processes

Article

An Improved Artificial Electric Field Algorithm for Multi-Objective Optimization

Hemant Petwal * and **Rinkle Rani**

Department of Computer Science and Engineering, Thapar Institute of Engineering and Technology, Patiala 147004, Punjab, India; raggarwal@thapar.edu
* Correspondence: hemant.petwal@thapar.edu

Received: 28 February 2020; Accepted: 12 April 2020; Published: 14 May 2020

Abstract: Real-world problems such as scientific, engineering, mechanical, etc., are multi-objective optimization problems. In order to achieve an optimum solution to such problems, multi-objective optimization algorithms are used. A solution to a multi-objective problem is to explore a set of candidate solutions, each of which satisfies the required objective without any other solution dominating it. In this paper, a population-based metaheuristic algorithm called an artificial electric field algorithm (AEFA) is proposed to deal with multi-objective optimization problems. The proposed algorithm utilizes the concepts of strength Pareto for fitness assignment and the fine-grained elitism selection mechanism to maintain population diversity. Furthermore, the proposed algorithm utilizes the shift-based density estimation approach integrated with strength Pareto for density estimation, and it implements bounded exponential crossover (BEX) and polynomial mutation operator (PMO) to avoid solutions trapping in local optima and enhance convergence. The proposed algorithm is validated using several standard benchmark functions. The proposed algorithm's performance is compared with existing multi-objective algorithms. The experimental results obtained in this study reveal that the proposed algorithm is highly competitive and maintains the desired balance between exploration and exploitation to speed up convergence towards the Pareto optimal front.

Keywords: artificial electric field algorithm; strength Pareto; multi-objective optimization problems; recombination operator; fine-grained elitism selection; shift-based density estimation

1. Introduction

Most realistic problems in science and engineering consist of diverse competing objectives that require coexisting optimization to obtain a solution. Sometimes, there are several distinct alternatives [1,2] solutions for such problems rather than a single optimal solution. Targeting objectives are considered the best solutions as they are superior to other solutions in decision space. Generally, realistic multiple objective problems (MOOPs) require a long time to evaluate each objective function and constraint. In solving such realistic MOOPs, the stochastic process shows more competence and suitability than conventional methods. The evolutionary algorithms (EA) that mimic natural biological selection and evolution process are found to be an efficient approach to solve MOOPs. These approaches evidence their efficiency to solve solutions simultaneously and explore an enormous search space with adequate time. Genetic algorithm (GA) is an acclaimed approach among bio-inspired EA. In recent years, GA has been used to solve MOOPs, called non-dominated sorting genetic algorithm (NSGA-II) [3]. NSGA II is recognized as a recent advancement in multi-objective evolutionary algorithms. Besides, particle swarm optimization is another biologically motivated approach which is extended to multi-objective particle swarm optimization (MOPSO) [4,5], bare-bones multi-objective particle swarm optimization (BBMOPSO) [6], and non-dominated sorting particle swarm optimization (NSPSO) [7] to solve multi-objective optimization problems. These multi-objective approaches in one computation can produce several evenly distributed candidate solutions rather than a single solution.

However, based on Coulomb's law of attraction electrostatic force and law of motion principle, a new algorithm called an artificial electric field algorithm [8] (AEFA) was proposed. AEFA mimics the interaction of charged particles under the attraction electrostatic force control and follows an iterative process. In a multi-dimensional search space, charged particles move towards heavier charges and converge to the heaviest charged particle. Due to the strong capability of exploration and exploitation, AEFA proved to be more robust and effective than artificial bee colony (ABC) [8], ant colony optimization (ACO) [8], particle swarm optimization (PSO) [8], and bio-geography based optimization (BBO) [8]. AEFA received growing attention from researchers in the development of a series of algorithms in several areas, such as parameter optimization [9], capacitor bank replacement [10], and scheduling [11]. There have been no research contributions found in recent years that have extended AEFA to solve MOOPs. In this paper, a modest contribution for improving AEFA to solve MOOP is made. The proposed algorithm (1) adopted the concept of "strength Pareto" for the fitness refinement of the charged particles, (2) introduced a fine-grained elitism selection mechanism based on the shift-based density estimation (SDE) approach to improve population diversity and manage the external population set, (3) utilized SDE with "strength Pareto" for density estimation, and (4) utilized bounded exponential crossover (BEX) [12] and polynomial mutation operator (PMO) [13,14] to improve global exploration, the local exploitation capability, and the convergence rate.

This paper is organized in the following way: Section 2 covers an overview of the existing literature on multi-objective optimization, Section 3 describes the preliminaries and background algorithms, Section 4 describes the proposed multi-objective method in detail, Section 5 presents results and performance of the proposed algorithm in comparison to existing optimization techniques, and Section 6 sums up the findings of this research in concluding remarks.

2. Related Work

Since a single solution cannot optimize multiple objectives at once, the problem is formulated as multi-objective optimization. Nobahari et al. [1] proposed a non-dominated sorting based gravitational search algorithm (NSGSA). Several multi-objective evolutionary algorithms (MOEAs) [15–18] are proposed in the past years. These algorithms have proven their ability to find various Pareto-optimal solutions in one single computation. Srinivas and Deb [19] proposed a non-dominated Sorting genetic algorithm (NSGA) where elitism was used to achieve a better convergence. Zitzler and Thiele [20] introduced the concept strength Pareto evolutionary algorithm (SPEA) with elitism selection. They proposed to maintain an external population to keep all the non-dominated solutions obtained in each generation. In each generation, the external and internal population were combined as a set, then based on the number of dominated solutions, all the non-dominated solutions in the combined set were assigned a fitness value. The fitness value decided the rank of the solutions that directed the exploration process towards the nondominated solution. Rudolph [21] proposed a simple elitist [22] multi-objective EA based on a comparison between the parent population and the child population. At each iteration, the candidate parent solutions were compared with the child non-dominated solution, and a final non-dominated solution set was formed to participate as the parent population for the next iteration. Although the proposed algorithm ensured the convergence to the Pareto-optimal front, it suffered from population diversity loss. All population-based evolutionary algorithms help in maintaining population diversity and convergence for multi-objective optimization [23]. In recent years, many significant contributions have been made that centered around the development of hybrid [24] algorithms and many-objective optimization algorithms [25,26]. Zhao and Cao [27] proposed an external memory-based particle swarm optimization (MOPSO) algorithm in which, besides the initial search population, two external memories were used to store global and local best individuals. Also, a geographic-based technique was utilized to maintain population diversity. Zhang and Li [28] combined traditional mathematical approaches with EA and introduced a decomposition-based MOEA,

MOEA/D [29]. Martinez and Coello [30] extended MOEA/D and proposed a constraint-based selection mechanism to solve constrained MOPs. Gu et al. [31] proposed a dynamic weight design method based on the projections of non-dominated solutions to produce evenly distributed non-dominated solutions. Zhang [32] introduced a novel and efficient approach for non-dominated sorting to solve MOEAs. Chong and Qiu [33] proposed a novel opposition-based self-adaptive hybridized differential evolution algorithm to handle the continuous MOPs. Cheng et al. [34] proposed a many-objective optimization for high-dimensional objective space in which a reference vector guided an evolutionary algorithm introduced to maintain balance between diversity and convergence of the solutions. Hassanzadeh and Rouhani [35] proposed a multi-objective gravitational search algorithm (MOGSA). Yuan et al. [36] extended the gravitational search algorithm (GSA) to strength Pareto based multi-objective gravitational search algorithm (SPGSA). A review of existing literature is presented in Table 1.

Table 1. Summary of existing MOOPs.

Author(s)	Objective/Work Done	Technique Proposed/Used	Performance Parameters	Research Gap(s) Identified
Nobahari, et.al. [1]	Proposed a multi-objective gravitational search based on non-dominated sorting for power transformer design	Non-dominated sorting gravitational search algorithm (NSGSA)	Normalized arithmetic mean	The Algorithm lacks scalability when dealing with complex cases of power transformer design.
Srinivas, and Deb [19]	Proposed a multi-objective genetic algorithm based on non-dominated sorting	Non-dominated sorting genetic algorithm (NSGA)	Chi-square test	The proposed algorithm shows a slow convergence rate.
Deb and Jain [25]	Proposed an evolutionary approach for solving many-objective optimization	Reference-based non dominated sorting approach (NSGA3)	1. Benchmark functions 2. Mean 3. Standard deviation 4. Inverted generational distance (IGD)	Recombination operator can be improved to enhance population diversity
Zhang and Li [28]	Proposed a multi-objective evolutionary algorithm based on decomposition	The multi-objective evolutionary algorithm based on decomposition (MOEA/D)	1. C-metric 2. D-metric	The penalty parameter used in the proposed algorithm is statically initialized. For an extremely lower or higher value, the performance of the penalty method decreases
Martinez and Coello [30]	Proposed a decomposition-based multi-objective evolutionary algorithm (eMOEA/D-DE) for constraint MOOP	A new selection mechanism based on ε-constraint method	1. Hypervolume 2. Feasibility ratio	Constraint parameters (ε, δ) used in the algorithm are statically initialized. It can be initialized dynamically.
Gu et.al. [31]	Proposed a multi-objective evolutionary algorithm based on the projection of the current non-dominated solutions and equidistance interpolation	Dynamic weight design method with MOEA/D	1. Benchmark functions 2. Mean 3. Standard deviation 4. Inverted generational distance (IGD)	The algorithm lacks efficacy to solve complex higher-dimensional problems.
Zhang, et al. [32]	Proposed a novel and computationally efficient approach to non-dominated sorting	Efficient non-dominated sort (ENS)	1. Number of dominance comparison 2. Execution time	The efficiency of the algorithm decreases with an increase in the number of objectives

Table 1. *Cont.*

Author(s)	Objective/Work Done	Technique Proposed/Used	Performance Parameters	Research Gap(s) Identified
Chong and Qiu [33]	Proposed MOO algorithm to solve multi-objective traveling salesman problem	Self-adaptive differential algorithm with a decomposition-based framework (D-OSADE)	1. Benchmark functions 2. Mean 3. Standard deviation 4. Inverted generational distance (IGD)	The algorithm becomes less effective as the number of salesmen increases
Cheng et al. [34]	Proposed an evolutionary algorithm for many-objective optimization	Reference vector guided evolutionary algorithm (RVEA)	1. Benchmark functions 2. Mean 3. Standard deviation 4. Wilcoxon rank-sum test	The reference vector is static. The selection type of reference vector to be used in many-objective optimization is not considered
Hassanzade and Rouhani [35]	Proposed a multi-objective algorithm based on gravitational force	Multi-objective gravitational search algorithm (MOGSA)	1. Spacing metric 2. Generational distance metric	The algorithm suffers premature convergence in solving complex higher-dimensional problems.
Yuan et al. [36]	Proposed a multi-objective gravitational search based on the concept of strength Pareto	Strength Pareto gravitational search (SPGSA)	1. Convergence metric 2. Space metric 3. Generational distance metric 4. Diversity metric	Population diversity can be further improved.

Our Contribution

In this paper, an improved artificial electric field algorithm is proposed to solve multi-objective optimization problems. The concept of strength Pareto is used in the optimization process of the proposed algorithm to refine fitness assignment. The shift-based density estimation (SDE) technique with fine-grained elitism selection approach and SDE with strength Pareto are applied to improve the population diversity and density estimation, respectively. The bounded exponential crossover (BEX) operator and polynomial mutation operator (PMO) are used to reduce the possibility of the solution trapping in local optima. The proposed algorithm is validated using different benchmark functions and compared with existing multi-objective optimization techniques.

3. Preliminaries and Background

This section briefly discusses the basic concepts of multi-objective optimization, artificial electric field algorithm, shift-based density estimation technique, and recombination and mutation operators.

