JOURNAL OF INFORMATION SCIENCE AND ENGINEERING 34, 353-369 (2018)
DOI: 10.6688/JISE.201803_34(2).0003

Multi-Objective Optimization using Co-Variance Guided Artificial Bee Colony

DIVYA KUMAR AND K. K. MISHRA
Department of Computer Science and Engineering
Motilal Nehru National Institute of Technology Allahabad
Allahabad, 211004 India
E-mail: {divyak; kkm}@mnnit.ac.in

Use of statistics and co-variance principles in the domains of soft computing is an emerging and intriguing research area. Co-variance based optimization algorithms have faster convergence rate and remarkable efficiency as they possess fair information about the gradient of objective function. Artificial Bee Colony (ABC) is one of the highly researched and utilized optimization technique, based on the foraging behavior of honey bee swarms. In this article, we have proposed a novel ABC based algorithm for multi-objective optimization. The proposed Multi-objective Co-variance based ABC (M-CABC), is the first algorithm in its class which works on the coalition of statistical co-variance and ABC. The performance of the proposed algorithm has been gauged on both unconstrained and constrained multi-objective benchmark functions on the basis of error rate, generational distance and spacing metrics. The results have shown that M-CABC has eminently converged close to optimal pareto front while successfully maintaining high diversity in the solutions. The article concludes with the observatory remarks on the performance of M-CABC, which is consistent in solving low as well as high dimension multi-objective benchmark problems when it has been compared with other traditionally established multi-objective optimization algorithms.

Keywords: meta-heuristics, swarm intelligence, artificial bee colony, C-variance, multi-objective optimization, non-dominated sorting

1. INTRODUCTION

Nature inspired algorithms has gained a tremendous success in solving single objective optimization problems. This success had provided an impetus for the use of these algorithms for Multi-objective optimization as well. In the last decade a large number of nature inspired algorithms have been proposed and are being deployed for solving multi-objective problems not only related to computer science engineering but for ecology, biology and medical treatment also [1]. Multi-objective problems are usually accompanied with few constraints over the decision variables. Introduction of the constraints in multi-objective optimization, though pulls the problem into more realistic and real-world scenario, it however perplexes the problem many-fold. Notwithstanding with assorted developments in the field of nature inspired multi-objective optimizers, we still lack unified algorithms which works well for constrained and un-constrained multi-objective problems simultaneously [2, 3].

Nature inspired algorithms pertains to a class of algorithms whose search strategies model the natural biological evolution trends. This phenomenon includes natural selection, reproduction, genetic inheritance and the Darwinian strife for survival. These algo-

Received May 4, 2016; revised July 9, 2016; accepted September 16, 2016.
Communicated by Tzung-Pei Hong.

rithms act upon a set of candidate solutions in parallel by interpreting the similarities among them to change their values. Nature inspired algorithms are robust and most comported choice for Multi-objective Optimization (MOO). They are preferred over other mathematical alternatives because of the following reasons: (i) they never fail even if Pareto optimal front is concave or discontinuous; (ii) they result in several trade-off solutions instead of one, that too in every iteration; (iii) they can efficiently deal with the constraints over objective function (iv) although gradient information of the objective function enormously assist in convergence, nature inspired algorithms do not require this gradient information also.

Though the concern for evolutionary and nature inspired multi-objective optimization has been growing day by day but Swarm Intelligence (SI) based algorithms for MOO are still attenuated. SI is a sub-class of nature inspired algorithms; it can be succinctly defined as the combined and cooperative behavior of stochastic, dispersed and self-organized swarms. Ant colonies, bird flocks, beehives, fish schools, glowworm luminescence, bats echolocation and self-propelled particle, are some of the well known examples existing in the nature which have inherent optimization characteristics. These algorithms are inspired by the cooperative and reconciling behaviour of stochastic but self-organizing swarms. The three main SI approaches are: Ant Colony Optimization (ACO), inspired by ant colony, proposed by Dorigo *et al.* in [4]; Particle Swarm Optimization (PSO), inspired by bird flocking, proposed by Kennedy and Eberhart in [5]; Artificial Bee Colony (ABC), motivated by natural honey bees is proposed by Dervis Karaboga in [6].

The Artificial Bee Colony (ABC) algorithm is a stochastic optimization technique which is a collective arrangement of three modules namely: employed bee, onlooker bee and scout bee. ABC is based on intelligent foraging behaviour of natural honeybee swarms. The basic structure of Artificial Bee Colony (ABC) algorithm, is described in algorithm 1. The qualitative as well as quantitative assessment of ABC algorithm was done by Dervis Karaboga *et al.* in [7, 8]. The performance was compared against the latest state of the art evolutionary algorithms which include Genetic Programming (GP), Evolution Strategy (ES), Evolutionary Programming (EP), Differential Evolution (DE) and Particle Swarm Optimization (PSO). From the simulation results it can be ascertained that for benchmark multi-dimensional as well as for benchmark multi-modal numeric problems, the performance of ABC algorithm is fairly better or at least comparable to those of the mentioned algorithms with an added advantage of employing fewer control parameters. It was also concluded that ABC can be efficiently employed to solve engineering problems with high dimensionality.

A good number of ABC variants have also been proposed in the past for solving real life multi-objective optimization problems. Khorsandi *et al.* in [9] and Adaryani *et al.* in [10] have deployed multi-objective ABC algorithm for solving optimal power flow problem. Li *et al.* in [11] have proposed a discrete ABC algorithm for solving multi-objective job-shop scheduling problem involving maintenance activities. Yahya *et al.* in [12] have used a multi-objective version of ABC algorithm for layout planning of a construction land site. Hancer *et al.* in [13] proposed a multi-objective ABC algorithm for feature selection problem which involves two conflicting objectives: Minimizing the number of selected features and maximizing the classification accuracy. Basu in [14] has used the ABC algorithm for solving multi-area economic dispatch problem with various tie line

constraints. Author has also concluded that ABC based approach is more promising than other nature inspired algorithms. Naidu *et al.* in [15] have used a weighted sum ABC based multi-objective optimizer for load frequency control in a two point connected re-heat thermal power arrangement.

Though we earlier stated that the use of gradient specific information about objective function is never required by any evolutionary or swarm intelligence algorithm but gradient information when coupled with meta-heuristics always yield fruitful results in terms of faster convergence and precise results because gradient information helps us in locating minima and maxima efficiently [16]. In the past, co-variance matrices were used in the population based approaches to approximate the gradient information [17, 19]. In these approaches, new points were generated in the search space in accordance with the co-variance matrices and evolutionary operators.

