Soft Comput (2013) 17:1911–1928
DOI 10.1007/s00500-013-1032-8

METHODOLOGIES AND APPLICATION

Memetic search in artificial bee colony algorithm

Jagdish Chand Bansal · Harish Sharma · K. V. Arya · Atulya Nagar

Published online: 30 March 2013

Abstract Artificial bee colony (ABC) optimization algorithm is relatively a simple and recent population based probabilistic approach for global optimization. ABC has been outperformed over some Nature Inspired Algorithms (NIAs) when tested over benchmark as well as real world optimization problems. The solution search equation of ABC is significantly influenced by a random quantity which helps in exploration at the cost of exploitation of the search space. In the solution search equation of ABC, there is a enough chance to skip the true solution due to large step size. In order to balance between diversity and convergence capability of the ABC, a new local search phase is integrated with the basic ABC to exploit the search space identified by the best individual in the swarm. In the proposed phase, ABC works as a local search algorithm in which, the step size that is required to update the best solution, is controlled by Golden Section Search approach. The proposed strategy is named as Memetic ABC (MeABC). In MeABC, new solutions are generated around the best solution and it helps to enhance the exploitation capability of ABC. MeABC is established as a modified ABC algorithm through experiments over 20 test problems of different complexities and 4 well known engineering optimization problems.

Keywords Artificial bee colony · Swarm intelligence · Exploration-exploitation · Memetic algorithm

Communicated by G. Acampora.

J. C. Bansal · H. Sharma (✉) · K. V. Arya
ABV-Indian Institute of Information Technology
and Management, Gwalior, India
e-mail: harish.sharma0107@gmail.com

J. C. Bansal
e-mail: jcbansal@gmail.com

K. V. Arya
e-mail: kvarya@gmail.com

A. Nagar
Department of Computer and Mathematical Sciences,
Centre for Applicable Mathematics and Systems Science
(CAMSS), Liverpool Hope University, Liverpool L16 9JD, UK
e-mail: nagara@hope.ac.uk

1 Introduction

Swarm Intelligence has become an emerging and interesting area in the field of nature inspired techniques that is used to solve optimization problems during the past decade. It is based on the collective behavior of social creatures. Swarm based optimization algorithms find solution by collaborative trial and error. Social creatures utilize their ability of social learning to solve complex tasks. Peer to peer learning behavior of social colonies is the main driving force behind the development of many efficient swarm based optimization algorithms. Researchers have analyzed such behaviors and designed algorithms that can be used to solve nonlinear, nonconvex or discrete optimization problems. Previous research (Dorigo and Di Caro 1999; Kennedy and Eberhart 1995; Price et al. 2005; Vesterstrom and Thomsen 2004) have shown that algorithms based on swarm intelligence have great potential to find solutions of real world optimization problems. The algorithms that have emerged in recent years include ant colony optimization (ACO) (Dorigo and Di Caro 1999), particle swarm optimization (PSO) (Kennedy and Eberhart 1995), bacterial foraging optimization (BFO) (Passino 2002) etc.

Artificial bee colony (ABC) optimization algorithm introduced by Karaboga (2005) is a recent addition in this category. This algorithm is inspired by the behavior of honey bees when seeking a quality food source. Like any other population based optimization algorithm, ABC consists of a population of potential solutions. The potential solutions are food sources of honey bees. The fitness is determined in terms of the quality (nectar amount) of the food source. ABC is relatively a simple, fast and population based stochastic search technique in the field of nature inspired algorithms.

There are two fundamental processes which drive the swarm to update in ABC: the variation process, which enables exploring different areas of the search space, and the selection process, which ensures the exploitation of the previous experience. However, it has been shown that the ABC may occasionally stop proceeding toward the global optimum even though the population has not converged to a local optimum (Karaboga and Akay 2009). It can be observed that the solution search equation of ABC algorithm is good at exploration but poor at exploitation (Zhu and Kwong 2010). Therefore, to maintain the proper balance between exploration and exploitation behavior of ABC, it is highly required to develop a local search approach in the basic ABC to exploit the search region.

In this paper, a new local search strategy has been integrated with the basic ABC. In the proposed local search strategy, ABC's position update process is modified to exploit the search space in the vicinity of the best solution of the current swarm. The step size of the position update process in ABC is iteratively reduced and controlled by the Golden Section Search (GSS) (Kiefer 1953) strategy.

In past, GSS strategy has been incorporated as a local search strategy with many nature inspired algorithms like differential evolution algorithm (DE), PSO etc. Mininno and Neri proposed a memetic differential evolution algorithm in noisy optimization in which they integrated the GSS strategy with the basic DE algorithm (Mininno and Neri 2010). Oh and Hori develop an optimization strategy named, Golden Section Search driven PSO (Oh and Hori 2006) in which at a time only one particle is updated using the GSS strategy. But in the proposed strategy, GSS is not applied as a local search strategy but applied to fine tune the control parameter ϕ of the ABC's position update process (refer Sect. 3) during the local search phase.

Rest of the paper is organized as follows: Sect. 2 describes brief review on memetic approach. The ABC algorithm is explained in Sect. 3. Memetic ABC (MeABC) is proposed and tested in Sect. 4. In Sect. 5, performance of the proposed strategy is analyzed. The superiority of MeABC over other considered algorithms is verified through its application to various engineering optimization problems in Sect. 6. Finally, in Sect. 7, paper is concluded.

2 Brief review on memetic approach

In the field of optimization, memetic computing is an interesting approach to solve the complex problems (Ong et al. 2010). Memetic is synonymous to *memes* which can be described as "instructions for carrying out behavior, stored in brains" (Susan 1999). Memetic computing is defined as "... a paradigm that uses the notion of *memes* as units of information encoded in computational representations for the purpose of problem solving" (Ong et al. 2010). Memetic Computing can be seen then as a subject which studies complex structures composed of simple modules (*memes*) which interact and evolve adapting to the problem in order to solve it (Neri et al. 2012). A good survey on Memetic Computing can be found in (Ong et al. 2010; Neri et al. 2012, Chen et al. 2011). Memetic Algorithms can be seen as an aspect of the realization or condition based subset of Memetic computing (Chen et al. 2011). The term "Memetic Algorithm" (MA) was first presented by Moscato in (Moscato 1989) as a population based algorithm having local improvement strategy for search of solution. MAs are hybrid search methods that are based on the population-based search framework (Fogel and Michalewicz 1997; Eiben and Smith 2003) and neighbourhood-based local search framework (LS) (Hoos and Stützle 2005). Popular examples of population-based methods include Genetic Algorithms and other Evolutionary Algorithms while Tabu Search and Simulated Annealing (SA) are two prominent local search representatives. The main role of memetic algorithm in evolutionary computing is to provide a local search to establish exploitation of the search space. Local search algorithms can be categorized as (Neri et al. 2012):

- stochastic or deterministic behavior
- single solution or multi-solution based search
- steepest descent or greedy approach based selection.

A local search is thought of as an algorithmic structure converging to the closest local optimum while the global search should have the potential of detecting the global optimum. Therefore, to maintain the proper balance between exploration and exploitation behavior of an algorithm, it is highly required to incorporate a local search approach in the basic population based algorithm to exploit the search region.

Generally, population based search algorithms like genetic algorithm (GA) (Goldberg 1989), evolution strategy (ES) (Beyer and Schwefel 2002), differential evolution (DE) (Price et al. 2005), ant colony optimization (ACO) (Dorigo and Di Caro 1999), particle swarm optimization (PSO) (Kennedy 2006), artificial immune system (Dasgupta 2006), artificial bee colony (Karaboga 2005) etc. are stochastic in nature (Yang 2011). In recent years, researchers are hybridizing the local search procedures with the population based algorithms to improve the exploitation capability of the population based algorithms (Neri and Tirronen 2009; Caponio et al. 2009; Mininno and Neri 2010; Wang et al. 2009; Valenzuela and Smith 2002; Ishibuchi et al. 2003; Ong et al. 2003). Further, MAs have been successfully applied to solve a wide range of complex optimization problems like multiobjective optimization (Tan 2005; Knowles et al. 2008; Goh et al. 2009), continuous optimization (Ong et al. 2003; Ong and Keane 2004), combinatorial optimization (Ishibuchi et al. 2003; Tang et al. 2009; Repoussis et al. 2009), bioinformatics (Richer et al. 2009; Gallo et al. 2009), flow shop scheduling (Ishibuchi et al. 2003), scheduling and routing (Brest et al. 2006), machine learning (Ishibuchi and Yamamoto 2004; Caponio et al. 2007; Ruiz-Torrubiano and Suárez 2010), etc.

Ong and Keane (Ong and Keane 2004) introduced strategies for MAs control that decide at runtime which local search method is to be chosen for the local refinement of the solution. Further, they proposed multiple local search procedures during a MA search in the sprit of Lamarckian learning. Further, Ong et al. (2006) described a classification of *memes* adaptation in adaptive MAs on the basis of the mechanism used and the level of historical knowledge on the *memes* employed. Then the asymptotic convergence properties of the adaptive MAs are analyzed according to the classification. Nguyen et al. (2009) presented a novel probabilistic memetic framework that models MAs as a process involving the decision of embracing the separate actions of evolution or individual learning and analyzed the probability of each process in locating the global optimum. Further, the framework balances evolution and individual learning by governing the learning intensity of each individual according to the theoretical upper bound derived while the search progresses.

In past, very few efforts have been done to incorporate a local search with ABC. Kang et al. (2011) proposed a Hooke Jeeves Artificial Bee Colony algorithm (HJABC) for numerical optimization. In HJABC, authors incorporated a local search technique which is based on Hooke Jeeves method (HJ) (Hooke and Jeeves 1961) with the basic ABC. Further, Mezura-Montes and Velez-Koeppel (2010) introduced a variant of the basic ABC named Elitist Artificial Bee Colony. In their work, the authors integrated two local search strategies. The first local search strategy is used when 30, 40, 50, 60, 70, 80, 90, 95 and 97% of function evaluations have been completed. The purpose of this is to improve the best solution achieved so far by generating a set of 1000 new food sources in its neighbourhood. The other local search works when 45, 50, 55, 80, 82, 84, 86, 88, 90, 91,92, 93, 94, 95, 96, 97, 98, and 99 % of function evaluations have been reached.

Fister et al. (2012) proposed a memetic ABC for Large-Scale Global Optimization. In the proposed approach, ABC is hybridized with two local search heuristics: the Nelder-Mead algorithm (NMA) (Rao and Rao 2009) and the random walk with direction exploitation (RWDE) (Rao and Rao 2009). The former is attended more towards exploration, while the latter more towards exploitation of the search space. The stochastic adaptive rule as specified by Neri (Cotta and Neri 2012) is applied for balancing the exploration and exploitation.