3.1. Multi-Objective Optimization

In general, a multiple-objective problem (MOOP) involving m diverse objectives is mathematically defined as

$$P = \{P_1;\ P_2;\ \ldots\ ;\ P_d\}$$

where P represents the solution to MOOP, and d represents the dimension of the decision boundary. A MOOP can be a minimization problem, a maximation problem, and a combination of both. The target objectives in MOOP are defined as

Minimize/Maximize:

$$\mathrm{F}(P) = [\mathrm{F}_i(P),\ i = 1,\ 2\ldots.m]$$

Subject to the constraints:

$$\begin{cases} G_a(P) \le 0,\ a = 1,2,\ldots.A \text{ and,} \\ H_b(P)\ \le 0,\ b = 1,2,\ldots.B \end{cases}$$

where $F_i(P)$ represents the ith objective function of P^{th} solution. An MOOP either can have no or more than one constraint, e.g., $G_a\left(\vec{P}\right)$ and $H_b\left(\vec{P}\right)$ are considered as the a^{th} and b^{th} optional inequality and equality constraints, respectively. A and B represent the total number of inequality and equality constraints, respectively. The MOOP is then processed to calculate the value of the solution (P) for which $F(P)$ satisfies the desired optimization. Unlike single-objective optimization (SOO), MOOP considers multiple conflicting objectives and produces a set of solutions in the search space. These solutions are processed by the concept of Pareto dominance theory [36], defined as follows:

Considering an MOOP, a vector $\vec{x} = x_1;\ x_2;\ \ldots\ ;\ x_m$ shows domination on vector $\vec{y} = y_1;\ y_2;\ \ldots\ ;\ y_m$, if and only if

$$\forall\, i \in \{1 \ldots .m\},\ x_i \leq y_i\ \ \exists\, i\ \in\ \{1 \ldots .m\}\ :\ x_i < y_i$$

where m represents the objective space dimension. A solution $\vec{x}\ \in X$ (Universe) is called Pareto optimal if and only if no other solution $y \in Y$ dominates it. In such a case, solutions $\left(\vec{x}\ \right)$ are said to be non-dominated solutions. All these non-dominated solutions are composed to a set called Pareto-optimal set.

3.2. Artificial Electric Field Algorithm (AEFA)

Artificial electric field algorithm (AEFA) is a population-based meta-heuristic algorithm, which mimics the Coulomb's law of attraction electrostatic force and law of motion. In AEFA, the possible candidate solutions of the given problem are represented as a collection of the charged particles. The charge associated with each charged particle helps in determining the performance of each candidate solution. Attraction electrostatic force causes each particle to attract towards one another resulting in global movement towards particles with heavier charges. A candidate solution to the problem corresponds to the position of charged particles and fitness function, which determines their charge and unit mass. The steps of AEFA are as follows:

Step 1. Initialization of Population.

A population of P candidate solutions (charged particles) is initialized as follows:

$$CP_i = \left(CP_i^1, CP_i^2, CP_i^k \ldots\ldots\ldots CP_i^D\ \right),\ \forall\, i = 1,2,3 \ldots p \tag{1}$$

where CP_i^k represents the position of ith charged particle in the kth dimension, and D is the dimensional space.

Step 2. Fitness Evaluation.

A fitness function is defined as a function that takes a candidate solution as input and produces an output to show how well the candidate solution fits with respected to the considered problem. In AEFA, the performance of each charged particle depends on the fitness value during each iteration. The best and worst fitness are computed as follows:

$$Best\ (T) = min_{i=1}^{n}\ Fitness\ (T) \tag{2}$$

$$Worst\ (T) = max_{i=1}^{n}\ Fitness\ (T) \tag{3}$$

where $Fitness_i\ (T)$ and n represent the fitness value of ith charged particle and total number of charged particles in the population, respectively. $Best\ (T)$, $Worst\ (T)$ represent the best fitness and the worst fitness respectively of all charged particles at time T.

Step 3. Computation of Coulomb's Constant.

At time t, Coulomb's constant is denoted by $K(t)$ and computed as follows:

$$K(t) = K_0 * \exp\left(-\alpha \frac{iter}{maxiter}\right) \tag{4}$$

Here, K_0 represents the initial value and is set to 100. α is a parameter and is set to 30. *iter* and *maxiter* represent current iteration and maximum number of iterations, respectively.

Step 4. Compute the Charge of Charged Particles.

At time T, the charge of ith charged particle is represented by $Q_i(T)$. It is computed based on the current population's fitness as follows:

$$Q_i(T) = \frac{q_i(T)}{\sum_{i=1}^{n} q_i(T)} \tag{5}$$

where

$$q_i(T) = exp\left(\frac{fitness_{cp_i}(T) - Worst\ (T)}{Best\ (T) - Worst(T)}\right) \tag{6}$$

Here, $Fitness_{cp_i}$ is the fitness of the ith charged particle. $q_i(T)$ helps in determining the total charge $Q_i(T)$ acting on the ith charged particle at time T.

Step 5. Compute the Electrostatic Force and Acceleration of the Charged Particles.

1. The electrostatic force exerted by the jth charged particle on the ith charged particle in the D^{th} dimension at time T is computed as:

$$F_{ij}^{D}(T) = K(t)\frac{\left(Q_i(T) * Q_j(T)\right) * \left(P_j^D(T) - X_j^D(T)\right)}{R_{ij}(T) + \varepsilon} \tag{7}$$

$$F_i^D(T) = \sum_{j=1,\ j\neq i}^{N} rand\ ()\ * F_{ij}^D(T) \tag{8}$$

where $Q_i(T)$ and $Q_j(T)$ are the charges of ith and jth charged particles at any time T. ε is a small positive constant, and $R_{ij}(T)$ is the distance between two charged particles i and j. $P_j^D(T)$ and $X_j^D(T)$ are the global best and current position of the charged particle at time T. $F_i^D(T)$ is the net force exerted on ith charged particle by all other charged particles at time T. *rand* () is uniform random number generated in the [0, 1] interval.

2. The acceleration $a_i^D(T)$ of ith charged particle at time T in D^{th} dimension is computed using the Newton law of motion as follows:

$$a_i^D(T) = \frac{Q_i(T) * E_i^D(T)}{M_i^D(T)},\ E_i^D(T) = \frac{F_i^D(T)}{Q_i(T)} \tag{9}$$

where $E_i^D(T)$ and $M_i^D(T)$ represent respectively the electric field and unit mass of ith charged particle at time T and in D^{th} dimension.

Step 6. Update velocity and position of charged particles.

At time T, the position and velocity of ith charged particle in D^{th} dimension ae updated as follows:

$$vel_i^D(T+1) = rand_i *\ vel_i^D(T) + a_i^D(t) \tag{10}$$

$$CP_i^D(T+1) = CP_i^D(T) + vel_i^D(t) \tag{11}$$

where $vel_i^D(T)$ and $CP_i^D(T)$ represent the velocity and position of the ith charged particle in D^{th} dimension at time T, respectively. $rand()$ is a uniform random number in the interval [0, 1].

3.3. Shift-Based Density Estimation

In order to maintain good convergence and population diversity, the shift-based density estimation (SDE) [37] was introduced. While determining the density of an individual (P_i), unlike traditional density estimation approaches, SDE shifts other individuals in the population P to a new position. The SDE utilizes the convergence comparison between current and other individuals for each objective. For example, if $P_j \in P$ outperforms $P_k \in P$ on any objective, the corresponding objective value of P_J is shifted to the position of P_k on the same objective. In contrast, the value (position) of the objective remains unchanged. The process is described as

$$\mathrm{F}_i^{sd}\left(P_j\right) = \begin{cases} \mathrm{F}_i'(P_k), \text{ if } \mathrm{F}_\mathrm{i}'\left(P_j\right) < \mathrm{F}_\mathrm{i}'(\mathrm{P}_k) \\ \mathrm{F}_i'\left(P_j\right), \; otherwise \end{cases} \tag{12}$$

where $\mathrm{F}_i^{sd}\left(P_j\right)$ represent the shifted objective value of $\mathrm{F}_\mathrm{i}'\left(P_j\right)$, and $F^{sd}\left(P_j\right) = \left(F_1^{sd}\left(P_j\right), F_2^{sd}\left(P_j\right), F_3^{sd}\left(P_j\right), \ldots F_m^{sd}\left(P_j\right)\right)$ represents shifted objective vector of $F\left(P_j\right)$. The steps followed in density estimation are explained in Algorithm 1.

Algorithm 1 Density Estimation for Non-Domination Solutions

```
Input: Non-dominated solutions $\left(P_{I_{c_1}}, P_{I_{c_1}}, \ldots . P_{I_{cn}}\right)$
Output: Density of each solution
1.  Shift the position of non-dominated solutions using Equation (12)
2.  Calculate the distance between two adjacent non-dominated solutions as follows:
        $d_{\;P_{I_{c_i}},\, P_{I_{c_{i+1}}}} = \sqrt{\sum_{j=1}^{n}\left(Fit\left(P_{I_{c_i,j}}\right) - Fit\left(P_{I_{c_{i+1,j}}}\right)\right)^2}$
3.  Find the kth minimum value ($\sigma_{\;P_{I_{c_k}}}$) in $\{d_{P_{Ici}}, d_{P_{Ici+1}}, P_{I_{c_i}} \in \mathrm{P} \cap P_{I_{c_i}} \neq P_{I_{c_{i+1}}}\}$
4.  Compute $SDE\left(P_{I_{c_i}}\right) = \frac{1}{\sigma P_{I_{c_k}} + 2}$
```

3.4. Recombination and Mutation Operators

In AEFA, movement (acceleration) of one charged particle towards another is determined by the total electrostatic force exerted by other particles on it. In this paper, bounded exponential crossover (BEX) [12] and polynomial mutation operator (PMO) [13,14] are used with AEFA to enhance the acceleration in order to increase convergence speed further.

3.4.1. Bounded Exponential Crossover (BEX)

The performance of MOOP highly depends upon the solution generated by the crossover operator. A crossover operator produces solutions that reduce the possibility of being trapped in local optima. Bounded exponential crossover (BEX) is used to generate the offspring solutions in the interval $\left[P_i^l, P_i^u\right]$. BEX involves an additional factor α that depends on the interval $\left[P_i^l, P_i^u\right]$ and the parent solution position. The steps to generate offspring solutions in the interval $\left[P_i^l, P_i^u\right]$ and related to each pair of the parent x_i and y_i are given in Algorithm 2.

3.4.2. Polynomial Mutation Operator (PMO)

In order to avoid premature convergence in MOOP, the mutation operator is used. In this paper, the polynomial mutation operator (PMO) is used with the proposed algorithm. PMO includes two control parameters: mutation probability of a parent solution (pm) and the magnitude of the expected solution (η_m). For an individual $P_i^{(t)} = (P_1^{(t)}, P_2^{(t)} \ldots\ldots P_m^{(t)})$, PMO is computed as follows:

$$P_i^{(t+1)} = P_i^{(t)} + \delta * (yu - yd) \tag{13}$$

where $P_i^{(t+1)}$ represents the decision variable after the mutation, and $P_i^{(t)}$ represents the decision variable before the mutation. yu and yd represent the lower and upper bounds of the decision variable, respectively. δ represents a small variation obtained by

$$\delta = \begin{cases} \left[\ 2r_2 + (1-2r_2) * \left(\frac{\max\left(yu - x_i^t\ ,\ x_i^t - yd\right)}{yu - yd}\right)^{\eta m+1}\ \right]^{\frac{1}{\eta m+1}} - 1 \text{ if } \mathrm{r}_2 \le 0.5 \\ 1 - \left[\ 2r_2 + (1-2r_2) * \left(\frac{\max\left(yu - x_i^t\ ,\ x_i^t - yd\right)}{yu - yd}\right)^{\eta m+1}\ \right]^{\frac{1}{\eta m+1}}, \text{otherwise} \end{cases} \tag{14}$$

