution by higher-level strategies employing trial-and-error, exploration, and exploitations. Particle Swarm Optimization (PSO) (Eberhart and Kennedy, 1995), Evolutionary Algorithms (EA) including Genetic Algorithm (GA) (Holland, 1975), Ant Colony Optimization (ACO) (Bilchev and Parmee, 1995; Dorigo and Blum, 2005), and the Bee Algorithm (BA) (Pham et al., 2006) are among the most popular metaheuristic algorithms. Evolutionary algorithms and swarm intelligence-based algorithms are two main categories of population-based optimization (Karaboga and Akay, 2009). Genetic algorithms, Differential Evolution (DE) (Storm and Price, 1995), (Meng and Pan, 2016; Meng et al., 2018) and evolutionary strategy (ES) (Rechenberg, 1965; Schwefel, 1965) have been the most popular techniques in evolutionary computation. Particle swarm optimization and Bees Algorithm are the most popular examples of swarm intelligence optimization. Two advantages of the different categories, i.e. evolutionary algorithms and swarm intelligence-based algorithms are presented below:

1. Particularly useful in multi-modal and multi-objective optimization problems;
2. Hybridize algorithms to each other;

A number of optimization algorithms can combine with each other and produce the hybrid algorithm with the synergy of both algorithm's advantages and elimination of their disadvantages.

Global optimization can be applied to various branches of science, economics and engineering (Bomze et al., 1997; Gergel, 1997; Horst and Tuy, 1996; Li et al., 2015; Rizk-Allah et al., 2016, 2018). Generally, solving nonlinear optimization problems can be classified into deterministic and stochastic methods (Li et al., 2015; Arora et al., 1995; Pardalos et al., 2000; Younis and Dong, 2010). In deterministic methods, optimization problems are solved by creating deterministic progression of convergence at the global optimal solution. This method requires unfailing mathematical specification and responsively depends on the initial conditions. On the contrary, in the stochastic methods including heuristic and meta-heuristic methods, new points are randomly generated (Younis and Dong, 2010). The efficiency of the optimization algorithms is usually determined by their ability in finding the global best solution by the minimum cost usually corresponding to the number of function evaluations. Exploration and exploitation are two main strategies to find the global best solution. Poor exploring and very fast convergence of algorithms increase the chance of getting trapped in local minima. Furthermore, very slow converging and increasing function evaluations

* Corresponding author.
E-mail address: Ghanbarzadeh.A@scu.ac.ir (A. Ghanbarzadeh).

https://doi.org/10.1016/j.engappai.2018.04.012
Received 22 May 2017; Received in revised form 8 April 2018; Accepted 10 April 2018
Available online 30 May 2018

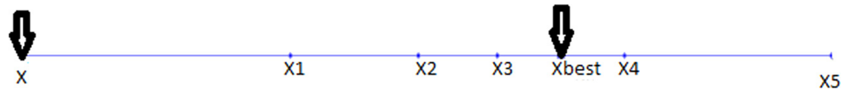

Fig. 1. Generating new points using FI method.

is not economical. The balance between exploration and exploitation is crucial to improve the efficiency of optimization algorithms (Li et al., 2015). Over the past decade, meta-heuristic algorithms such as GA (Price et al., 2005; Yang et al., 2007; Chelouah and Siarry, 2000) ACO, PSO (Jiang et al., 2007) and the artificial bee colony (ABC) have shown considerable successes in optimization algorithms (Ghanbari and Rhati, 2017). Previous researches show that ABC and GA have better exploration and slower convergence. However, ACO and PSO converge faster with more possibility of getting trapped in local optima (Alshamlan et al., 2015; Premalatha and Natarajan, 2009; Fidanova et al., 2014; Meng and Pan, 2016). A nonlinear optimization problem can be formulated as a D-dimensional problem of the following type: (Gergel, 1997; Nguyen et al., 2014).

$$f(x) = \begin{cases} \min & f(x) \\ s.t. & 1 \le x \le u \end{cases} \tag{1}$$

The objective function is defined by $f(x)$ and the D dimensional vector of variables is $x = (x1, x2, \dots, xD)$; lower and upper limits of variables are defined by $l = (l1, l2, \dots, lD)$ and $u = (u1, u2, \dots, uD)$.

2. Fibonacci indicator in the stock market

2.1. Fibonacci ratios

Leonardo Fibonacci is an Italian mathematician who found a sequence in 12–13th century. In this sequence, each number is generated by summing two previous numbers as follows: $\{0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, \dots\}$. In this sequence, each number is approximately 61.8% greater than the preceding number. By dividing one number in the sequence by the two and three places to the right, 38.2% and 23.6% will be found, respectively. Eqs. (2)–(7) are used to generate Fibonacci ratios.

$$F(1)/F(2) = 0 \tag{2}$$

$$F(2)/F(3) = 100\% \tag{3}$$

$$F(3)/F(4) = 50\% \tag{4}$$

$$\lim_{n\to\infty} (F(N)/F(N+1)) = 61.8\% \tag{5}$$

$$\lim_{n\to\infty} (F(N)/F(N+2)) = 38.2\% \tag{6}$$

$$\lim_{n\to\infty} (F(N)/F(N+3)) = 23.6\% \tag{7}$$

2.2. Fibonacci retracement and Fibonacci time zone

Fibonacci retracement and Fibonacci time zone are two well-known indicators used in finance technical analysis to predict possible maximum and minimum price of each stock and suggests suitable time to buy or sell in the future. Fibonacci retracement predicts the possible local minimum and maximum price of each stock by taking the minimum and maximum points on a chart and dividing the vertical distances proportional to Fibonacci ratios. For example assume m and M are minimum and maximum price of a specific stock, respectively.

$$\text{Fibonacci percentages} = [0, 23.6\%, 38.2\%, 50\%, 61.8\%, 100\%] \tag{8}$$

$$\text{Possible local minimum or maximum price } (i) = m + (M - m) \times \text{Fibonacci percentages } (i), i = 1, 2, \dots, 6 \tag{9}$$

Fibonacci time zone in summary predicts the periods in which the price of a specific stock will get a significant amount of movement by choosing the starting point and dividing the horizontal axis corresponding to Fibonacci series.

3. Fibonacci indicator algorithm

Here, we present a novel evolutionary optimization algorithm using a combination of two popular tools of technical traders in the stock marketing, Fibonacci retracement and Fibonacci time zone. Assume Fibonacci ratios are on arbitrary axis as shown in Fig. 1, so adding 50% to Fibonacci ratios arranges points more symmetrically around 100%. In this article, the act of generating new points proportional to Fibonacci series + 50% between the arbitrary point (x) and the independent variable of the best gained solution is named FI method. Fig. 1, shows FI method between x and x best.

$$\text{Fibonacci percentages} + 50\% = [50, 73.6, 88.2\%\ 100\%\ 111.8\%\ 150\%] \tag{10}$$

Assume the independent variable is time and the fitness of benchmark function is its price. In the real world, price chart of each stock is available and traders use Fibonacci indicators to determine the proper time to speculate in the stock markets by predicting local minimum and maximum prices.