This manuscript is dedicated to conjoin statistical co-variance principles with Artificial Bee Colony meta-heuristic to form an ideal multi-objective optimizer which works equivalently well for constrained and unconstrained multi-objective problems. In this pursuit the rest of the article is organized as follows: firstly we have detailed the fundamental aspects of multi-objective optimization and statistical co-variance principles. Subsequently we have presented our novel Multi-objective Co-variance guided Artificial Bee Colony (M-CABC) algorithm in detail. Then we have discussed three metrics that have been extensively used in the MOO literature to assert the performance of any multi-objective optimizer. These metrics are error rate, generational distance and spacing. The article concludes with results and discussions over the adept performance of M-CABC on various constrained and unconstrained benchmark functions.

Algorithm 1: ABC General Scheme

1: Initialization phase.
2: Repeat step 2.1 through 2.4 until (termination condition)
  2.1: Employed bee phase. *//search new points in whole search space.*
  2.2: Onlooker bee phase. *//according to the information shared by employed bees,*
    //exploit food sources containing high nectar amount.
  2.3: Scout bee phase. *//search new points instead of those which cannot be further evolved.*
  2.4: Memorize the best solution achieved so far.
3: Return the memorized best solution.

2. FUNDAMENTALS OF MULTI-OBJECTIVE OPTIMIZATION

When a problem involves simultaneous optimization of multiple conflicting objectives, it is known as a Multi-Objective Optimization Problem (MOOP) [20, 21]. A MOOP can be defined and formulated as follows: Given an n-dimensional vector of decision variables $\vec{X} = \{x_1, x_2, ..., x_n\}$, in solution space ξ, we have to ascertain a particular vector $\vec{X}^*$, or a set of trade-off vectors that minimizes/maximizes a given set of k objective functions:

$$F(\vec{X}) = \{f_1(\vec{X}), f_2(\vec{X}), \ldots, f_k(\vec{X})\}$$

where f_i is ith objective function and f_i: $\mathbb{R}^n \rightarrow \mathbb{R}$. The solution space ξ is usually restricted by constraints bounds on decision variables. These constraints are of the form $g(\vec{X}) \leq 0$ or $h(\vec{X}) = 0$. An another function F: $\vec{X} \rightarrow \vec{Y}$ is used to measure the fitness of any specific solution by converting it to an objective vector $\vec{Y} = \{y_1, y_2, ..., y_k\}$ in the objective space Ψ. The mapping takes place between an n-dimensional solution vector in solution space and a k-dimensional objective vector in objective vector space. For multi-criteria decision making problems, objectives are generally conflicting with trade-offs among them *i.e.* optimizing a solution with respect to a single objective can result in unacceptable results (or depreciated results) with respect to another objectives. As a consequence of this characteristic it becomes highly desirable to have multiple trade-off solutions, with respect to multiple conflicting objectives. This gives the power to choose the most suitable solution according to the other external conditions.

2.1 The Pareto Optimality

The solution of multi-objective optimization problems is contained in a set of trade-off solutions which is also known as Pareto optimal set. This set contains all non dominated solutions and it exemplifies the best possible trade-offs among the best solutions of various objectives. A feasible solution $\vec{X}^* \in \xi$ can be stated as Pareto optimal if there exists no other solution $\vec{X} \in \xi$ which dominates $\vec{X}^*$ [22]. Any solution $\vec{S} = \{s_1, s_2 \ .., s_n\}$ dominates a solution $\vec{Z} = \{z_1, z_2, ..., z_n\}$, or $\vec{S}$ has a better non-dominated rank r_s, such that $r_s \prec r_z$ where r_z and r_s are the rank of solution $\vec{Z}$ and $\vec{S}$ respectively, in a minimization context, iff both conditions 1 and 2 are true:

Condition 1: The solution $f(\vec{S})$ is as good as $f(\vec{Z})$ for all objectives:

$$\forall j \in [1 \ldots k] f_j(\vec{S}) \leq f_j(\vec{Z}) \tag{1}$$

Condition 2: The solution $f(\vec{S})$ is strictly better than $f(\vec{Z})$ for at least one objective:

$$\exists j \in [1 \ldots k] f_j(\vec{S}) < f_j(\vec{Z}) \tag{2}$$

2.2 Evolutionary and Swarm Based Multi-Objective Optimizers

The two classical, most cited and researched algorithms for multi-objective optimization are SPEA2 [23] and NSGA-2 [24]. In SPEA-2 the ranking of candidate solutions is done on the basis of the number of dominating and dominated solutions. First, it calculates the strength S of every individual based on the number of solutions it dominates then it calculates the rawfitness, R, of every individual based on the strength, S, of solutions dominating it. Secondly it uses a density function, D, to maintain the diversity in the population. D value of any individual is based on its distance from its kth nearest neighbor. The R and D, values finally define the fitness of individual. NSGA-2 by Srinivas and Deb ranks an element on the basis of non-dominated sorting and uses a crowding distance operator to maintain the diversity in the population. Both SPEA-2 and NSGA-2 are elitist algorithms. Although these algorithms are computationally expensive, they are well tested on numerous multi-objective benchmark problems [25].

Swarm based multi-objective optimizers have also been proposed in the past. These

algorithms are well studied and used in academia as well as in industry. Three of the most cited and researched swarm based algorithms are: Multi-objective Artificial Bee Colony (MOABC) [26], proposed by Akbari *et al.* is a multi-objective optimizer grounded on the operators of ABC. It uses the concept of pareto dominance to ascertain the behavior of employed and onlooker bees dealing with multiple objectives in the problems. This algorithm maintains an extraneous archive of non-dominated solution vectors and overlay a virtual grid on them to maintain the diversity. The solutions in the grids decide the flying trajectories of the bees dynamically. MOABC shows a good combination of exploitation and exploration. It has been thoroughly tested on different benchmarks and produced extremely competitive results.

Robic *et al.* in [27] proposed an extended version of DE for multi-objective optimization with three different approaches. They named it Differential Evolution (DE) for Multi-objective Optimization (DEMO). In their proposed work they created the candidate solution *C* according to DE operators and binary crossover. The solution *C* was later compared with its parent *P*. If *C* dominates *P*, *C* replaces *P*. If *P* dominates *C*, *C* is discarded. Otherwise if both are equal then *C* is added to the population. Population pool is truncated if it exceeds the size. In their second approach, *C* is compared with its nearest neighbour in the population, following the same replacement strategy. In their third approach *C* is treated with the nearest neighbor in the decision space, again with the same replacement strategy. The authors evaluated the performance of DEMO on various *ZDT* benchmarks and achieved good convergence and diversity values.

