Information Sciences Letters

Volume 12
Issue 7 *Jul. 2023*

Article 45

2023

# Optimizing Firefly Algorithm for Directional Overcurrent Relay Coordination: A case study on the Impact of Parameter Settings

Tareq Foqha
*Department of Electrical Engineering, Faculty of Engineering and Information Technology, Arab American University, Jenin P.O. Box 240, Palestine*, tariq.foqha@ptuk.edu.ps

Samer Alsadi
*Department of Electrical Engineering, Faculty of Engineering and Information Technology, Palestine Technical University-Kadoorie, Tulkarm P305, Palestine*, tariq.foqha@ptuk.edu.ps

Ali Elrashidi
*Department of Electrical Engineering, College of Engineering, University of Business and Technology, Jeddah, 21448, Kingdom of Saudi Arabia*, tariq.foqha@ptuk.edu.ps

Nael Salman
*Department of Computer Systems Engineering, Faculty of Engineering and Information Technology, Palestine Technical University-Kadoorie, Tulkarm P305, Palestine*, tariq.foqha@ptuk.edu.ps

Recommended Citation
Foqha, Tareq; Alsadi, Samer; Elrashidi, Ali; and Salman, Nael (2023) "Optimizing Firefly Algorithm for Directional Overcurrent Relay Coordination: A case study on the Impact of Parameter Settings," *Information Sciences Letters*: Vol. 12 : Iss. 7 , PP -.
Available at: https://digitalcommons.aaru.edu.jo/isl/vol12/iss7/45

Information Sciences Letters
*An International Journal*

http://dx.doi.org/10.18576/isl/120745

# Optimizing Firefly Algorithm for Directional Overcurrent Relay Coordination: A case study on the Impact of Parameter Settings

*Tareq Foqha[1,*], Samer Alsadi[2], Ali Elrashidi[3] and Nael Salman[4]*

[1]Department of Electrical Engineering, Faculty of Engineering and Information Technology, Arab American University, Jenin P.O. Box 240, Palestine
[2]Department of Electrical Engineering, Faculty of Engineering and Information Technology, Palestine Technical University-Kadoorie, Tulkarm P305, Palestine
[3]Department of Electrical Engineering, College of Engineering, University of Business and Technology, Jeddah, 21448, Kingdom of Saudi Arabia
[4]Department of Computer Systems Engineering, Faculty of Engineering and Information Technology, Palestine Technical University-Kadoorie, Tulkarm P305, Palestine

Received: 15 Jan. 2023, Revised: 22 Mar. 2023, Accepted: 28 May 2023.
Published online: 1 Jul. 2023.

**Abstract:** The Firefly Algorithm is a nature-inspired optimization algorithm that has been shown to be effective for solving a variety of problems. In this paper, we study the Firefly Algorithm for the IEEE 3-bus network coordination problem. We examine the impact of key parameters on the algorithm's performance, including generation number, population size, absorption coefficient, and randomization parameter. The results showed that increasing the generation number improves the quality of the solution, but the benefits decrease at a certain point. Feasibility also improves with higher generations, but a balance between solution quality and feasibility becomes apparent at very high generations. Objective function evaluations and computation time increase linearly with the number of generations. Larger population sizes yield better solution quality and feasibility, but a balance is observed at very high population sizes. The randomization parameter has a modest influence on the results, but extreme values can impact solution quality, feasibility, and computation time. The absorption coefficient, on the other hand, has a significant impact on convergence and quality. The study offers guidance for parameter optimization and adaptive techniques.

**Keywords:** Optimization algorithms, Firefly algorithm, Parameters setting, Directional overcurrent relays, MATLAB.

## 1 Introduction

Optimization involves selecting the optimal solution from a set of alternatives. Nature-inspired metaheuristic algorithms, particularly those based on swarm intelligence, have gained significant attention in the past decade for their ability to address complex optimization problems effectively [1]. Metaheuristic algorithms require of setting the values of several algorithm components and parameters. These parameters values have great impact on performance and efficacy of the algorithm [2,3]. Therefore, it is essential to investigate the algorithm parameters influence on the performance of the developed metaheuristic algorithms [4].

The Firefly algorithm (FA) is a popular metaheuristic algorithm in the field of swarm intelligence optimization. It has demonstrated impressive search capabilities when applied to various optimization problems [5]. The algorithm emulates the social behavior of fireflies by incorporating their flashing patterns and attraction characteristics [6,7]. It is known for its ease of comprehension and implementation. However, researches indicated that it has a tendency to premature convergence. To address this issue, scholars recommend relaxing the constraint of keeping the algorithm's parameters constant [8].

The Firefly algorithm's performance is influenced by several parameters, including the number of generations, population size, randomization parameter ($\alpha$), and absorption coefficient ($\gamma$). The $\alpha$ parameter controls the level of randomness and diversity in the solutions generated [1]. The $\gamma$ parameter plays an essential role in determining the

[*]Corresponding author e-mail: tariq.foqha@ptuk.edu.ps

variations of attractiveness and the convergence speed of the algorithm [7,9]. The number of fireflies corresponds to the population size, and a larger population enhances the algorithm's exploration capability. However, this also results in increased computational time [10–12]. The number of generations determines the total iterations of the algorithm. Increasing the number of generations allows for a longer exploration time, enabling the algorithm to converge towards the optimal solution. However, excessively large numbers of generations can lead to longer computational times without significant improvements in performance [13].

In [14], the authors analyzed the impact of key parameters on the Firefly Algorithm's performance. They investigated the randomization parameter, absorption coefficient, and population size using benchmark functions. Optimal values for the absorption coefficient and randomization parameter were identified, emphasizing the importance of parameter selection. Increasing the population size improved solution quality, but with longer computation time. Authors suggest reasonable ranges for the randomization parameter ([0.1, 0.2]) and optimal values for the absorption coefficient ([1, 30]). For simpler problems, a smaller population size (20 to 40) is suitable, while more complex problems may require a population size not exceeding 50.

In [15], the Firefly Algorithm was analyzed for performance and success rate using benchmark functions, leading to parameter selection guidelines. Optimal parameter sets were identified for different functions, highlighting the balance between exploration and exploitation. The algorithm was evaluated on four benchmark functions, revealing the importance of selecting a suitable number of fireflies. Functions with single minima perform well, while those with multiple minima pose challenges. The algorithm's convergence and dynamic behavior were analyzed, providing qualitative guidelines for parameter selection. Results indicate better performance with a small number of fireflies and lower parameter values.

The effectiveness of a genetic algorithm (GA) for parameter identification in an E. coli fed-batch cultivation process was examined by the authors of [4] in relation to the effect of population size. They examined populations with fixed generations and chromosome counts ranging from 5 to 200. The results showed that 100 chromosomes for 200 generations was the ideal population size for accurate parameter estimation in a manageable amount of time. Beyond 100 individuals, there was no improvement in solution accuracy, but there was a significant increase in computation time. This emphasizes the importance of choosing an appropriate population size in GA-based parameter identification problems in order to balance solution quality and computational efficiency.

The directional overcurrent relays (DOCRs) coordination problem in power system protection is a complex optimization problem, which can be highly constrained, nonlinear, and non-convex [16]. In this paper, Firefly Algorithm is applied to solve this problem in the IEEE 3-bus network. The aim of the study is to determine how various algorithmic factors, such as the population size, absorption coefficient, generation number and randomization parameter, affect the algorithm's performance.

The main contributions of this paper are summarized as follows:

- Investigate the application of the Firefly Algorithm for solving the coordination problem in the IEEE 3-bus network.
- Analyze the effect of the algorithm parameters on the performance of the Firefly Algorithm.
- Comprehensive analysis of the Firefly Algorithm's performance, considering solution quality, feasibility, computational effort, and efficiency.

The rest of this paper is organized as follows. **Section 2** provides a comprehensive explanation of the Firefly Algorithm, including its working principles and its parameters, along with an analysis of how these parameters influence the algorithm's performance. The directional overcurrent relay coordination problem is described in **Section 3** along with its related formulations. In **Section 4**, the results obtained from applying the Firefly Algorithm to the IEEE 3-bus system are presented and analyzed, examining the impact of algorithm parameters on its performance. Finally, **Section 5** concludes the study by summarizing the main findings and contributions. It also highlights possible directions for future research.