Fei Kang et al. (2011) presented a novel hybrid Hooke Jeeves ABC (HJABC) algorithm with intensification search based on the Hooke Jeeves pattern search and the ABC. In the HJABC, two modification are proposed, one is the fitness (fit_i) calculation function of basic ABC is changed and calculated by Eq. (1) and another is that a Hooke Jeeves local search is incorporated with the basic ABC.

$$fit_i = 2 - SP + \frac{2(SP-1)(p_i-1)}{NP-1}, \qquad (1)$$

here p_i is the position of the solution in the whole population after ranking, $SP \in [1.0, 2.0]$ is the selection pressure. A medium value of $SP = 1.5$ can be a good choice and NP is the number of solutions.

Neri et al. (2011) proposed an unconventional memetic computing strategy for solving continuous optimization problems characterized by memory limitations. The proposed algorithm, unlike employing an explorative evolutionary framework and a set of local search algorithms, employs multiple exploitative search within the main framework and performs a multiple step global search. The proposed local memetic approach is based on a compact evolutionary framework. Iacca et al. (2012) proposed a counter-tendency approach for algorithmic design for memetic computing algorithms. Further Kang et al. (2011) described a Rosenbrock ABC (RABC) that combines Rosenbrock's rotational direction method with ABC for accurate numerical optimization. In RABC, exploitation phase is introduced in the ABC using Rosenbrock's rotational direction method.

3 Artificial bee colony (ABC) algorithm

The ABC algorithm is relatively recent swarm intelligence based algorithm. The algorithm is inspired by the intelligent food foraging behavior of honey bees. In ABC, each solution of the problem is called food source of honey bees.

The fitness is determined in terms of the quality of the food source. In ABC, honey bees are classified into three groups namely employed bees, onlooker bees and scout bees. The numbers of employed bees are equal to the onlooker bees. The employed bees are the bees which searches the food source and gather the information about the quality of the food source. Onlooker bees which stay in the hive and search the food sources on the basis of the information gathered by the employed bees. The scout bee searches new food sources randomly in places of the abandoned foods sources. Similar to the other population-based algorithms, ABC solution search process is an iterative process. After, initialization of the ABC parameters and swarm, it requires the repetitive iterations of the three phases namely employed bee phase, onlooker bee phase and scout bee phase. Each of the phase is described as follows:

3.1 Initialization of the swarm

The parameters for the ABC are the numbers of food sources, the number trials after which a food source is considered to be abandoned and the termination criteria. In the basic ABC, the numbers of food sources are equal to the employed bees or onlooker bees. Initially, a uniformly distributed initial swarm of *SN* food sources, where each food source x_i ($i = 1, 2, ..., SN$) is a *D*-dimensional vector, generated. Here *D* is the number of variables in the optimization problem and x_i represent the *i*th food source in the swarm. Each food source is generated as follows:

$$x_{ij} = x_{minj} + rand[0,1](x_{maxj} - x_{minj}) \tag{2}$$

where x_{minj} and x_{maxj} are bounds of x_i in *j*th direction and *rand* [0,1] is a uniformly distributed random number in the range [0, 1].

3.2 Employed bee phase

In employed bee phase, employed bees modify the current solution (food source) based on the information of individual experience and the fitness value of the new solution. If the fitness value of the new solution is higher than that of the old solution, the bee updates her position with the new one and discards the old one. The position update equation for *i*th candidate in this phase is

$$x'_{ij} = x_{ij} + \phi_{ij}(x_{ij} - x_{kj}) \tag{3}$$

where $k \in \{1, 2, ..., SN\}$ and $j \in \{1, 2, ..., D\}$ are randomly chosen indices. *k* must be different from *i*. ϕ_{ij} is a random number between [−1, 1].

3.3 Onlooker bees phase

After completion of the employed bees phase, the onlooker bees phase starts. In onlooker bees phase, all the employed bees share the new fitness information (nectar) of the new solutions (food sources) and their position information with the onlooker bees in the hive. Onlooker bees analyze the available information and select a solution with a probability, $prob_i$, related to its fitness. The probability $prob_i$ may be calculated using following expression (there may be some other but must be a function of fitness):

$$prob_i = \frac{fitness_i}{\sum_{i=1}^{SN} fitness_i} \tag{4}$$

where $fitness_i$ is the fitness value of the solution *i*. As in the case of the employed bee, it produces a modification on the position in its memory and checks the fitness of the candidate source. If the fitness is higher than that of the previous one, the bee memorizes the new position and forgets the old one.

3.4 Scout bees phase

If the position of a food source is not updated up to predetermined number of cycles, then the food source is assumed to be abandoned and scout bees phase starts. In this phase, the bee associated with the abandoned food source becomes scout bee and the food source is replaced by a randomly chosen food source within the search space. In ABC, predetermined number of cycles is a crucial control parameter which is called *limit* for abandonment.

Assume that the abandoned source is x_i. The scout bee replaces this food source by a randomly chosen food source which is generated as follows

$$x_{ij} = x_{minj} + rand[0,1](x_{maxj} - x_{minj}), \quad for\, j \in \{1, 2, ..., D\} \tag{5}$$

where x_{minj} and x_{maxj} are bounds of x_i in *j*th direction.

3.5 Main steps of the ABC algorithm

Based on the above explanation, it is clear that there are three control parameters in ABC search process: the number of food sources *SN* (equal to number of onlooker or employed bees), the value of *limit* and the maximum number of iterations. The pseudo-code of the ABC is shown in Algorithm 1 (Karaboga and Akay 2009):

Algorithm 1 Artificial Bee Colony Algorithm:

Initialize the parameters;
while Termination criteria is not satisfied **do**
  Step 1: Employed bee phase for generating new food sources.
  Step 2: Onlooker bees phase for updating the food sources depending on their nectar amounts.
  Step 3: Scout bee phase for discovering the new food sources in place of abandoned food sources.
  Step 4: Memorize the best food source found so far.
end while
Output the best solution found so far.

4 Memetic artificial bee colony algorithm

Dervis Karaboga and Bahriye Akay (Karaboga and Akay 2009) compared the different variants of ABC for global optimization and found that the ABC shows poor performance and remains inefficient in exploring the search space. Exploration of the large area of search space and exploitation of the near optimal solution region may be balanced by maintaining the diversity in early and later iterations for any random number based search algorithm. In ABC, any potential solution updates itself using the information provided by a randomly selected potential solution within the current swarm. In this process, a step size, which is a linear combination of a random number $\phi_{ij} \in [-1, 1]$, current solution and a randomly selected solution, are used. Now the quality of the updated solution highly depends upon this step size. If the step size is too large, which may occur if the difference of current solution and randomly selected solution is large with high absolute value of ϕ_{ij}, then updated solution can surpass the true solution and if this step size is too small then the convergence rate of ABC may significantly decrease. A proper balance of this step size can enhance the exploration and exploitation capability of the ABC simultaneously. But, since this step size consists of random component so the balance can not be done manually.

The another way of avoiding the situation of skipping true solution while maintaining the speed of convergence is the incorporation of some memetic search into the basic ABC process. The memetic search algorithm, in case of large step sizes, can search within the area that is jumped by the basic ABC. During the iterations, memetic algorithm exhibits very strong exploitation capability due to executing efficient local search on solutions (Wang et al. 2009).

In this paper, a new local search phase is introduced within the ABC. In the proposed phase, ABC algorithm works as a local search algorithm in which only the best individual of the current swarm updates itself in its neighbourhood. The proposed strategy in ABC is hereby, named as Memetic Search Phase (MSP) and the entire algorithm is named as Memetic ABC (*MeABC*). In MSP, the step size, required to update the best individual in the current swarm is controlled by the Golden Section Search (GSS) approach (Kiefer 1953).

It is clear from the position update Eq. (3) of ABC that the step size of an individual depends upon the random component ϕ and the difference between the individual and a randomly selected individual. Therefore, the random component ϕ is an important parameter which decides direction and step size of an individual. In the MSP, the GSS strategy is used to fine tune the value of ϕ dynamically and iteratively, in order to exploit the region nearby best solution.

Original GSS approach finds the optima of a unimodal continuous function without using any gradient information of the function. GSS processes the interval $[a = -1.2, b = 1.2]$ and generates two intermediate points:

$$F_1 = b - (b - a) \times \psi, \tag{6}$$

$$F_2 = a + (b - a) \times \psi, \tag{7}$$

where $\psi = 0.618$ is the golden ratio. The pseudo-code of GSS algorithm is shown in Algorithm 2:

Algorithm 2 Golden Section Search Pseudo Code:

```
Input Optimization function Minf(x) subject to a ≤ x ≤ b and a termination condition;
while Termination condition do
  Compute F1 = b − (b − a) × ψ and F2 = a + (b − a) × ψ;
  Calculate f(F1) and f(F2);
  if f(F1) < f(F2) then
    b = F2 and the solution lies in the range [a, b];
  else
    a = F1 and the solution lies in the range [a, b];
  end if
end while
```

In this paper, *MSP* is applied in each iteration as a local search technique to find the best suitable value of *ABC* parameter ϕ_{ij} corresponding to the best food position. More specifically, in every cycle of the ABC, the best solution updates its position until the step size is equal or less than a predefined limit to avoid the stagnation and loss of computational efficiency. Here, the step size is controlled using the GSS, which iteratively decreases the range of ϕ_{ij} for the best particle of the current swarm. Here the local search in the space where ϕ_{ij} varies, may be seen as the minimization of ϕ_{ij} over the variable F (= F_1 *or* F_2) of objective function f in the direction determined by x_{best} (the best solution) and x_k (a randomly selected solution). At first the range of ϕ_{ij} is set to [a, b] where $a = -1.2$ and $b = 1.2$, then it is reduced using the Eqs. (6) and (7) iteratively. Therefore, the local search attempts to solve the minimization problem given in Eq. (8):

$$\min f(\phi) \quad in\,[a, b]; \tag{8}$$

Here, it is assumed that the optimization problem under consideration is of minimization type. The pseudo-code of the proposed memetic search strategy in ABC is shown in Algorithm 3.