Coello *et al.* in [28] proposed, Multiple Objective Particle Swarm Optimization (MOPSO), a variant of PSO algorithm for multi-objective optimization problems. In this variant, the concept of *Global Best* location was replaced by the group of non-dominated leader particles. Leader particles are maintained in an external archive of fixed size, which is different from the other particles. In each iteration, the velocity of every particle depends on its *Particle_Best* location and its distance from a dynamically chosen leader. The chosen leader is the one which is most distant and diverse in the population hypercubes. *Particle_Best* is updated if new location is not dominated by the older one. After every iteration, the leaders are also updated with elitism. The results of proposal show that MOPSO is one of the best algorithms for multi-objective optimization.

3. USAGE OF GRADIENT IN DETERMINISTIC OPTIMIZATION

Deterministic optimization algorithm starts with an initial guess x_0 of the decision variables and iterates to produce a sequence of improved estimates $x_1, x_2,\ldots, x_k$ using the first order derivatives of the objective function and the search direction p to proceed in a promising direction which should yield $f(x_k) < f(x_k - 1)$ [29]. The deterministic optimization functions can have a high convergence rate if we start with right approximations of the roots and the objective function is smooth enough [30]. The theme of deterministic optimization is showcased in Eq. (4) as Taylor expansion [31]. In this expansion any function f is expressed as the sum of infinite series of its derivatives. Here the objective is to calculate $f(x + p)$ at a point $(x + p)$ near to x, successively and then reach the root x^* such that $f(x^*) = 0$.

$$f(x+p) = f(x) + \nabla f(x)^T p + \tfrac{1}{2!} p^T \nabla^2 f(x) p + \ldots \quad (4)$$

The roots of any objective function can be approximated using the formula in Eq. (5) which is Newton-Raphson Method [32], deduced from Taylor expansion. Here $\nabla f(x_k)$ is the gradient of $f(x)$ at point x_k. If we wish to minimize any function, we should make $\nabla f(x_k) = 0$, so our goal is to find out the root of $\nabla f(x_k)$. In this case the value of x_{k+1} can be determined according to Eq. (6) and search direction p_k of Eq. (4) can be written as shown in Eq. (7).

$$x_{k+1} = x_k - \frac{f(x_k)}{\nabla f(x_k)} \quad (5)$$

$$x_{k+1} = x_k - \frac{\nabla f(x_k)}{\nabla f(x_k)} \quad (6)$$

$$p_k = x_{k+1} - x_k = \nabla^2 f(x_k)^{-1} \cdot \nabla f(x_k) \quad (7)$$

$\nabla^2 f(x)$ is an approximation of symmetric hessian matrix H [33], which can be competently written as $H_{i,j} = \frac{\partial^2 f}{\partial x_i \partial x_j}$. Thus all the deterministic optimization methods expect gradient (the first order derivatives) and hessian (the second order derivative) in advance to find a new location x_{k+1} to converge to the local/global optima.

4. GRADIENT APPROXIMATION IN META-HEURISTICS

The objective functions of the problems that are solved using meta-heuristic algorithms are hard enough to be differentiated. Hence finding gradient is a tough task. We may however use the statistical co-variance principles, as an alternative way, to approximate the gradient or hessian [16, 17]. For an n-dimensional objective function problem, a co-variance matrix C of size $n \times n$ is an approximation of search direction p, used in Eq. (7), in a population of size N. The co-variance matrix for a population of N members, each of which is of n dimension, can be formed using the Eq. (8).

$$C_{i,j} = \frac{1}{N-1} \sum_{k=1}^{N} (x_{ik} - \bar{x}_l)(x_{jk} - \bar{x}_j) \quad \forall i, j \in [1, \ldots, n] \quad (8)$$

5. THE MULTI-OBJECTIVE CABC (M-CABC)

Every multi-objective evolutionary algorithm comprises of two fundamental phases, one is to find the non-dominated rank of a solution over another and second is to produce child population using the parent population. In M-CABC, we have used the basic NSGA-2 algorithm [24] for ranking purposes. Once the ranking procedure for any multi-objective optimizer is determined, the only task that remains is to frame a procedure for selecting parents and generating offspring population from selected parent pool. The original NSGA-2 algorithm uses tournament selection and genetic operators for parent selection and child generation respectively. In our algorithm we have used co-variance guided M-CABC operators for selection as well as for the generation of offspring popu-

lation. Adopting the basic iterative structure of ABC as detailed in Algorithm 1, the M-CABC phases are explained in following sub-sections:

5.1 Initialization Phase

The algorithm starts with the initialization phase. In this phase the variables are assigned their initial values. One important variable is the number of bees acting in the system. As described in the traditional ABC algorithm [6], we have kept equal number of employed and onlookers bees in our algorithm. From various experiments we have found that optimum number of employed or onlooker bees is 5 times the number of dimensions n in the problem. A set of bees lesser than $5n$ bees have poor convergence and more than $5n$ bees results in premature convergence with more number of fitness evaluations. Hence for an n dimensional problem there occurs $5n$ employed bees and $5n$ onlooker bees. In the initialization phase $5n$ employed bees are formed randomly using the Eq. (9). ub_i and lb_i are the respective lower bound and upper bound on ith dimension. The random bees are then organized in k Pareto fronts $\{F_1, F_2, \ldots, F_k\}$ using the NSGA-2 algorithm [24] and co-variance matrix C is initialized as identity matrix.

$$x_i^m = lb_i + rand(0,1) * (ub_i - lb_i), \quad \forall i \in [1,...,n] \quad \forall m \in [1,...,5n] \tag{9}$$

5.2 Employed Bee Phase

The employed bee phase of ABC algorithm stands for exploration of the search space. In this phase a new temporary bee t is formed for each mth employed bee x^m by mutating any dimension d using the Eq. (10). x^r is any random neighbor of x^m and x^{best} is any bee taken from front F_1. $z \in (0, 1)$ appears to another control parameter. In the early iterations z is kept high so as to have more randomness and to avoid premature convergence whereas in the higher iterations the value of z is kept low for smooth and faster convergence. A greedy selection, as described in sub-section 2.1, is applied between x and t to form x of next iteration.

$$t_d^m = x_d^m + z(x_d^m - x_d^r) + (1-z)(x_d^m - x_d^{best}) \tag{10}$$

5.3 Onlooker Bee Phase

Onlooker bee phase represents exploitation of good solutions. In the onlooker bee phase of traditional ABC [6], more number of onlooker bees are attracted towards better solutions. An analogous strategy is followed in the proposed algorithm. Here we are sending more number of onlookers to the better front. To distribute $5n$ available onlooker bees among k fronts $\{F_1, F_2, \ldots, F_i, \ldots, F_k\}$, $\left\lceil \frac{5n}{2^i} \right\rceil$ number of onlooker bees are send to ith front. If $5n > 2^k$ then remaining bees are again directed from front F_1. If $5n < 2k$ then uncovered fronts are ignored. Once the onlooker bees are distributed on the fronts, they exploit the fronts using the Eq. (11). Here z is the control parameter as described in the employed bee phase. M is the mean solution of front F_i. $B_{n\times n}$ and $D_{n\times n}$ are the eigen decomposition of co-variance matrix C, which is formed according to Eq. (8), using the members of front F_i. $r_{n\times 1}$ is any random vector. After the creation of onlooker bees, the next step is to create the bee hive according to the Eq. (12). NSGA-2 algorithm [24] is

again applied over the hive (now of size 10*n*) to select best 5*n* employed bees for the next iteration.

$$X = M + zBDr \tag{11}$$

$$hive = \{employed\ bees \cup onlooker\ bees\} \tag{12}$$

5.4 Scout Bee Phase

Scout bee phase represents the negative feedback corresponding to the positive feedback of onlooker bees. This phase is very simple. Here, the bees whose performance is not improving since last 10*n* iterations are re-initialized using the Eq. (9). The rationale behind using 10*n* is to wait for at least 10 iterations per dimension for a better change. A value lesser than 10*n* prevents faster convergence and a value greater than 10*n* results more number of wasteful, fitness evaluations.