## 2 Firefly Algorithm

The Firefly Algorithm (FA) was initially developed by Xin She Yang at Cambridge University in late 2007 and 2008. It draws inspiration from the flashing patterns and behavior exhibited by fireflies [1]. The algorithm fundamentally relies on three idealized rules, which are as follows:

- Fireflies are unisex so that one firefly will be attracted to other fireflies regardless of their gender.

- The brightness of a firefly is influenced or determined by the landscape of the objective function to be optimized.
- The attractiveness of a firefly is proportional to its brightness and they both decrease with distance (*r*) as shown in Equation (1). Consequently, when two fireflies are compared, the one with lower brightness will move towards the brighter firefly. If there is no firefly brighter than a particular one, it will move randomly within the search space [7,17–19].

$$I \propto \frac{1}{r^2} \tag{1}$$

Equation (2) describes the relationship between light intensity and distance (r) when light passes through a medium with a light absorption coefficient γ [8].

$$I = I_0\, e^{-\gamma r^2} \tag{2}$$

where $I_0$ represents the source's light intensity. The brightness, $\beta$, can also be expressed as in Equation (3).

$$\beta = \beta_0\, e^{-\gamma r^2} \tag{3}$$

where $\beta_0$ is the attractiveness at *r* equal to 0, which is the maximum attractiveness that a firefly can have. The distance between any two Fireflies *i* and *j* whose positions are $x_i$ and $x_j$ is given by the Cartesian distance as follows:

$$r_{i,j} = \sqrt{\sum_{m=1}^{D} (x_{i,m} - x_{j,m})^2} \tag{4}$$

where, *D* is the number of dimensions, $x_{i,m}$ refers to the *m-th* component of the spatial coordinate of the *i-th* firefly, while $x_{j,m}$ refers to the *m-th* component of the spatial coordinate of the *j-th* firefly [20]. The movement of the *i-th* firefly towards more attractive *j-th* firefly is calculated as:

$$x_i^{new} = x_i^{old} + \beta_0\, e^{-\gamma r^2}\left(x_i^{old} - x_j\right) + \alpha\,(rand - 0.5) \tag{5}$$

Here, he first term represents the current location of the *i-th* firefly. The second term represents the attraction exerted by one firefly on another. The third term, involving the parameter α, introduces randomization into the movement of the firefly. This randomization is achieved using the random number generator Rand. The new position of the i-th firefly is denoted as $x_i^{new}$, while $x_i^{old}$ represents its previous position [6].

The pseudo-code in **Fig. 1** illustrates the structure of the standard firefly algorithm. As can be seen, the algorithm consists of the following components: a firefly representation, an initialization (line 3), a moving operator (line 8), and an objective function (lines 4 and 10).

```
Firefly Algorithm
1.   Initialize FA: number of fireflies (n), generation number of FA
     (nGer), absorption coefficient (γ), initial attractiveness (β0), and the
     randomness strength (α).
2.   Objective function f(x),        x = (x1, ..., xD)^T
3.   Generate initial population of fireflies xi (i = 1, 2, ..., n)
4.   Light intensity Ii at xi is determined by f(xi)
5.       while (t< nGer)
6.           for i =1: n all n fireflies
7.               for j =1: i all n fireflies
8.               if (Ij > Ii),
                 Move firefly i towards j in D-dimension; end if
9.               Attractiveness varies with distance r via $e^{-\gamma r^2}$
10.              Evaluate new solutions and update light intensity
11.              end for j
12.          end for i
13.  Rank the fireflies and find the current best
14   t = t + 1
15.  end while
16.  Postprocess results and visualization
```

**Fig. 1:** Pseudo code of the firefly algorithm.

The movement of the entire population of solutions in the Firefly Algorithm is influenced by two factors. Firstly, it depends on where the global best solution, which is the most comprehensively ideal solution thus far, is located. Secondly, it is also influenced by the positions of local better (brighter) solutions within the neighborhood (as indicated in line 13 of the algorithm). In addition, a termination condition is necessary to ensure the appropriate termination of the Firefly Algorithm (line 5). The algorithm incorporates a criterion, referred to as the maximum number of generations (nGen), which stops the algorithm once it reaches the specified maximum number of generations or iterations. This criterion helps determine when the algorithm should cease its iterations [21–23].

The optimization algorithm begins by assigning an initial solution to each agent in a population. These agents represent fireflies in a firefly swarm. The algorithm then proceeds iteratively, updating the positions of the fireflies based on their light intensity and their proximity to other fireflies. If the light intensity of a firefly, represented by agent i, is lower than the light intensity of another firefly, represented by agent j at position $x_j$, then firefly i moves towards firefly j. The objective is for the fireflies to converge towards the brightest light source, which represents the optimal solution to the optimization problem. The fitness values of the fireflies are used to rank them, with higher fitness indicating better solutions. If a firefly discovers a better solution than the current global best solution, the global best is updated accordingly. The movement of each firefly is influenced by a randomization parameter α, which is a random number uniformly distributed between 0 and 1. This parameter introduces randomness into the movement of the fireflies, allowing exploration of the solution space. The algorithm continues iterating and updating the positions of the fireflies until convergence is achieved, aiming to find the optimal solution to the optimization problem [24,25]. The main steps in the firefly algorithm are depicted in **Fig. 2.**

In the Firefly Algorithm, the selection of parameters plays a crucial role in achieving good performance. Two important parameters are γ in Equation (3) and the α in Equation (5). The algorithm typically does not consider the knowledge or information gained by the fireflies during the search when selecting these parameters. However, the effectiveness of the search process and the quality of the solution found are greatly influenced by the correct selection of these parameters. Consequently, it is important to carefully choose appropriate values for γ and α [26,27]. In this work, the authors investigated the effects of varying γ and α, as well as the number of generations and fireflies, on the algorithm's performance. The study attempts to obtain an understanding of how parameter settings affect the algorithm's efficacy and adjust them accordingly by examining these aspects. The effect of each parameter on the Firefly Algorithm's performance will be discussed in the sections that follow.

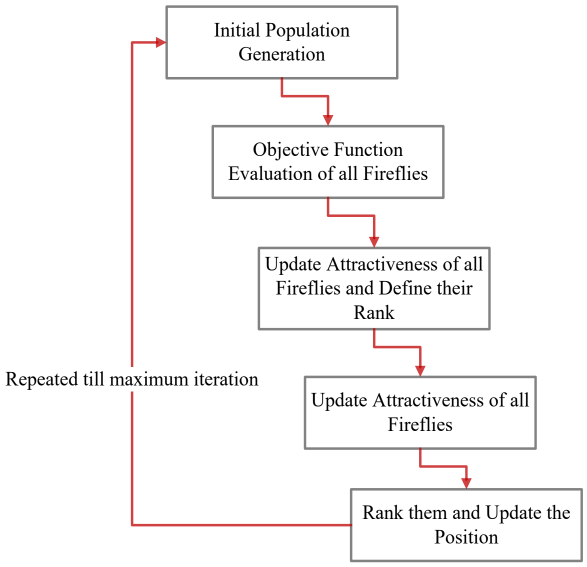
Initial Population Generation
Objective Function Evaluation of all Fireflies
Update Attractiveness of all Fireflies and Define their Rank
Repeated till maximum iteration
Update Attractiveness of all Fireflies
Rank them and Update the Position

**Fig. 2:** Working of the firefly algorithm.

## *2.1 Effect of Randomization Parameter (α)*

The α parameter controls the algorithm's level of randomness and, to some extent, the variety of possible solutions [1]. A higher α value results in more random movements, facilitating exploration of the solution space. Conversely, a lower α value promotes exploitation of already promising areas by reducing randomness [28]. The effect of α on algorithm performance is dependent on the problem's characteristics and the stage of the search process. Initially, a higher α value may aid in escaping local optima, while a lower α value towards the end of the search can refine the solutions found.

Adjusting α during the search process can lead to improved exploration-exploitation balance.