Algorithm 3 Memetic Search Phase (MSP):

```
Input optimization function Minf(x), a and b;
Select the best solution x_best in the swarm.
while (|a − b| < ϵ) do
  Compute F1 = b − (b − a) × ψ and F2 = a + (b − a) × ψ;
  Generate a new solution x_new1 using F1 and x_new2 using F2 by Algorithm 4.
  Calculate f(x_new1) and f(x_new2).
  if f(x_new1) < f(x_new2) then
    b = F2
    if f(x_new1) < f(x_best) then
      x_best = x_new1;
    end if
  else
    a = F1
    if f(x_new2) < f(x_best) then
      x_best = x_new2;
    end if
  end if
end while
```

Algorithm 4 New solution generation:

Input F and best solution x_{best} ;
for $j = 1$ to D **do**
  if $U(0,1) > p_r$ **then**
    $x_{newj} = x_{bestj} + F(x_{bestj} - x_{kj})$;
  else
    $x_{newj} = x_{bestj}$;
  end if
end for
Return x_{new}

In Algorithm 3 and 4, ϵ determines the termination of local search. p_r is a perturbation rate (a number between 0 and 1) which controls the amount of perturbation in the best solution, $U(0,1)$ is a uniformly distributed random number between 0 and 1, D is the dimension of the problem and x_k is a randomly selected solution within swarm. See Sect. 5.2 for details of these parameter settings.

In *MeABC*, only the best solution of the current swarm updates itself in its neighbourhood. Here, it should be noted that GSS is not applied as a local search strategy but the ABC position update process is modified by adjusting the value of ϕ for exploiting the neighboring area of the best solution. Figures 1, 2 and 3 show the effect of memetic search phase, used to update an individual in two dimensional search space for *Easom's* function (f_{19}), refer Table 1.

Figure 1 shows iterative change through iterations in the range of ϕ_{ij}. Figure 2 shows position change behavior of the best solution. Figure 3 shows iterative reduction of the step size of the best solution.

The proposed *MeABC* consists of four phases: employed bee phase, onlooker bee phase, scout bee phase and memetic search phase out of which employed bee phase, onlooker bee phase and scout bee phase are similar to the basic ABC except the position update equation of an individual. The position update equation of MeABC is given in equation 9. The inspiration behind the development of this position update is PSO (Kennedy and Eberhart 1995) and Gbest Guided ABC (GABC) (Zhu and Kwong 2010). Due to the insertion of best individual, now other individuals are benefitted from the information of best solution.

$$x'_{ij} = x_{ij} + \phi_{ij}(x_{ij} - x_{kj}) + \psi_{ij}(x_{bestj} - x_{ij}), \tag{9}$$

here, ψ_{ij} is a uniform random number in [0, C], where C is a non-negative constant and a parameter in MeABC.

The working of the memetic search phase is explained in Algorithm 3. The pseudo-code of the *MeABC* algorithm is shown in Algorithm 5.

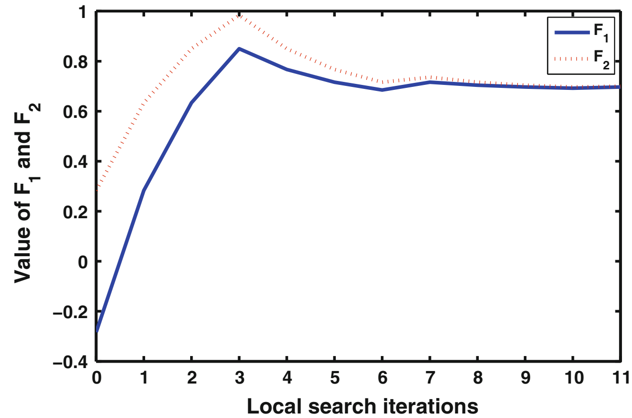

Fig. 1 Range of ϕ_{ij} during *MSP* for f_{19} in two dimension search space

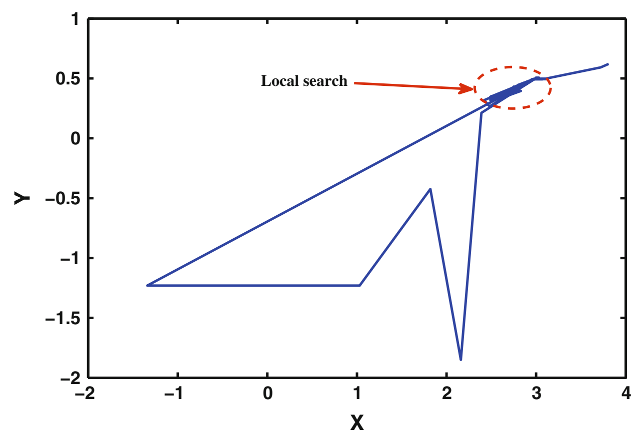

Fig. 2 Best solution movement in the two dimension search space for f_{19}

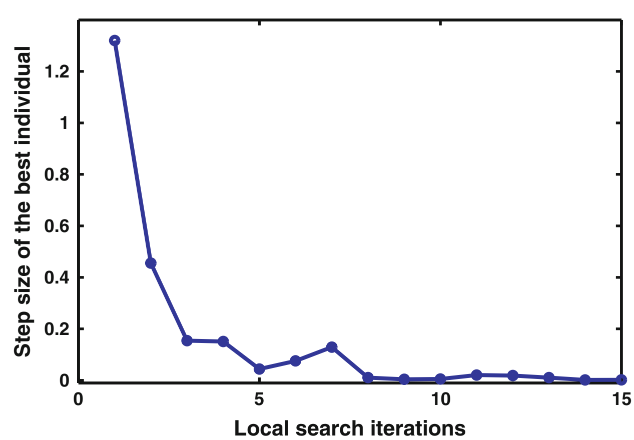

Fig. 3 Best solution step size during *MSP*, in the two dimension search space for f_{19}

Table 1 Test problems

Test problem	Objective function	Search range	Optimum value	D	Acceptable error
Zakharov	$f_1(x) = \sum_{i=1}^{D} x_i{}^2 + (\sum_{i=1}^{D} \frac{ix_i}{2})^2 + (\sum_{i=1}^{D} \frac{ix_1}{2})^4$	[−5.12, 5.12]	$f(0) = 0$	30	1.0E−02
Salomon Problem	$f_2(x) = 1 - cos(2\pi\sqrt{\sum_{i=1}^{D} x_i^2}) + 0.1(\sqrt{\sum_{i=1}^{D} x_i^2})$	[−100, 100]	$f(0) = 0$	30	1.0E−01
Sum of different powers	$f_3(x) = \sum_{i=1}^{D} \lvert x_i \rvert^{i+1}$	[−1, 1]	$f(0) = 0$	30	1.0E−05
Quartic function	$f_4(x) = \sum_{i=1}^{D} ix_i^4 + \text{random}[0, 1)$	[−1.28, 1.28]	$f(0) = 0$	30	1.0
Inverted cosine wave	$f_5(x) = -\sum_{i=1}^{D-1} \left(\exp\left(\frac{-(x_i^2+x_{i+1}^2+0.5x_ix_{i+1})}{8}\right) \times \mathrm{I}\right)$	[−5 5]	$f(0) = -D + 1$	10	1.0E−05
Neumaier 3 Problem (NF3)	$f_6(x) = \sum_{i=1}^{D} (x_i - 1)^2 - \sum_{i=2}^{D} x_ix_{i-1}$	$[-D^2, D^2]$	$f(0) = -(D \times (D + 4)(D - 1))/6.0$	10	1.0E−01
Levy montalvo 1	$f_7(x) = \frac{\pi}{D}(10\sin^2(\pi y_1) + \sum_{i=1}^{D-1}(y_i - 1)^2(1 + 10sin^2(\pi y_{i+1}))$ where $y_i = 1 + \frac{1}{4}(x_i + 1)$ $+ (y_D - 1)^2)$,	[−10, 10]	$f(-1) = 0$	30	1.0E−05
Levy montalvo 2	$f_8(x) = 0.1(\sin^2(3\pi x_1) + \sum_{i=1}^{D-1}(x_i - 1)^2(1 + \sin^2(3\pi x_{i+1})) + (x_D - 1)^2(1 + \sin^2(2\pi x_D))$	[−5, 5]	$f(1) = 0$	30	1.0E−05
Beale function	$f_9(x) = (1.5 - x_1(1 - x_2))^2 + (2.25 - x_1(1 - x_2^2))^2 + (2.625 - x_1(1 - x_2^3))^2$	[−4.5, 4.5]	$f(3, 0.5) = 0$	2	1.0E−05
Colville function	$f_{10}(x) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2 + 90(x_4 - x_3^2)^2 + (1 - x_3)^2 + 10.1[(x_2 - 1)^2 + (x_4 - 1)^2] + 19.8(x_2 - 1)(x_4 - 1)$	[−10, 10]	$f(1) = 0$	4	1.0E−05
Kowalik function	$f_{11}(x) = \sum_{i=1}^{11}\left(a_i - \frac{x_1(b_i^2+b_ix_2)}{b_i^2+b_ix_3+x_4}\right)^2$	[−5, 5]	$f(0.1928, 0.1908, 0.1231, 0.1357) = 3.07E{-04}$	4	1.0E−05
Shifted Rosenbrock	$f_{12}(x) = \sum_{i=1}^{D-1}(100(z_i^2 - z_{i+1})^2 + (z_i - 1)^2) + f_{bias}$, $z = x - o + 1$, $x = [x_1, x_2, ...x_D]$, $o = [o_1, o_2, ...o_D]$	[−100, 100]	$f(o) = f_{bias} = 390$	10	1.0E−01
Shifted Sphere	$f_{13}(x) = \sum_{i=1}^{D} z_i^2 + f_{bias}$, $z = x - o$, $x = [x_1, x_2, ...x_D]$, $o = [o_1, o_2, ...o_D]$	[−100, 100]	$f(o) = f_{bias} = -450$	10	1.0E−05
Shifted Rastrigin	$f_{14}(x) = \sum_{i=1}^{D}(z_i^2 - 10\cos(2\pi z_i) + 10) + f_{bias} z = (x - o), x = (x_1, x_2, ...x_D),$ $o = (o_1, o_2, ...o_D)$	[−5, 5]	$f(o) = f_{bias} = -330$	10	1.0E−02
Shifted Schwefel	$f_{15}(x) = \sum_{i=1}^{D}(\sum_{j=1}^{i} z_j)^2 + f_{bias}$, $z = x - o$, $x = [x_1, x_2, ...x_D]$, $o = [o_1, o_2, ...o_D]$	[−100, 100]	$f(o) = f_{bias} = -450$	10	1.0E−05
Shifted Griewank	$f_{16}(x) = \sum_{i=1}^{D} \frac{z_i^2}{4000} - \prod_{i=1}^{D} \cos(\frac{z_i}{\sqrt{i}}) + 1 + f_{bias}, z = (x - o), x = [x_1, x_2, ...x_D], o = [o_1, o_2, ...o_D]$	[−600, 600]	$f(o) = f_{bias} = -180$	10	1.0E−05
Shifted Ackley	$f_{17}(x) = -20\exp(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} z_i^2}) - \exp(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi z_i)) + 20 + e + f_{bias},$ $z = (x - o), x = (x_1, x_2, ...x_D), o = (o_1, o_2, ...o_D)$	[−32, 32]	$f(o) = f_{bias} = -140$	10	1.0E−05
Goldstein-Price	$f_{18}(x) = (1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2))(30 + (2x_1 - 3x_2)^2(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2))$	[−2, 2]	$f(0, -1) = 3$	2	1.0E−14
Easom's function	$f_{19}(x) = -\cos x_1 \cos x_2 e^{((-(x_1-\pi)^2-(x_2-\pi)^2))}$	[−10, 10]	$f(\pi,\pi) = -1$	2	1.0E−13
Meyer and Roth Problem	$f_{20}(x) = \sum_{i=1}^{5}(\frac{x_1x_3t_i}{1+x_1t_i+x_2v_i} - y_i)^2$	[−10, 10]	$f(3.13, 15.16, 0.78) = 0.4E{-04}$	3	1.0E−03

Algorithm 5 Memetic search in ABC (*MeABC*):

Initialize the parameters;
while Termination criteria **do**
  Step 1: Employed bee phase for generating new food sources.
  Step 2: Onlooker bees phase for updating the food sources depending on their nectar amounts.
  Step 3: Scout bee phase for discovering the new food sources in place of abandoned food sources.
  Step 4: Apply memetic search phase using Algorithm 3
end while
Print best solution.