6. PERFORMANCE METRICS

The qualitative performance comparison of various evolutionary multi-objective algorithms is very difficult and complex. This is primarily because of three reasons: (i) we work on many solutions simultaneously, (ii) the evolutionary algorithms are stochastic in nature and many readings are required to conclude any fact about the efficiency of the algorithm [34] and (iii) the goal is to optimize various objectives, all together [35] possibly under constraints. Irrespective of these difficulties there are some basic issues, which may be taken into consideration for measuring the performance of multi-objective optimizers. These issues are:

(A) Number of elements in the pareto optimal set.

The algorithm under examination should results in maximum number of non-dominated solutions. Error Ratio or *ER*, proposed by Van in [36] tackles this issue. *ER* represents the percentage of the solutions, produced by multi-objective optimizer, which are not contained in the actual pareto front. Mathematically, it is described in Eq. (13) for a population of size *N*. $e_i = 0$ if *i*th candidate solution is present in the actual pareto front otherwise $e_i = 1$. The lesser the value of *ER*, the more ideal will be the optimization algorithm.

$$ER = \frac{\sum_{i=1}^{n} e_i}{N} \tag{13}$$

(B) The distance between the obtained pareto front and actual pareto front.

The algorithm should minimize this distance measure. Generational Distance or *GD*, proposed by Van *et al.* in [37] tackles this issue. Let's suppose that F is the pareto front produced by the algorithm under test then *GD* represents the proximity level of the *F* with actual pareto front as defined in Eq. (14). *M* is the number of solutions in *F*. d_i is the Euclidian distance between *i*th member of *F* and its nearest neighbor in actual pareto front. $GD \rightarrow 0$, if the algorithm approximates the actual pareto front *i.e.* maximum mem-

bers of F are contained in actual pareto front. This value thus depicts nearness/faraway lineaments of obtained pareto front with respect to actual pareto front.

$$GD = \frac{\sqrt{\sum_{i=1}^{M}(d_i)^2}}{M} \tag{14}$$

(C) The distribution of solutions in the obtained pareto front.

The algorithm should maximize the spread of solutions on the obtained pareto front. The Spacing metric (SP) is proposed by Scott in [38] to tackle this issue. It represents the spread or distribution of solutions on obtained pareto front. In a D dimensional objective space of objective functions $\{f^1, f^2, ..., f^D\}$, dnn_i, is defined in Eq. (15), is the distance of ith solution from its nearest neighbor in the objective space. SP metric for any M membered front F is defined in Eq. (16) in which $\overline{dnn}$ is the average of all dnn_i. $SP \to 0$, if all the members on pareto front are equidistant.

$$dnn_i = \min_{\forall j \in [1...m], j \neq i} \{\sum_{k=1}^{D} | f_i^k - f_j^k |\} \tag{15}$$

$$SP = \sqrt{\frac{1}{M-1}\sum_{i=1}^{M}(\overline{dnn} - dnn_i)^2} \tag{16}$$

7. TESTING OF M-CABC ON MULTI-OBJECTIVE BENCHMARKS

To measure the effectiveness of M-CABC, we have tested it on two sets of benchmark functions (i) Unconstrained Benchmark Functions and (ii) Constrained Benchmark Functions. ZDT (Zitzler-Deb-Thiele's) Test Suite [35], containing six functions (ZDT1, ZDT2, ZDT3, ZDT4, ZDT5, ZDT6), is used as unconstrained benchmark. These functions are predominately used standard test problems, in evolutionary literature. Because of its binary encoding ZDT5 is usually an omitted test case, accordingly we have also excluded this function from our analysis.

Constrained multi-objective problems are much difficult to optimize because constraints prevent MOEA to converge towards actual pareto-front and restricts to have a discrete valued front. However the real strength of any evolutionary algorithm is justified when the algorithm is able to handle multiple constraints effectively. To deal with the constraints we have used the approach suggested by Jimenez *et al.* in [39]. According to this approach, in the binary tournament selection we compare the solutions as follows: if both the solutions are feasible then we select the non-dominated one; if one is feasible and another is infeasible, we select the feasible one; if both the solutions are infeasible, we select the one with lower constraint violations. We have used following five constraint benchmark functions in our analysis: Binh and Korn function (BINH) proposed by Binh *et al.* in [40], Constr-Ex function (CNSTR) as described in [24], SRN proposed by Srinivas *et al.* in [41], TNK, proposed by Tanaka in [42] and OSY function proposed by Osyczka *et al.* in [43].

For a concrete performance analysis, we have compared our results with the results of five well established multi-objective optimizers: (i) MOABC [26], (ii) DEMO [27], (iii) MOPSO [28], (iv) NSGA-2 [24] and (v) SPEA-2 [23]. Results show that M-CABC performs significantly better than, or at least comparable to, these well-established evolu-

tionary algorithms. Each algorithm is executed five times for every benchmark function and the best performance is selected for tabulation. For each of the test, a set of 200 solutions is evolved till 1,000 generations.

7.1 Results

The results of M-CABC, which are presented in the current sub-section validate its adept performance. Graphs in Figs. 1 and 2, reflect the convergence of M-CABC, MO-ABC, DEMO, MOPSO, NSGA-2 and SPEA-2, towards the actual pareto optimal front on un-constrained and constrained benchmark functions. From these graphs, we have following two observations: First, it is clear that all the six algorithms have converged towards the true pareto front. However M-CABC is the only one successful to reach the vicinity of the actual pareto front. ZDT4 is the only benchmark function for which the performance of our algorithm is slightly foiled. Second observation is about the spread of solutions to cover the entire pareto optimal front. From the figure it is also clear that there is no point on the pareto optimal front which is not addressed by the non-dominated solutions of M-CABC algorithm. Table 1 shows the comparison of all six algorithms on the basis of *ER* (error ratio), *GD* (generational distance) and *SP* (spacing metrics). The *ER* value is 1.0 every time for all the algorithms. This is because of the real numbered decision space. Although the ER value is not 0.0 but all the algorithms possess a very less generational distance.