### *2.2 Effect of Absorption coefficient (γ)*

The γ parameter in FA determines the rate at which the light intensity diminishes with distance. A higher γ value implies faster absorption of light intensity. The effect of γ on algorithm performance is twofold. A lower γ value allows fireflies to retain their brightness over longer distances, promoting exploration of the solution space. On the other hand, a higher γ value leads to more rapid decay of light intensity, encouraging fireflies to converge towards brighter regions and exploit the best solutions. The choice of γ depends on the problem at hand, and striking the right balance is important for achieving optimal performance.

It can be deduced that a constant light absorption coefficient γ [9] gives rise to two distinct scenarios with limited outcomes:

- When γ approaches 0: In this scenario, the attractiveness of other fireflies becomes constant, implying that every firefly can be seen by all the others. In this case, the FA behaves similarly to a classical Particle Swarm Optimization (PSO) algorithm. The movement of fireflies is solely determined by the positions and velocities of other fireflies, without any decay or diminishing effect on their attractiveness. This can lead to a more global exploration of the solution space, as fireflies are not influenced by the decay of light intensity with distance.
- When γ approaches ∞: In this situation, the attractiveness between fireflies becomes negligible and tends towards zero. Fireflies are unable to determine the direction of movement based on the attractiveness of other fireflies, resulting in random flight behavior. Essentially, the FA becomes a pure random search algorithm, where fireflies move randomly without any specific guidance from their interactions. As a result, the algorithm loses its ability to exploit information from other fireflies and relies solely on chance encounters for exploration.

As can be seen, the parameter γ plays an essential role in defining the variations of attractiveness in the FA. Its value significantly impacts the convergence speed of the algorithm [7].

### *2.3 Effect of Number of Fireflies/Population Size (n)*

The convergence of any nature-inspired optimization algorithm is significantly impacted by the size of the population it operates on. The number of fireflies represents the size of the population within the algorithm. A larger number of fireflies enhances the exploration capability of the algorithm by covering a larger portion of the solution space. With more fireflies, there is a higher chance of finding better solutions and escaping local optima. However, increasing the number of fireflies also raises the computational time of the algorithm. It is vital to strike a balance between the population size and computational resources [10,11,29]. Hence, this study also examines the influence of changing population sizes on the efficiency of the Firefly Algorithm.

### *2.4 Effect of Number of Generations (nGer)*

The number of generations determines how many iterations or steps the algorithm will take during the search process. Increasing the number of generations allows for a longer exploration time, giving the algorithm more opportunities to converge towards the optimal solution. However, a very large number of generations may lead to excessive computational time without significant improvements in performance. The computational efforts and performance of an algorithm are largely determined by the number of iterations required to achieve an optimal solution with a given accuracy. A superior algorithm should aim to minimize computation and iterations, resulting in more efficient performance [13,30]. The choice of the number of generations depends on the complexity of the problem, the convergence characteristics of the algorithm, and the available computational resources [31].

## 3 Coordination Problem of Directional Overcurrent Relays (DOCRs)

Protective relays have a vital role in maintaining the reliability of the electric power supply by quickly detecting faults, isolating the faulty parts, and keeping the unaffected sections in service, ensuring the continuity of power supply [32]. Overcurrent protection is the primary protection of the power distribution systems. In interconnected systems, the directional overcurrent protection is used to avoid the disconnection of unnecessary parts due to fault currents flowing in both directions. Non-directional overcurrent relays are unable to fulfill this purpose, which is why directional overcurrent relays (DOCRs) are utilized [33].

Coordinating protective devices within a protection system is an essential process to ensure the robustness and reliability of a power system [34]. The primary objective of relay coordination is to effectively protect the power system against abnormal conditions. By promptly disconnecting faulty sections, the coordination ensures the uninterrupted operation of the healthy sections, thereby guaranteeing the continued service and reliability of the power system [35]. It can be shortly defined as: "the quality of selectivity among protective devices" [36]. Therefore, accurate relay coordination ensures the proper selection of primary relays responsible for clearing faults within their designated zones [37,38]. Backup relays, on the other hand, are designed to respond to faults outside their designated zones and are activated only if the primary relays exceed the permissible time delay. This time delay is referred to as the coordination time interval (CTI) [39]. The relays must be properly coordinated so that there is no misoperation. Also, to not lead to unnecessary tripping of breakers and the isolation of sections of the power system that are not actually faulted. This may be accomplished by appropriately determining the operating time of the relays [40,41]. The DOCRs have two settings, the time multiplier setting (TMS) and the plug setting (PS) [42]. The operating time of the relay is directly impacted by these factors [43]. The total operating time of the DOCRs should be minimized to reduce power network interruption. To ensure that the selectivity study is valid, the specified CTI between the primary and backup protections must be maintained [44]. However, the DOCR coordination problem is formulated as an optimization problem. The goal is to reduce the total operating time of the relays within the system for various fault scenarios. This optimization process considers multiple constraints and boundary limits, including relay settings and selectivity constraints. The goal is to achieve an optimal coordination scheme that ensures effective protection while handling these constraints [45–47].

The coordination problem is recognized as a complex optimization problem with multiple constraints, nonlinearity, and non-convexity. However, it can be formulated as a linear programming problem. In this formulation, the time multiplier setting of the relays is treated as the control variable, while the plug setting is considered a fixed value within predefined boundaries [48]. Alternatively, the coordination problem can be formulated as a nonlinear programming problem by considering both the time multiplier setting and plug setting as decision variables. These variables can be either continuous or discrete in nature. However, the inclusion of discrete settings introduces additional complexity to the problem since it restricts the number of feasible solutions that can be explored [49].

Researchers typically divide their studies on solving the coordination problem of DOCRs into two main subjects. The first one involves modeling the DOCRs coordination problem to be solved using optimization algorithms. The second one focuses on optimizing the developed model for the DOCRs coordination problem. To achieve optimal coordination of DOCRs, several steps are typically taken. Firstly, the specifications of each component in the system, such as the network type, network topology, and fault location, are identified. Secondly, a load flow analysis is performed to determine the maximum load current that the protected zone can handle. The third step involves identifying all relay pairs involved in the coordination process to ensure proper selectivity and minimize disruptions. Next, the smallest severe faults are identified through short-circuit analysis, which is essential for setting the relay plug settings within permissible limits. Finally, an optimization method is applied to determine the optimum relay settings for effective coordination [50–52]. The following sections describe the formulation of the coordination of DOCRs as an optimization problem.

## *3.1 Objective Function Formulation*

The purpose of DOCRs coordination is to detect the failure in a minimum time. To achieve this, authors of this study have utilized objective function aims for minimizing the total operating times of all primary DOCRs in the system. The objective function is presented by the following equation:

$$OF = \min \sum_{i=1}^{n} T_i \tag{6}$$

where, n is the number of primary relays in the network; T represents the operating time of the *i-th* primary relay. The operating time of the DOCRs for all objective functions is determined using Equation (7), as specified in both IEC/BS and ANSI/IEEE standards. In this equation, the pickup current ($I_P$) and short-circuit current (Isc) are known quantities.

$$T = TMS\left[\frac{\beta}{(\frac{I_{SC}}{PS})^{\alpha}-1} + \delta\right] \tag{7}$$

$$PS = \frac{Ip}{CTR} \tag{8}$$

where, α,β and γ are scalar quantities vary depending on the type of characteristics utilized for DOCRs. In this article, the IEC standard inverse type is used where α = 0.02, β = 0.14 and $\delta$ = 0. TMS is the time multiplier setting of the relay, Isc is the fault current flowing through the relay, and PS is the plug setting of the relay. In general, the plug setting represents the ratio of the pickup current ($I_P$) to the current transformer ratio (CTR).

## *3.2 Constraints Formulation*

The minimization of the objective function is subjected to various sets of constraints. One set of these constraints concerns the relay characteristic, and the other set is concerned with assuring selectivity. The following sections outline each type of constraint and how the authors address them.

### *3.2.1 Relay Settings Constraints*

The constraints related to the relay characteristics, such as relay operation time, TMS, and PS, are important considerations in the design and operation of relay systems. These parameters are essential for achieving effective coordination and protection of electrical systems.