In FIA the algorithm tries to find the minimum fitness of problem in variable intervals without using chart. The population size is the number of points that algorithm uses to start searching. Evaluating the benchmark cost functions in chosen points determines their fitness; all points save and sort based on fitness value. The best point sets as "xbest" and its fitness sets as "gbest". However, in each step the data of "xbest" and "gbest" are shared among all traders.

3.1. Searching phase

Searching phase is the main part of Fibonacci indicator algorithm. In this phase algorithm generates new points between the first chosen points from the best to the worst fitness, respectively and the shared xbest. As soon as a better fitness is met, the worst point should replace with xbest. This procedure continues until all the first chosen points are renewed after replacing all of them with new founded points it is suggested to do not omit any xbest in iterations. After making decision about adding or replacing newly founded points, algorithm cares on objective function. In single variable problems new points are generated from the worst point (worst xbest in previous iterations) to the newest xbest but in multivariable problems this procedure should just carry out for the specific percent of iterations (p%). Also, in $(100-p)$% of iterations, one point (assume "x") should generate by crossing over between xbest of previous iterations; it means each variable gains from xbest of one of the previous iterations randomly.

For example for a 3 dimensional function if m xbest saved, x generates by crossing over as you see below.

$$\text{xbest} = \text{xbest}^m = (\text{xbest}1^m, \text{xbest}2^m, \text{xbest}3^m) \tag{11}$$

$x = (\text{xbest}1^i, \text{xbest}2^j, \text{xbest}3^k)$ and i, j, k are integer numbers randomly chosen from 1 to m.

New points are generated between "x" and xbest by FI method. Crossing over and using xbest of previous iterations from the worst to the best fitness helps the algorithm to do not trap in local optimums. In both case of single variable problems and multi variable problems, as soon as a better point is met, xbest saves and if after specific function evaluations (determines by parameter C) a better solution did not found, all of the saved data except the latest xbest should vanish and the algorithm restarts. Fig. 2, exposes the flowchart of FIA and the Pseudo-code of the Fibonacci Indicator Algorithm is shown in Algorithm 1.

Algorithm 1 Pseudo-code of the Fibonacci Indicator Algorithm
Initialization
1. Determine the population size, counter $\max(C)$ and set $I = 0$
Iteration
2. **While** predefined criterion not met
    3. **While** $I < C$ **do** searching phase
    4. **If** better solution found accept and decide to add or replace the worst solution and set $I = 0$
    5. **Else** $I = I + 1$ // **end if**
    6. **End** while
    7. Save xbest, go to line 1 and add xbest to starting points
End

4. FIA performance analysis

In this section the performance of FIA is investigated by employing Rastrigin function on the interval $[-100, 100]$ which is a challenging multimodal function. The experiments carried out to analyze the influence of population size and parameters P and C, on the 2 and 30 dimensions cases.

4.1. Exploration and exploitation

Exploration and exploitation are two main characteristics in meta-heuristic algorithms.

Exploration is related to global search and we are interested in exploring the search space looking for good solutions as well as exploitation is related to local search and the algorithm want to refine the solution and try to avoid big jumps on the search space.

Random based parameters are usually used to perform exploration. But the exploitation focuses on the best found solutions.

FIA explores and exploits in the search space simultaneously by employing FI method between gained solutions (from the worst one to the best one) and the best solution ever met. Also crossing over and selecting parameter C to restarting the algorithm as it expressed before, improves the FIA's search ability.

4.2. Influence of population size and problem dimension on FIA

We employed different population size on 2 and 30 dimensions Rastrigin benchmark functions to investigate the influence of both population size and problem dimension. Parameters C and P are set to 50 and 50% respectively. Fig. 3, and Fig. 4, expose the influence of varying the population size on 2 and 30 dimensions problem, respectively. Results are the average of 30 independent runs and algorithm terminates after 10000* (problems dimension) function evaluations. The proper number of population size for 2 and 30 dimensions Rastrigin function is 6 and 10, respectively. Obviously determining the proper number of population size for each problem helps FIA to find the better solution.

4.3. Sensitivity analysis of parameters C and P

As mentioned, for each problem FIA uses fixed values for parameter C and also the probability parameter P to improve the exploration ability. In this section some experiments on Rastrigen benchmark function with dimensions 2 and 30 have been carried out to expose the effect of parameters on FIA. We set the number of maximum function evaluations to 10000* (problem dimension) and results are the average of 30 independent runs. At first we set the parameter P to 50% to investigate the effect of C on FIA. As it is shown in Fig. 5, and Fig. 6, the search capability of FIA is influenced by C and the convergence efficiency can be controlled by adjusting it. On the other hand, each function depend on its dimension and complexity needs a different value of C. Setting C to INF eliminates the influence of C on the algorithm and results show the necessity of that. As shown in Fig. 5, and Fig. 6, the proper value of C is 10 and 120 in cases 2 and 30 dimensions respectively. Obviously the algorithm should more try and evaluate functions more times as the dimension and complexity of problems increases. However we demonstrated the influence of parameter P on our algorithms search ability by setting the population size = 6 and $C = 10$ for 2 dimensions and also population size = 10 and $C = 120$ for 30 dimensions Rastrigin function and varying the value of parameter P. As it is shown in Fig. 7, and Fig. 8, the best value of P decreases from 50% to 25% when the problem dimension increases from 2 to 30. Actually increasing the probability of crossing over carries out by decreasing P and it improves FIA's exploration ability what is need when the dimension of problems increases.

5. Experimental framework

5.1. Experimental settings

We compare the efficiency of FIA with some optimization algorithms such as PSO variants, DE variants, PSO, ABC, ABC-PS, TLBO, HPA, HS, firefly algorithm (FA) (Yang, 2010), GSA, gray wolf optimizer and (GWO) (Mirjalili et al., 2014) by carrying out experiments on some well-known benchmark functions. All results of mentioned algorithms were directly taken from Meng et al. (2016), Gao et al. (2015), Li et al. (2015) and Rakhshani and Rahati (2017). However 2 engineering applications of FIA are presented to test the FIA by two engineering benchmark problems. The performance of FIA is comprised to that of some well-known methods directly taken from Rizk-Allah (2017).

In order to show the ability of the FIA in some benchmark functions that the global optima were not the coordinate origin, the global optima were shifted to A^* by using $f(x - A^*)$ to test the FIA however other algorithms were tested on original form of benchmark functions.

In each experiment the number of runs and also stopping criterion of FIA (a fixed number of function evaluations) are set according to what reference proposed. The FIA was implemented in Matlab2014 Version 8.3.0.532 environment under windows 10 operating system. All simulations were conducted on an Intel(R) Core™ i7, 2 GHz CPU and 6 GB of RAM.

5.2. Mathematical optimization problem

For comparison FIA is compared with swarm based algorithms, differential based algorithms and some recently proposed algorithms. A large test set is used to validate new algorithms. Benchmark functions including uni-modal, complex multi-modal and also multi-dimensional functions are listed in Table 1. All functions should be minimized.