The proposed algorithm has very less *GD* values because of the perfect exploitation and exploration balance of M-CABC operators. For some of the benchmark functions our algorithm has achieved an ace *GD*, which is of the order of 10^{-5} or 10^{-4}. For the benchmark functions ZDDTx, BINH and SRN, M-CABC has outperformed all the established algorithms in terms of both *GD* and *SP* metrics. For CNSTR function M-CABC has achieved the best *GD* value only. For TNK and OSY, M-CABC has achieved best *SP* value only. The algorithm has worked very well because it explores (or freely moves in) the search space according to two control criteria: (i) distance of the candidate solution from any other random candidate solution. This value helps in the global convergence of whole population simultaneously as all candidate solutions try to move according to one another; (ii) distance of a candidate solution from the best solution seen so far. This parameter provides a good movement direction to every candidate solution. And when the search space is properly explored we quickly converge and reach the locality of optimal solutions after proper exploitation.

For a proper and accurate exploitation, we have used the concepts of statistical co-variance between solutions. The co-variance matrices C, B and D aids in faster convergence. The matrix $C_{d\times d}$ for a d dimensional decision space represents an n dimensional ellipsoid containing the most promising solutions belonging to the non-dominated front. This ellipsoid has a mean M. We have used these concepts (in sub-section 5.3) to form onlooker bees for a front around the mean solution M of that front, using the formulae $Onlooker\ Bee = M + zBDr$. All the values in this equation are fixed accept r which is a randomly created vector of size d *i.e.* $r[1\ldots d]$: $r \in (0, 1)$. The uniformity in the r values helps in attaining a front on which the solutions are uniformly distributed. This is the reason we get a pareto optimal front with uniformly distributed solutions, which is highly desirable in multi-objective optimization. As another positive effect, this has reduced the

value of spacing metric also. In all our experiments with M-CABC the value of spacing metric is of the order of 10^{-3} for unconstrained benchmark functions and of the order of 10^{-1} at least, for constrained benchmark functions. The spacing metric for all ZDTx, BINH, SRN, TNK and OSY is even better than that of NSGA-2, a well known algorithm which maintains the diversity in population by its crowding distance operators. These results are presented in the Table 1. The quality of the solutions is improved by 50.3% in terms of *GD* and 43.6% in terms of *SP* over the algorithms which do not use co-variance. The proposed algorithm has more time and space complexity as it performs non-dominated sorting and co-variance calculations in every iteration. At the time of experiments it has also been observed that convergence rate of the algorithms slows down when large number of fronts are obtained. This is because of the formation of in-accurate co-variance matrices when data size is less.

Table 1. ER, GD and SP metric values of M-CABC, MOABC, MOPSO, DEMO, NSGA-2 and SPEA-2 for both constrained and un-constrained benchmark functions.

Test Functions	Metric	Algorithms					
		M-CABC	MOABC	DEMO	MOPSO	NSGA-2	SPEA-2
ZDT1	GD×10^{-4}	**1.97**	11.06	30.71	23.74	23.45	46.71
	SP×10^{-3}	**2.99**	4.34	3.28	7.94	4.35	3.75
ZDT2	GD×10^{-3}	**1.08**	3.18	5.18	1.59	4.56	3.98
	SP×10^{-3}	**1.45**	3.58	4.50	3.25	4.71	2.86
ZDT3	GD×10^{-4}	**1.49**	9.95	16.06	3.84	7.26	30.14
	SP×10^{-3}	**2.11**	5.68	6.66	6.74	2.98	2.99
ZDT4	GD×10^{-2}	**2.84**	149.86	108.67	46.27	137.43	212.69
	SP×10^{-3}	**2.36**	10.13	4.08	10.92	10.92	10.13
ZDT6	GD×10^{-5}	**2.53**	151.74	115.45	91.11	20.29	62.95
	SP×10^{-3}	**1.61**	2.72	2.53	3.04	2.22	2.14
BINH	GD×10^{-1}	**1.14**	2.32	14.35	13.76	13.76	2.23
	SP×10^{-1}	**3.39**	4.46	4.55	3.88	3.88	4.46
CNSTR	GD×10^{-4}	**1.62**	2.31	7.62	4.95	5.08	4.67
	SP×10^{-2}	2.53	3.75	3.39	2.05	2.10	**2.02**
SRN	GD×10^{-2}	**1.09**	4.09	2.66	3.03	1.66	1.20
	SP×10^{-1}	**5.34**	7.05	9.39	7.57	8.13	5.65
TNK	GD×10^{-4}	4.51	3.81	4.83	**3.55**	6.48	6.03
	SP×10^{-3}	**2.17**	4.14	3.41	3.33	2.27	2.14
OSY	GD×10^{-1}	3.09	1.91	4.83	**1.57**	1.91	4.80
	SP×10^{-1}	**3.35**	5.28	4.88	7.32	5.28	4.88
ALL	ER	1.00	1.00	1.00	1.00	1.00	1.00

Finally *GD* and *SP* metric values for each candidate solution is presented as whisker plots in the Fig. 3. This is done to have an in-depth statistical analysis of proposed M-CABC algorithm. From these values and statistical analysis, it can be concluded that the algorithm has converged fully for all ZDTx functions except ZDT4. A remarkable performance is observed for ZDT2 and ZDT6 functions. The similar is the case with con-

strained benchmark functions. M-CABC Algorithm converged well for all benchmark functions and there occurs a noteworthy performance for CNSTR and TNK functions. For spacing metric, little outliers, $\leq 10\%$, are observed for each benchmark function. There encounters a case of $SP \to 0$ for rest of the candidate solutions. Specifically, for ZDT2, ZDT6, CNSTR and TNK the points on the resulting non-dominated front are well distributed.