The protective relaying system is designed to operate within a predefined time range, where it has a specific threshold time for initiating the tripping action. Moreover, there exists a maximum time limit that imposes a restriction on the duration for which the relaying system can remain operational [53,54]. The bounds on the operation time of relays could be stated as:

$$T_{i,min} \leq T_i \leq T_{i,max} \tag{9}$$

where the superscripts "min" and "max" represent the minimum and maximum values of the corresponding variables, respectively, while $T_i$ denotes the operation time of relay i. The maximum time is determined by the critical clearing time and the allowable thermal limit of the protected equipment, while the minimum time is influenced by the relay's manufacturing specifications [55].

To achieve an optimal setting result for TMS, it is necessary to determine its upper and lower bounds. The limitations of TMS can be defined as follows:

$$TMS_{i,min} \leq TMS_i \leq TMS_{i,max} \tag{10}$$

where $TMS_{i,min}$ and $TMS_{i,max}$ are the minimum and maximum values of TMS of the ith relay, respectively. $TMS_{i,max}$ represents the maximum allowable value that can be set for the relay. Setting TMS too high may result in delayed or ineffective operation during fault conditions, while $TMS_{i,min}$ signifies the minimum acceptable value that can be set for the relay. Setting TMS too low may lead to unnecessary tripping or improper coordination with upstream relays. The relay manufacturer provides the minimum and maximum values for the TMS of the relays. [49,56].

The PS value determines the current level at which a protective relay triggers its operation. It serves as an indicator of the relay's sensitivity towards detecting faults. The PS value is established by considering factors such as the minimum fault current, the desired level of protection, and the full load current. Accurate setting of the PS is vital to prevent unnecessary tripping or failure to trip during a fault. The definition of the PS parameter is as follows:

$$PS_{i,min} \leq PS_i \leq PS_{i,max} \tag{11}$$

where $PS_{i,min}$ and $PS_{i,max}$ are the minimum and maximum values of PS of the ith relay, respectively. $PS_{i,min}$ represents the lowest current level at which the relay will operate. It should be set to a value that is equal to or greater than the maximum overload current. This ensures that the relay is sensitive enough to detect and respond to fault conditions where the current exceeds the maximum overload level. It is defined as in the following equation:

$$PS_{i,min} = \frac{OLF * I_{i-L,max}}{CTR_i} \tag{12}$$

where OLF stands for the overload factor, which is dependent on the protected element, $I_{i-L,max}$ is the maximum load current.

The maximum value, $PS_{i,max}$, represents the highest current level at which the relay will still operate. It should be set to the minimum fault current or less. This ensures that the relay will activate and initiate the appropriate protection measures whenever a fault current exceeds the specified threshold. Setting the maximum value lower than the minimum fault current may result in the relay failing to detect and respond to lower magnitude faults, potentially leading to inadequate protection. On the other hand, setting the maximum value lower than the minimum fault current may cause false tripping and unnecessary interruptions in normal operation. $PS_{i,max}$ is determined as follows [53,57]:

$$PS_{i,max} = \frac{2I_{i-f,min}}{3CTR_i} \tag{13}$$

where $I_{i-f,min}$ is the minimum fault current that must be detected by that relay.

### 3.2.2 Relay Settings Constraints

In order to ensure a reliable protection system, it is common practice for each primary protection to have its own backup protection. This ensures that if the primary protection fails, the backup scheme can intervene and provide the necessary protection. To facilitate proper coordination between these protection schemes, a predefined coordination time interval (CTI) is established. The CTI represents the time delay between the initiation of the primary protection scheme and the backup scheme being activated. This is important to comply with the selectivity requirement of the primary and backup relays [47]. The following equation represents the coordination constraint:

$$T_{j,k} - T_{i,k} \geq CTI \tag{14}$$

where $T_{j,k}$ and $T_{i,k}$ are the operating times of backup relay (Rj) and primary relay (Ri), respectively, for fault at k, and CTI is the coordination time interval (also known as the minimum allowable discrimination margin between Ri and Rj) given to the *i-th* primary relay.

### 3.3 Constraint Handling Technique

To handle the constrained functions, the penalty method is used in this paper. This method involves augmenting the objective function with a penalty term to discourage infeasible solutions that violate the constraints. In the case of the DOCRs coordination problem, both the relay coordination constraints and the relay characteristic constraints are incorporated into the objective function using the penalty method, as depicted in Equation (15). If any of the constraints are violated, a penalty value is added to the objective function. Given that the objective function is of the minimization type, a large penalty factor is utilized.

$$OF = \min \sum_{i=1}^{n} T_i + \sum_{k=1}^{m} penalty\ (k) \tag{15}$$

where *m* is the number of relay pairs, the penalty term *penalty (k)* is given by the following equation:

$$penalty\ (k) = \begin{cases} 0 & if(T_{j,k} - T_{i,k} \geq CTI) \\ \partial & otherwise \end{cases} \tag{16}$$

where $\partial$ is the penalty factor for penalty method to make the value of the objective function more significant during minimization [58].

## 4 Numerical Experiments

In this study, the coordination problem of the IEEE 3-bus network is solved using the Firefly Algorithm. The objective is to demonstrate the impact of changing the parameters of the FA on its performance. Specifically, the study focuses on analyzing the effects of varying the number of generations, the number of fireflies (population size), the absorption coefficient (γ), and the randomization parameter (α) on the algorithm's performance.

The study intends to assess their impact on the convergence, solution quality, and computing efficiency of the algorithm by systematically changing these parameters and carrying out tests. The algorithm's total number of iterations is determined by the number of generations, allowing for a longer exploration period. The size of the population, or the number of fireflies, has an impact on the algorithm's ability to explore and the variety of solutions produced. The rate of light intensity decrease is determined by the absorption coefficient, which also affects firefly attraction. The level of randomness in firefly movements is controlled by the randomization parameter.

The study aims to demonstrate how different parameter settings affect the FA's ability to find high-quality solutions for the IEEE 3-bus network DOCRs coordination problem through comparative analysis and performance evaluation. By analyzing the effects of these parameters, the study aims to improve the FA and provide suggestions for selecting suitable parameter values for similar coordination problems in power systems. The results were achieved through the development of a precise simulation program using MATLAB software.

**Fig. 3** depicts the IEEE 3-bus network configuration which consists of three buses, three power generators, three branches, and six DOCRs. This network serves as the test case for evaluating the performance of the Firefly Algorithm in solving the coordination problem associated with this specific power system configuration.

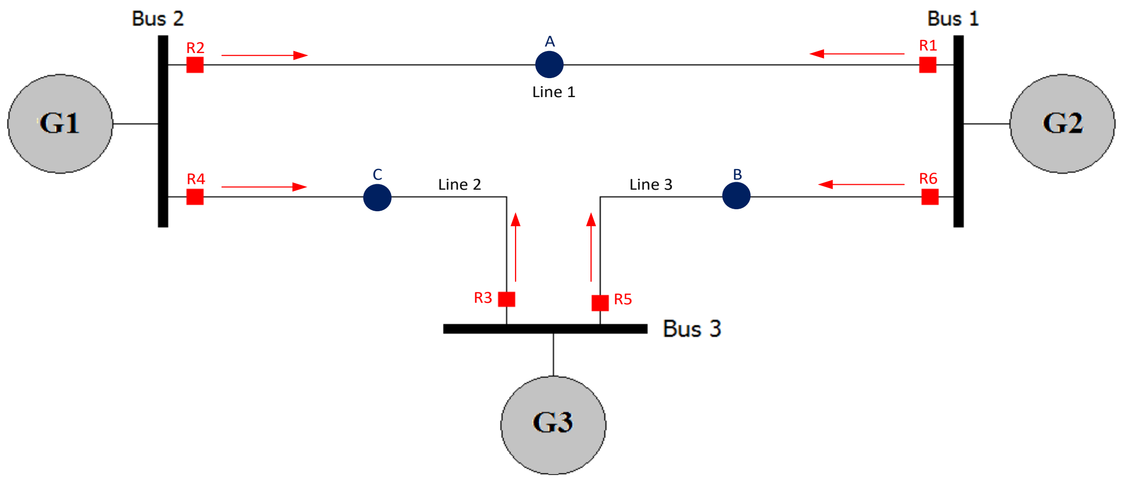

**Fig. 3:** IEEE 3-bus Network.

In this case, the coordination problem is expressed as a non-linear programming problem, where the pickup setting and time multiplier setting are considered as design variables within the range of [1.5, 5.0] and [0.1, 1.1] respectively, both represented as continuous values. Tables 1 and 2 present the results of three-phase short circuits and the current transformer ratio (CTR) of the relays in the IEEE 3-bus system respectively. The problem is subjected to 30 constraints, including 6 inequality conditions for minimum and maximum operating times, 6 inequality conditions for selectivity criteria, and 6 side constraints for both TMS and PS.