5 Experimental results and discussion

5.1 Test problems under consideration

In order to analyze the performance of *MeABC*, 20 different global optimization problems (f_1 to f_{20}) are selected (listed in Table 1). These are continuous optimization problems and have different degrees of complexity and multimodality. Test problems $f_1 - f_{11}$ and $f_{18} - f_{20}$ are taken from (Ali et al. 2005) and test problems $f_{12} - f_{17}$ are taken from (Suganthan et al. 2005) with the associated offset values.

5.2 Experimental setting

To prove the efficiency of *MeABC*, it is compared with *ABC* and some recent algorithms namely Gbest-guided ABC (*GABC*) (Zhu and Kwong 2010), Best-So-Far ABC (*BSFABC*) (Banharnsakun et al. 2011), Modified ABC (*MABC*) (Akay et al. 2010), Hooke Jeeves ABC (*HJABC*) (Kang et al. 2011), Opposition based lévy flight ABC (OBLFABC) (Harish et al. 2012) and Scale factor local search DE (*SFLSDE*) (Neri and Tirronen 2009). To test *MeABC*, *ABC*, *GABC*, *BSFABC*, *MABC*, *HJABC*, *OBLFABC* and *SFLSDE* over considered problems, following experimental setting is adopted:

- Colony size $NP = 50$ (Diwold et al. 2011; El-Abd 2011),
- $\phi_{ij} = rand[-1, 1]$,
- Number of food sources $SN = NP/2$,
- $limit = 1500$ (Karaboga and Akay 2010; Akay et al. 2010),
- $C = 1.5$ (Zhu and Kwong 2010),
- The stopping criteria is either maximum number of function evaluations (which is set to be 200,000) is reached or the acceptable error (mentioned in Table 1) has been achieved,
- The number of simulations/run =100,
- In order to investigate the effect of the parameter p_r, described by Algorithm 4 on the performance of *MeABC*, its sensitivity with respect to different values of p_r in the range [0.1, 1], is examined in the Figure 4. It is clear that the test problems are very sensitive towards p_r and value 0.4 gives comparatively better results. Therefore $p_r = 0.4$ is selected for the experiments in this paper.
- Value of termination criteria in memetic search phase is set to be $\epsilon = 0.01$.
- Parameter settings for the algorithms *GABC*, *BSFABC*, *MABC*, *HJABC*, *OBLFABC* and *SFLSDE* are similar to their original research papers.

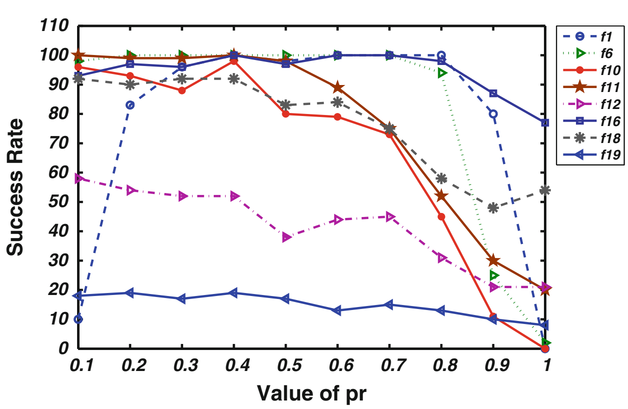

Fig. 4 Effect of parameter p_r on success rate

5.3 Results comparison

Numerical results with experimental settings of subsection 5.2 are given in Table 2. In Table 2, standard deviation (*SD*), success rate (*SR*), mean error (*ME*) and average number of function evaluations (*AFE*) are reported. Table 2 shows that most of the time *MeABC* outperforms in terms of reliability, efficiency and accuracy as compared to the other considered algorithms. Some more intensive analyses based on performance indices and boxplots have been carried out for the results of *MeABC* and considered algorithms.

Figure 5 shows the convergence characteristics in terms of the error of the median run of each algorithm for

Table 2 Comparison of the results of test problems

Test function	Measure	MeABC	ABC	GABC	BSFABC	MABC	HJABC	OBLFABC	SFLSDE
f_1	SD	5.10E−04	1.52E+01	1.89E+01	1.22E+01	1.02E−01	5.94E−02	1.70E+01	7.72E−04
	ME	9.58E−03	9.73E+01	9.73E+01	8.49E+01	1.46E−01	8.94E−02	1.08E+02	9.11E−03
	AFE	94564.52	200000	200000.01	200000	200005.52	198146	200030.9	131560.85
	SR	100	0	0	0	0	4	0	100
f_2	SD	3.28E−02	6.25E−02	3.38E−02	6.58E−02	3.46E−02	3.45E−02	4.62E−02	3.91E−02
	ME	9.24E−01	9.56E−01	9.32E−01	9.53E−01	9.31E−01	9.12E−01	9.34E−01	9.23E−01
	AFE	18209.52	149071.35	75922.34	184747.81	27739.5	20266.59	86086.26	23117.77
	SR	100	68	98	74	100	100	94	100
f_3	SD	2.99E−06	2.79E−06	2.73E−06	2.54E−06	1.99E−06	2.94E−06	2.94E−06	1.97E−06
	ME	5.24E−06	5.84E−06	5.60E−06	5.92E−06	7.51E−06	5.44E−06	6.56E−06	7.20E−06
	AFE	4738.72	15619.5	9290.5	14278	9422	4989.58	7693.7	8966.62
	SR	100	100	100	100	100	100	100	100
f_4	SD	4.02E−01	5.26E−01	6.08E−01	5.56E−01	4.35E−01	4.62E−01	5.14E−01	1.53E−01
	ME	9.20E+00	1.17E+01	1.05E+01	9.96E+00	9.87E+00	9.01E+00	9.59E+00	1.51E+00
	AFE	200017.89	200040.14	200020.76	200034.71	200013.78	200025.36	200030.13	199773.56
	SR	0	0	0	0	0	0	0	0
f_5	SD	2.05E−06	4.32E−05	2.58E−06	1.98E−01	1.61E−06	1.71E−06	1.74E−06	7.95E−01
	ME	8.14E−06	1.10E−05	6.94E−06	6.16E−02	8.42E−06	8.61E−06	8.31E−06	6.78E−01
	AFE	37002.11	74479.03	45459.4	124146.79	64730.3	79129.95	26016.21	125272
	SR	100	99	100	80	100	99	100	47
f_6	SD	9.50E−03	9.54E−01	1.75E+00	5.91E+00	9.57E−02	1.24E−02	1.27E−02	1.36E−02
	ME	9.01E−02	1.08E+00	1.45E+00	4.67E+00	1.13E−01	8.59E−02	8.80E−02	8.49E−02
	AFE	21903.53	198665.08	195087.89	200022.97	134358.49	48774.77	18549.06	22721.63
	SR	100	4	8	0	97	100	100	100
f_7	SD	7.97E−07	2.24E−06	1.99E−06	2.41E−06	7.37E−07	6.05E−07	1.58E−06	8.26E−07
	ME	9.16E−06	7.21E−06	7.84E−06	6.98E−06	9.17E−06	9.24E−06	8.31E−06	9.13E−06
	AFE	11770.12	19614.5	13030.5	26863	22548.5	19214.65	15241.73	21966.27
	SR	100	100	100	100	100	100	100	100
f_8	SD	7.56E−07	2.13E−06	1.83E−06	2.41E−06	8.14E−07	6.49E−07	1.78E−06	8.06E−07
	ME	9.10E−06	7.35E−06	8.10E−06	7.13E−06	9.06E−06	9.18E−06	8.31E−06	9.17E−06
	AFE	13031.58	22016	14283	28673.5	20985.5	17368.82	17270.74	24387.23
	SR	100	100	100	100	100	100	100	100
f_9	SD	2.90E−06	1.73E−06	2.92E−06	6.07E−05	3.06E−06	2.99E−06	2.83E−06	2.87E−06
	ME	5.14E−06	8.58E−06	5.14E−06	2.19E−05	5.24E−06	4.76E−06	7.62E−06	5.02E−06
	AFE	2688.15	15768.28	9344.1	50222.41	10082.84	4839.56	7022.41	3002.58
	SR	100	100	100	92	100	100	100	100
f_{10}	SD	2.42E−03	1.07E−01	1.24E−02	2.99E−02	7.72E−03	2.52E−03	1.60E−02	2.12E−03
	ME	7.03E−03	1.67E−01	1.58E−02	2.18E−02	1.26E−02	7.14E−03	1.50E−02	6.82E−03
	AFE	30813.41	198058.11	154523.83	155548.21	144033.7	43566.59	120442.76	9719.26
	SR	100	2	42	47	54	100	69	100
f_{11}	SD	2.03E−05	7.32E−05	3.57E−05	8.16E−05	6.84E−05	5.58E−05	9.67E−06	2.11E−04
	ME	8.17E−05	1.69E−04	9.27E−05	1.45E−04	1.90E−04	1.18E−04	9.82E−05	5.48E−04
	AFE	47100.43	178355.83	98389.57	140918.92	191449.61	127096.42	68245.49	171243.91
	SR	100	23	90	51	10	58	97	17
f_{12}	SD	7.83E−02	9.44E−01	7.56E−02	5.63E+00	9.67E−01	7.42E−01	2.87E+00	7.63E−01
	ME	1.03E−01	6.79E−01	9.30E−02	2.96E+00	6.80E−01	5.79E−01	6.40E−01	2.48E−01
	AFE	103949.03	175270.8	100594.41	185221.92	163969.65	151927.05	62464.38	62586.87
	SR	97	24	93	13	39	46	88	96