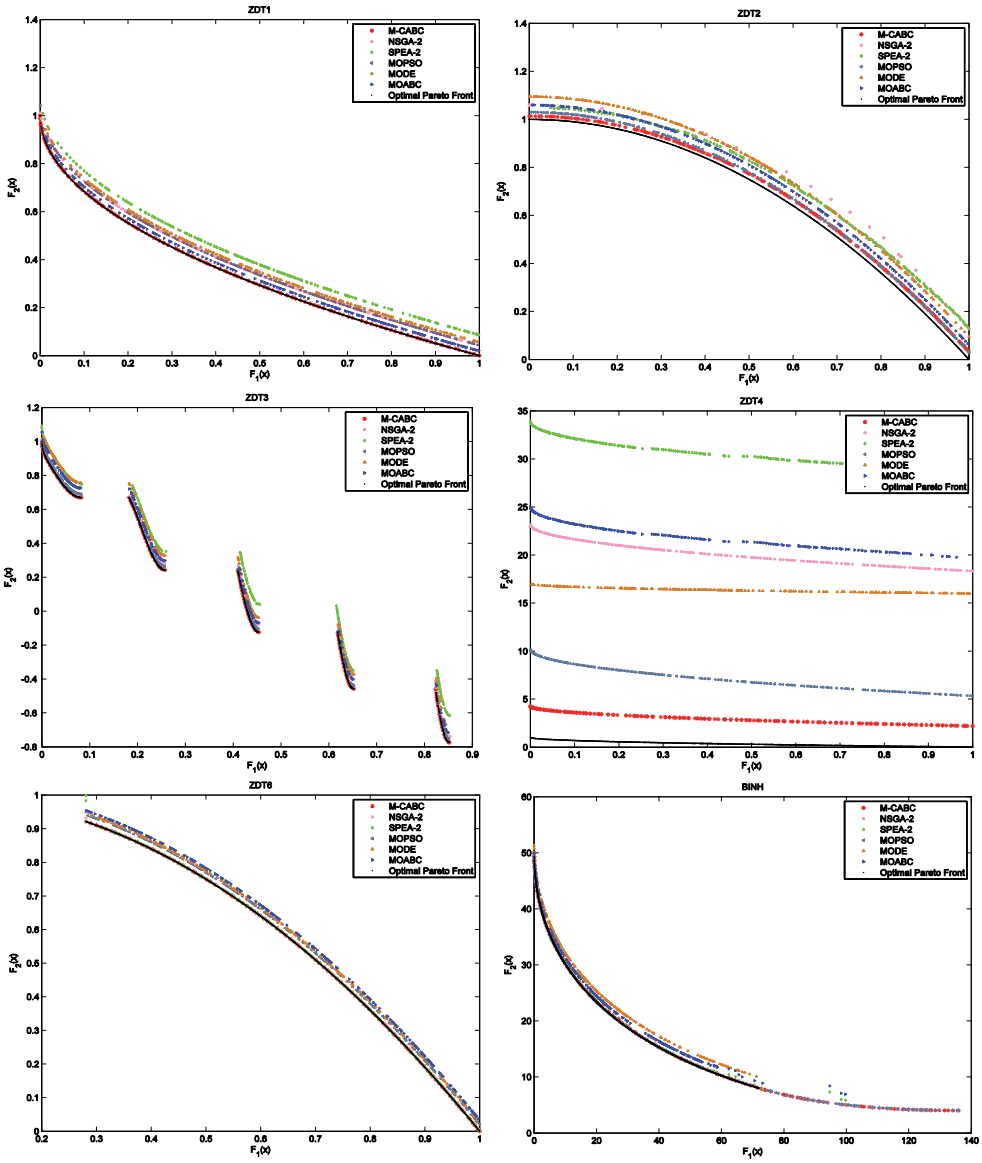

Fig. 1. Non-dominated solutions obtained with M-CABC, NSGA-2, SPEA-2, MOPSO, MODE and MOABC on ZDT1, ZDT2, ZDT3, ZDT4, ZDT6 and BINH benchmark functions.

8. CONCLUSION

In this paper, we have detailed the concepts pertaining to multi-objective optimization and pareto-optamility. We also presented a concise survey of evolutionary and swarm based multi-objective optimizers. Then, we have introduced a fast and efficient, ABC algorithm based multi-objective optimizer which works on the principle of statistical co-variance. We called this algorithm M-CABC and have described its working in detail. M-CABC is later on compared with other well-known multi-objective optimizers (MOABC, DEMO, MOPSO, NSGA-2, SPEA-2) on the basis of error ratio, generational distance and spacing metric. The simulation result establishes the fact that the demonstrated approach is extremely competitive and it can be conceived as a viable alternative to solve multi-objective optimization problems. The quality of the solutions is improved by 50.3% in terms of generational distance and 43.6% in terms of spacing over the algorithms which do not use co-variance. From this quantitative analysis of the results, we can state that results seem to be best in the category and place the M-CABC algorithm in the list of highly successful algorithms. As a part of future work this algorithm may be applied on real world problems and a further performance assessment can be done.

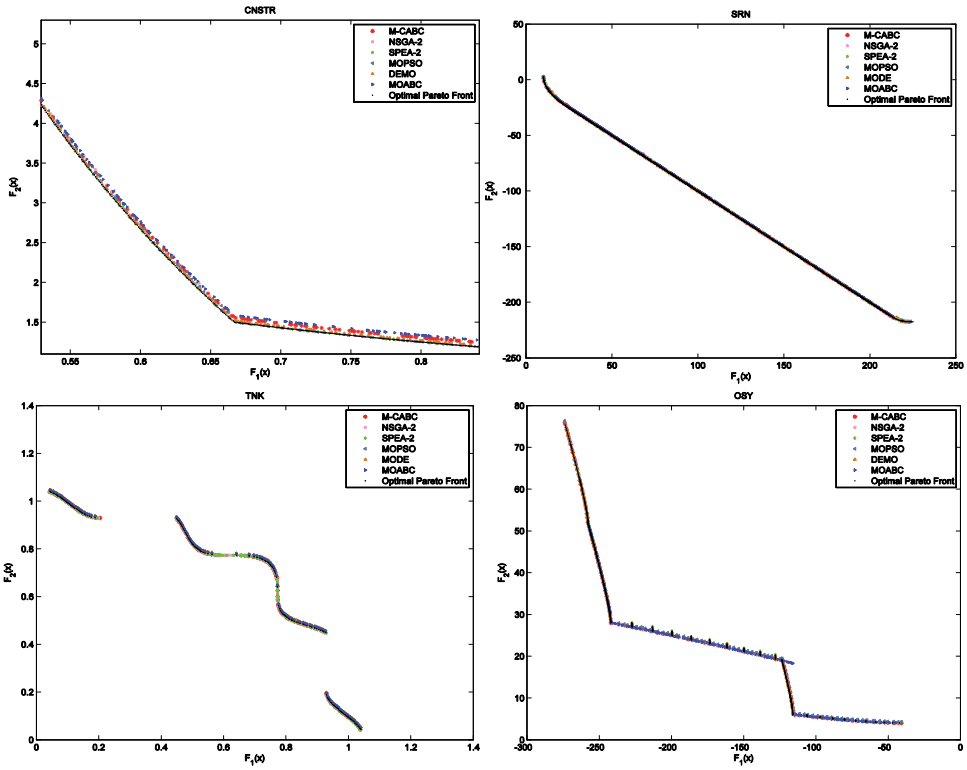

Fig. 2. Non-dominated solutions obtained with M-CABC, NSGA-2, SPEA-2, MOPSO, MODE and MOABC on CNSTR, SRN, TNK and OSY benchmark functions.