**Table 1:** Three-phase short circuit current for IEEE 3-bus network.

| P/B Paris | Primary Relay | Short-circuit current (A) | Backup Relay | Short-circuit current (A) |
|---|---|---|---|---|
| 1 | 1 | 1978.90 | 5 | 175.00 |
| 2 | 2 | 1525.70 | 4 | 545.00 |
| 3 | 3 | 1683.90 | 1 | 617.22 |
| 4 | 4 | 1815.40 | 6 | 466.17 |
| 5 | 5 | 1499.66 | 3 | 384.00 |
| 6 | 6 | 1766.30 | 2 | 145.34 |

**Table 2:** Three-phase short circuit current for IEEE 3-bus network.

| Relay Number | CTR |
|---|---|
| 1,4 | 300/5 |
| 2,3,5 | 200/5 |
| 6 | 400/5 |

## *4.1 Effect of Changing Number of Generations*

The study initially examines the impact of varying the number of generations on the performance of the FA. The objective is to evaluate the effectiveness of the FA across different generations. A range of generation values, ranging from 100 to 2000 with an increment of 100, is considered. The number of fireflies in the algorithm is kept constant at 20, the absorption coefficient is set to 1, and the randomization parameter is set to 0.2.

The study assesses the performance of the FA based on several criteria:

- Solution quality: The objective function value is used as a measure of the quality of the obtained solutions. Lower objective function values indicate better solutions.
- Feasibility of solutions: The number of constraint violations is counted to evaluate the feasibility of the solutions. A lower number of violations indicates more feasible solutions.
- Number of objective function evaluations: The study considers the number of times the objective function is evaluated during the algorithm's execution. This metric reflects the computational effort required by the algorithm.
- Computation time: The elapsed time needed to execute the FA. It provides insights into the algorithm's efficiency and computational speed.

By comparing these metrics across the different numbers of generations, the study aims to determine the optimal number of generations that leads to improved solution quality, increased feasibility, reduced computational effort, and reasonable computation time.

**Table 3:** Simulation results for different number of generations.

| **Criteria/Number of Generations** | **100** | **200** | **300** | **400** | **500** | **600** | **700** | **800** | **900** | **1000** |
|---|---|---|---|---|---|---|---|---|---|---|
| Objective function value | 1.8605 | 1.8246 | 1.7247 | 1.7242 | 1.7242 | 1.6889 | 1.6509 | 1.632 | 1.4528 | 1.4052 |
| Number of violations | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 2 | 0 | 1 |
| Number of objection function evaluations | 2020 | 4020 | 6020 | 8020 | 10020 | 12020 | 14020 | 16020 | 18020 | 20020 |
| Computation time | 4.9482826 | 5.382310 | 5.499888 | 5.669091 | 5.798593 | 6.012650 | 6.183932 | 6.341228 | 6.532285 | 6.780342 |
| **Criteria/Number of Generations** | **1100** | **1200** | **1300** | **1400** | **1500** | **1600** | **1700** | **1800** | **1900** | **2000** |
| Objective function value | 1.4052 | 1.4015 | 1.4015 | 1.3974 | 1.3879 | 1.3879 | 1.3879 | 1.3758 | 1.3758 | 1.3758 |
| Number of violations | 2 | 2 | 1 | 1 | 2 | 2 | 1 | 3 | 2 | 2 |
| Number of objection function evaluations | 22020 | 24020 | 26020 | 28020 | 30020 | 32020 | 34020 | 36020 | 38020 | 40020 |
| Computation time | 7.0222451 | 7.270984 | 7.603479 | 7.931434 | 8.244370 | 8.538376 | 8.858966 | 9.164575 | 9.457850 | 9.943240 |

Table 3 presents the results obtained for various criteria at different numbers of generations in the FA. It is shown that as the number of generations increases, the objective function value generally decreases. This indicates that with more generations, the algorithm is able to converge towards better solutions. However, after a certain point (around 1600 generations), the objective function value seems to reach a plateau, indicating that further rising the number of generations does not significantly improve the solution quality. It is also shown that the number of constraint violations varies with the number of generations. At lower generations (100, 200), there is one violation, but as the number of generations increases, the algorithm produces feasible solutions with zero violations. However, there is a slight increase in violations at higher generations (1700, 1800,1900), indicating a potential balance between solution quality and feasibility. Also, the number of objective function evaluations increases linearly with the number of generations. This is expected as each generation involves evaluating the objective function for all individuals in the population. Computation time generally increases with the number of generations, indicating that more iterations require more computational resources. However, the increase in computation time is not proportional to the number of generations, indicating that the algorithm may reach a point of diminishing returns in terms of computational efficiency. The results indicate that increasing the number of generations in the FA can lead to improved solution quality and feasibility. However, there is a balance to be struck, as very high numbers of generations may not significantly improve the results but would increase the computational time. It is shown that from the results at number of generation equal to 900 is the best because it gives less objective function value with no violations.

The analysis reveals several key findings:

- It is observed that as the number of generations increases, the objective function value decreases as demonstrated in **Fig. 4**. This indicates that with more generations, the algorithm converges towards better solutions. However, after reaching around 1600 generations, the objective function value stabilizes, indicating that further increasing the number of generations does not significantly enhance the solution quality.
- The number of constraint violations varies with the number of generations. At lower generations (100, 200), there is one violation, but as the number of generations increases, the algorithm consistently produces feasible solutions with zero violations as shown in **Fig. 5**. However, a slight increase in violations is observed in higher generations (1700, 1800, 1900), indicating a potential balance between solution quality and feasibility.