Here, r_2 is a random number in [0, 1] interval, which is compared with a predefined threshold value (0.5), and η_m represents the mutation distribution index.

```
Algorithm 2 Bounded Exponential Crossover (BEX)
Input: Parent solutions $x = x_1,\ x_2, \ldots .x_m$ and $y = y_1,\ y_2, \ldots .y_m$ and scaling parameter λ > 0
Output: Offspring solutions
While $i \le$ m do
    1. Generate uniformly distributed random number $u_i \in U(0,1)$
    2. Compute $\beta_i^x$ and $\beta_i^y$ which is derived by inverting the bounded exponential distribution,
       $\beta_i^x = \begin{cases} \lambda \log\left\{\exp\left(\frac{x_i^l - x_i}{\lambda(y_i - x_i)}\right) + u_i\left(1 - \exp\left(\frac{x_i^l - x_i}{\lambda(y_i - x_i)}\right)\right)\right\}, \text{ if } r_i \le 0.5 \\ -\lambda\ \log\left\{1 - u_i\left(1 - \exp\left(\frac{x_i^l - x_i}{\lambda(y_i - x_i)}\right)\right)\right\}, \text{ if } r_i > 0.5 \end{cases}$
       $\beta_i^y = \begin{cases} \lambda \log\left\{\exp\left(\frac{x_i^l - y_i}{\lambda(y_i - x_i)}\right) + u_i\left(1 - \exp\left(\frac{x_i^l - y_i}{\lambda(y_i - x_i)}\right)\right)\right\}, \text{ if } r_i \le 0.5 \\ -\lambda\ \log\left\{1 - u_i\left(1 - \exp\left(\frac{y_i^l - x_i}{\lambda(y_i - x_i)}\right)\right)\right\}, \text{ if } r_i > 0.5 \end{cases}$
       where $r_i \in U(0,1)$ represents uniformly distributed random variable and $x_i^l,\ x_i^u$ are the lower and upper bounds of the ith decision variable.
    3. Offspring solutions are generated using
       $\varepsilon_i = x_i + \beta_i^x\ (y_i - x_i),\ \eta_i = y_i + \beta_i^x\ (y_i - x_i)$
end while
```

4. Proposed Algorithm

In this section, the proposed multi-objective optimization method is described in detail. The algorithm starts with parameter initialization. Then, a population of candidate solutions is generated. Further, by computing fitness values for each candidate solution, the best solution is selected. The population is iteratively updated until the termination conditions are satisfied and the optimal solution is returned. The proposed algorithm is presented in Algorithm 3. The explanation of the symbols used in the proposed algorithm is presented in Table 2. The steps followed in the proposed algorithm are as follows.

4.1. Population Generation

In the proposed algorithm, there are two populations, i.e., searching population (P_n) and the external population $\left(P_{I_c}^*\right)$. The searching population, which contains initial candidate solutions, computes the non-dominated solutions and stores them in the external population. The process is performed iteratively.

Algorithm 3 Proposed Multi-Objective Optimization Algorithm

```
Input: Searching population of size $(n)$, External Population of size $(m)$
Output: Non-dominated set of charged particles $\left(ND_{C_p}\right)$
Begin
1.  Initialize searching population ( $P_n$ ) of charged particles $CP$.
2.  Initialize external population $\overline{P^*} = \varnothing$ and set iteration counter $I_c = 1$
While ($I_c < I_{cmax}$)
    For each $CP \in P_{I_c}\ U\ \overline{P^*_{I_c}}$
1.  Compute the fitness $F(CP)$ using Equation (15)
2.  Computer the density using shift-based density estimation (Section 3.3)
    end for
    For each $CP \in P_{I_c}\ U\ \overline{P^*_{I_c}}$ do
        If $F(CP) < 1$ then
        $P^*_{I_{c+1}} = P^*_{I_{c+1}}\ U\ \{CP\}$
      end if
    end for
    If $(P^*_{I_{c+1}} < m)$ then
     $\overline{P_{I_c+1}} = \overline{P_{I_c+1}}\ U\ \left(\left(P_{I_c} U\ P^*_{I_c}\right)\left[1 : m - \left|\overline{P_{I_c+1}}\right|\right]\right)$
    else
        delete non-dominated solution from $P^*_{I_c}$ using Equation (17) as described in Section 4.3
     end if
a.  Select charged particles (CP) into the mating pool $P_{I_c+1}$ from $P_{I_c}\ \ U\ P^*_{I_{c+1}}$
b.  Evaluate charge, update the velocity and position of $P_{I_C+1}$ as defined in Section 3.2 and obtain the new
    position of $CP$
c.  Apply crossover and mutation operator (using Section 3.4) on population $P_{I_c+1}$
end while
    For each $CP \in \overline{P_{t+1}}$ do
     If $CP$ is a non-dominated solution then
        $ND_{C_p} = ND_{C_p} * \{CP\}$ ;
     end if
    end for
```

4.2. Fitness Evaluation

The traditional AEFA is not suitable for the MOOPs due to the definition of charge. According to Equations (7) and (8), the charge of a particle is related to fitness value. Thus, the multi-objective fitness assignment used in the SPEAII [38] is introduced to evaluate the charge of the AEFA algorithm for two or more objectives in MOOPs. The multi-objective fitness assignment of the proposed algorithm is calculated by using the formula as follows:

$$F(i) = R(i) + D(i),\ R(i) = \sum_{j\ \in \mathrm{P}_t \mathrm{U}\ \mathrm{P}'_t} s(j) \text{ and } s(j)|\ \{\ i|i \in\ \mathrm{P}_t\mathrm{U}\ \mathrm{P_t}^*\ \}| \tag{15}$$

where $F(i)$ is the fitness value of the charged particle i. $R(i)$ means the raw fitness value of the charged particle i, and $D(i)$ represents the additional density information of the charged particle i. The raw fitness exhibits the strength of each charged particle by assigning a rank to each charged particle. In MOOP, to avoid such conditions where more than more solution (charged particle) shows a similar rank, an additional density measure is used for distinguishing the difference between them. The SDE technique is used to describe the additional density (crowding distance) of i.

4.3. A Fine-Grained Elitism Selection Mechanism

Elitism selection is a critical process in multi-objective optimization. It helps in the determination of the best fit solution for the next generation. The selection process affects the Pareto front output. If a better solution is lost in the selection, it cannot be found again. The selection of a better solution that dominates other solutions helps in removing worse solutions. In this study, a modified fine-grained elitism selection method that utilizes the shift-based density estimation (SDE) approach is proposed to improve population diversity. Similarly, to the non-dominated solution selection mechanism, the proposed method also selects the non-dominated solution as follows:

$$P^*_{I_C+1} = \left\{ CP \middle| CP\ P_n\ U\ P^*_{I_C+1} \right\} \wedge F(CP) < 1$$

As the external population size (m) is fixed, the non-dominated solutions are copied to the external population set $(P^*_{I_C+1})$ until the current size of the external population is less than m. In contrast, the non-dominated solutions $(P^*_{I_C+1} - m)$ are discarded from the external population set. In order to discard solutions and maintain diversity, the proposed method deletes the solution with the most crowded region computed using the SDE approach, as described in Section 3.3. The steps used to compute the shared crowding distance are as follows.

1. The distance between two adjacent particles P_{I_c} and $P^*_{I_C+1}$ is computed as Algorithm 1.
2. For each particle, an additional density estimation (shared crowding distance) is computed as follows:

$$\sigma_{crowd} = \frac{SDE\ (P^*_{I_c})}{|m|} \tag{16}$$

When the size of the non-dominated solution size exceeds external population size (i.e., $P^*_{I_c} > m$), the proposed method selects the k^{th} charged particle from the external population $(P^*_{I_c})$ using Equation (17) and deletes it. This process is performed iteratively until $|P^*_{I_c}| = m$.

$$\sigma_{k_{crowd}} > \frac{\sum_{i=1}^{|P^*_{I_c}|} \sigma_{crowd(P^*_{I_c})_i}}{|m|} \tag{17}$$

Here, the solution (charged particle) whose density is greater than the average shared crowing distance of all non-dominated solutions in the external population is considered as the worst solution and is deleted from the external population set. Then, the average crowding distance is recomputed and used in a similar manner until $P^*_{I_c} = m$.

Table 2. Symbols used in the proposed algorithm.

Symbol	Definition
n	Initial searching population size
P_n	Searching population
m	External population size
P_m	External population
$CP_i^D\,(T)$	Position of ith charged particle (candidate solution) in D^{th} dimension at time T
$vel_i^D\,(T)$	velocity of ith charged particle in D^{th} dimension at time T
ND_{C_p}	Non-dominated set of charged particles (candidate solution)
$Fitness\,(T)$	Objective fitness function
$Best\,(T)$	Charged particle with best fitness at time T
$Worst\,(T)$	Charged particle with worst fitness at time T
K	Coulomb's constant
$Q_i(T)$	Total charge on a ith charged particle at time T
$q_i(T)$	Small charge of ith charged particle to determine the total charge acting on ith charged particle
F_{ij}	Force exerted by jth charge particle on ith charge particle
SDE	Shift based density estimation
I_{cmax}	Maximum number of iterations
$d_{P_{Ic_i},P_{Ic_{i+1}}}$	Distance between two non-dominated solutions (P_{Ic_i} and $P_{Ic_{i+1}}$)
σ_{crowd}	Shared crowding distance for each CP
σ_{kcrowd}	Shared crowding distance of k^{th} CP chosen for deletion from the external population set

5. Experimental Results and Discussion

This section is further divided into three sub-sections. Section 5.1 discusses the performance comparison of the AEFA with existing evolutionary approaches. Section 5.2 gives a performance comparison of the proposed algorithm with existing multi-objective optimization algorithms, and Section 5.3 presents a sensitivity analysis of the proposed algorithm.

5.1. Performance Comparison of the AEFA With Existing Evolutionary Approaches

At first, the performance of AEFA is evaluated on 10 benchmark functions. These benchmarks are taken as three unimodal (UM), three low-dimensional multimodal (LDMM), and four high-dimensional multimodal (HDMM) functions. All these functions belong to the minimization problem. The description of the benchmark functions is given in Table 3. Then, the performance of the AEFA is compared with existing evolutionary approaches. For performance comparison, four statistical performance indices are considered: average best-so-far solution, mean best-so-far solution, the variance of the best-so-far solution, and average run-time. During each iteration, the algorithm updates the best-so-far solution and records running time. The performance indices are computed by analyzing the obtained best-so-far solution and running time.

5.1.1. Parameter Setting

For experimental analysis, the parameters, i.e., population size (P_n, P_m), the maximum Coulomb's constant (K_0), and the maximum number of iterations (I_{cmax}), are initialized in Table 4.

5.1.2. Results and Discussion

The performance of AEFA is compared with existing evolutionary optimization algorithms: backtracking search algorithm (BSA) [36], cuckoo search algorithm (CK) [36], artificial bee colony (ABC) algorithm [36], and gravitational search algorithm (GSA) [36]. The results are demonstrated in Table 5 and Figure 1. Figure 1 presents the difference between the AEFA and other existing algorithms based on the values obtained by performance measures on benchmark functions. For a better representation, all the results (except Figure 1d–h) are formulated in logarithm scale (base 10). A larger logarithmic value represents the minimum values of performance measures obtained for different functions.

The results in Table 5 and Figure 1 demonstrate that the AEFA obtained minimum values for all UM and LDMM functions. The obtained values competed with the values of GSA and outperformed the value of ABC, CK, and BSA, which reveals that AEFA maintains a balance between exploration and exploitation and performs better in comparison to existing approaches. However, for HDMM (except benchmark F10) functions, AEFA performs worse as compared to ABC, and BSA, which implies that the AEFA suffers from the loss of diversity resulting in premature convergence in solving complex HDMM problems. It is concluded from results that the original AEFA faces challenges in solving higher-dimensional multimodal problems.

Table 3. Benchmark functions used for performance comparison of evolutionary algorithms.