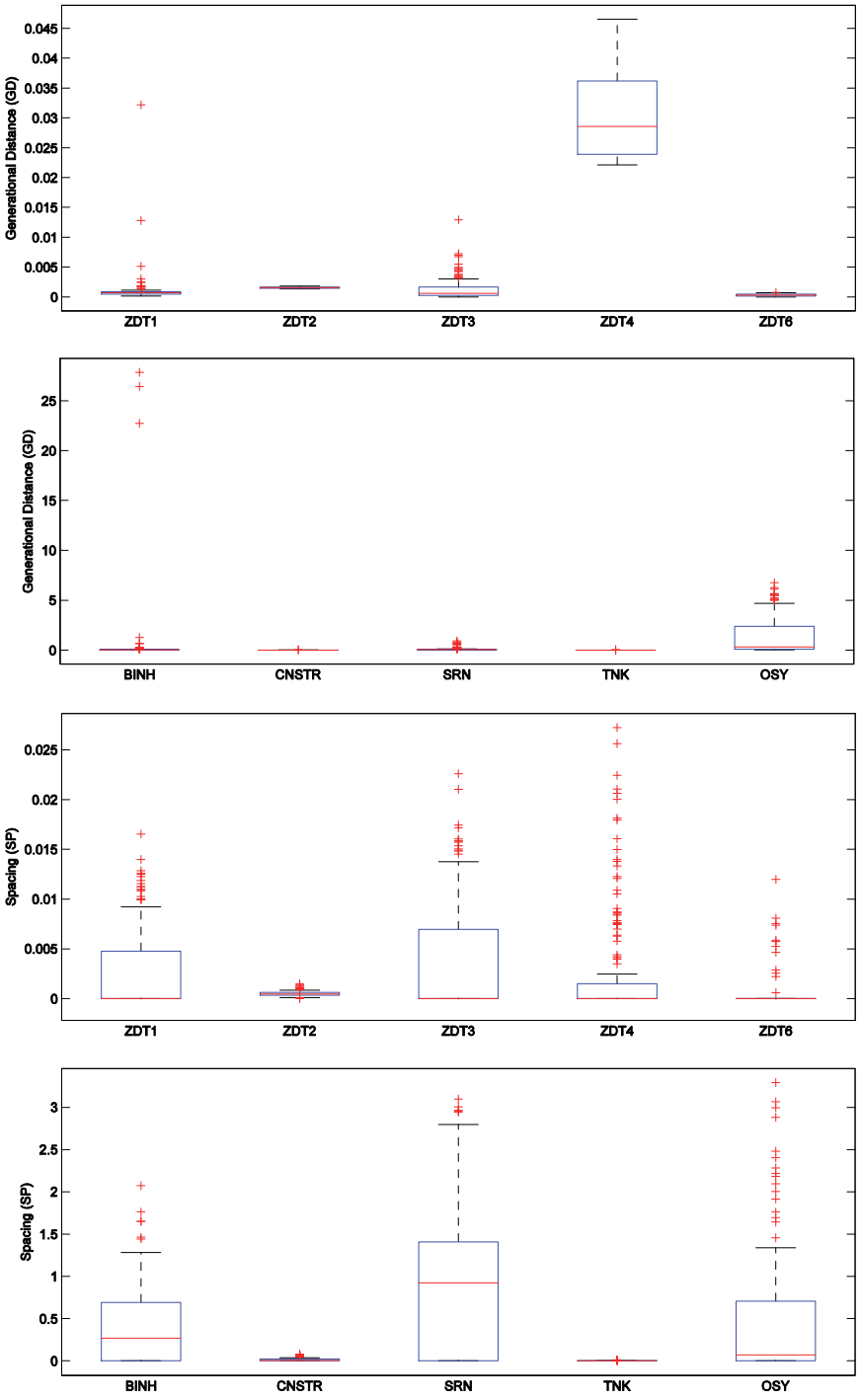

Fig. 3. Statistical whisker plots of the obtained GD and SP metric values of the whole population of candidate solutions for both constrained and un-constrained benchmark functions.

REFERENCES

1. C. A. C. Coello and G. B. Lamont, *Applications of Multi-Objective Evolutionary Algorithms*, World Scientific, Vol. 1, 2004.
2. K. Deb, A. Pratap, and T. Meyarivan, "Constrained test problems for multi-objective evolutionary optimization," *Evolutionary Multi-Criterion Optimization*, Springer, 2001, pp. 284-298.
3. E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach," *IEEE Transactions on Evolutionary Computation*, Vol. 3, 1999, pp. 257-271.
4. M. Dorigo and M. Birattari, "Ant colony optimization," *Encyclopedia of Machine Learning*, Springer, 2010, pp. 36-39.
5. J. Kennedy, "Particle swarm optimization," in *Encyclopedia of Machine Learning*, Springer, 2010, pp. 760-766.
6. D. Karaboga, "An idea based on honey bee swarm for numerical optimization," Technical Report No. tr06, Computer Engineering Department, Erciyes University, 2005.
7. D. Karaboga and B. Basturk, "On the performance of Artificial Bee Colony (ABC) algorithm," *Applied Soft Computing*, Vol. 8, 2008, pp. 687-697.
8. D. Karaboga and B. Akay, "A comparative study of artificial bee colony algorithm," *Applied Mathematics and Computation*, Vol. 214, 2009, pp. 108-132.
9. A. Khorsandi, S. Hosseinian, and A. Ghazanfari, "Modified artificial bee colony algorithm based on fuzzy multi-objective technique for optimal power flow problem," *Electric Power Systems Research*, Vol. 95, 2013, pp. 206-213.
10. M. R. Adaryani and A. Karami, "Artificial bee colony algorithm for solving multi-objective optimal power flow problem," *International Journal of Electrical Power & Energy Systems*, Vol. 53, 2013, pp. 219-230.
11. J.-Q. Li, Q.-K. Pan, and M. F. Tasgetiren, "A discrete artificial bee colony algorithm for the multi-objective flexible job-shop scheduling problem with maintenance activities," *Applied Mathematical Modelling*, Vol. 38, 2014, pp. 1111-1132.
12. M. Yahya and M. Saka, "Construction site layout planning using multi-objective artificial bee colony algorithm with levy flights," *Automation in Construction*, Vol. 38, 2014, pp. 14-29.
13. E. Hancer, B. Xue, M. Zhang, D. Karaboga, and B. Akay, "A multi-objective artificial bee colony approach to feature selection using fuzzy mutual information," in *Proceedings of IEEE Congress on Evolutionary Computation*, 2015, pp. 2420-2427.
14. M. Basu, "Artificial bee colony optimization for multi-area economic dispatch," *International Journal of Electrical Power & Energy Systems*, Vol. 49, 2013, pp. 181-187.
15. K. Naidu, H. Mokhlis, and A. Bakar, "Multiobjective optimization using weighted sum artificial bee colony algorithm for load frequency control," *International Journal of Electrical Power & Energy Systems*, Vol. 55, 2014, pp. 657-667.
16. A. Auger and N. Hansen, "Tutorial cma-es: evolution strategies and covariance matrix adaptation," in *GECCO* (*Companion*), 2012, pp. 827-848.
17. N. Hansen, "The cma evolution strategy: A tutorial," *Vu le*, Vol. 29, 2005, pp. 1-39.