Table 2 continued

Test function	Measure	MeABC	ABC	GABC	BSFABC	MABC	HJABC	OBLFABC	SFLSDE
f_{13}	SD	1.92E−06	2.37E−06	2.02E−06	2.53E−06	1.72E−06	1.89E−06	2.29E−06	1.69E−06
	ME	7.86E−06	6.87E−06	7.28E−06	6.99E−06	8.05E−06	7.62E−06	7.72E−06	7.95E−06
	AFE	5535.34	9069	5586.5	18028	8731.5	7941.75	6718.35	12234.46
	SR	100	100	100	100	100	100	100	100
f_{14}	SD	1.24E+01	1.15E+01	1.02E+01	1.43E+01	9.16E+00	1.34E+01	1.16E+01	1.56E+01
	ME	8.23E+01	8.67E+01	8.47E+01	1.22E+02	8.27E+01	8.47E+01	9.01E+01	1.14E+02
	AFE	200012.07	200011.96	200007.18	200036.47	200015.62	200056.91	200031	199771.79
	SR	0	0	0	0	0	0	0	0
f_{15}	SD	3.33E+03	3.46E+03	3.48E+03	8.42E+03	2.88E+03	3.01E+03	3.07E+03	6.66E+03
	ME	1.05E+04	1.10E+04	1.10E+04	2.69E+04	9.76E+03	1.01E+04	1.12E+04	2.27E+04
	AFE	200025.38	200025.28	200018.02	200036.25	200015.8	200029.99	200032.45	199768.59
	SR	0	0	0	0	0	0	0	0
f_{16}	SD	7.35E−04	2.55E−03	9.15E−06	5.71E−03	1.89E−03	2.46E−06	2.86E−03	7.35E−04
	ME	7.95E−05	8.38E−04	5.56E−06	4.22E−03	5.23E−04	7.48E−06	1.09E−03	8.20E−05
	AFE	41069.37	80839.77	42393.56	112424.09	81447.47	63630.7	72268.18	43725.22
	SR	99	90	99	62	93	100	86	99
f_{17}	SD	1.28E−06	1.71E−06	1.48E−06	1.83E−06	1.02E−06	9.38E−07	1.50E−06	9.99E−07
	ME	8.64E−06	7.96E−06	8.38E−06	8.03E−06	9.00E−06	8.88E−06	8.43E−06	8.79E−06
	AFE	10010.84	16833	9353.5	31072.5	14167.57	15113.51	11605.06	18202.59
	SR	100	100	100	100	100	100	100	100
f_{18}	SD	4.38E−15	9.98E−07	4.37E−15	4.08E−15	4.06E−15	3.92E−15	4.40E−15	4.79E−14
	ME	4.73E−15	2.36E−07	5.05E−15	6.60E−15	4.94E−15	4.34E−15	5.36E−15	5.68E−14
	AFE	3001.46	102407.85	3862.8	13795.04	13702.76	12325.91	4188.2	116882.18
	SR	100	71	100	100	100	100	100	43
f_{19}	SD	7.75E−14	9.48E−05	3.79E−13	2.82E−14	1.18E−03	3.76E−05	3.30E−14	2.72E−14
	ME	1.50E−14	2.26E−05	8.68E−14	3.86E−14	8.34E−04	9.02E−06	4.30E−14	5.01E−14
	AFE	37595.23	190415.59	45276.99	4582.08	200024.99	186876.19	13219.66	8217.09
	SR	100	10	99	100	0	15	100	100
f_{20}	SD	2.72E−06	2.84E−06	2.83E−06	1.22E−05	2.84E−06	2.80E−06	2.88E−06	5.33E−04
	ME	1.95E−03	1.95E−03	1.95E−03	1.95E−03	1.95E−03	1.95E−03	1.95E−03	1.16E−03
	AFE	3228.44	25398.62	4168.08	20077.65	8489.06	5193.96	5652.2	595.98
	SR	100	100	100	99	100	100	100	100

functions on which *ABC* and *MeABC* algorithms achieved 100 % success rate within the specified maximum function evaluations (to carry out fair comparison of convergence rate). It can be observed that the convergence of *MeABC* is relatively better than *ABC*.

MeABC and the considered algorithms are compared through *SR*, *ME* and *AFE* in Table 2. First *SR* is compared for all these algorithms and if it is not possible to distinguish the algorithms based on *SR* then comparison is made on the basis of *AFE*. *ME* is used for comparison if it is not possible on the basis of *SR* and *AFE* both. Outcome of this comparison is summarized in Table 3. In Table 3, '+' indicates that the *MeABC* is better than the considered algorithm and '−' indicates that the proposed algorithm is not better than considered algorithms. The last row of Table 3 establishes the superiority of *MeABC* over the considered algorithms.

For the purpose of comparison in terms of consolidated performance, boxplot analyses have been carried out for all the considered algorithms. The empirical distribution of data is efficiently represented graphically by the boxplot analysis tool (Williamson et al. 1989). The boxplots for *MeABC*, *ABC*, *GABC*, *BSFABC*, *MABC*, *HJABC*, *OBLFABC* and *SFLSDE* are shown in Fig. 6. It can be observed from Fig. 6 that *MeABC* performs better than the basic *ABC* and the considered algorithms as interquartile range and median are low comparatively.

Further, to compare the considered algorithms, by giving weighted importance to the success rate, the mean error and the average number of function evaluations,

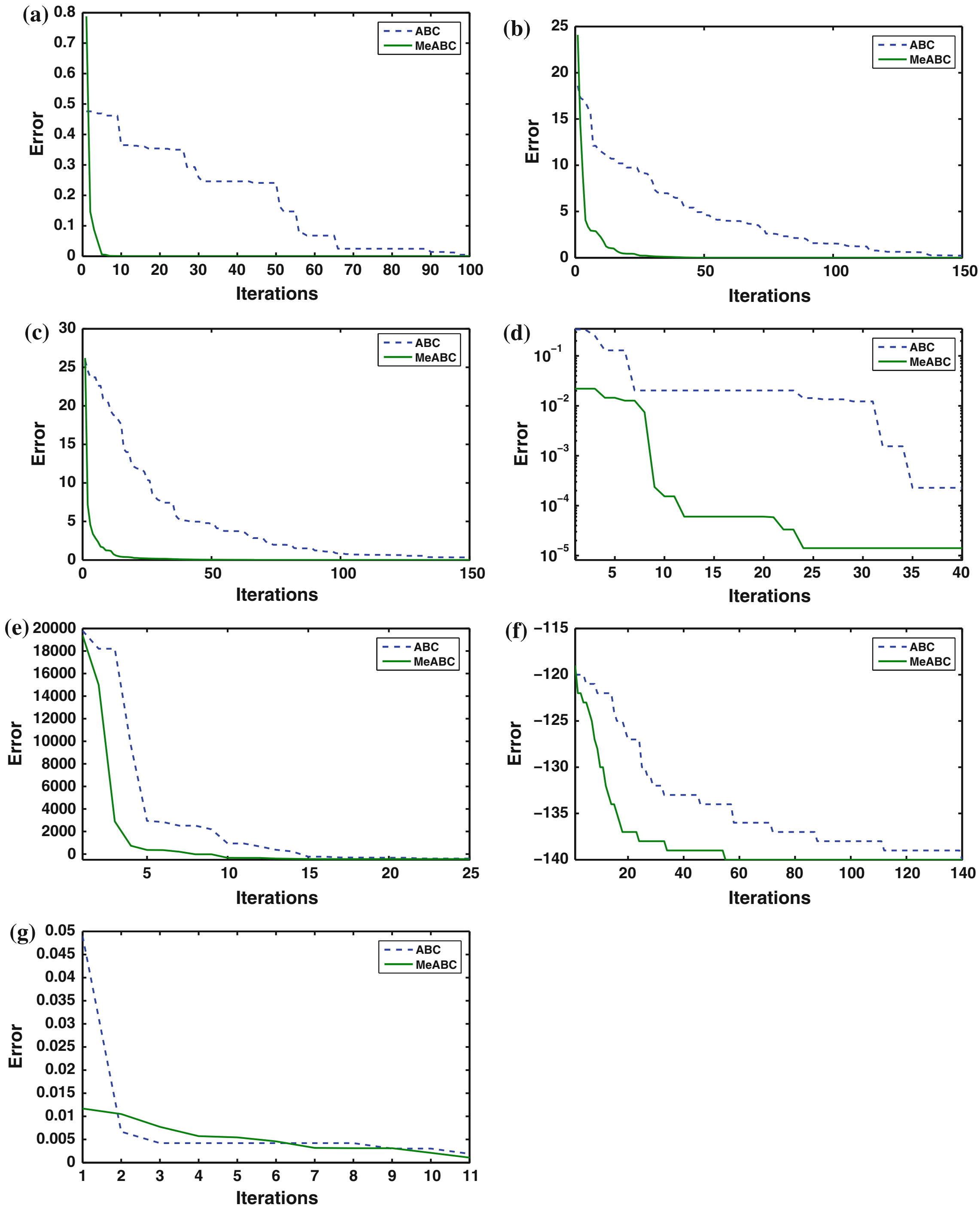

Fig. 5 Convergence characteristics of ABC and MeABC for functions **a** f_3, **b** f_7, **c** f_8, **d** f_9, **e** f_{13}, **f** f_{17}, **g** f_{20}

performance indices (*PI*) are calculated (Thakur Deep 2007). The values of *PI* for the *MeABC*, *ABC*, *GABC*, *BSFABC*, *MABC*, *HJABC*, *OBLFABC* and *SFLSDE* are calculated by using following equations:

$$PI = \frac{1}{N_p}\sum_{i=1}^{N_p}(k_1\alpha_1^i + k_2\alpha_2^i + k_3\alpha_3^i)$$

where $\alpha_1^i = \frac{Sr^i}{Tr^i}$; $\alpha_2^i = \begin{cases} \frac{Mf^i}{Af^i}, & \text{if } Sr^i > 0. \\ 0, & \text{if } Sr^i = 0. \end{cases}$; and $\alpha_3^i = \frac{Mo^i}{Ao^i}$ $i = 1, 2, ..., N_p$

- Sr^i = Successful simulations/runs of ith problem.
- Tr^i = Total simulations of ith problem.
- Mf^i = Minimum of average number of function evaluations used for obtaining the required solution of ith problem.