Problems with one local optimum point are called uni-modal, and the ability of getting rid of the local minima is tested by Multi-modal functions. Also, increasing the dimension of search spaces increases the problem difficulty. Poor exploration of the algorithm leads to a trap in the local minima. On the other hand, strong exploration plays an important role in flat search spaces because flatness of the search space does not lead the algorithm toward the minima (Karaboga and Akay, 2009).

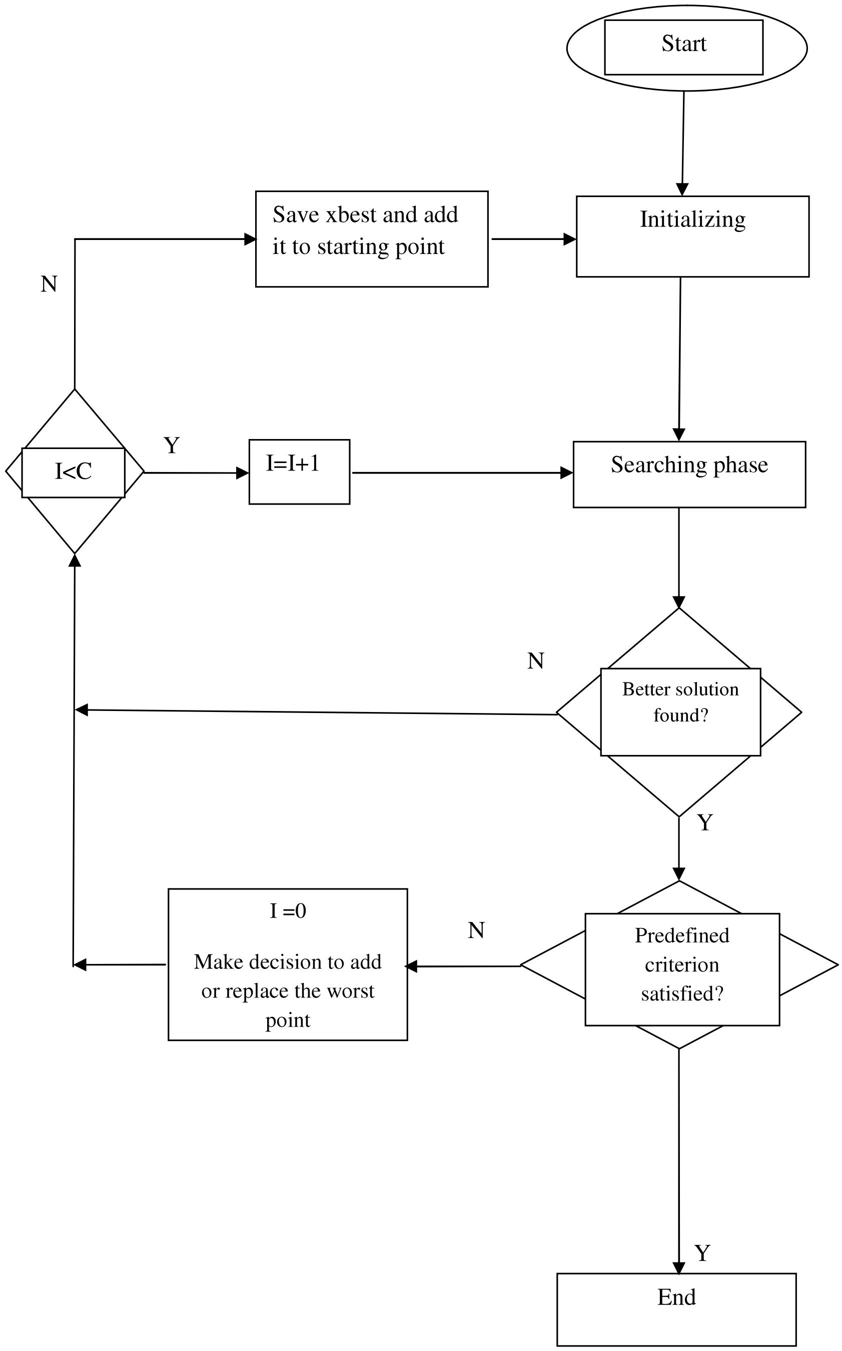

Fig. 2. Flowchart of FIA for different objective functions.

5.2.1. Experiment 1

The performance of proposed FIA is tested on 5 well-known unimodal and multimodal benchmark functions. Table 2 shows the comparison of the results obtained by FIA with those of PSO variants such as iw PSO, ccPSO, ccPSO-local, FIPS_Uring, CLPSO and DNLPSO. All data of these algorithms are taken directly from Meng et al. (2016). In the FIA we set the population size to 6, parameter C to 8 and parameter P to 50%. The maximum number of function evaluations is set to 100,000 and results are the average of 20 times run independently.

5.2.2. Experiment 2

In the second experiment we compared the performance of FIA with DE extensions and PSO extensions and also classical CS, MCS, GA, GSA, FA. We set the population size of FIA to 8 and parameters P and C to 25% and 150 respectively. Stopping criterion is set as 40,000 function evaluations and the algorithm is run 30 times independently in order to have fair comparison to data shown in Tables 6 and 7 what taken directly from Rakhshani and Rahati (2017).

5.2.3. Experiment 3

In this section we tested the FIA by employing some 100 dimensional unimodal and multimodal benchmark functions to make a comparison with PSO, ABC, HPA and ABC-PS. In each algorithm the maximum number of function evaluations is set to D*10000 (D is the functions dimension). All data are taken directly from Li et al. (2015) and we set the population size of FIA to 10 and parameters P and C to 25% and 200, respectively. In the comparison table the results less than 10^{-60} for unimodal functions and 10^{-20} for multimodal functions are assumed to be zero.

Table 1
The considered benchmark functions (d: Dimension, U: Unimodal, M: Multimodal, R1: (Rao, 2016), R2: (Li et al., 2015), R3: (Gao et al., 2015), R4: (Rakhshani and Rahati, 2017), R5: (Meng et al., 2016).