Benchmark Function	Type	Variable Bound	Objective Function	Dimension(s)
F1	UM	$-100 \le x_i \le 100$ $i = 1, 2, 3 \ldots n$	$F(x) = \sum_{i=1}^{n} x_i^2$	30
F2	UM	$-10 \le x_i \le 10$ $i = 1, 2 \ldots n$	$F(x) = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} x_i$	30
F3	UM	$-100 \le x_i \le 100$ $i = 1, 2 \ldots n$	$F(x) = \max \{\lvert x_i \rvert, 1 \le i \le n\}$	30
F4	LDMM	$0 \le x_i \le 1$ $i = 1, 2 \ldots n$	$F(x) = -\sum_{i=1}^{4} c_i \exp(-\sum_{j=1}^{6} a_{ij} \left(x_j - p_{ij}\right)^2)$	6
F5	LDMM	$0 \le x_i \le 100$ $i = 1, 2 \ldots n$	$F(x) = -\sum_{i=1}^{7} \left[(X - a_i)(X - a_i)^T + c_i\right]^{-1}$	4
F6	LDMM	$0 \le x_i \le 100$ $i = 1, 2 \ldots n$	$F(x) = -\sum_{i=1}^{10} \left[(X - a_i)(X - a_i)^T + c_i\right]^{-1}$	4
F7	HDMM	$-500 \le x_i \le 500$ $i = 1, 2 \ldots n$	$F(x) = \sum_{i=1}^{n} -x_i \sin\left(\sqrt{\lvert x_i \rvert}\right)$	30
F8	HDMM	$-5.12 \le x_i \le 5.12$ $i = 1, 2 \ldots n$	$F(x) = \sum_{i=1}^{n} \left[x_i^2 - 10cos(2\pi x_i) + 10\right]$	30
F9	HDMM	$-600 \le x_i \le 600$ $i = 1, 2 \ldots n$	$F(x) = \frac{1}{40000} \sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	30
F10	HDMM	$-50 \le x_i \le 50$ $i = 1, 2 \ldots n$	$F(x) = 0.1\{\sin(3\pi x_i) + \sum_{i=1}^{n} (x_i - 1)^2 + 1$ $+ \sin^2(3\pi x_i + 1)] + (x_n - 1)^2$ $* [1 + \sin^2(2\pi x_n)]\} + \sum_{i=1}^{u} u(x_i, 10, 100, 4),$ $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & x_i > a \\ k(-x_i - a)^m & x_i < -a \end{cases}$	30

Table 4. Parameters used to evaluate AEFA.

Description	Parameter	Value
Population size	P_n, P_m	50
Initial value used in Coulomb's constant	K_0	100
Maximum number of iterations	I_{cmax}	300 for $F4 - F6$ and 1000 for the rest of the benchmark functions

Table 5. Performance comparison between AEFA and existing evolutionary optimization algorithms.

Benchmark	Optimization Algorithm	Statistical Performance Indices			
		Average Best	Mean Best	Variance Best	Average Run Time
F1	AEFA	$\mathbf{3.21 \times 10^{-16}}$	$\mathbf{2.54 \times 10^{-18}}$	$\mathbf{3.98 \times 10^{-16}}$	9.48×10^{0}
	GSA	3.33×10^{-14}	2.68×10^{-17}	4.79×10^{-14}	9.71×10^{0}
	ABC	4.20×10^{-9}	1.52×10^{-9}	8.40×10^{-9}	1.63×10^{1}
	CK	9.26×10^{-3}	9.44×10^{-3}	2.35×10^{-2}	3.88×10^{0}
	BSA	4.87×10^{-3}	3.28×10^{-3}	4.53×10^{-3}	$\mathbf{1.28 \times 10^{0}}$
F2	AEFA	$\mathbf{1.26 \times 10^{-8}}$	$\mathbf{2.00 \times 10^{-8}}$	$\mathbf{1.10 \times 10^{-8}}$	9.72×10^{0}
	GSA	1.55×10^{-8}	2.08×10^{-8}	1.16×10^{-8}	9.95×10^{0}
	ABC	3.22×10^{-6}	2.99×10^{-6}	1.35×10^{-6}	1.75×10^{1}
	CK	1.62×10^{0}	1.30×10^{0}	8.15×10^{-1}	4.12×10^{0}
	BSA	1.73×10^{-2}	1.49×10^{-2}	9.11×10^{-3}	$\mathbf{1.98 \times 10^{0}}$
F3	AEFA	$\mathbf{2.68 \times 10^{-10}}$	$\mathbf{2.98 \times 10^{-9}}$	$\mathbf{8.78 \times 10^{-11}}$	9.01×10^{0}
	GSA	3.34×10^{-9}	3.13×10^{-9}	9.05×10^{-10}	8.98×10^{0}
	ABC	6.02×10^{1}	6.15×10^{1}	9.39×10^{0}	1.71×10^{1}
	CK	3.21×10^{0}	3.22×10^{0}	9.39×10^{0}	3.92×10^{0}
	BSA	4.52×10^{0}	4.65×10^{0}	1.05×10^{0}	$\mathbf{1.74 \times 10^{0}}$
F4	AEFA	$\mathbf{-3.30 \times 10^{0}}$	$\mathbf{-3.30 \times 10^{0}}$	$\mathbf{4.38 \times 10^{-16}}$	2.44×10^{0}
	GSA	-3.32×10^{0}	-3.32×10^{0}	4.08×10^{-16}	2.24×10^{0}
	ABC	-3.32×10^{0}	-3.32×10^{0}	2.33×10^{-9}	7.24×10^{0}
	CK	-3.32×10^{0}	-3.32×10^{0}	4.80×10^{-5}	1.32×10^{0}
	BSA	-3.32×10^{0}	-3.32×10^{0}	1.48×10^{-4}	$\mathbf{6.74 \times 10^{-1}}$
F5	AEFA	$\mathbf{-1.02 \times 10^{1}}$	$\mathbf{-1.02 \times 10^{1}}$	$\mathbf{2.68 \times 10^{-16}}$	2.34×10^{0}
	GSA	-1.04×10^{1}	-1.04×10^{1}	2.97×10^{-15}	2.29×10^{0}
	ABC	-1.04×10^{1}	-1.04×10^{1}	1.38×10^{-3}	8.23×10^{0}
	CK	-1.04×10^{1}	-1.04×10^{1}	2.69×10^{-3}	1.57×10^{0}
	BSA	-1.04×10^{1}	-1.04×10^{1}	3.50×10^{-2}	$\mathbf{1.04 \times 10^{0}}$
F6	AEFA	$\mathbf{-1.05 \times 10^{1}}$	$\mathbf{-1.05 \times 10^{1}}$	$\mathbf{1.28 \times 10^{-15}}$	2.40×10^{0}
	GSA	-1.05×10^{1}	-1.05×10^{1}	1.47×10^{-15}	2.46×10^{0}
	ABC	-1.05×10^{1}	-1.05×10^{1}	7.06×10^{-4}	9.34×10^{0}
	CK	-1.05×10^{1}	-1.05×10^{1}	2.97×10^{-3}	2.97×10^{-3}
	BSA	-1.05×10^{1}	-1.05×10^{1}	6.17×10^{-2}	$\mathbf{9.07 \times 10^{-1}}$
F7	AEFA	-2.79×10^{3}	-2.68×10^{3}	4.18×10^{2}	2.98×10^{0}
	GSA	-2.84×10^{3}	-2.84×10^{3}	4.22×10^{2}	3.01×10^{0}
	ABC	-1.08×10^{4}	-1.07×10^{4}	2.97×10^{2}	5.86×10^{0}
	CK	-8.77×10^{3}	-8.79×10^{3}	$\mathbf{2.44 \times 10^{2}}$	4.60×10^{0}
	BSA	$\mathbf{-1.11 \times 10^{4}}$	$\mathbf{-1.11 \times 10^{4}}$	2.49×10^{2}	$\mathbf{2.57 \times 10^{0}}$
F8	AEFA	1.28×10^{1}	1.28×10^{1}	3.56×10^{0}	1.06×10^{1}
	GSA	1.53×10^{1}	1.54×10^{1}	4.01×10^{0}	1.10×10^{1}
	ABC	$\mathbf{1.06 \times 10^{-1}}$	$\mathbf{2.24 \times 10^{-6}}$	$\mathbf{3.05 \times 10^{-1}}$	1.89×10^{1}
	CK	8.38×10^{1}	8.40×10^{1}	1.05×10^{1}	4.80×10^{0}
	BSA	2.84×10^{1}	2.77×10^{1}	4.18×10^{0}	$\mathbf{2.62 \times 10^{0}}$
F9	AEFA	4.08×10^{0}	3.19×10^{0}	1.80×10^{0}	9.70×10^{0}
	GSA	4.15×10^{0}	3.56×10^{0}	1.81×10^{0}	9.74×10^{0}
	ABC	$\mathbf{4.01 \times 10^{-4}}$	$\mathbf{2.81 \times 10^{-8}}$	$\mathbf{1.79 \times 10^{-3}}$	2.44×10^{1}
	CK	1.03×10^{-1}	9.68×10^{-2}	9.6×10^{-2}	5.32×10^{0}
	BSA	3.21×10^{-2}	1.87×10^{-2}	3.65×10^{-2}	$\mathbf{2.42 \times 10^{0}}$
F10	AEFA	$\mathbf{5.00 \times 10^{-4}}$	$\mathbf{1.96 \times 10^{-18}}$	$\mathbf{2.30 \times 10^{-3}}$	1.12×10^{1}
	GSA	5.49×10^{-4}	2.10×10^{-18}	2.46×10^{-3}	1.07×10^{1}
	ABC	2.10×10^{-3}	1.17×10^{-3}	2.45×10^{-3}	1.27×10^{1}
	CK	1.50×10^{-1}	1.49×10^{-1}	6.82×10^{-2}	7.82×10^{0}
	BSA	4.98×10^{-4}	4.37×10^{-4}	3.60×10^{-4}	$\mathbf{4.35 \times 10^{0}}$

5.2. Performance Comparison of the Proposed Algorithm with Existing Multi-Objective Optimization Algorithms

As shown in Table 5, the AEFA faces challenges in solving MOOP. These challenges are addressed through the proposed algorithm. The proposed algorithm is evaluated with six benchmark functions.

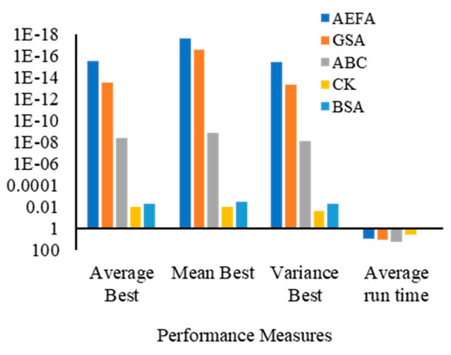
The description of benchmark functions is presented in Table 6. The performance of the proposed algorithm is compared with the existing MOOP based on four performance measures: generational distance metric (GD) [2], diversity metric (DM) [3], converge metric (CM) [3], and spacing metric (SM) [39]. DM measures the extent of spread attained in the obtained optimal solution. CM measures the convergence to the obtained optimal Pareto front. GD measures the proximity of the optimal solution to the Pareto optimal front. SM demonstrates how evenly the optimal solutions are distributed among themselves. Further, the performance of the proposed algorithm is compared with existing MOOPs in terms of recombination and mutation operators.

5.2.1. Parameter Setting

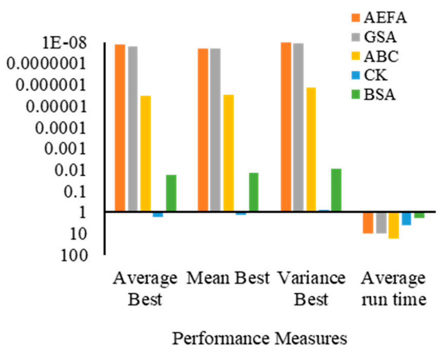
For experimental analysis, the parameters, i.e., initial population size (P_n), external population size (P_m), the maximum Coulomb's constant (K_0), the maximum number of iterations (I_{cmax}), initial crossover probability (P_{co}), final crossover probability (P_{c1}), initial mutation probability (P_{m0}), and final mutation probability (P_{m1}), are initialized and presented in Table 7.