Table 3 Summary of Table 2 outcome

Function	MeABC Vs ABC	MeABC Vs GABC	MeABC Vs VSFABC	MeABC Vs MABC	MeABC Vs HJABC	MeABC Vs OBLFABC	MeABC Vs SFLSDE
f_1	+	+	+	+	+	+	+
f_2	+	+	+	+	+	+	+
f_3	+	+	+	+	−	+	+
f_4	+	+	+	+	−	+	−
f_5	+	+	+	+	+	−	+
f_6	+	+	+	+	+	−	+
f_7	+	+	+	+	+	+	+
f_8	+	+	+	+	+	+	+
f_9	+	+	+	+	+	+	+
f_{10}	+	+	+	+	+	+	−
f_{11}	+	+	+	+	+	+	+
f_{12}	+	+	+	+	+	+	+
f_{13}	+	−	+	+	+	+	+
f_{14}	+	+	+	−	+	+	+
f_{15}	+	+	+	−	−	+	+
f_{16}	+	+	+	+	−	+	+
f_{17}	+	−	+	+	+	+	+
f_{18}	+	+	+	+	+	+	+
f_{19}	+	+	−	+	+	−	−
f_{20}	+	+	+	+	+	+	−
Total number of + sign	20	18	19	18	16	17	16

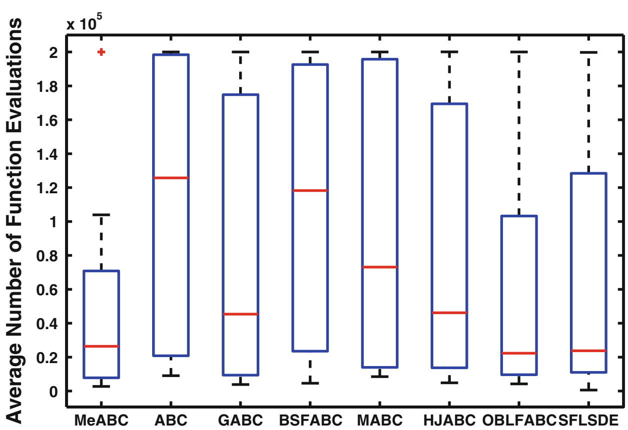

Fig. 6 *Boxplots* graphs for average number of function evaluation

- Af^i = Average number of function evaluations used for obtaining the required solution of ith problem.
- Mo^i = Minimum of mean error obtained for the ith problem.
- Ao^i = Mean error obtained by an algorithm for the ith problem.
- N_p = Total number of optimization problems evaluated.

The weights assigned to the success rate, the average number of function evaluations and the mean error are represented by k_1, k_2 and k_3 respectively where $k_1 + k_2 + k_3 = 1$ and $0 \le k_1, k_2, k_3 \le 1$. To calculate the *PI*s, equal weights are assigned to two variables while weight of the remaining variable vary from 0 to 1 as given in (Thakur Deep 2007). Following are the resultant cases:

1. $k_1 = W, k_2 = k_3 = \frac{1-W}{2}, 0 \le W \le 1;$
2. $k_2 = W, k_1 = k_3 = \frac{1-W}{2}, 0 \le W \le 1;$
3. $k_3 = W, k_1 = k_2 = \frac{1-W}{2}, 0 \le W \le 1$

The graphs corresponding to each of the cases (1), (2) and (3) for the considered algorithms are shown in Fig. 7a, b, and c respectively. In these figures the weights k_1, k_2 and k_3 are represented by horizontal axis while the *PI* is represented by the vertical axis.

In case (1), average number of function evaluations and the mean error are given equal weights. *PIs* of the considered algorithms are superimposed in Fig. 7a for comparison of the performance. It is observed that *PI* of *MeABC* is higher than the considered algorithms. In case (2), equal weights are assigned to the success rate and average number of function evaluations and in case (3), equal weights are assigned to the success rate and the mean

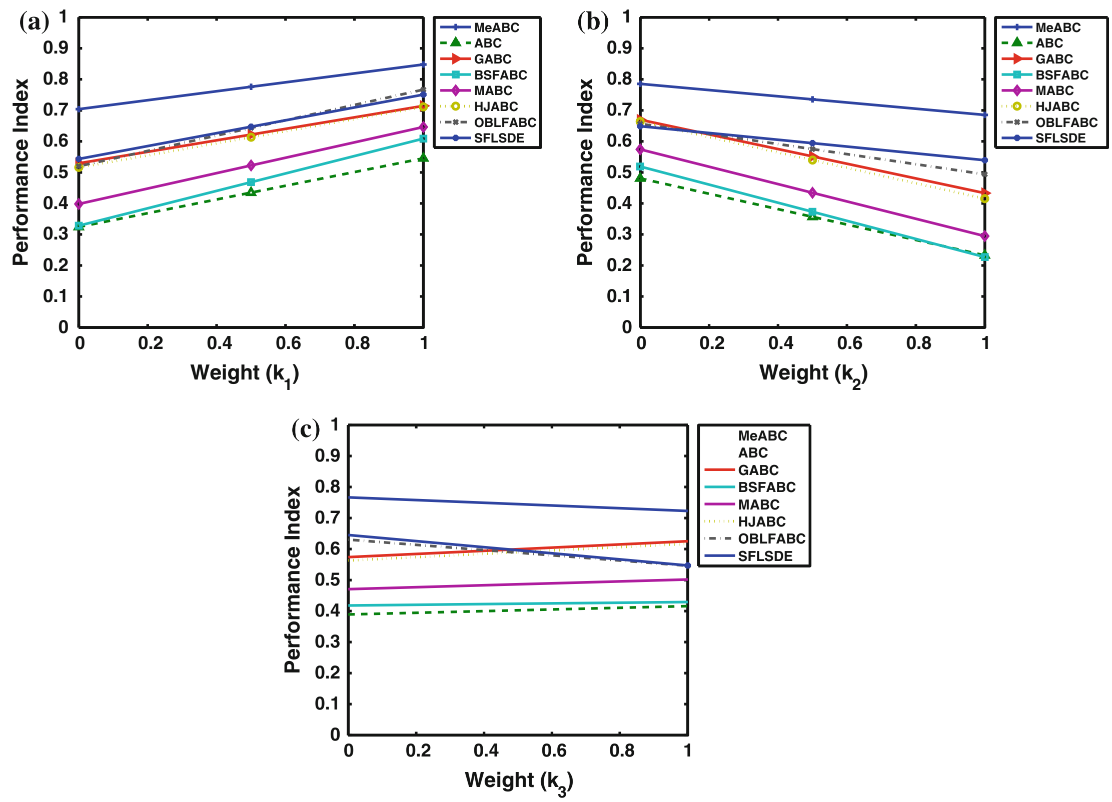

Fig. 7 Performance index for test problems; **a** for case (1), **b** for case (2) and **c** for case (3)

error. It is clear from Fig. 7b, c that the algorithms perform same as in case (1).

6 Applications of *MeABC* to engineering optimization problems

To see the robustness of the proposed strategy, four real world engineering optimization problems, namely, Lennard-Jones (Clerc M. List based pso for real problems. http://clerc.maurice.free.fr/pso/ListBasedPSO/ListBasedPSO28PSOsite29.pdf and 16 2012), parameter estimation for frequency-modulated (FM) sound waves (Das and Suganthan 2010), Compression Spring (Onwubolu and Babu 2004; Sandgren 1990) and Welded beam design optimization problem (Ragsdell and Phillips 1976; Mahdavi et al. 2010) are also solved. The considered engineering optimization problems are described as follows:

6.1 Lennard-Jones

The function to minimize is a kind of potential energy of a set of N atoms. The position X_i of the atom i has three coordinates, and therefore the dimension of the search space is $3N$. In practice, the coordinates of a point X are the concatenation of the ones of the X_i. In short, we can write $X = (X_1, X_2, ..., X_N)$, and we have then

$$E_1(\mathbf{X}) = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \left(\frac{1}{\|X_i - X_j\|^{2\alpha}} - \frac{1}{\|X_i - X_j\|^{\alpha}} \right)$$

In this study $N = 5$, $\alpha = 6$, and the search space is [2,2] (Clerc M. List based pso for real problems. http://clerc.maurice.free.fr/pso/ListBasedPSO/ListBasedPSO28PSOsite29.pdf and 16 2012).

6.2 Frequency-modulated (FM) sound wave

Frequency-modulated (FM) sound wave synthesis has an important role in several modern music systems. The parameter optimization of an FM synthesizer is a six dimensional optimization problem where the vector to be optimized is $\mathbf{X} = \{a_1, w_1, a_2, w_2, a_3, w_3\}$ of the sound wave given in Eq. (10). The problem is to generate a sound (1) similar to target (2). This problem is a highly complex multimodal one having strong epistasis, with minimum value $f(\mathbf{X}_{sol}) = 0$. This problem has been tackled using genetic algorithms (GAs) in (Akay et al. 2010; Ali et al. 2005). The expressions for the estimated sound and the target sound waves are given as:

$$y(t) = a_1 sin(w_1 t\theta + a_2 sin(w_2 t\theta + a_3 sin(w_3 t\theta))) \quad (10)$$

$$y_0(t) = (1.0)sin((5.0)t\theta - (1.5)sin((4.8)t\theta + (2.0)sin((4.9)t\theta))) \quad (11)$$

respectively where $\theta = 2\pi/100$ and the parameters are defined in the range $[-6.4, 6.35]$. The fitness function is the summation of square errors between the estimated wave (1) and the target wave (2) as follows:

$$E_2(\mathbf{X}) = \sum_{i=0}^{100} (y(t) - y_0(t))^2$$

Acceptable error for this problem is $1.0E{-}05$, i.e. an algorithm is considered successful if it finds the error less than acceptable error in a given number of generations.