No	Range	A*	Function	Type	Formulation	Reference
1	[−100, 100]	80	Step	U	$f(x)=\sum_{i=1}^{d}\left(\lvert x_i+0.5\rvert\right)^2$	R3
2	[−100, 100]	80	Sphere	U	$f(x)=\sum_{i=1}^{d}x_i^2$	R4
3	[−10, 10]	8	Sum Squares	U	$f(x)=\sum_{i=1}^{d}ix_i^2$	R4
4	[−1.28, 1.28]	0.5	Quartic	U	$f(x)=\sum_{i=1}^{d}ix_i^4$	R4
5	[−1.2, 1.2]	0.2	Rosenbrock	U	$f(x)=100\left(x_1^2-x_2\right)^2+\left(1-x_1\right)^2$	R1
6	[−100, 100]	80	discus function	U	$f(x)=10^6.x_1^2+\sum_{i=2}^{d}ix_i^2$	R2
7	[−100, 100]	0	Rot-discus function	U	$f(z)=10^6.z_1^2+\sum_{i=2}^{d}z_i^2-1100$ $z=T_{\mathrm{osz}}(M_1(X-0))$	R5
8	[−600, 600]	500	Griewangk	M	$f(x)=\frac{\sum_{i=1}^{d}x_i^2}{4000}-\prod_{i=1}^{d}\cos\left(\frac{x_i}{\sqrt{i}}\right)+1$	R4
9	[−100, 100]	0	Rot- Griewangk	M	$f(x)=\frac{\sum_{i=1}^{d}z_i^2}{4000}-\prod_{i=1}^{d}\cos\left(\frac{z_i}{\sqrt{i}}\right)+1-(500)\ z=\Lambda^{100}M_1(\frac{600(X-0)}{100})$	R5
10	[−32, 32]	0	Ackely	M	$f(x)=-20.\exp\left[-0.2\sqrt{\frac{1}{d}\sum_{i=1}^{d}x_i^2}\right]-\exp\left[\frac{1}{d}\sum_{i=1}^{d}\cos(2\pi x_i)\right]+(20+\mathrm{e})$	R4
11	[−100, 100]	0	Rot- Ackely	M	$f(z)=-20.\exp\left[-0.2\sqrt{\frac{1}{d}\sum_{i=1}^{d}z_i^2}\right]-\exp\left[\frac{1}{d}\sum_{i=1}^{d}\cos\left(2\pi z_i\right)\right]+(20+\mathrm{e})-700,\ z=\Lambda^{10}M_2T_{\mathrm{asy}}^{0.5}(M_1(X-0))$	R5
12	[−50, 50]	0	penalized2	M	$f(x)=\frac{1}{d}\left\{\sin^2\pi x_1+\sum_{i=1}^{d-1}\left(x_i-1\right)^2\left[1+\sin^2(3\pi x_{i+1})\right]+\left(x_d-1\right)^2\left[1+\sin^2(2\pi x_{i+1})\right]\right\}+\sum_{i=1}^{n}u(x_i,5,100,4)$ $u\left(x_i,a,k,m\right)=\begin{cases}k\left(x_i-a\right)^m, x_i>a\\ 0,-a\le x_i\le a\\ k\left(-x_i-a\right)^m, x_i<-a\end{cases}$	R4
13	[−10, 10]	8	Alpine	M	$f(x)=\sum_{i=1}^{d}x_i\sin\left(x_i\right)+0.1.x_i$	R4
14	[−5.12, 5.12]	0	Rastrigin	M	$f(x)=10+\sum_{i=1}^{d}\left[z_i^2-10\cos(2\pi z_i)\right]-300\ z=\Lambda^{10}T_{\mathrm{asy}}^{0.2}(T_{\mathrm{osz}}(\frac{5.12(X-0)}{100}))$	R5
15	[−100, 100]	0	Rot- Rastrigin	M	$f(x)=10+\sum_{i=1}^{d}\left[z_i^2-10\cos\left(2\pi xz_i\right)\right]-300\ z=M_1\Lambda^{10}M_2T_{\mathrm{asy}}^{0.2}(M_1T_{\mathrm{osz}}^{0.2}(\frac{5.12(X-0)}{100}))$	R5
16	[−5, 10]	0	Rosenbrock	M	$f(x)=\sum_{i=1}^{d-1}\{100\left(x_i^2-x_{i+1}\right)^2+\left(1-x_i\right)^2\}$	R4
17	[−100, 100]	0	expanded schaffer's function	M	$f(x)=g\left(x_1,x_2\right)+g\left(x_2,x_3\right)+\ldots+g\left(x_{D-1},x_D\right)+g(x_D,x_1)$ $g(x,y)=0.5+\frac{\sin^2\left(\sqrt{x^2+y^2}\right)-0.5}{\left(1+0.001(x^2+y^2)\right)^2}$	R2
18	[−5, 10]	3	Rosenbrock	M	$f(x)=\sum_{i=1}^{d-1}\{100\left(x_i^2-x_{i+1}\right)^2+\left(1-x_i\right)^2\}$	R1
19	[−100, 100]	0	Schaffer	M	$f(x)=0.5+\frac{\sin^2\left(\sum_{i=1}^{d}x_i^2\right)-0.5}{\left(1+0.001\left(\sum_{i=1}^{d}x_i^2\right)\right)^2}+\lvert x_n-1\rvert\left[1+\sin^2(3\pi x_n)\right]$	R1
20	[−100, 100]	80	HGBat	M	$f(x)=\left\lvert\left(\sum_{i=1}^{d}x_i^2\right)^2-\left(\sum_{i=1}^{d}x_i\right)^2\right\rvert^{\frac{1}{2}}+\frac{0.5\sum_{i=1}^{d}x_i^2+\sum_{i=1}^{d}x_i}{D}$	R2

Table 2
Statistical results of 20 runs. Mean: Mean of the Best values, StdDev: Standard Deviation of the best values RANK: Rank of each algorithm.
Source: All data of PSO variants are taken from Meng et al. (2016).

NO	Function	*d*		FIA	iw PSO	ccPSO	ccPSO-local	FIPS_Uring	CLPSO	DNLPSO
1	Sphere	10	Mean	0	0	$5.6843e^{-01}4$	0	0	0	0
			StdDev	0	0	$1.0101e^{-13}$	0	0	0	0
			RANK	**1**	**1**	**7**	**1**	**1**	**1**	**1**
2	Rot- Ackley's function	10	Mean	$2.\ 0573e^{+01}$	$2.0213e^{+01}$	$2.0215e^{+01}$	$2.0225e^{+01}$	$2.0248e^{+01}$	$2.0220e^{+01}$	$2.0293e^{+01}$
			StdDev	$2.6329e^{-02}$	$5.7059e^{-02}$	$4.4255e^{-02}$	$4.5937e^{-02}$	$4.3374e^{-02}$	$5.7137e^{-02}$	$1.3372e^{-01}$
			RANK	**1**	**2**	**3**	**6**	**5**	**4**	**7**
3	Rot-Griewangk	10	Mean	$9.47561e^{-02}$	$4.9989e^{-01}$	$3.4498e^{-01}$	$1.0603e^{-01}$	$6.3301e^{-01}$	$3.1252e^{-01}$	$3.19071e^{-01}$
			StdDev	$1.2312e^{-02}$	$2.9794e^{-01}$	$1.8439e^{-01}$	$4.2707e^{-02}$	$5.8004e^{-02}$	$8.0042e^{-02}$	$1.7876e^{-01}$
			RANK	**1**	**6**	**5**	**2**	**3**	**3**	**4**
4	Rot-Rastrigins	10	Mean	$6.34727e^{00}$	$1.5064e^{01}$	$1.6018e^{01}$	$7.4621e^{00}$	$2.7347e^{01}$	$5.5914e^{00}$	$1.6440e^{01}$
			StdDev	$9.4482e^{-01}$	$5.5123e^{00}$	$6.7319e^{00}$	$2.7675e^{00}$	$3.4252e^{00}$	$1.6202e^{00}$	$8.9530e^{00}$
			RANK	**2**	**4**	**5**	**3**	**7**	**1**	**6**
5	Rot- discus function	10	Mean	$8.3125e^{-05}$	$4.5055e^{-02}$	$1.9519e^{-04}$	$1.8738e^{01}$	$1.9171e^{03}$	$1.8794e^{03}$	1.7469e03
			StdDev	$1.0125e^{-05}$	$5.1575e^{-02}$	$5.7786e^{-04}$	$1.7684e^{01}$	$5.1283e^{02}$	$5.9328e^{02}$	3.8752e03
			RANK	**1**	**3**	**2**	**4**	**5**	**6**	**7**