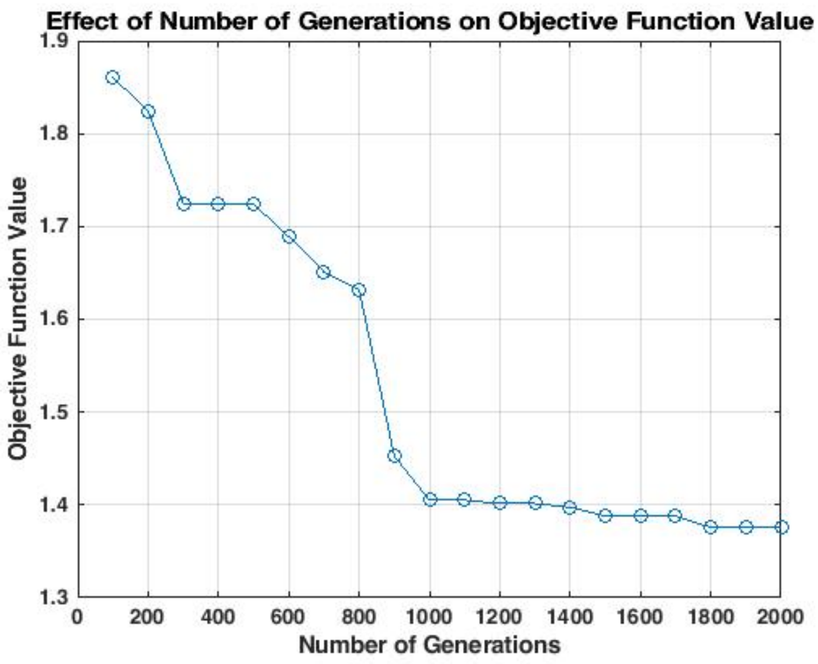

**Fig. 4:** Effect of changing number of generations on the objective function value.

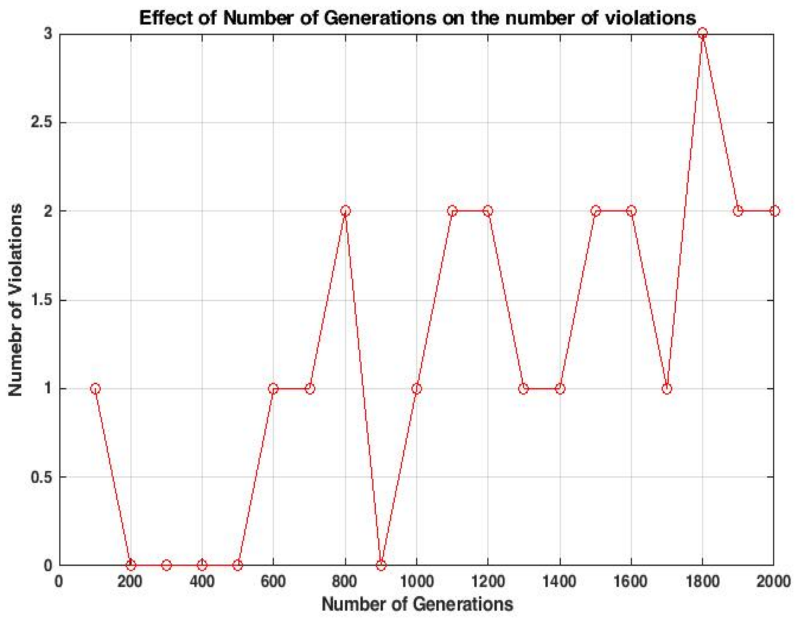

**Fig. 5:** Effect of changing number of generations on the number of violations.

- The number of objective function evaluations increases linearly with the number of generations as illustrated in **Fig. 6**. This is expected since each generation involves evaluating the objective function for all individuals in the population.
- The computations conducted in this paper were executed on a computer system equipped with a 1.8 GHz Intel Core i3 processor and 8 GB of RAM. The authors developed the source code using Matlab, version 2021a. As shown in **Fig.** 7, the results of the study demonstrate a linear relationship between the number of generations and computation time. The computation time increases along with the number of generations, indicating the need for more computational resources to carry out more iterations. This result illustrates that as the number of generations increases, the algorithm's computational efficiency decreases.

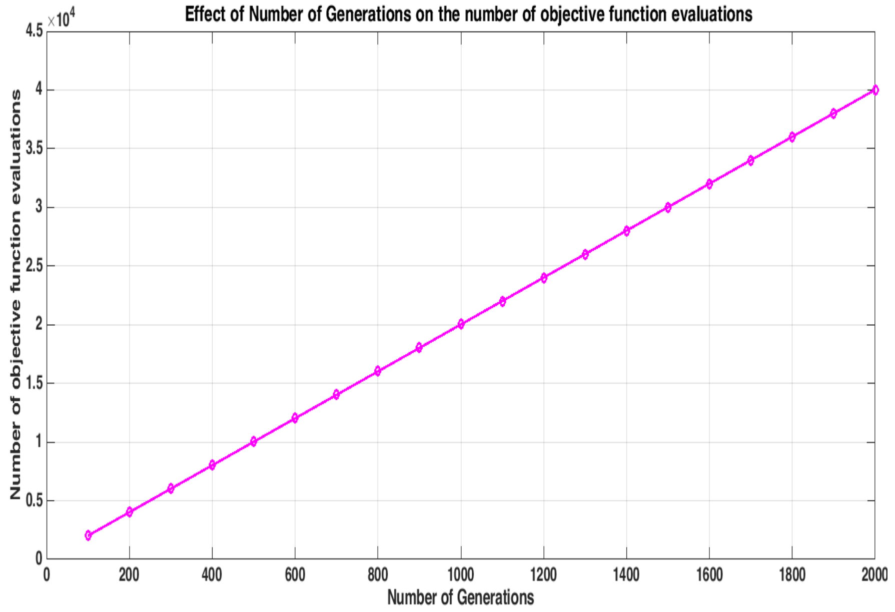

**Fig. 6:** Effect of changing number of generations on the number of objective function evaluation.

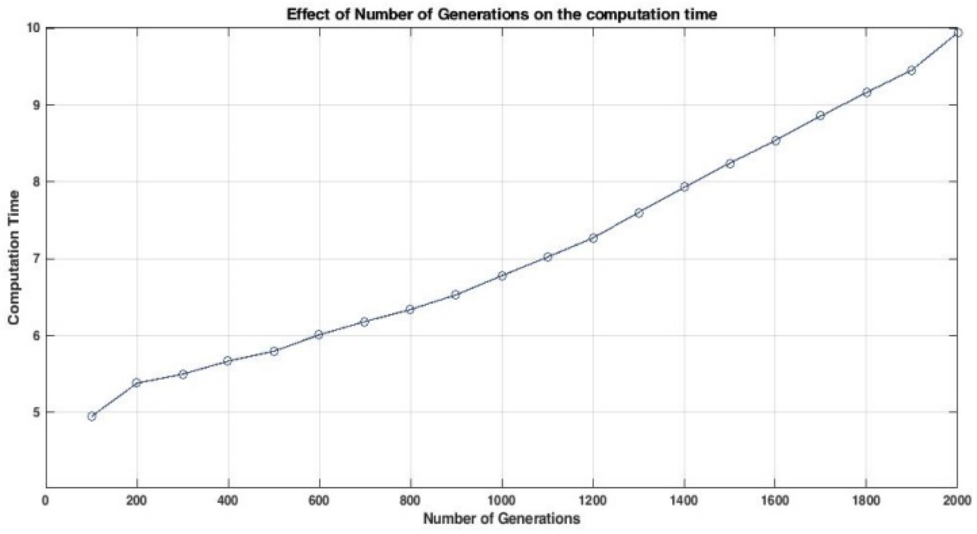

**Fig. 7:** Effect of changing number of generations on the computation time.

In conclusion, the results demonstrated that increasing the number of generations in the FA can lead to improved solution quality and feasibility. However, a balance must be struck, as excessively high numbers of generations may not significantly enhance the results but would increase the computational time. It is clear from the results that 900 generations produce the best results because they produce a lower objective function value with no violation of the constraints.

## *4.2 Effect of Changing the Population Size*

This section presents an analysis of the influence of population size on the performance of the FA. The objective is to evaluate the efficiency of the FA under different population sizes. To investigate this effect, a range of values for the number of fireflies is considered, ranging from 10 to 200 with a step size of 10. The remaining parameters of the algorithm are kept constant, with the number of generations set to 900 (determined from the previous analysis), the absorption coefficient set to 1, and the randomization parameter set to 0.2. Table 4 provides the results obtained for different criteria at varying population sizes in the FA. The analysis reveals several key findings:

- The objective function value decreases as the number of fireflies increases as illustrated in **Fig. 8**. This indicates that a larger population size enables the algorithm to explore the solution space more effectively, leading to improved solutions. However, there is a slight increase in the objective function value at the number of fireflies 90, 160, 190 and 200, indicating that very large population sizes may not necessarily lead to further improvement.