6.3 Compression spring

The considered third engineering optimization application is compression spring problem (Onwubolu and Babu 2004; Sandgren 1990). This problem minimizes the weight of a compression spring, subject to constraints of minimum deflection, shear stress, surge frequency, and limits on outside diameter and on design variables. There are three design variables: the wire diameter x_1, the mean coil diameter x_2, and the number of active coils x_3. This is a simplified version of a more difficult problem. The mathematical formulation of this problem is:

$$x_1 \in \{1, \ldots, 70\}\ granularity 1$$
$$x_2 \in [0.6; 3]$$
$$x_3 \in [0.207; 0.5]\ granularity\ 0.001$$

and four constraints

$$g_1 := \frac{8C_f F_{max} x_2}{\pi x_3^3} - S \leq 0$$
$$g_2 := l_f - l_{max} \leq 0$$
$$g_3 := \sigma_p - \sigma_{pm} \leq 0$$
$$g_4 := \sigma_w - \frac{F_m ax - F_p}{K} \leq 0$$

with

$$C_f = 1 + 0.75\frac{x_3}{x_2 - x_3} + 0.615\frac{x_3}{x_2}$$
$$F_{max} = 1000$$
$$S = 189000$$
$$l_f = \frac{F_{max}}{K} + 1.05(x_1 + 2)x_3$$
$$l_{max} = 14$$
$$\sigma_p = \frac{F_p}{K}$$
$$\sigma_{pm} = 6$$
$$F_p = 300$$
$$K = 11.5 \times 10^6 \frac{x_3^4}{8x_1 x_2^3}$$
$$\sigma_w = 1.25$$

Table 4 Comparison of the results of test problems

Test function	Algorithm	SD	ME	AFE	SR
E_1	MeABC	3.37E−04	9.09E−04	53,516.6	95
	ABC	1.27E−04	8.59E−04	69,676.78	100
	GABC	5.74E−04	1.10E−03	101,719.41	76
	BSFABC	3.64E−04	9.78E−04	161,599.53	79
	MABC	1.56E−01	4.74E−01	200,032.7	0
	HJABC	1.28E−04	8.53E−04	60,196.31	90
	OBLFABC	1.03E−04	9.08E−04	20,101.4	100
	SFLSDE	4.27E−03	1.24E−03	128,146.36	68
E_2	MeABC	2.43E+00	6.24E−01	131,494.58	79
	ABC	5.38E+00	5.80E+00	198,284.91	1
	GABC	4.96E+00	3.42E+00	186,455.4	18
	BSFABC	4.96E+00	1.03E+01	200,028.93	0
	MABC	2.83E+00	2.55E+00	200,023.02	0
	HJABC	2.41E+00	1.14E+00	197,071.77	3
	OBLFABC	3.79E+00	1.62E+00	159,614.82	45
	SFLSDE	8.91E−01	8.75E+00	199,773.14	0
E_3	MeABC	2.37E−03	1.71E−03	123,440.3	62
	ABC	1.17E−02	1.36E−02	187,602.32	10
	GABC	9.50E−03	8.64E−03	189,543.56	11
	BSFABC	3.08E−03	3.02E−02	200,031.13	0
	MABC	6.59E−03	5.28E−03	181,705.01	15
	HJABC	1.53E−03	1.17E−03	109,737.22	70
	OBLFABC	4.43E−03	3.27E−03	135,098	58
	OBLFABC	4.43E−03	3.27E−03	135,098	58
	SFLSDE	3.68E−01	5.36E−02	24,538.12	93
E_4	MeABC	4.43E−03	9.51E−02	26966.28	100
	ABC	8.75E−02	2.52E−01	200,017.84	1
	GABC	9.22E−03	9.91E−02	116,903.66	68
	BSFABC	5.12E−03	9.46E−02	53,885.62	98
	MABC	4.91E−03	9.36E−02	32,049.47	100
	HJABC	5.64E−03	9.34E−02	20,297.88	100
	OBLFABC	1.90E−02	1.03E−01	96,331.12	80
	SFLSDE	4.76E−03	9.36E−02	2,970	100

and the function to be minimized is

$$E_3(\mathbf{X}) = \pi^2 \frac{x_2 x_3^2 (x_1 + 2)}{4}$$

The best known solution is (7, 1.386599591, 0.292), which gives the fitness value $f^* = 2.6254$. Acceptable error for this problem is $1.0E{-}04$.

6.4 Welded beam design optimization problem

The problem is to design a welded beam for minimum cost, subject to some constraints (Ragsdell and Phillips 1976; Mahdavi et al. 2007). The objective is to find the minimum fabricating cost of the welded beam subject to constraints

Table 5 Summary of Table 4 outcome

Function	MeABC Vs ABC	MeABC Vs GABC	MeABC Vs BSFABC	MeABC Vs MABC	MeABC Vs HJABC	MeABC Vs OBLFABC	MeABC Vs SFLSDE
E_1	−	+	+	+	+	−	+
E_2	+	+	+	+	+	+	+
E_3	+	+	+	+	−	+	−
E_4	+	+	+	+	−	+	−

on shear stress τ, bending stress σ, buckling load P_c, end deflection δ, and side constraint. There are four design variables: x_1, x_2, x_3 and x_4. The mathematical formulation of the objective function is described as follows:

$$E_4(\mathbf{x}) = 1.10471x_1^2x_2 + 0.04811x_3x_4(14.0 + x_2)$$

subject to:

$$g_1(\mathbf{x}) = \tau(\mathbf{x}) - \tau_{max} \le 0$$
$$g_2(\mathbf{x}) = \sigma(\mathbf{x}) - \sigma_{max} \le 0$$
$$g_3(\mathbf{x}) = x_1 - x_4 \le 0$$
$$g_4(\mathbf{x}) = \delta(\mathbf{x}) - \delta_{max} \le 0$$
$$g_5(\vec{x}) = P - P_c(\mathbf{x}) \le 0$$
$$0.125 \le x_1 \le 5, 0.1 \le x_2, x_3 \le 10 \ and \ 0.1 \le x_4 \le 5$$

where

$$\tau(\mathbf{x}) = \sqrt{\tau'^2 - \tau'\tau''\frac{x_2}{R} + \tau''^2},$$
$$\tau' = \frac{P}{\sqrt{2}x_1x_2}, \tau'' = \frac{MR}{J}, M = P(L + \frac{x_2}{2}),$$
$$R = \sqrt{\frac{x_2{}^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2},$$
$$J = 2/\left(\sqrt{2}x_1x_2\left[\frac{x_2{}^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2\right]\right),$$
$$\sigma(\mathbf{x}) = \frac{6PL}{x_4x_3{}^2}, \delta(\mathbf{x}) = \frac{6PL^3}{Ex_4x_3{}^2},$$
$$P_c(\mathbf{x}) = \frac{4.013Ex_3x_4{}^3}{6L^2}\left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right),$$
$$P = 6{,}000\,lb, L = 14\,in., \delta_{max} = 0.25\,in.,$$
$$\sigma_{max} = 30{,}000\,psi,$$
$$\tau_{max} = 13{,}600\,psi, E = 30 \times 10^6\,psi, G = 12 \times 10^6\,psi.$$

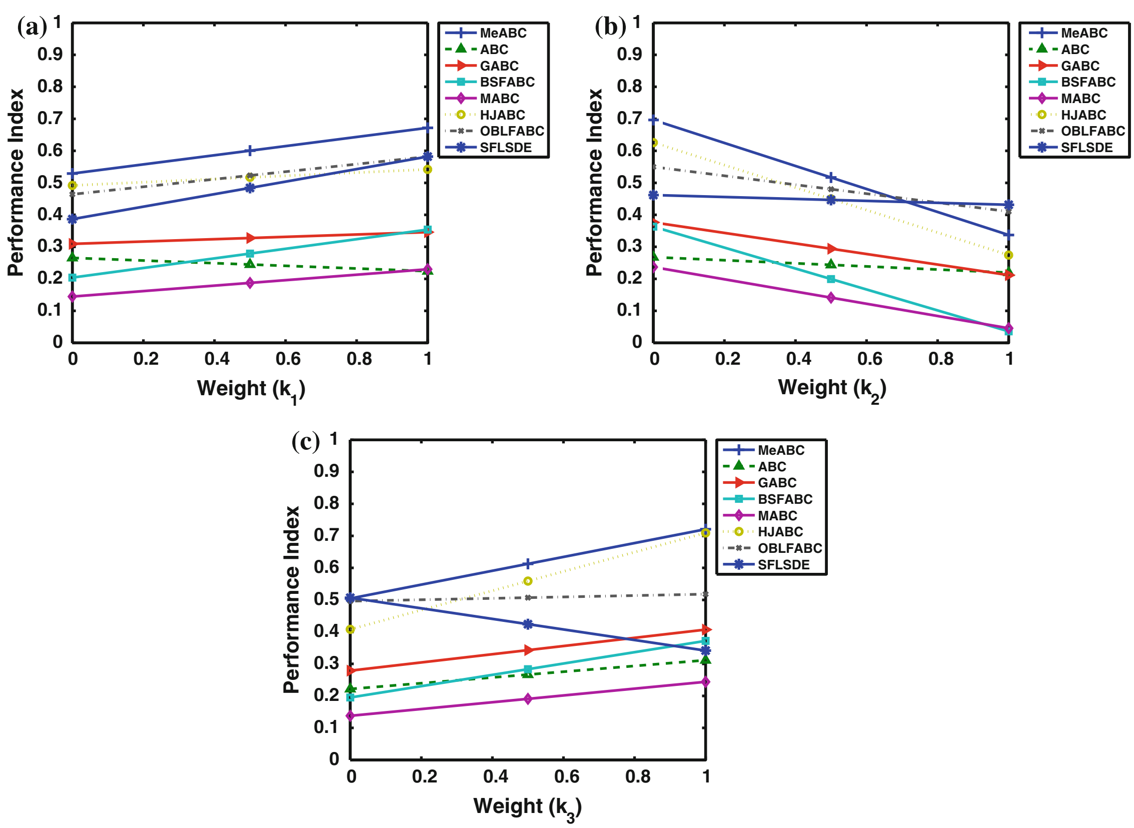

Fig. 8 Performance index for engineering optimization problems; **a** for case (1), **b** for case (2) and **c** for case (3)

The best known solution is (0.205730, 3.470489, 9.036624, 0.205729), which gives the function value 1.724852. Acceptable error for this problem is 1.0E−01.

6.5 Experimental results

To solve the constraint optimization problems (E_1 and E_4), a penalty function approach is used in the experiments. In this approach the search is modified by converting the original problem into an unconstrained optimization problem by adding a penalty term in case of constraints violation as shown below:

$$f(x) = f(x) + \beta$$

where, $f(x)$ is the original function value and β is the penalty term which is set to 10^3.

Table 4 shows the experimental results of the considered algorithms on the engineering optimization problems. It is clear from Table 4 that the inclusion of memetic strategy in the basic *ABC* performs better than the considered algorithms.

Further, the algorithms are compared through *SR*, *ME* and *AFE*. On the basis of results shown in Table 4, the results of comparison are given in Table 5. It is clear from Table 5 that the *MeABC* performs better than the considered algorithms for the considered engineering optimization problems.

The algorithms are also compared on the basis of performance indices (*PI*). The *PI* are calculated same as described in Sect. 5.3 and the results for each case are shown in Fig. 8. It is observed from Fig. 8 that the inclusion of the proposed approach enhance the performance of the basic *ABC* significantly.

7 Conclusion

In this paper, a new phase, namely, memetic search phase is introduced in ABC. The so obtained modified ABC is named as Memetic search in *ABC*. In memetic search phase, the ABC algorithm also works as a local search algorithm in which Golden Section Search algorithm is used to fine tune the control parameter ϕ. In the memetic search phase new solutions are generated in the neighbourhood of the best solution depending upon a newly introduced parameter, perturbation rate. With the help of experiments over test problems and well known engineering optimization applications, it is shown that the inclusion of the proposed strategy in the basic *ABC*, improves the reliability, efficiency and accuracy as compare to their original versions.