5.2.2. Results and Discussion

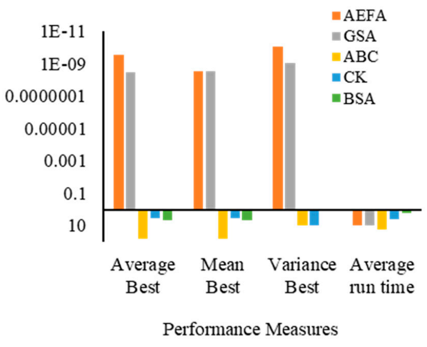
The performance of the proposed algorithm is analyzed in three steps: (1) the proposed algorithm is evaluated on SCH, FON, and ZDT benchmarks and then compared with NSGA II [3], NSPSO [7], BCMOA [40], and SPGSA [36] based on CM, DM, and GD metrics; the (2) proposed algorithm is evaluated on MOP5 and MOP6 benchmarks and then compared with NSGSA [36], MOGSA [36], SMOPSO [36], MOGA II [36], and SPGSA [36] based on GD and SM metrics; and (3) the efficacy of using the recombination operator is validated by evaluating the proposed algorithm on six benchmarks and then compared with the existing MOOPs. Experiments are performed 10 times on each benchmark function, and results are recorded in terms of mean and variance, contributing to the robustness of the algorithm. The results are presented in Tables 8–12 and Figures 2–6. For better representation, all the results in Figures 2–6 are prepared using logarithmic scale, where a high logarithmic value corresponds to a minimum output value (mean and variance). Results in Table 8 and Figure 2 demonstrate that the proposed algorithms obtained minimum values of CM metric as compared to NSGSA [36], MOGSA [36], SMOPSO [36], MOGA II [36], and SPGSA [36] which proves that the proposed algorithm ensures better convergence and keeps a good balance between exploration and exploitation of low-dimensional as well as high-dimensional problems. Figure 3 and Table 9 show that the proposed algorithm obtained the minimum value of variance (in DM metric) for all the benchmark functions as compared to NSGA II, NSPSO, BCMOA, and SPGSA, which validates the efficacy and robustness of the proposed algorithm. Results in Table 10 (GD metric) and Figure 4 demonstrate that the proposed algorithm performed better in comparison to existing NSGA II, NSPSO, BCMOA, and SPGSA. GD metric validates the proximity of obtained optimal solution to the optimal Pareto front, thereby ensuring the effectiveness of the proposed algorithm. Table 11 and Figure 5 demonstrate that the proposed algorithm achieves better values for GD and SM metrics than the compared algorithm. GD values of the proposed algorithm show more accurate results than the compared algorithm, and SM values show that the proposed algorithm attains a better convergence accuracy for all benchmark functions, which shows that the proposed algorithm can produce uniformly distributed, non-dominated optimal solutions. Table 12 and Figure 6a–c show that the proposed algorithm is compared with SPEA2, SPGSA, SPGSA_NRM, and a self-variant of MOOP without utilizing BEX and PMO operators. Results indicate that involving BEX and PMO operators in the proposed algorithm significantly improves the algorithm optimization performance in comparison to the compared algorithm. Contrary, in the absence of these operators, the proposed algorithm suffers from premature convergence. So, it is concluded from all four metrics (Tables 8–11) and Table 12 that the proposed algorithm shows better efficacy, accuracy, and robustness in searching true Pareto fronts across the global search space in comparison to the existing MOOP algorithm. Further, involving BEX and PMO operators, the proposed algorithm maintains desirable diversity in searching the true Pareto front.



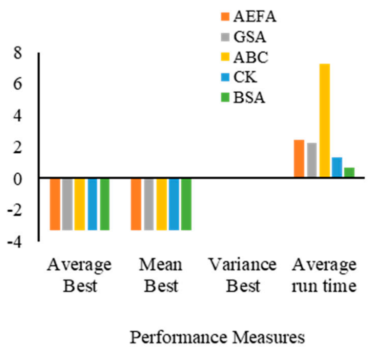
(a)



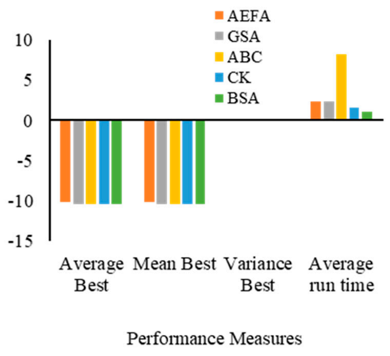
(b)



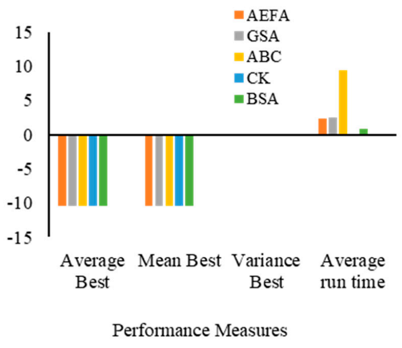
(c)



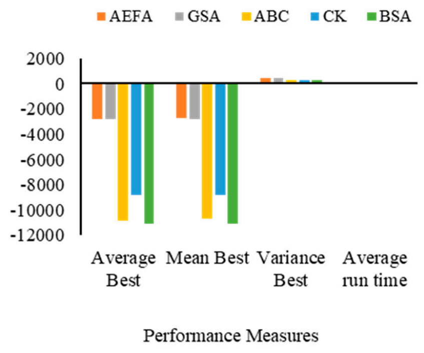
(d)



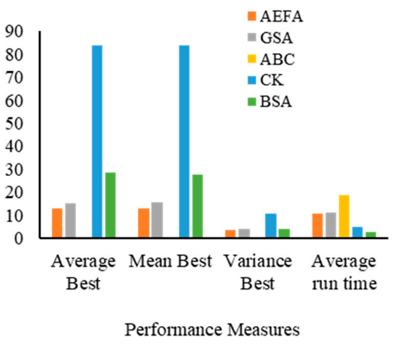
(e)



(f)

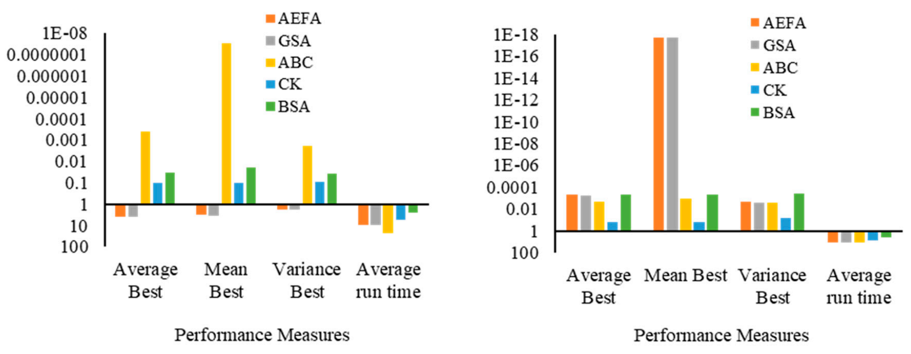


(g)



(h)

Figure 1. *Cont.*



(i) (j)

Figure 1. Difference among algorithms based on the values obtained by performance measures on benchmark functions. (**a**) Performance measurement for F1 benchmark function. (**b**) Performance measurement for F2 benchmark function. (**c**) Performance measurement for F3 benchmark function. (**d**) Performance measurement for F4 benchmark function. (**e**) Performance measurement for F5 benchmark function. (**f**) Performance measurement for F6 benchmark function. (**g**) Performance measurement for F7 benchmark function. (**h**) Performance measurement for F8 benchmark function. (**i**) Performance measurement for F9 benchmark function. (**j**) Performance measurement for F10 benchmark function.

Table 6. Benchmark functions used for MOOP performance comparison.

Benchmark Function	Type	Variable Bound	Objective Function	Dimension(s)
SCH [3]	Bi-Objective (Low dimension)	$-3 \le x_i \le 3$	Minimize $F_1(x) = x^2$ Minimize $F_2(x) = (x-2)^2$	1
FON [3]	Bi-Objective (Low dimension)	$-4 \le x_i \le 4$ $i = 1, 2 \ldots n$	Minimize $F_1(x) = 1 - \exp\left(-\sum_{i=1}^{n}\left(x_i - \frac{1}{\sqrt{n}}\right)^2\right)$ Minimize $F_2(x) = 1 - \exp\left(-\sum_{i=1}^{n}\left(x_i + \frac{1}{\sqrt{n}}\right)^2\right)$	3
ZDT1 [3]	Bi-Objective (High dimension)	$0 \le x_i \le 1$ $i = 1, 2 \ldots n$	Minimize $F_1(x) = x_1$ Minimize $F_2(x) = g\left(1 - \sqrt{\frac{f1}{g}}\right)$, $g = 1 + 9\sum_{i=2}^{n}\left(\frac{x_i}{n-1}\right)$	30
ZDT2 [3]	Bi-Objective (High dimension)	$0 \le x_i \le 1$ $i = 1, 2 \ldots n$	Minimize $F_1(x) = x_1$ Minimize $F_2(x) = g\left(1 - \left(\frac{f1}{g}\right)^2\right)$, $g = 1 + 9\sum_{i=2}^{n}\left(\frac{x_i}{n-1}\right)$	30
MOP5 [5]	Tri-Objective (Low dimension)	$-30 \le x, y <= 30$	Minimize $F_1(x, y) = \frac{1}{2}\left(x^2 + y^2\right) Sin\left(x^2 + y^2\right)$ Minimize $F_2(x, y) = \frac{1}{8}(3x - 2y + 4) + \frac{(x-y+1)^2}{27} + 15$ Minimize $F_3(x, y) = \frac{1}{x^2+y^2+1} - 1.1e^{-}\left(x^2 + y^2\right)$	2
MOP6 [5]	Bi-Objective (Low dimension)	$0 \le x, y <= 1$	Minimize $F_1(x, y) = x$ Minimize $F_2(x, y) = (1 + 10y)$ $*\left[1 - \left(\frac{x}{1+10y}\right)^2 - \left(\frac{x}{1+10y}\right) sin(8\pi x)\right]$	2
DTLZT2 [13]	Tri-Objective (High dimension)	$0 \le x_i \le 1$ $i = 1, 2 \ldots n$	Minimize $F_1(x) = Cos\left(\frac{\pi}{2}x_1\right)Cos\left(\frac{\pi}{2}x_2\right)(1 + g(x))$ Minimize $F_2(x) = Cos\left(\frac{\pi}{2}x_1\right)Sin\left(\frac{\pi}{2}x_2\right)(1 + g(x))$ Minimize $F_3(x) = Sin\left(\frac{\pi}{2}x_2\right)(1 + g(x))$, where $g = \sum_{i=3}^{n}(x_i - 0.5)^2$	12

Table 7. Parameters used in the proposed algorithm.

Description	Parameter	Value
Population (charged particles) size	P_n	100
External Population size	P_m	100 for SCH, FON, ZDT 800 for MOP functions
Initial value used in Coulomb's constant	K_0	100
The maximum number of iterations	I_{cmax}	100 for SCH and FON functions 250 for ZDT and MOP functions
Initial crossover probability	P_{co}	1.0
Final crossover probability	P_{c1}	0.0
Initial mutation probability	P_{m0}	0.01
Final mutation probability	P_{m1}	0.001

Table 8. Performance comparison of the proposed algorithm with existing MOOPs based on SCH, FON, and ZDT functions using the CM metric.

Benchmark Function	Parameters	Algorithm				
		Proposed Algorithm	SPGSA	BCMOA	NSGA II	NSPSO
SCH	Mean	$\mathbf{1.48 \times 10^{-1}}$	1.65×10^{-1}	7.60×10^{-1}	3.8×10^{-1}	8.6×10^{-1}
	Variance	1.22×10^{-4}	1.3×10^{-4}	1.00×10^{-3}	1.00×10^{-3}	8.60×10^{-4}
FON	Mean	$\mathbf{1.52 \times 10^{-1}}$	1.61×10^{-1}	4.85×10^{-1}	4.14×10^{-1}	5.81×10^{-1}
	Variance	2.52×10^{-4}	2.4×10^{-4}	1.00×10^{-4}	9.80×10^{-4}	3.16×10^{-2}
ZDT1	Mean	$\mathbf{1.5 \times 10^{-1}}$	1.61×10^{-1}	5.98×10^{-1}	4.06×10^{-1}	6.38×10^{-1}
	Variance	1.42×10^{-4}	1.4×10^{-4}	4.10×10^{-3}	1.26×10^{-3}	2.69×10^{-3}
ZDT2	Mean	$\mathbf{1.61 \times 10^{-1}}$	1.63×10^{-1}	6.89×10^{-1}	4.39×10^{-1}	5.80×10^{-1}
	Variance	1.46×10^{-4}	1.5×10^{-4}	8.13×10^{-3}	1.19×10^{-3}	1.08×10^{-3}

Table 9. Performance comparison of the proposed algorithm with existing MOOPs based on SCH, FON, and ZDT functions using the DM metric.