18. Y. Wang, H.-X. Li, T. Huang, and L. Li, "Differential evolution based on covariance matrix learning and bimodal distribution parameter setting," *Applied Soft Computing*, Vol. 18, 2014, pp. 232-247.
19. S. Ghosh, S. Das, S. Roy, S. M. Islam, and P. N. Suganthan, "A differential covariance matrix adaptation evolutionary algorithm for real parameter optimization," *Information Sciences*, Vol. 182, 2012, pp. 199-219.
20. X. Yu and M. Gen, *Introduction to Evolutionary Algorithms*, Springer, London, 2010.
21. A. Konak, D. W. Coit, and A. E. Smith, "Multi-objective optimization using genetic algorithms: A tutorial," *Reliability Engineering & System Safety*, Vol. 91, 2006, pp. 992-1007.
22. C. Coello, C. Dhaenens, and L. Jourdan, "Multi-objective combinatorial optimization: Problematic and context," in *Advances in Multi-Objective Nature Inspired Computing*, ser. Studies in Computational Intelligence, C. Coello Coello, C. Dhaenens, and L. Jourdan, Eds. Springer Berlin Heidelberg, Vol. 272, 2010, pp. 1-21.
23. E. Zitzler, M. Laumanns, and L. Thiele, "Spea2: Improving the strength pareto evolutionary algorithm," in *Proceedings of Evolution Methods Design*, *Optimisation and Control With Applications to Industrial Problems*, 2001, pp. 95-100.
24. K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, Vol. 6, 2002, pp. 182-197.
25. S. Huband, P. Hingston, L. Barone, and L. While, "A review of multiobjective test problems and a scalable test problem toolkit," *IEEE Transactions on Evolutionary Computation*, Vol. 10, 2006, pp. 477-506.
26. R. Akbari, R. Hedayatzadeh, K. Ziarati, and B. Hassanizadeh, "A multi-objective artificial bee colony algorithm," *Swarm and Evolutionary Computation*, Vol. 2, 2012, pp. 39-52.
27. T. Robic and B. Filipic, "Demo: Differential evolution for multiobjective optimization," in *Proceedings of International Conference on Evolutionary Multi-Criterion Optimization*, 2005, pp. 520-533.
28. C. C. Coello and M. S. Lechuga, "Mopso: A proposal for multiple objective particle swarm optimization," in *Proceedings of IEEE Congress on Evolutionary Computation*, Vol. 2, 2002, pp. 1051-1056.
29. A. Brondsted and R. Rockafellar, "On the subdifferentiability of convex functions," in *Proceedings of the American Mathematical Society*, Vol. 16, 1965, pp. 605-611.
30. J. Snyman, *Practical Mathematical Optimization: An Introduction to Basic Optimization Theory and Classical and New Gradient-Based Algorithms*, Springer Science & Business Media, USA, Vol. 97, 2005.
31. M. Abramowitz and I. A. Stegun, *Handbook of Mathematical Functions: with Formulas*, *Graphs*, *and Mathematical Tables*, Courier Corporation, USA, No. 55, 1964.
32. F. Cajori, "Historical note on the newton-raphson method of approximation," *The American Mathematical Monthly*, Vol. 18, 1911, pp. 29-32.
33. G. Jastrebski, D. V. Arnold, *et al.*, "Improving evolution strategies through active covariance matrix adaptation," in *Proceedings of IEEE Congress on Evolutionary Computation*, 2006, pp. 2814-2821.
34. C. A. C. Coello, "Recent trends in evolutionary multiobjective optimization," in *Evolutionary Multiobjective Optimization*, Springer, 2005, pp. 7-32.

35. E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms: Empirical results," *Evolutionary Computation*, Vol. 8, 2000, pp. 173-195.
36. D. A. Van Veldhuizen, "Multiobjective evolutionary algorithms: classifications, analyses, and new innovations," DTIC Document, Technical Report No. ADA364478, 1999.
37. D. A. Van Veldhuizen and G. B. Lamont, "Multiobjective evolutionary algorithm research: A history and analysis," Citeseer, Technical Report No. 98-03, 1998.
38. J. R. Schott, "Fault tolerant design using single and multicriteria genetic algorithm optimization," DTIC Document, Technical Report No. ADA296310, 1995.
39. F. Jimenez, A. F. Gomez-Skarmeta, G. Sanchez, and K. Deb, "An evolutionary algorithm for constrained multi-objective optimization," in *Proceedings of IEEE Congress on Evolutionary Computation*, Vol. 2, 2002, pp. 1133-1138.
40. T. T. Binh and U. Korn, "Mobes: A multiobjective evolution strategy for constrained optimization problems," in *Proceedings of the 3rd International Conference on Genetic Algorithms*, Vol. 25, Citeseer, 1997, p. 27.
41. N. Srinivas and K. Deb, "Muiltiobjective optimization using nondominated sorting in genetic algorithms," *Evolutionary Computation*, Vol. 2, 1994, pp. 221-248.
42. M. Tanaka, H. Watanabe, Y. Furukawa, and T. Tanino, "Ga-based decision support system for multicriteria optimization," in *Proceedings of IEEE International Conference on Systems, Man and Cybernetics*, Vol. 2, 1995, pp. 1556-1561.
43. A. Osyczka and S. Kundu, "A new method to solve generalized multicriteria optimization problems using the simple genetic algorithm," *Structural Optimization*, Vol. 10, 1995, pp. 94-99.

Divya Kumar received his M.Tech. degree in Software Engineering from MNNIT Allahabad, India. He is currently working as an Assistant Professor at MNNIT Allahabad and simultaneously pursuing his Ph.D. in Swarm Intelligence under the guidance of Dr. K. K. Mishra. His research interests include software engineering, soft computing and algorithm design and analysis.

K. K. Mishra received his Ph.D. degree in Computer Science and Engineering from MNNIT Allahabad, India. He is currently serving as an Assistant Professor at MNNIT Allahabad, India and a Visiting Professor at University of Missouri, St. Louis. His research interests include automata theory, machine learning, evolutionary and swarm intelligence.