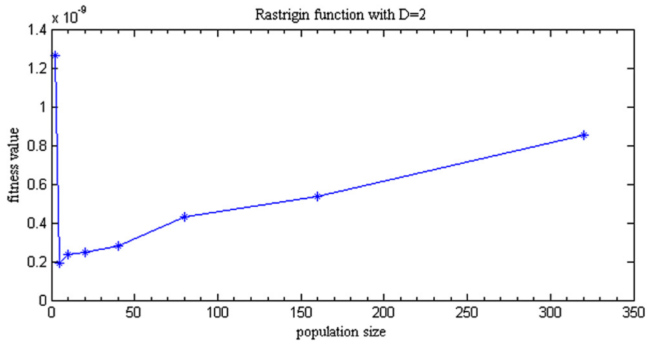

Fig. 3. Influence of population size on 2 dimensions Rastrigin benchmark function.

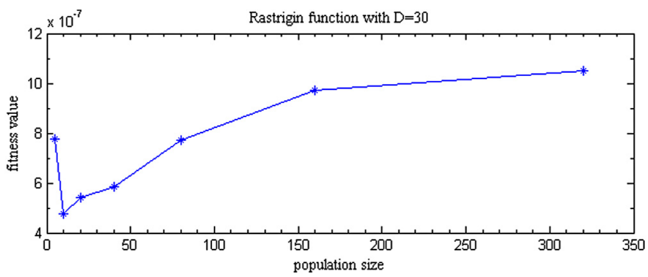

Fig. 4. Influence of population size on 30 dimensions Rastrigin benchmark function.

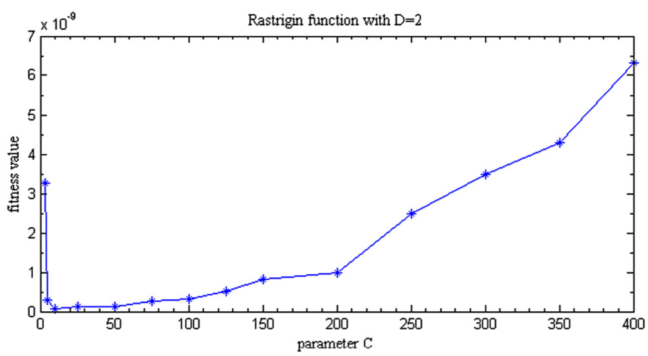

Fig. 5. Results of the different C values for 2 dimensions Rastrigin benchmark function.

5.3. Engineering design problem

In this section two **Engineering design problems are proposed** to examine the robustness of the FIA in contrast some well known algorithms Rizk-Allah (2017).

In the FIA the population size is set to 6, and parameters C and P are set to 8 and 50% respectively. The results are the average of 20 times run independently.

5.3.1. The gear train design problem

Here algorithms are used to find the optimum number of tooth for four gears of a train shown in Fig. 9 the cost function is the cost of gear ratio and FIA finds the optimum value of parameters $x1, x2, x3$ and $x4$.

The objective function can mathematically be stated as:

$$Min\ F(X) = \left[\frac{1}{6.931} - \frac{x3x2}{x1x4}\right]^2 \tag{12}$$

$$12 \le xi \le 60,\ \ i = 1, 2, 3, 4$$

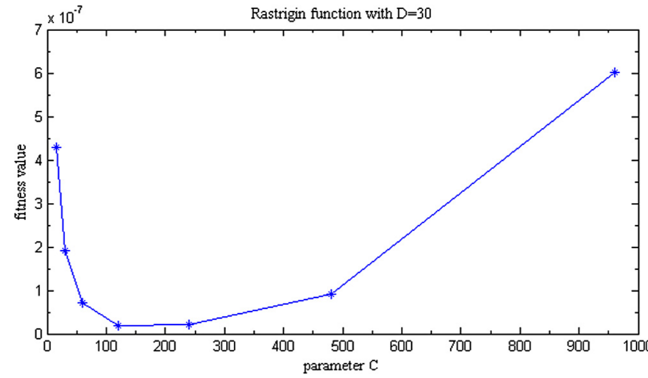

Fig. 6. Results of the different C values for 30 dimensions Rastrigin benchmark function.

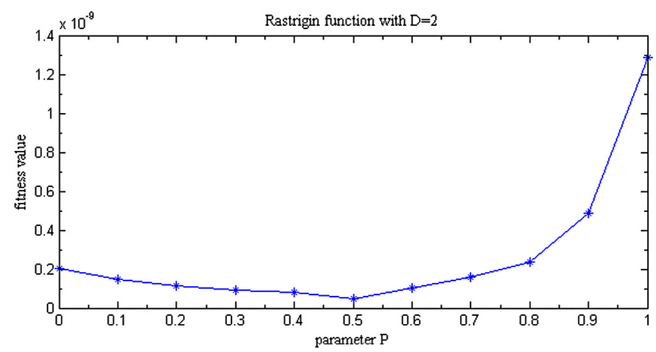

Fig. 7. Results of the different P values for 2 dimensions Rastrigin benchmark function.

5.3.2. Cantilever beam

Fig. 10 shows the schematic of a cantilever beam that used to test the FIA performance.

The thickness of each square-shaped hollow elements are supposed to be constant and

The goal of this problem is to minimize the weight of the beam.

So five optimum structural parameters $x1, x2, x3, x4$ and $x5$ are tried to find however the problem contains one vertical displacement constraint.

Node 1 is supported rigidly and a vertical load is applied to node 6 the free end of the beam. The objective function is:

$$Min\ F(x) = 0.06224(x1 + x2 + x3 + x4 + x5) \tag{13}$$

$$g(x) = \frac{61}{x1^3} + \frac{37}{x2^3} + \frac{19}{x3^3} + \frac{7}{x4^3} + \frac{1}{x5^3} \le 1$$

$$0.01 \le xi \le 100, i = 1, 2, 3, 4, 5$$

5.4. Experimental results and discussion

In experiment 1, FIA was compared to some PSO variants such as the canonical PSO, Inertia Weighted PSO (iwPSO), Constriction Coefficients PSO (ccPSO), Fully Informed Particle Swarm (FIPS), Comprehensive Learning PSO (CLPSO), and Dynamic Neighborhood Learning PSO (DNLPSO). The mean best values and standard deviations were given in Table 2 and as can be seen except for Rot-Rastrigins FIA was the best algorithm. CLPSO was better than FIA in Rot-Rastrigins benchmark function. In experiment 2, as can be seen in Tables 3 and 4, unimodal benchmark functions (Sum Squares, Step and Quartic functions) according to the mean and standard deviation of results, FIA gives more accurate range for true global optimum on Sum Square function, but GSA on Quartic function and CS and MCS on step function are performed

Table 3
Statistical results of 30 runs. Mean: Mean of the Best values, StdDev: Standard Deviation of the best values RANK: Rank of each algorithm.
Source: All data of CS, MSC, FA, GSA algorithms are taken from Rakhshani and Rahati (2017).