Benchmark Function	Parameters	Algorithm				
		Proposed Algorithm	SPGSA	BCMOA	NSGA II	NSPSO
SCH	Mean	$\mathbf{2.75 \times 10^{-3}}$	$\mathbf{3.24 \times 10^{-3}}$	3.28×10^{-3}	3.14×10^{-3}	3.40×10^{-1}
	Variance	1.89×10^{-8}	2.98×10^{-8}	2.16×10^{-8}	4.64×10^{-8}	7.40×10^{-4}
FON	Mean	$\mathbf{1.55 \times 10^{-3}}$	$\mathbf{1.72 \times 10^{-3}}$	2.77×10^{-3}	2.36×10^{-3}	2.84×10^{-1}
	Variance	1.30×10^{-8}	1.10×10^{-8}	3.41×10^{-8}	1.24×10^{-8}	9.04×10^{-2}
ZDT1	Mean	$\mathbf{1.02 \times 10^{-3}}$	$\mathbf{1.17 \times 10^{-3}}$	1.19×10^{-3}	4.02×10^{-3}	3.81×10^{-1}
	Variance	1.98×10^{-9}	2.66×10^{-9}	3.41×10^{-8}	3.14×10^{-7}	3.22×10^{-3}
ZDT2	Mean	$\mathbf{7.90 \times 10^{-5}}$	$\mathbf{8.06 \times 10^{-4}}$	8.37×10^{-4}	2.71×10^{-3}	4.59×10^{-1}
	Variance	4.22×10^{-11}	6.37×10^{-10}	1.25×10^{-8}	7.69×10^{-8}	3.24×10^{-3}

Table 10. Performance comparison of the proposed algorithm with existing MOOPs based on SCH, FON, and ZDT functions using the GD metric.

Benchmark Function	**Parameters**	**Algorithm**				
		Proposed Algorithm	**SPGSA**	**BCMOA**	**NSGA II**	**NSPSO**
SCH	Mean Variance	$\mathbf{2.69 \times 10^{-4}}$ 2.02×10^{-10}	3.78×10^{-4} 3.06×10^{-10}	3.78×10^{-4} 1.57×10^{-10}	3.68×10^{-4} 3.4×10^{-10}	4.5×10^{-4} 2.6×10^{-8}
FON	Mean Variance	$\mathbf{4.05 \times 10^{-5}}$ 3.12×10^{-11}	2.13×10^{-4} 3.06×10^{-10}	3.62×10^{-4} 8.03×10^{-10}	2.94×10^{-4} 4.2×10^{-10}	3.6×10^{-4} 1.8×10^{-9}
ZDT1	Mean Variance	$\mathbf{3.12 \times 10^{-5}}$ 4.37×10^{-11}	2.4×10^{-4} 4.76×10^{-10}	2.02×10^{-4} 3.64×10^{-9}	5.56×10^{-4} 7.95×10^{-9}	4.3×10^{-4} 3.4×10^{-8}
ZDT2	Mean Variance	$\mathbf{6.16 \times 10^{-5}}$ 1.14×10^{-11}	9.71×10^{-5} 1.56×10^{-11}	1.00×10^{-4} 1.97×10^{-10}	4.18×10^{-4} 9.64×10^{-9}	3.5×10^{-4} 1.6×10^{-8}

Table 11. Performance comparison of the proposed algorithm with existing MOOPs based on MOP5 and MOP6 functions using the GD/SM metric.

Benchmark Function	**Metric**	**Algorithm**					
		Proposed Algorithm	**SPGSA**	**NSGSA**	**MOGSA**	**SMOPSO**	**MOGA II**
MOP5	GD SM	$\mathbf{2.86 \times 10^{-6}}$ $\mathbf{4.86 \times 10^{-3}}$	3.9×10^{-5} 2.18×10^{-2}	1.0×10^{-4} 3.4×10^{-2}	1.2×10^{-3} 1.8×10^{-1}	$\mathbf{1.11 \times 10^{-2}}$ 3.96×10^{-1}	7.2×10^{-1} 2.0×10^{-1}
MOP6	GD SM	$\mathbf{7.02 \times 10^{-7}}$ $\mathbf{4.86 \times 10^{-3}}$	5.01×10^{-6} 5.34×10^{-2}	3.0×10^{-5} 5.7×10^{-2}	2.45×10^{-5} 1.80×10^{-1}	$\mathbf{2.98 \times 10^{-4}}$ 1.16×10^{-1}	1.00×10^{-3} 9.42×10^{-1}

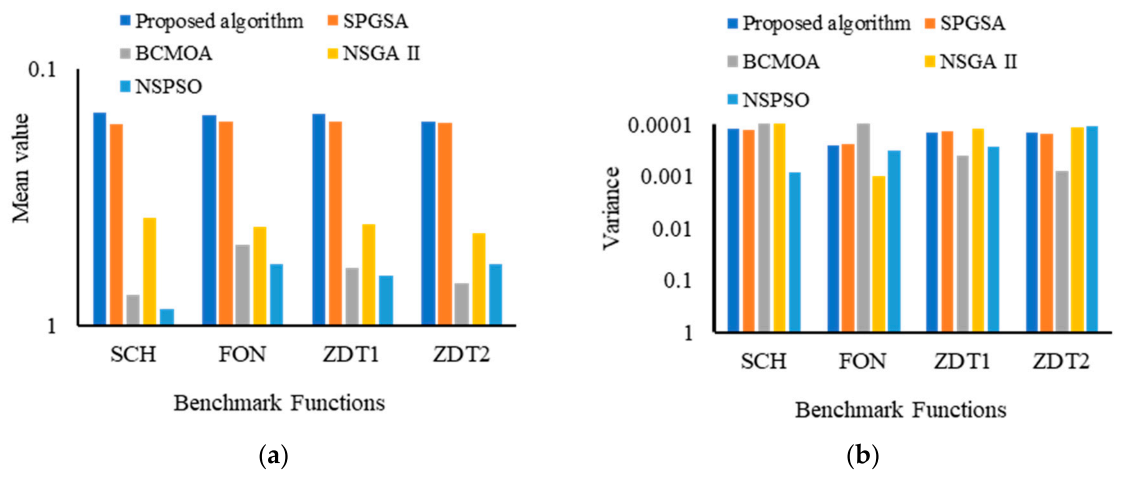
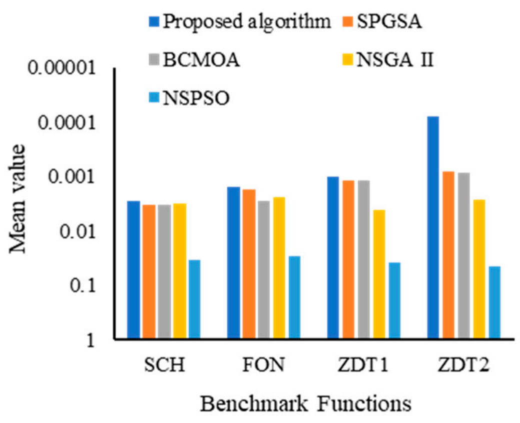


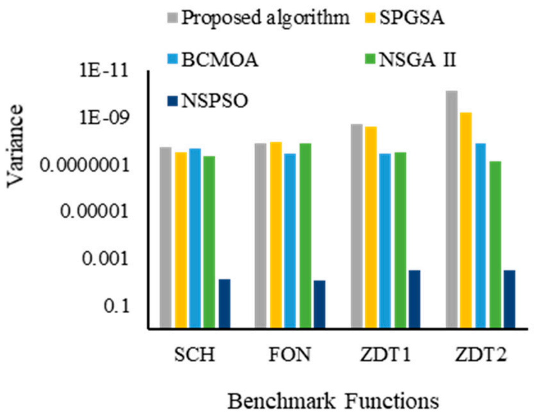
Figure 2. Performance comparison of the proposed algorithm with existing MOOPs based on SCH, FON, and ZDT functions using the CM metric. (**a**) Performance measurement based on mean value; (**b**) performance measurement based on variance.

Table 12. Performance comparison of the proposed algorithm with existing MOOPs based on recombination and mutation operators.

Performance Metric	Benchmark Functions	Proposed Algorithm	Proposed Algorithm (without *BEX* and *PMO* Operator)	SPEA2	SPGSA	SPGSA_ES
CM metric	SCH	$\mathbf{2.87 \times 10^{-3}}$	3.11×10^{-3}	3.31×10^{-3}	3.21×10^{-3}	3.23×10^{-3}
	FON	$\mathbf{1.45 \times 10^{-3}}$	1.58×10^{-3}	1.86×10^{-3}	1.60×10^{-3}	1.67×10^{-3}
	ZDT1	$\mathbf{1.07 \times 10^{-3}}$	1.17×10^{-3}	1.31×10^{-3}	1.17×10^{-3}	1.43×10^{-3}
	ZDT2	$\mathbf{5.08 \times 10^{-4}}$	7.28×10^{-4}	8.68×10^{-4}	7.84×10^{-4}	9.37×10^{-4}
	MOP5	$\mathbf{4.18 \times 10^{-3}}$	5.11×10^{-3}	4.01×10^{-2}	3.65×10^{-2}	6.16×10^{-2}
	MOP6	$\mathbf{6.18 \times 10^{-6}}$	4.11×10^{-5}	9.86×10^{-5}	9.84×10^{-5}	1.19×10^{-4}
	DTLZ2	$\mathbf{1.88 \times 10^{-4}}$	5.01×10^{-3}	7.97×10^{-3}	8.01×10^{-3}	8.33×10^{-3}
GD metric	SCH	$\mathbf{4.08 \times 10^{-5}}$	4.98×10^{-5}	4.03×10^{-4}	3.76×10^{-4}	3.78×10^{-4}
	FON	$\mathbf{1.81 \times 10^{-4}}$	1.95×10^{-4}	2.37×10^{-4}	1.98×10^{-4}	2.01×10^{-4}
	ZDT1	$\mathbf{2.31 \times 10^{-4}}$	2.43×10^{-4}	2.63×10^{-4}	2.50×10^{-4}	2.60×10^{-4}
	ZDT2	$\mathbf{6.25 \times 10^{-5}}$	7.11×10^{-5}	9.41×10^{-5}	9.36×10^{-5}	9.78×10^{-5}
	MOP5	$\mathbf{5.28 \times 10^{-4}}$	2.70×10^{-3}	1.90×10^{-2}	1.76×10^{-2}	2.13×10^{-2}
	MOP6	$\mathbf{1.12 \times 10^{-5}}$	1.32×10^{-5}	1.32×10^{-5}	1.30×10^{-5}	1.55×10^{-5}
	DTLZ2	$\mathbf{3.77 \times 10^{-4}}$	4.01×10^{-4}	1.09×10^{-3}	1.10×10^{-3}	1.18×10^{-3}
SM metric	SCH	$\mathbf{5.48 \times 10^{-3}}$	6.11×10^{-3}	20.08×10^{-2}	1.36×10^{-2}	1.05×10^{-2}
	FON	$\mathbf{3.12 \times 10^{-3}}$	4.22×10^{-3}	3.66×10^{-3}	3.00×10^{-3}	3.51×10^{-3}
	ZDT1	$\mathbf{7.81 \times 10^{-4}}$	8.11×10^{-4}	3.66×10^{-3}	3.18×10^{-3}	3.21×10^{-3}
	ZDT2	$\mathbf{5.23 \times 10^{-4}}$	5.89×10^{-4}	3.61×10^{-3}	3.11×10^{-3}	3.58×10^{-3}
	MOP5	$\mathbf{6.08 \times 10^{-3}}$	6.28×10^{-3}	6.09×10^{-1}	6.10×10^{-1}	6.15×10^{-1}
	MOP6	$\mathbf{2.90 \times 10^{-3}}$	3.09×10^{-3}	6.01×10^{-3}	3.10×10^{-3}	3.47×10^{-3}
	DTLZ2	$\mathbf{7.16 \times 10^{-3}}$	8.01×10^{-3}	5.74×10^{-2}	5.87×10^{-2}	2.3×10^{-2}

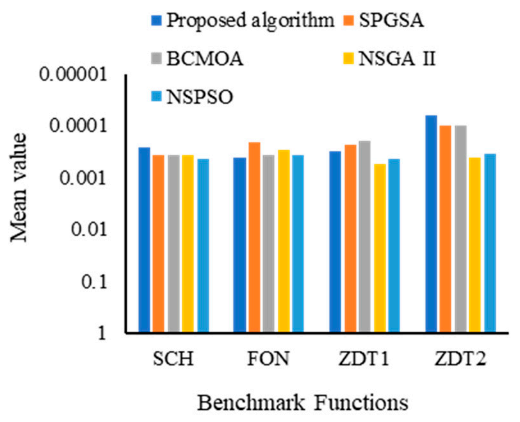


(**a**)

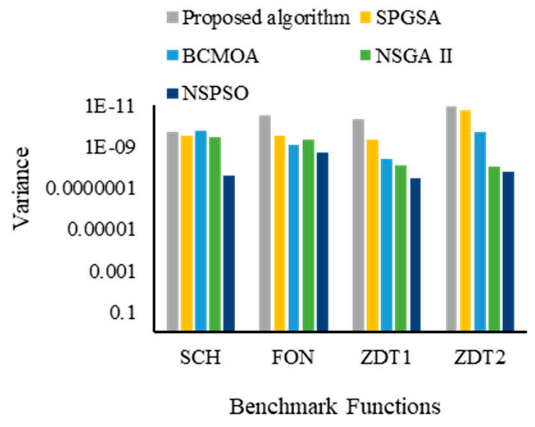


(**b**)

Figure 3. Performance comparison of the proposed algorithm with existing MOOPs based on SCH, FON, and ZDT functions using the DM metric (**a**). Performance measurement based on mean value; (**b**) performance measurement based on variance.