References

Akay B, Karaboga D (2010) A modified artificial bee colony algorithm for real-parameter optimization. Inf Sci. doi: 10.1016/j.ins.2010.07.015

Ali MM, Khompatraporn C, Zabinsky ZB (2005) A numerical evaluation of several stochastic algorithms on selected continuous global optimization test problems. J Global Optim 31(4): 635–672

Banharnsakun A., Achalakul T, Sirinaovakul B (2011) The best-so-far selection in artificial bee colony algorithm. Appl Soft Comput 11(2):2888–2901

Beyer HG, Schwefel HP (2002) Evolution strategies—a comprehensive introduction. Nat comput Springer 1(1):3–52

Brest J, Zumer V, Maucec MS (2006) Self-adaptive differential evolution algorithm in constrained real-parameter optimization. In: IEEE Congress on Evolutionary Computation 2006. CEC 2006. IEEE, pp 215–222

Caponio A, Cascella GL, Neri F, Salvatore N, Sumner M (2007) A fast adaptive memetic algorithm for online and offline control design of pmsm drives. Syst Man Cybernet Part B: Cybernet IEEE Trans 37(1):28–41

Caponio A, Neri F, Tirronen V (2009) Super-fit control adaptation in memetic differential evolution frameworks. Soft Comput-A Fusion Found, Methodol Appl 13(8):811–831

Chen X, Ong YS, Lim MH, Tan KC (2011) A multi-facet survey on memetic computation. IEEE Trans Evol Comput 15(5):591–607

Clerc M (2012) List based pso for real problems. http://clerc.maurice.free.fr/pso/ListBasedPSO/ListBasedPSO28PSOsite29.pdf, 16 July 2012

Cotta C, Neri F (2012) Memetic algorithms in continuous optimization. Handbook of Memetic Algorithms, pp 121–134

Das S, Suganthan PN (2010) Problem definitions and evaluation criteria for CEC 2011 competition on testing evolutionary algorithms on real world optimization problems. Jadavpur University, Kolkata, India, and Nangyang Technological University, Singapore, Tech. Rep, 2010

Dasgupta D (2006) Advances in artificial immune systems. Comput Intell Mag IEEE 1(4):40–49

Diwold K, Aderhold A, Scheidler A, Middendorf M (2011) Performance evaluation of artificial bee colony optimization and new selection schemes. Memet Comput 3(3):149–162

Dorigo M, Di Caro G (1999) Ant colony optimization: a new meta-heuristic. In: Evolutionary Computation, 1999. CEC 99. Proceedings of the 1999 Congress on, vol 2. IEEE

Eiben AE, Smith JE (2003) Introduction to evolutionary computing. Springer, Belin

El-Abd M (2011) Performance assessment of foraging algorithms vs. evolutionary algorithms. Inf Sci 182(1):243–263

Fister I, Fister Jr I, Brest J, Zumer V (2012) Memetic artificial bee colony algorithm for large-scale global optimization. Arxiv preprint arXiv:1206.1074

Fogel DB, Michalewicz Z (1997) Handbook of evolutionary computation. Taylor & Francis, London

Gallo C, Carballido J, Ponzoni I (2009) Bihea: a hybrid evolutionary approach for microarray biclustering. In: Advances in Bioinformatics and Computational Biology, LNCS, vol 5676. Springer, Heidelberg, pp 36–47

Goh CK, Ong YS, Tan KC (2009) Multi-objective memetic algorithms, vol. 171. Springer, Berlin

Goldberg DE (1989) Genetic algorithms in search, optimization, and machine learning. Addison-Wesley, Reading, MA

Hooke R, Jeeves TA (1961) "Direct search" solution of numerical and statistical problems. J ACM (JACM) 8(2):212–229
Hoos, HH Stützle T (2005) Stochastic local search: Foundations and applications. Morgan Kaufmann
Iacca G, Neri F, Mininno E, Ong YS, Lim MH (2012) Ockham's razor in memetic computing: three stage optimal memetic exploration. Inf Sci: Int J 188:17–43
Ishibuchi H, Yamamoto T (2004) Fuzzy rule selection by multi-objective genetic local search algorithms and rule evaluation measures in data mining. Fuzzy Sets Syst 141(1):59–88
Ishibuchi H, Yoshida T, Murata T (2003) Balance between genetic search and local search in memetic algorithms for multiobjective permutation flowshop scheduling. IEEE Trans Evol Comput 7(2):204–223
Kang F, Li J, Ma Z (2011) Rosenbrock artificial bee colony algorithm for accurate global optimization of numerical functions. Inf Sci 181(16):3508–3531
Kang F, Li J, Ma Z, Li H (2011) Artificial bee colony algorithm with local search for numerical optimization. J Softw 6(3):490–497
Karaboga D (2005) An idea based on honey bee swarm for numerical optimization. Technical Report. TR06, Erciyes University Press, Erciyes
Karaboga D, Akay B (2009) A comparative study of artificial bee colony algorithm. Appl Math Comput 214(1):108–132
Karaboga D, Akay B (2010) A modified artificial bee colony (abc) algorithm for constrained optimization problems. Appl Soft Comput
Kennedy J (2006) Swarm intelligence. Handbook of Nature-Inspired and Innovative Computing, pp 187–219
Kennedy J, Eberhart R (1995) Particle swarm optimization. In: Neural Networks, 1995. Proceedings, IEEE International Conference on, vol. 4. IEEE, pp 1942–1948
Kiefer J (1953) Sequential minimax search for a maximum. In: Proceedings of American Mathematical Society, vol. 4, pp 502–506
Knowles J, Corne D, Deb K (2008) Multiobjective problem solving from nature: From concepts to applications (Natural computing series). Springer, Berlin
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, Velez-Koeppel RE (2010) Elitist artificial bee colony for constrained real-parameter optimization. In 2010 Congress on Evolutionary Computation (CEC2010), IEEE Service Center, Barcelona, Spain, pp 2068–2075
Mininno E, Neri F (2010) A memetic differential evolution approach in noisy optimization. Memet Comput 2(2):111–135
Moscato P (1989) On evolution, search, optimization, genetic algorithms and martial arts: towards memetic algorithms. Caltech concurrent computation program, C3P Report, 826:1989
Neri F, Cotta C, Moscato P (2012) Handbook of memetic algorithms, vol. 379. Springer, Berlin
Neri F, Iacca G, Mininno E (2011) Disturbed exploitation compact differential evolution for limited memory optimization problems. Inf Sci 181(12):2469–2487
Neri F, Tirronen V (2009) Scale factor local search in differential evolution. Memet Comput Springer 1(2):153–171
Nguyen QH, Ong YS, Lim MH (2009) A probabilistic memetic framework. IEEE Trans Evol Comput 13(3):604–623
Oh S, Hori Y (2006) Development of golden section search driven particle swarm optimization and its application. In SICE-ICASE, 2006. International Joint Conference. IEEE, pp 2868–2873
Ong YS, Keane A.J (2004) Meta-lamarckian learning in memetic algorithms. IEEE Trans Evol Comput 8(2):99–110
Ong YS, Lim M, Chen X (2010) Memetic computationpast, present and future [research frontier]. Comput Intell Mag IEEE 5(2):24–31
Ong YS, Lim MH, Zhu N, Wong KW (2006) Classification of adaptive memetic algorithms: a comparative study. Syst Man Cybernet, Part B: Cybernet, IEEE Trans 36(1):141–152
Ong YS, Nair PB, Keane A.J (2003) Evolutionary optimization of computationally expensive problems via surrogate modeling. AIAA J 41(4):687–696
Onwubolu GC, Babu BV (2004) New optimization techniques in engineering. Springer, Berlin
Passino KM (2002) Biomimicry of bacterial foraging for distributed optimization and control. Control Syst Mag IEEE 22(3):52–67
Price KV, Storn RM, Lampinen JA (2005) Differential evolution: a practical approach to global optimization. Springer, Berlin
Ragsdell KM, Phillips DT (1976) Optimal design of a class of welded structures using geometric programming. ASME J Eng Ind 98(3):1021–1025
Rao SS, Rao SS (2009) Engineering optimization: theory and practice. Wiley, New York
Repoussis PP, Tarantilis CD, Ioannou G (2009) Arc-guided evolutionary algorithm for the vehicle routing problem with time windows. Evol Comput IEEE Trans 13(3):624–647
Richer JM, Goëffon A, Hao JK (2009) A memetic algorithm for phylogenetic reconstruction with maximum parsimony. Evolutionary Computation, Machine Learning and Data Mining in Bioinformatics, pp 164–175
Ruiz-Torrubiano R, Suárez A (2010) Hybrid approaches and dimensionality reduction for portfolio selection with cardinality constraints. Comput Intell Mag IEEE 5(2):92–107
Sandgren E (1990) Nonlinear integer and discrete programming in mechanical design optimization. J Mech Des 112:223
Sharma H, Chand Bansal J, Arya KV (2012) Opposition based lTvy flight artificial bee colony. Memet Comput. doi:10.1007/s12293-012-0104-0, December (2012)
Suganthan PN, Hansen N, Liang JJ, Deb K, Chen YP, Auger A, Tiwari S (2005) Problem definitions and evaluation criteria for the CEC 2005 special session on real-parameter optimization. In CEC 2005
Susan J (1999) The meme machine. Oxford University Press, Oxford
Tan KC (2005) Eik fun khor, tong heng lee, multiobjective evolutionary algorithms and applications (advanced information and knowledge processing)
Tang K, Mei Y, Yao X (2009) Memetic algorithm with extended neighborhood search for capacitated arc routing problems. IEEE Trans Evol Comput 13(5):1151–1166
Thakur Deep M.K. (2007) A new crossover operator for real coded genetic algorithms. Appl Math Comput 188(1):895–911
Valenzuela J, Smith AE (2002) A seeded memetic algorithm for large unit commitment problems. J Heuristics 8(2):173–195
Vesterstrom J, Thomsen RA (2004) comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems. In: Evolutionary Computation, 2004. CEC2004. Congress on, vol. 2. IEEE, pp 1980–1987
Wang H, Wang D, Yang S (2009) A memetic algorithm with adaptive hill climbing strategy for dynamic optimization problems. Soft Comput-A Fusion Found Methodol Appl 13(8):763–780
Williamson DF, Parker RA, Kendrick JS (1989) The box plot: a simple visual method to interpret data. Ann Intern Med 110(11):916
Yang XS (2011) Nature-inspired metaheuristic algorithms. Luniver Press, UK
Zhu G, Kwong S (2010) Gbest-guided artificial bee colony algorithm for numerical function optimization. Appl Math Comput 217(7):3166–3173