NO	Range	*D*	Function	Min		FIA	CS	MSC	FA	GSA
1	[−10, 10]	50	Sum Squares	0	MEAN STD RANK	7.93^{-016} $6.83e^{-016}$ 1	$2.75e^{-001}$ $3.35e^{-001}$ 4	$7.39e^{-005}$ $5.04e^{-005}$ 3	$1.52e^{+000}$ $1.30e^{+000}$ 5	$7.26e^{-015}$ $2.82e^{-015}$ 2
2	[−100, 100]	50	Step	0	MEAN STD RANK	$5.833e^{-14}$ $1.455e^{-14}$ 3	$3.39e^{+001}$ $5.35e^{+001}$ 5	$0.00e^{+000}$ $0.00e^{+000}$ 1	$0.00e^{+000}$ $0.00e^{+000}$ 1	2.20^{+000} 2.80^{+000} 4
3	[−1.28, 1.28]	50	Quartic	0	MEAN STD RANK	$5.963e^{-17}$ $6.737e^{-17}$ 2	$1.66e^{-004}$ $1.35e^{-004}$ 5	$7.27e^{-015}$ $9.52e^{-015}$ 3	$5.20e^{-007}$ $5.23e^{-007}$ 4	$1.57e^{-031}$ $1.19e^{-031}$ 1
4	[−10, 10]	50	Alpine	0	MEAN STD RANK	$4.634e^{-009}$ $2.559e^{-009}$ 1	$9.05e^{+000}$ $3.46e^{+000}$ 5	$3.41e^{-003}$ $1.82e^{-003}$ 3	$1.36e^{-001}$ $1.23e^{-001}$ 4	$1.86e^{-004}$ $5.80e^{-004}$ 2
5	[−100, 100]	50	Schaffer	0	MEAN STD RANK	$1.635e^{-003}$ $5.362e^{-003}$ 1	$4.52e^{-001}$ $1.42e^{-002}$ 4	$3.13e^{-003}$ $2.85e^{-013}$ 2	$4.98e^{-001}$ $1.01e^{-003}$ 5	$3.26e^{-001}$ $2.71e^{-002}$ 3
6	[−5, 10]	50	Rosenbrock	0	MEAN STD RANK	$5.342e^{-002}$ $2.699e^{-002}$ 1	$2.28e^{+002}$ $1.27e^{+002}$ 5	$5.28e^{+001}$ $2.00e^{+001}$ 2	$5.85e^{+001}$ $3.18e^{+001}$ 3	$7.62e^{+001}$ $3.50e^{+001}$ 4
7	[−50, 50]	50	Penalized2	0	MEAN STD RANK	$4.724e^{-006}$ $6.541e^{-006}$ 1	$3.44e^{+002}$ $6.66e^{+002}$ 5	$3.58e^{+000}$ $3.93e^{-001}$ 4	$5.15e^{-004}$ $1.47e^{-004}$ 2	$2.48e^{+000}$ $2.51e^{+000}$ 3

Table 4
Statistical results of 30 runs. Mean: Mean of the Best values, StdDev: Standard Deviation of the best values RANK: Rank of each algorithm.
Source: All data of ELPSO, CIWPSO, RIWPSO, CDE, ODE and EPSDE algorithms are taken from Rakhshani and Rahati (2017).

NO	Range	*D*	Function	Min		FIA	ELPSO	CIWPSO	RIWPSO	CDE	ODE	EPSDE
1	[−100, 100]	30	Sphere	0	MEAN STD RANK	$6.921e^{-029}$ $7.561e^{-029}$ 1	$5.244e^{-008}$ $1.643e^{-008}$ 3	$1.565e^{-007}$ $7.640e^{-008}$ 6	$4.810e^{-007}$ $7.240e^{-008}$ 7	$8.130e^{-008}$ $2.160e^{-008}$ 5	$7.540e^{-008}$ $2.850e^{-008}$ 4	$1.443e^{-014}$ $2.206e^{-014}$ 2
2	[−32, 32]	30	Ackey	0	MEAN STD RANK	$4.322e^{-12}$ $2.443e^{-12}$ 1	2.779e−001 $8.160e^{-002}$ 3	$2.560e^{+000}$ $9.076e^{-001}$ 7	$2.314e^{+000}$ $8.196e^{-001}$ 6	$4.567e^{-001}$ $2.154e^{-001}$ 4	$5.376e^{-001}$ $1.057e^{-001}$ 5	$7.570e^{-008}$ $8.176e^{-008}$ 2
3	[−600, 600]	30	Griewank	0	MEAN STD RANK	$5.721e^{-019}$ $2.662e^{-019}$ 1	$2.748e^{-004}$ $1.232e^{-004}$ 3	$1.240e^{-002}$ $6.500e^{-003}$ 7	$1.140e^{-002}$ $1.150e^{-002}$ 6	$3.220e^{-004}$ $1.930e^{-004}$ 4	$4.690e^{-004}$ $1.310e^{-004}$ 5	$7.421e^{-014}$ $2.065e^{-013}$ 2
4	[−5.12, 5.12]	30	Rastrigin	0	MEAN STD RANK	$3.134e^{-003}$ $7.886e^{-003}$ 1	$8.640e^{+000}$ $4.187e^{+000}$ 2	$1.900e^{+001}$ $7.784e^{+000}$ 5	$2.557e^{+001}$ $9.023e^{+000}$ 7	$9.255e^{+000}$ $4.945e^{+000}$ 3	$1.144e^{+001}$ $5.973e^{+000}$ 4	$2.359e^{+001}$ $2.934e^{+000}$ 6
5	[−5, 10]	30	Rosenbrock	0	MEAN STD RANK	$3.433e^{-006}$ $3.244e^{-006}$ 1	$5.817e^{+000}$ $1.384e^{+000}$	$8.688e^{+000}$ $2.329e^{+001}$ 4	$2.839e^{+001}$ $2.428e^{+001}$ 6	$5.852e^{+000}$ $1.745e^{+000}$ 3	$5.322e^{+000}$ $1.214e^{+000}$ 2	$1.543e^{+001}$ $2.016e^{+000}$ 5

Table 5
Statistical results of 30 runs. Mean: Mean of the Best values, StdDev: Standard Deviation of the best values RANK: Rank of each algorithm.
Source: All data of PSO, ABC and ABC-PS algorithms are taken from Li et al. (2015).