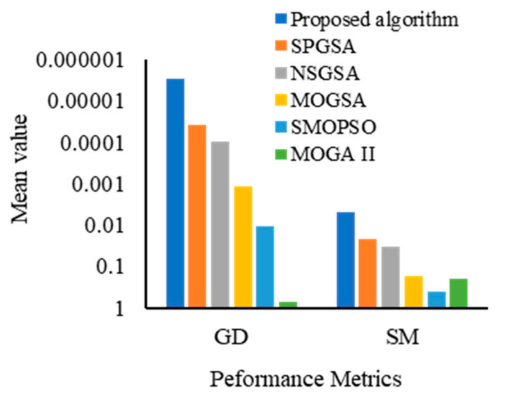


(**a**)

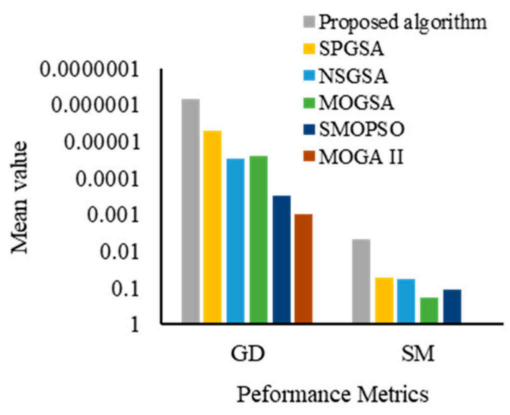


(**b**)

Figure 4. Performance comparison of the proposed algorithm with existing MOOPs based on SCH, FON, and ZDT functions using the GD metric. (**a**) Performance measurement based on mean value; (**b**) performance measurement based on variance.

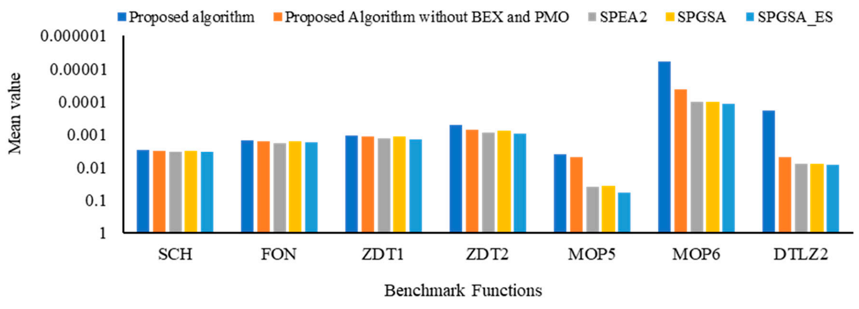


(**a**)

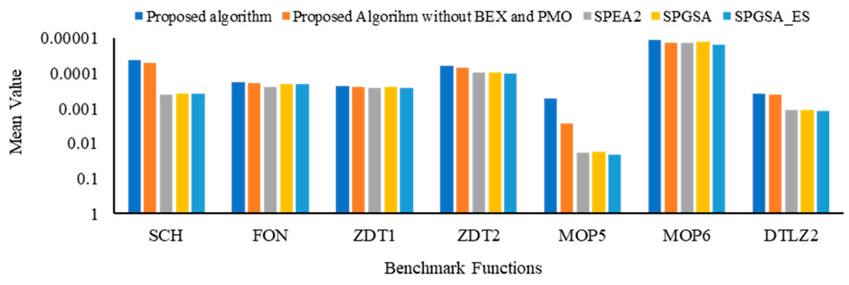


(**b**)

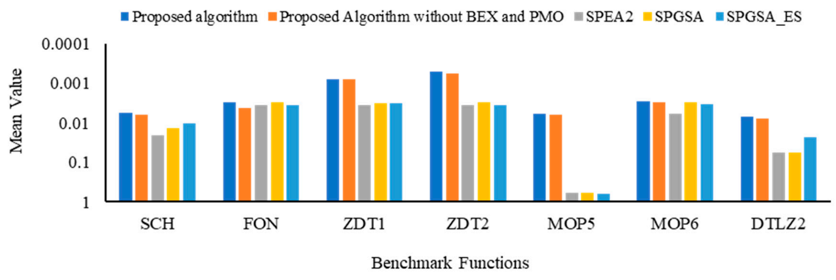
Figure 5. Performance comparison of the proposed algorithm with existing MOOPs based on MOP5 and MOP6 functions using the GD/SM metric. (**a**) Performance measurement for MOP5 benchmark function; (**b**) performance measurement for MOP6 benchmark function.



(**a**)



(**b**)



(**c**)

Figure 6. Performance comparison of the proposed algorithm with existing MOOPs based on recombination and mutation operators. (**a**) Performance comparison of the proposed algorithm with existing MOOPs based on recombination and mutation operators on the CM metric. (**b**) Performance comparison of the proposed algorithm with existing MOOPs based on recombination and mutation operators on the GD metric. (**c**) Performance comparison of the proposed algorithm with existing MOOPs based on recombination and mutation operators on the SM metric.

5.3. Sensitivity Analysis of the Proposed Algorithm

Sensitivity analysis is defined as the process that determines the influences of change in input variable on the robustness of outcome. In this paper, the robustness of the proposed algorithm is evaluated in two steps: (1) sensitivity analysis 1 and (2) sensitivity analysis 2. Sensitivity analysis 1 is performed for the different values of crossover operator (BEX), mutation operator (PMO), and initial value of coulomb's constant (K_0). Sensitivity analysis 2 is performed for the different values of initial population size and maximum number of iterations. The detailed description of the sensitivity analysis and related parameter settings is described in Table 13. Further, a combined performance score, described in Table 14, is used measure the performance of the proposed algorithm.

Table 13. Parameter settings for sensitivity analysis.

Analysis	Sensitivity Analysis Performed on	Parameter Setting	Setting of Variable Parameters	Number of Parameter Setting Combinations
Sensitivity Analysis 1	1. BEX Cross over 2. PMO Mutation 3. Initial value of Coulomb's constant (K_0)	1. Initial population size (P_n) = 100 2. External population (P_m) = 100 3. Maximum no. of iterations = 2000	1. Mutation = 0.01; 0.7; 0.9; 0.001 2. Crossover Rate = 0.7; 0.8; 0.9; 1.0 3. K_0 = 100, 300, 500	30
Sensitivity Analysis 2	Initial population Size	Mutation probability = 0.001	Population Size = 50; 100; 200; 400; 500	20
	Maximum no. of Iterations	Crossover probability = 1.0	Maximum no. of Iterations = 500; 1000; 2000	50

Table 14. Performance measure for sensitivity analysis.

Performance Measures	Overall Performance of Parameter Setting Combinations				
	Very Good	Good	Average	Poor	Very Poor
Combined performance Score	5.0–4.0	4.0–3.0	3.0–2.0	2.0–1.0	1.0–0.0

Results and Discussion

Sensitivity of mutation operator (PMO), crossover operator (BEX), initial population size, maximum number of iterations, and initial value of coulomb's constant (K_0) are analyzed using GD and SM metrics, and results are presented in Tables 15–19, respectively. Table 15 shows that performance of the PMO varied from "Average" to "Good", which means that selection of suitable values for mutation is required to achieve optimum results. Table 16 demonstrates that performance of the BEX varied from "Average" to "Very Good", which means that the sensitivity of BEX is low for the proposed algorithm. Table 17 demonstrates that performance of the population size varied from "Average" to "Very Good", which means that the sensitivity of the population is moderate for the proposed algorithm. Table 18 demonstrates that performance of maximum no. of iterations is "Good" for all experiments, which means that the proposed algorithm is less sensitive to maximum number of iterations. Table 19 demonstrates that performance of the initial value of Coulomb's constant (K_0) varies from "Good" to "Very Good", which means that the proposed algorithm is less sensitive to (K_0). It is concluded from all the tables (Tables 15–19) that the proposed algorithm works efficiently in all given scenarios, which shows its robustness.

Table 15. Sensitivity analysis of mutation parameter (PMO).

Value of Mutation Parameter	Value of Constant Parameters	Metrics		Combined Performance Score	Overall Performance
		GD	SM		
0.01	1. Initial population size = 100	3899	24.6	2.9	Average
0.07	2. External population size = 100	67	32.3	3.6	Good
0.09	3. Maximum no. of iterations = 2000	86	33.5	3.0	Average
0.001	4. Crossover value = 0.8	135	28.4	1.8	Very Poor
0.01	1. Population Size = 100	11,085	22	3.3	Good
0.07	2. Maximum no. of iterations = 2000	55	32	3.8	Good
0.09	3. Crossover value = 0.9	83	28.7	2.6	Average
0.001		71	32	3.4	Good

Table 16. Sensitivity analysis of crossover parameter (BEX).

Value of Crossover Parameter	Value of Constant Parameters	Metrics		Combined Performance Score	Overall Performance
		GD	SM		
0.7	1. Initial population size = 100	112	39	4.2	Very Good
0.8	2. External population size = 100, 800	45	28	3.2	Good
0.9	3. Maximum no. of iterations = 2000	60	35	3.0	Average
1.0	4. Mutation value = 0.001	36	30	2.6	Average
0.7	1. Initial population size = 100	4454	26	3.8	Good
0.8	2. External population size = 100, 800	10,121	23	3.6	Good
0.9	3. Maximum no. of iterations = 2000	3632	21	3.2	Good
1.0	4. Mutation value = 0.09	11,068	24	3	Good

Table 17. Sensitivity analysis of initial population size.

Initial Population Size	Value of Constant Parameters	Metrics		Combined Performance Score	Overall Performance
		GD	SM		
100	1. External population size = 100, 800	225	28	3.0	Average
200	2. Maximum no. of iterations = 1000	32	34	3.1	Good
400	3. Mutation value = 0.005	36	39	4.0	Very Good
500	4. Crossover Value = 0.9	38	30	3.6	Good
100	1. External population size = 100, 800	60	30	3.5	Good
200	2. Maximum no. of iterations = 1000	105	34	3.9	Good
400	3. Mutation value = 0.005	21	36	3.5	Good
500	4. Crossover Value = 0.9	21	35	3.2	Good

Table 18. Sensitivity analysis of maximum no. of iterations.

Maximum No of Iterations	Value of Constant Parameters	Metrics		Combined Performance Score	Overall Performance
		GD	SM		
500	1. Initial population size = 200	65	30	3.4	Good
1000	2. External population size = 100, 800	21	33	3.2	Good
2000	3. Mutation value = 0.09	30	34	3.2	Good
	4. Crossover Value = 0.7				
500	1. Initial population size = 500	50	32	3.3	Good
1000	2. External population size = 100, 800	51	34	3.6	Good
2000	3. Mutation value = 0.09	24	34	3.8	Good
	4. Crossover Value = 1.0				

Table 19. Sensitivity analysis of Coulomb's constant initial value.