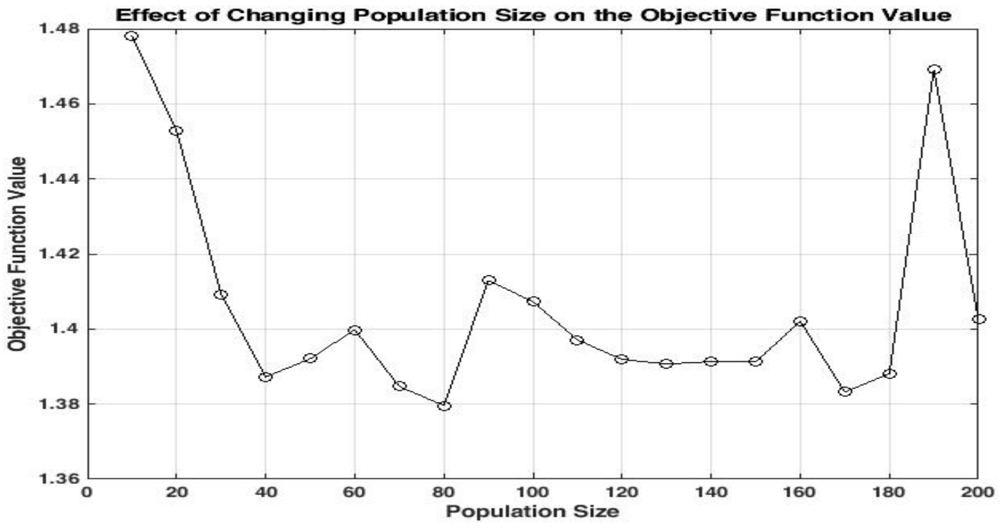

**Fig. 8:** Effect of changing population size on the objective function value.

- For a smaller population size of 10 fireflies, there were 2 constraint violations. However, by increasing the population size to 20 fireflies, the number of violations decreased to 0. This suggests that with a larger population, the algorithm was able to generate feasible solutions without any constraint violations. As the population size further increased from 30 to 200 fireflies, the number of violations ranged between 1 and 3. There was a slight increase in violations at population sizes of 130, 140, 160, and 180, reaching a peak of 3 violations as demonstrated in **Fig. 9**.

**Table 4:** Simulation results for different numbers of fireflies.

| **Criteria/Number of Fireflies** | **10** | **20** | **30** | **40** | **50** | **60** | **70** | **80** | **90** | **100** |
|---|---|---|---|---|---|---|---|---|---|---|
| Objective function value | 1.4781 | 1.4528 | 1.4093 | 1.3872 | 1.3921 | 1.3998 | 1.3847 | 1.3796 | 1.4129 | 1.4073 |
| Number of violations | 2 | 0 | 1 | 1 | 1 | 2 | 1 | 1 | 2 | 2 |
| Number of objection function evaluations | 9010 | 18020 | 27030 | 36040 | 45050 | 54060 | 63070 | 72080 | 81090 | 90100 |
| Computation time | 45.089 | 45.33124 | 45.64397 | 45.98139 | 46.40564 | 46.87077 | 47.42224 | 47.94415 | 48.74911 | 49.51412 |
| **Criteria/Number of Fireflies** | **110** | **120** | **130** | **140** | **150** | **160** | **170** | **180** | **190** | **200** |
| Objective function value | 1.3971 | 1.3919 | 1.3907 | 1.3913 | 1.3912 | 1.4021 | 1.3832 | 1.3881 | 1.4691 | 1.4025 |
| Number of violations | 2 | 1 | 2 | 3 | 1 | 1 | 3 | 2 | 2 | 1 |
| Number of objection function evaluations | 99110 | 108120 | 117130 | 126140 | 135150 | 144160 | 153170 | 162180 | 171190 | 180200 |
| Computation time | 50.225843 | 52.58907 | 53.35569 | 54.36403 | 55.46911 | 56.42490 | 57.53044 | 58.51649 | 60.79457 | 61.93083 |

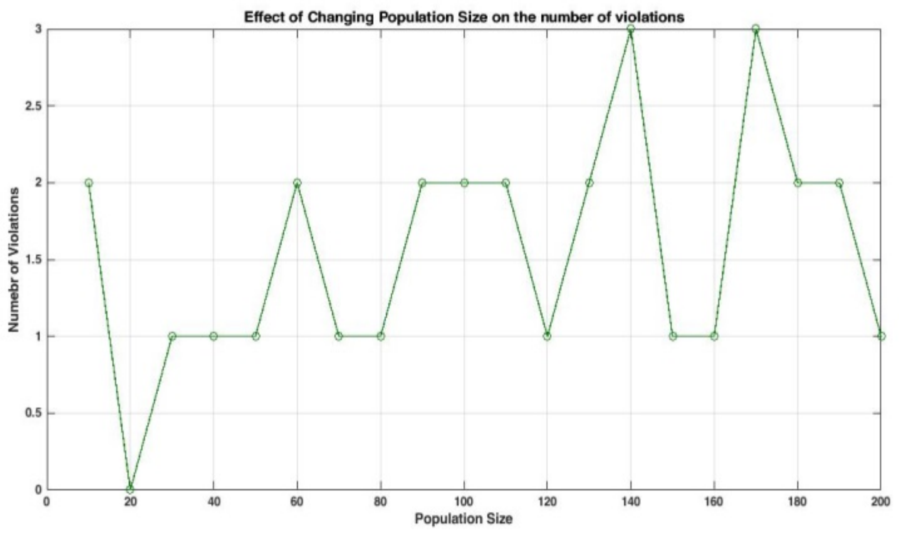

**Fig. 9**: Effect of changing population size on the number of violations.

- The results demonstrate that increasing the population size generally improves the feasibility of solutions, as evidenced by the reduction in violations from 2 to 0 when transitioning from 10 to 20 fireflies. However, at larger population sizes, there is a potential balance between solution quality and feasibility. This is evident from the slight increase in violations observed at population sizes of 130, 140, 160, and 180, reaching a peak of 3 violations. Therefore, selecting an optimal population size requires careful consideration of balancing solution quality and feasibility to achieve the desired results.
- As the population size increased from 10 to 200 fireflies, the number of objective function evaluations followed a linear relation as shown in **Fig. 10**. Specifically, the number of evaluations progressively increased in increments of 9010, reflecting the number of individuals in the population. For instance, with 10 fireflies, there were 9010 evaluations, and with 20 fireflies, the number of evaluations doubled to 18020. This linear relationship continued as the population size increased, with each additional 10 fireflies resulting in an additional 9010 evaluations. Consequently, at a population size of 200 fireflies, the algorithm required 180200 evaluations. These results indicate that the number of objective function evaluations is directly proportional to the population size in the FA. Therefore, larger population sizes demand more computational resources and increase the computational effort needed to obtain solutions.