NO	Range	*D*	Function	Min		FIA	PSO	ABC	ABC-PS
1	[−100, 100]	100	Discus function	0	MEAN STD RANK	$4.6526e^{-04}$ $3.3231e^{-04}$ 1	$5.2459e^{+04}$ $1.6172e^{+04}$ 4	$8.4910e^{+02}$ $4.7740e^{+02}$ 3	$3.9066e^{-03}$ $1.2206e^{-02}$ 2
2	[−100, 100]	100	Griewangk	0	MEAN STD RANK	$4.8932e^{-0011}$ $7.234e^{-011}$ 1	$2.6103e^{+00}$ $2.7011e^{-01}$ 3	$9.3039e^{+00}$ $1.0803e^{+01}$ 4	$1.0229e^{-08}$ $3.6532e^{-06}$ 2
3	[−100, 100]	100	Rastrigin	0	MEAN STD RANK	$7.2341e^{-05}$ $4.4331e^{-05}$ 1	$1.1197e^{+04}$ $2.7289e^{+03}$ 4	$1.6966e^{+00}$ $2.8824e^{+00}$ 3	$7.7497e^{-05}$ $5.5547e^{-04}$ 2
4	[−100, 100]	100	HGBat	0	MEAN STD RANK	$4.4288e^{-02}$ $2.7231e^{-02}$ 1	$6.1837e^{+03}$ $9.5735e^{+02}$ 4	$7.8231e^{+01}$ $6.4651e^{+01}$ 3	$5.0619e^{-01}$ $6.7441e^{-01}$ 2
5	[−100, 100]	100	Expanded schaffer's function	0	MEAN STD RANK	$6.3456e^{-03}$ $3.2145e^{-02}$ 1	$4.5550e^{+03}$ $2.0116e^{+02}$ 3	$7.4062e^{+03}$ $9.0657e^{+03}$ 4	$7.8203e^{-07}$ $6.9087e^{-03}$ 2

better than FIA. But FIA converges closer to the global optimum and provides a better standard deviation on presented multimodal benchmark functions, the results imply FIA provides a better performance to escape from the local minima and finding global optima especially in multimodal dimension benchmark functions. In experiment 3, we tested the performance of FIA for 100 dimensional benchmark functions with

Table 6
Comparisons among different methods for gear train design problem.
Source: All data of CS, MBA, ABC, and MOSCA algorithms are taken from Rizk-Allah (2017).

No	Algorithm	$x1$	$x2$	$x3$	$x4$	$F(x)$	Number of function evaluation
1	CS	43	16	19	49	2.7009E−012	5000
2	MBA	43	16	19	49	2.7009E−012	10000
3	ABC	19	16	44	49	2.78E−11	40000
4	MOSCA	49	19	16	43	2.7009E−12	900
5	FIA	49	19	16	43	2.700857148886513e−12	734

Table 7
Comparisons among different methods for cantilever design problem.
Source: All data of ALO, SOS, CS, and MOSCA algorithms are taken from Rizk-Allah (2017).

No	Algorithm	$x1$	$x2$	$x3$	$x4$	$x5$	$F(x)$	Number of function evaluation
1	ALO	6.01812	5.31142	4.48836	3.49751	2.158329	1.33995	14000
2	SOS	6.01878	5.30344	4.49587	3.49896	2.15564	1.33996	15000
3	CS	6.0089	5.3049	4.5023	3.5077	2.1504	1.33999	2500
4	MOSCA	6.02154	5.3101	4.4991	3.4943	2.1486	1.33652	4380
5	FIA	6.01612	5.31242	4.48436	3.49751	2.16333	1.33652	3173

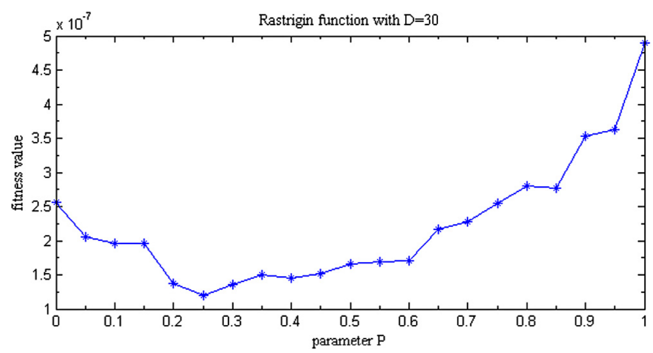

Fig. 8. Results of the different P values for 30 dimensions Rastrigin benchmark function.

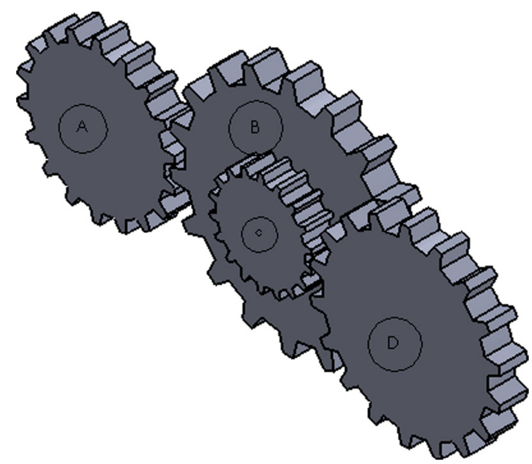

Fig. 9. The schematic of gear train.

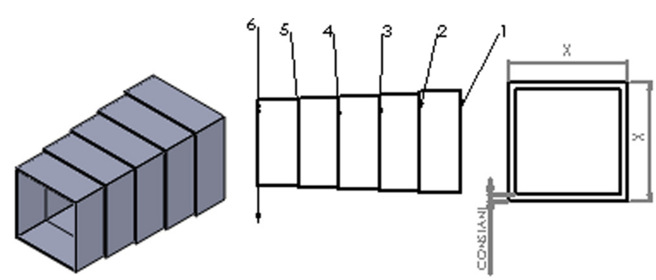

Fig. 10. The schematic of cantilever beam.

the original PSO, ABC and hybrid PS-ABC method. Discus function is a unimodal function and Griewangk, Rastrigin, HGBat and expanded Schaffer's function are 100 dimensional multimodal test functions. As the dimension of the function increases, the number of local optimum increase exponentially. As mentioned before the all data of PSO, ABC and hybrid PS-ABC are taken directly from Li et al. (2015) and according to Table 5 FIA performed better than other algorithms on mentioned functions. The rank and steadily of solutions obtained by FIA method exposes the efficiency and robustness of the algorithm in solving high-dimensional benchmark functions. According to The results of gear train design and cantilever beam design showed in Tables 6 and 7, the FIA outperforms the compared algorithms to solve the discrete and continuous real problems effectively with low computational cost.

6. Conclusions

In this paper, Fibonacci Indicator Algorithm is proposed. Predictions about the time of maximum and minimum cost of stocks were the basic motivation to develop this new optimization algorithm. The introduced algorithm was tested on two engineering design problems and also several benchmark cost functions with up to 100 dimensions. The comparison with algorithms such as DE extensions, PSO extensions, ABC, ABC-PS, CS, MCS, GSA and FA illustrated its efficiency and global optima achievement. Although the efficiency of the standard version of FIA can be improved by employing some effective heuristics, our aim was to compare the performance of the standard version of FIA to other famous population-based algorithms. In experiments engineering design problems and benchmark cost functions were used to comprise the ability of finding global minima and convergence of the FIA with some well-known classical and also state-of-the-art algorithm, however the noticeable fact is that the higher performance of FIA in the above-mentioned functions does not guarantee that FIA would be the best optimization algorithm ever. But, certainly it is suitable for many of optimization problems.

Acknowledgments

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.engappai.2018.04.012.