Initial Value of Coulomb's Constant	Value of Constant Parameters	Metrics		Combined Performance Score	Overall Performance
		GD	SM		
100	1. Initial population size = 200	85	31	3.6	Good
300	2. External population size = 100, 800	30	40	3.3	Good
	3. Maximum number of iterations = 1000				
500	4. Mutation value = 0.09	40	332	4.2	Very Good
	5. Crossover Value = 0.7				
100	1. Initial population size = 500	62	35	3.3	Good
300	2. External population size = 100, 800	48	39	3.9	Good
	3. Maximum number of iterations = 2000				
500	4. Mutation value = 0.09	38	31	4.0	Very Good
	5. Crossover Value = 1.0				

6. Conclusions and Future Work

In this paper, an improved population-based meta-heuristic algorithm, AEFA, is proposed to handle the multi-objective optimization problems. The proposed algorithm used strength Pareto dominance theory to refine fitness assignment and an improved fine-grained elitism selection based on the SDE mechanism to maintain population diversity. Further, the proposed algorithm used the SDE technique with strength Pareto dominance theory for density estimation, and it implemented bounded exponential crossover (BEX) and polynomial mutation operator (PMO) to avoid solutions trapping in local optima and enhance convergence. The experiments were performed in two steps. In the first step, the AEFA was evaluated on different low-dimensional and high-dimensional benchmark functions, and then the performance of the AEFA was compared with the existing evolutionary optimization algorithms (EOAs) based on four parameters: average best-so-far solution, mean best-so-far solution, the variance of the best-so-far solution, and average run-time. Experimental results show that for low-dimensional benchmark functions, AEFA performed better in comparison to existing EOAs, but for complex high-dimensional benchmark functions, AEFA suffered from loss of diversity and performed worse. In the second step, the proposed algorithm was evaluated on different benchmark functions, and then the performance of the proposed algorithm was compared with the existing evolutionary multi-objective optimization algorithms (EMOOPs) based on diversity, converge, generational distance, and spacing metrics. Experimental results prove that the proposed algorithm is not only able to find the optimal Pareto front (non-dominated solutions), but it also shows better performance than the existing multi-objective optimization techniques in terms of accuracy, robustness, and efficacy. In the future, the proposed work can be explored in various directions. One direction is to extend the proposed algorithm to solve various real-word multi-objective optimization problem. The proposed work can also be used to solve complex problems by hybridization of the proposed algorithm with another existing algorithm.

Author Contributions: Conceptualization, Methodology, Formal analysis, Validation, Writing—Original Draft Preparation, H.P.; Supervision, R.R. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Nobahari, H.; Nikusokhan, M.; Siarry, P. Non-dominated sorting gravitational search algorithm. In Proceedings of the 2011 International Conference on Swarm Intelligence, Cergy, France, 14–15 June 2011; pp. 1–10.
2. Van Veldhuizen, D.A.; Lamont, G.B. On measuring multiobjective evolutionary algorithm performance. In Proceedings of the 2000 Congress on Evolutionary Computation, La Jolla, CA, USA, 16–19 July 2000; Volume 1, pp. 204–211.
3. Deb, K.; Pratap, A.; Agarwal, S.; Meyarivan, T. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans. Evol. Comput.* **2002**, *6*, 182–197. [CrossRef]
4. Zhao, B.; Cao, Y.J. Multiple Objective Particle Swarm Optimization Technique for Economic Load Dispatch. *J. Zhejiang Univ. Sci. A* **2005**, *6*, 420–427.
5. Cagnina, L. A Particle Swarm Optimizer for Multi-Objective Optimization. *J. Comput. Sci. Technol.* **2005**, *5*, 204–210.
6. Zhang, Y.; Gong, D.W.; Ding, Z. A bare-bones multi-objective particle swarm optimization algorithm for environmental/economic dispatch. *Inf. Sci.* **2012**, *192*, 213–227. [CrossRef]
7. Li, X. A non-dominated sorting particle swarm optimizer for multiobjective optimization. In *Genetic and Evolutionary Computation Conference*; Springer: Berlin/Heidelberg, Germany, 2003; pp. 37–48.
8. Yadav, A. AEFA: Artificial electric field algorithm for global optimization. *Swarm Evol. Comput.* **2019**, *48*, 93–108.
9. Demirören, A.; Hekimoğlu, B.; Ekinci, S.; Kaya, S. Artificial Electric Field Algorithm for Determining Controller Parameters in AVR system. In Proceedings of the 2019 International Artificial Intelligence and Data Processing Symposium (IDAP), Malatya, Turkey, 21–22 September 2019; pp. 1–7.
10. Abdelsalam, A.A.; Gabbar, H.A. Shunt Capacitors Optimal Placement in Distribution Networks Using Artificial Electric Field Algorithm. In Proceedings of the 2019 the 7th International Conference on Smart Energy Grid Engineering (SEGE), Oshawa, ON, Canada, 12–14 August 2019; pp. 77–85.
11. Shafik, M.B.; Rashed, G.I.; Chen, H. Optimizing Energy Savings and Operation of Active Distribution Networks Utilizing Hybrid Energy Resources and Soft Open points: Case Study in Sohag, Egypt. *IEEE Access* **2020**, *8*, 28704–28717. [CrossRef]

12. Thakur, M.; Meghwani, S.S.; Jalota, H. A modified real coded genetic algorithm for constrained optimization. *Appl. Math. Comput.* **2014**, *235*, 292–317. [CrossRef]
13. Gong, M.; Jiao, L.; Du, H.; Bo, L. Multiobjective immune algorithm with nondominated neighbor-based selection. *Evol. Comput.* **2008**, *16*, 225–255. [CrossRef]
14. Liu, Y.; Niu, B.; Luo, Y. Hybrid learning particle swarm optimizer with genetic disturbance. *Neurocomputing* **2015**, *151*, 1237–1247. [CrossRef]
15. Fonseca, C.M.; Fleming, P.J. Genetic algorithms for multi-objective optimization: Formulation, discussion and generalization. In Proceedings of the 5th International Conference on Genetic Algorithms, San Mateo, CA, USA, 1 June 1993; pp. 416–423.
16. Horn, J.; Nafploitis, N.; Goldberg, D.E. A niched Pareto genetic algorithm for multi-objective optimization. In Proceedings of the 1st IEEE Conference on Evolutionary Computation, Orlando, FL, USA, 27–29 June 1994; pp. 82–87.
17. Tan, K.C.; Goh, C.K.; Mamun, A.A.; Ei, E.Z. An evolutionary artificial immune system for multi-objective optimization. *Eur. J. Oper. Res.* **2008**, *187*, 371–392. [CrossRef]
18. Zitzler, E. *Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications*; Ithaca: Shaker, OH, USA, 1999; Volume 63.
19. Srinivas, N.; Deb, K. Multi-Objective function optimization using non-dominated sorting genetic algorithms. *Evol. Comput.* **1995**, *2*, 221–248. [CrossRef]
20. Zitzler, E.; Thiele, L. Multiobjective optimization using evolutionary algorithms—A comparative case study. In *International Conference on Parallel Problem Solving from Nature*; Springer: Berlin/Heidelberg, Germany, 1998; pp. 292–301.
21. Rudolph, G. *Technical Report No. CI-67/99: Evolutionary Search under Partially Ordered Sets*; Dortmund Department of Computer Science/LS11, University of Dortmund: Dortmund, Germany, 1999.
22. Rughooputh, H.C.; King, R.A. Environmental/economic dispatch of thermal units using an elitist multiobjective evolutionary algorithm. In Proceedings of the IEEE International Conference on Industrial Technology, Maribor, Slovenia, 10–12 December 2003; Volume 1, pp. 48–53.
23. Deb, K. *Multi-Objective Optimization Using Evolutionary Algorithms*; John Wiley & Sons: New York, NY, USA, 2001; Volume 16.
24. Vrugt, J.A.; Robinson, B.A. Improved evolutionary optimization from genetically adaptive multimethod search. *Proc. Natl. Acad. Sci. USA* **2007**, *104*, 708–711. [CrossRef] [PubMed]
25. Deb, K.; Jain, H. An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: Solving problems with box constraints. *IEEE Trans. Evol. Comput.* **2013**, *18*, 577–601. [CrossRef]
26. Jain, H.; Deb, K. An evolutionary many-objective optimization algorithm using reference-point based nondominated sorting approach, part II: Handling constraints and extending to an adaptive approach. *IEEE Trans. Evol. Comput.* **2013**, *18*, 602–622. [CrossRef]
27. Zhao, B.; Guo, C.X.; Cao, Y.J. A multiagent-based particle swarm optimization approach for optimal reactive power dispatch. *IEEE Trans. Power Syst.* **2005**, *20*, 1070–1078. [CrossRef]
28. Zhang, Q.; Li, H. MOEA/D: A multiobjective evolutionary algorithm based on decomposition. *IEEE Trans. Evol. Comput.* **2007**, *11*, 712–731. [CrossRef]
29. Ma, X.; Liu, F.; Qi, Y.; Li, L.; Jiao, L.; Liu, M.; Wu, J. MOEA/D with Baldwinian learning inspired by the regularity property of continuous multiobjective problem. *Neurocomputing* **2014**, *145*, 336–352. [CrossRef]
30. Martinez, S.Z.; Coello, C.A.C. A multi-objective evolutionary algorithm based on decomposition for constrained multi-objective optimization. In Proceedings of the 2014 IEEE Congress on Evolutionary Computation (CEC), Beijing, China, 6–11 July 2014; pp. 429–436.
31. Gu, F.; Liu, H.L.; Tan, K.C. A multiobjective evolutionary algorithm using dynamic weight design method. *Int. J. Innov. Comput. Inf. Control* **2012**, *8*, 3677–3688.
32. Zhang, X.; Tian, Y.; Cheng, R.; Jin, Y. An efficient approach to nondominated sorting for evolutionary multiobjective optimization. *IEEE Trans. Evol. Comput.* **2014**, *19*, 201–213. [CrossRef]
33. Chong, J.K.; Qiu, X. An Opposition-Based Self-Adaptive Differential Evolution with Decomposition for Solving the Multiobjective Multiple Salesman Problem. In Proceedings of the 2016 IEEE Congress on Evolutionary Computation (CEC), Vancouver, BC, Canada, 24–29 July 2016; pp. 4096–4103.

34. Cheng, R.; Jin, Y.; Olhofer, M.; Sendhoff, B. A reference vector guided evolutionary algorithm for many-objective optimization. *IEEE Trans. Evol. Comput.* **2016**, *20*, 773–791. [CrossRef]
35. Hassanzadeh, H.R.; Rouhani, M. A multi-objective gravitational search algorithm. In Proceedings of the 2010 2nd International Conference on Computational Intelligence, Communication Systems and Networks, Liverpool, UK, 28–30 July 2010; pp. 7–12.
36. Yuan, X.; Chen, Z.; Yuan, Y.; Huang, Y.; Zhang, X. A strength pareto gravitational search algorithm for multi-objective optimization problems. *Int. J. Pattern Recognit. Artif. Intell.* **2015**, *29*, 1559010. [CrossRef]
37. Li, M.; Yang, S.; Liu, X. Shift-based density estimation for Pareto-based algorithms in many-objective optimization. *IEEE Trans. Evol. Comput.* **2013**, *18*, 348–365. [CrossRef]
38. Zitzler, E.; Laumanns, M.; Thiele, L. *TIK report 103: SPEA2: Improving the Strength Pareto Evolutionary Algorithm*; Computer Engineering and Networks Laboratory (TIK), ETH Zurich: Zurich, Switzerland, 2001.
39. Schott, J.R. *Fault Tolerant Design Using Single and Multicriteria Genetic Algorithm Optimization*; Air force inst of tech Wright-Patterson AFB: Cambridge, MA, USA, 1995.
40. Guzmán, M.A.; Delgado, A.; De Carvalho, J. A novel multiobjective optimization algorithm based on bacterial chemotaxis. *Eng. Appl. Artif. Intell.* **2010**, *23*, 292–301. [CrossRef]