References

Alshamlan, H.M., Badr, G.H., Alohali, Y.A., 2015. Genetic Bee Colony (GBC) algorithm: A new gene selection method for microarray cancer classification. Comput. Biol. Chem. 56, 49–60.

Arora, J.S., lwakeil, O.A., hahande, A.I., sieh, C.C., 1995. Global optimization methods for engineering applications: A review. Struct. Optim. 9 (3–4), 137–159.

Bilchev, G., Parmee, I.C., 1995. The ant colony metaphor for searching continuous design spaces. In: Fogarty, T.C. (Ed.), Evolutionary Computing: AISB Workshop Sheffield, UK, April 3–4, 1995 Selected Papers. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 25–39.

Bomze, I., Csendes, T., Horst, R., Pardalos, P.M., 1997. Developments in Global Optimization, Vol.18. Springer US.

Chelouah, R., Siarry, P., 2000. A continuous genetic algorithm designed for the global optimization of multimodal functions. J. Heuristics 6 (2), 191–213.

Dorigo, Marco, Blum, Christian, 2005. Ant colony optimization theory: A survey. Theoret. Comput. Sci. 344 (2–3), 243–278.

Eberhart, R.C., Kennedy, J., 1995. A new optimizer using particle swarm theory. In Proceedings of the Sixth International Symposium on Micro Machine and Human Science, vol. 1, pp. 39–43.

Fidanova, S., Paprzycki, M., Roeva, O., 2014. Hybrid GA-ACO Algorithm for a model parameters identification problem. In: 2014 Federated Conference on Computer Science and Information Systems.

Gao, Weifeng, Chan, Felix T.S., Huang, Lingling, Liu, Sanyang, 2015. Bare bones artificial bee colony algorithm with parameter adaptation and fitness-based neighborhood. Inform. Sci. 316, 180–200.

Gergel, V.P., 1997. A global optimization algorithm for multivariate functions with Lipschitzian first derivatives. J. Global Optim. 10, 257–281.

Ghanbari, S., Rhati, A., 2017. An improved artificial bee colony algorithm and its application to reliability optimization problems. Appl. Soft Comput. 62.

Holland, J.H., 1975. Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence. University of Michigan Press.

Horst, R., Tuy, H., 1996. Global Optimization: Deterministic Approaches. Springer Science & Business Media.

Jiang, Y., Hu, T., Huang, C., Wu, X., 2007. An improved particle swarm optimization algorithm. Appl. Math. Comput. 193, 231–239.

Karaboga, D., Akay, B., 2009. A comparative study of Artificial Bee Colony algorithm. Appl. Math. Comput. 214 (1), 108–132.

Li, Zhiyong, Wang, Weiyou, Yan, Yanyan, Li, Zheng, 2015. Ps-abc: a hybrid algorithm based on particle swarm and artificial bee colony for high-dimensional optimization problems. Expert Syst. Appl. 42 (22), 8881–8895.

Meng, Z., Pan, J.-S., 2016. Monkey King Evolution: A new memetic evolutionary algorithm and its application in vehicle fuel consumption optimization. Knowl.-Based Syst. 97, 144–157.

Meng, Z., Pan, J.S., Kong, L., 2018. Parameters with adaptive learning mechanism (palm) for the enhancement of differential evolution. Knowl.-Based Syst.

Meng, Z., Pan, J.S., Xu, H., 2016. Quasi-affine transformation evolutionary (quatre) algorithm: a cooperative swarm based algorithm for global optimization. Knowl.-Based Syst. 109, 104–121.

Mirjalili, Seyedali, Mirjalili, Seyed Mohammad, Lewis, Andrew, Optimizer, Grey Wolf, 2014. Grey wolf optimizer. Adv. Eng. Softw. 69, 46–61.

Nguyen, Tien Trong, Li, ZhiYong, Zhang, ShiWen, Truong, Tung Khac, 2014. A hybrid algorithm based on particle swarm and chemical reaction optimization. Expert Syst. Appl. 41 (5), 2134–2143.

Pardalos, P.M., Romeijn, H.E., Tuy, H., 2000. Recent developments and trends in global optimization. J. Comput. Appl. Math. 124 (1–2), 209–228.

Pham, D.T., Ghanbarzadeh, A., KoC, E., Otri, S., Rahim, S., Zaidi, M., 2006. The bees algorithm _ A novel tool for complex optimisation problems. In: Intelligent Production Machines and Systems. Elsevier Science Ltd, Oxford, pp. 454–459.

Premalatha, K., Natarajan, A.M., 2009. Hybrid PSO and GA for Global Maximization. pp. 597–608.

Price, K., Storn, R.M., Lampinen, J.A., 2005. Differential Evolution: A Practical Approach to Global Optimization. Springer Science & Business Media.

Rakhshani, Hojjat, Rahati, Amin, 2017. Snap-drift cuckoo search: A novel cuckoo search optimization algorithm. Appl. Soft Comput. 52, 771–794.

Rao, R.V., 2016. Jaya: a simple and new optimization algorithm for solving constrained and unconstrained optimization problems. Int. J. Ind. Eng. Comput. 7, 19–34.

Rechenberg, I., 1965. Cybernetic solution path of an experimental problem. In: Royal Aircraft Establishment Translation No. 1122, B. F. Toms, Trans. Ministry of Aviation, Royal Aircraft Establishment.

Rizk-Allah, R.M., 2017. Hybridizing sine cosine algorithm with multi-orthogonal search strategy for engineering design problems. J. Comput. Design Eng. http://dx.doi.org/10.1016/j.jcde.2017.08.002.

Rizk-Allah, R.M., EI-Sehiemy, R.A., Deb, S., Wang, G.G., 2016. A novel fruit fly framework for multi-objective shape design of tubular linear synchronous motor. J. Supercomput. 73 (3), 1–22.

Rizk-Allah, R.M., EI-Sehiemy, R.A., Wang, G.G., 2018. A novel parallel hurricane optimization algorithm for secure emission/economic load dispatch solution. Appl. Soft Comput. 63, 206–222.

Schwefel, H.P., 1965. Kybemetische Evolution Als Strategie Der Experimentellen Forschung in Der Stromungstechink (Master's thesis), Tehnical University of Berlin, Germany.

Storm, R., Price, K., 1995. Differential Evolutional Simple and Efficient Adaptive Scheme for Global Optimization over Continuous Spaces, Brekeley. International Computer Science Institute, CA.

Yang, X.-S., 2010. Firefly algorithm, levy flights and global optimization. In: Bramer, M., Ellis, R., Petridis, M. (Eds.), Research and Development in Intelligent Systems XXVI: Incorporating Applications and Innovations in Intelligent Systems XVII. Springer London, London, pp. 209–218.

Yang, Z., Tang, K., Yao, X., 2007. 2007 Differential evolution for high-dimensional function optimization. In: Proceedings of the IEEE Congress on Evolutionary Computation. CEC 2007. IEEE, pp. 3523–3530.

Younis, A., Dong, Z., 2010. Trends, features, and tests of common and recently introduced global optimization methods. Eng. Optim. 42 (8), 691–718.