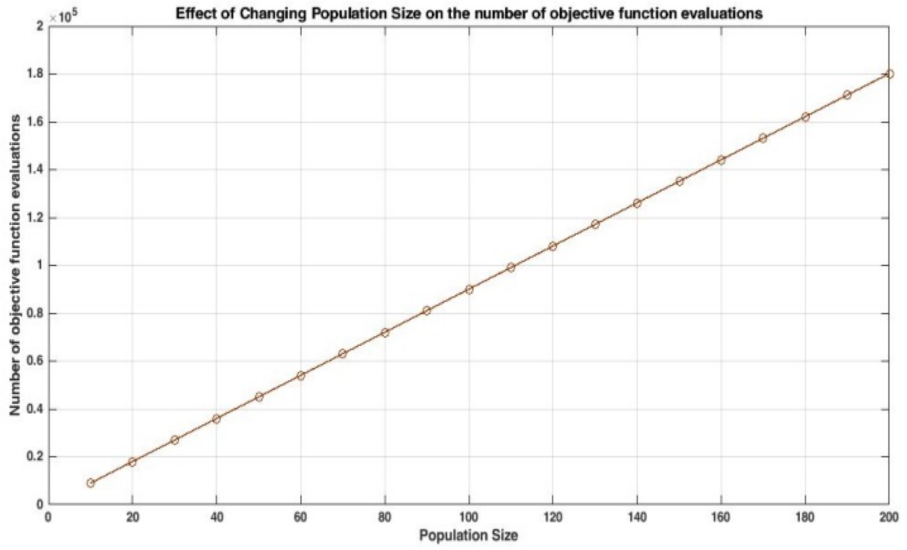

**Fig. 10:** Effect of changing population size on the number of function evaluation.

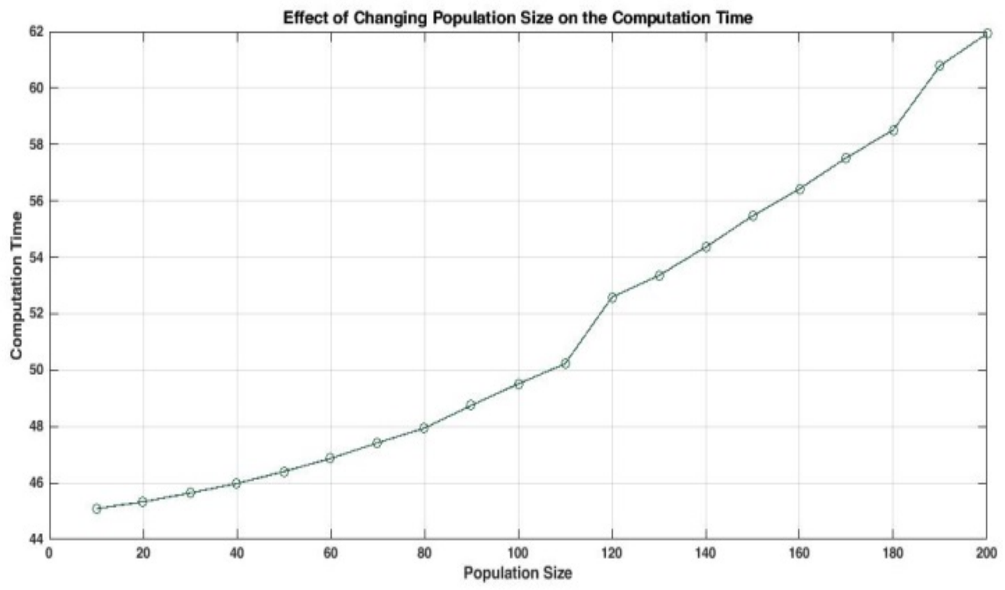

**Fig. 11:** Effect of changing population size on the computation time.

- It was found that the computing time increased together with the number of fireflies, as shown in **Fig. 11.** The computation time exhibited only minor changes at smaller population numbers (10 to 80 fireflies). A progressive increase in computation time was seen as the population size increased beginning at 80 fireflies. As the population became larger than 100 fireflies, this increase in computing time became more evident. For instance, the computation required 45.09 seconds when there were 10 fireflies. The computation took 61.93 seconds when there were 200 fireflies in the population. This shows that carrying out the process of optimization requires more time and computational resources when population numbers are bigger. It also demonstrated that, depending on a number of variables, including the complexity of the optimization issue and the way the method is implemented, the relationship between population size and computing time might not be strictly linear. However, the relation indicates that longer calculation times are associated with larger populations in the FA. Therefore, it is essential to take into account the balance between computation time and solution quality while using the FA.

Changing the population size in the FA has been studied, and the results provide some significant observations. As seen by the reduction of constraint violations from 2 to 0 when going from 10 to 20 fireflies, increasing the population size generally makes solutions more feasible. The minor increase in violations that is seen at higher population sizes (30 to 200 fireflies) indicates a possible balance between solution quality and its feasibility. A greater population size may enable a better exploration of the solution space and the finding of improved solutions, as the objective function value normally decreases as the number of fireflies grows. However, there is a slight increase in the objective function value at certain population sizes (90, 160, 190, and 200), indicating that very large population sizes may not necessarily lead to further improvements and could potentially hinder the algorithm's performance. Additionally, there is a linear relationship between the amount of objective function evaluations and population size, with an increase of 9010 evaluations for every additional 10 fireflies. This demonstrates the correlation between population size and computing effort, with larger populations requiring more time and resources to solve problems. Additionally, the computation time grows with the number of fireflies, particularly after 100 fireflies, indicating the increasing computing power needed for larger population sizes. Choosing an optimal population size requires consideration of calculation time and solution quality. In order to choose the best population size in the FA, it is necessary to strike a balance between time constraints, computing efficiency, and solution quality. The results indicate that growing population size generally enhances the feasibility and quality of the solutions, but that there are decreasing returns at a certain level.

## *4.3 Effect of Changing the Randomization Parameter (α)*

In this section, the authors examine the impact of varying the randomization parameter on the performance of the FA. The objective is to assess the efficiency of the FA using different values of α. To investigate this effect, a range of α values from 0.1 to 0.9 with a step size of 0.1 is explored. The other parameters of the algorithm are held constant: the number of generations is set to 900, the number of fireflies is set to 20 (as determined from the previous analysis), and the absorption coefficient is set to 1. The results obtained for various criteria at different α values are summarized in Table 5. The criteria considered include the objective function value, number of violations, number of objective function evaluations, and computation time. The analysis reveals several key findings:

**Table 5:** Simulation results for different values of randomization parameter.

| **Criteria/Value of randomization parameter.** | **0,1** | **0.2** | **0.3** | **0.4** | **0.5** | **0.6** | **0.7** | **0.8** | **0.9** |
|---|---|---|---|---|---|---|---|---|---|
| Objective function value | 1.4022 | 1.4528 | 1.4274 | 1.6041 | 1.7807 | 1.7716 | 766231855.3 | 1.7694 | 1.8441 |
| Number of violations | 3 | 0 | 0 | 1 | 1 | 0 | 2 | 2 | 2 |
| Number of objection function evaluations | 18020 | 18020 | 18020 | 18020 | 18020 | 18020 | 18020 | 18020 | 18020 |
| Computation time | 87.87 | 88.07 | 88.32 | 88.59 | 88.78 | 88.93 | 89.083 | 89.4325 | 90 |

- The objective function value varies with different values of α. It is evident that the best objective function value is achieved when α is set to 0.3. This indicates that a moderate level of randomization contributes to improving the solution quality. The objective function value gradually increases as α deviates from 0.3 as shown in **Fig. 12**. When α is set to 0.7, an extremely large objective function value of 766231855.3 is obtained, suggesting that this value of α adversely affects the algorithm's performance. Deviating too much from this optimal value (e.g., α = 0.1 or α = 0.9) leads to higher objective function values, indicating that excessive or insufficient randomization may impact the algorithm's performance.

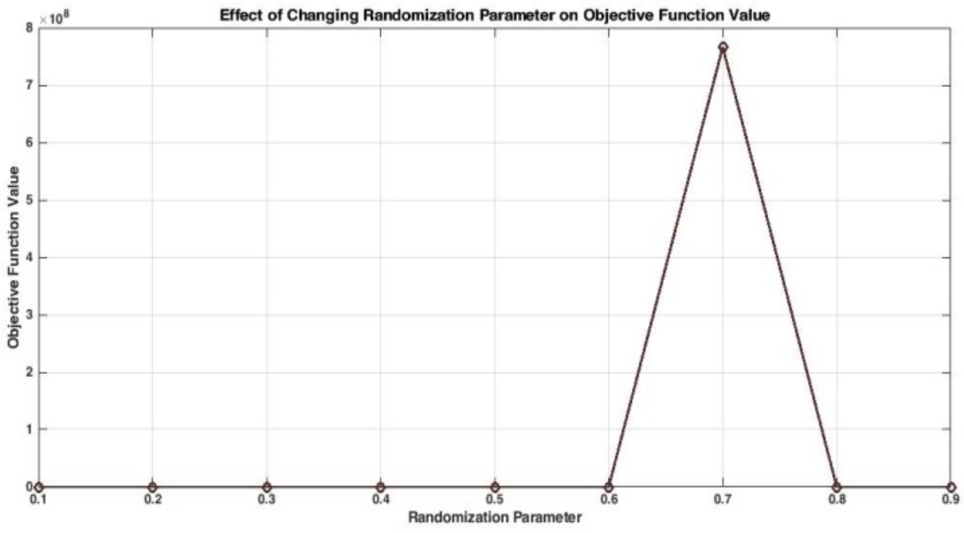

**Fig. 12:** Effect of changing randomization parameter on the objective function value.

- The number of violations indicates the extent to which the solutions generated by the FA satisfy the problem's constraints for different values of the α. **Fig. 13** visually shows the effect of changing α on the algorithm's performance. When α is set to 0.2 and 0.3, the FA produces solutions with zero violations, indicating that it successfully maintains constraint satisfaction. These values of α strike a balance between exploration and exploitation, allowing the algorithm to effectively explore the solution space while handling to the constraints.

For α values of 0.4 and 0.5, we observed one violation in each case. This indicates that a slight increase in α may introduce some balance between solution quality and constraint satisfaction, resulting in a single violation. At α = 0.6, the FA once again generates solutions without violating any constraints, indicating its ability to find feasible solutions while exploring the search space. However, as α increases to 0.7, 0.8, and 0.9, we observed two violations for each value. This indicates that higher values of α may lead to a decrease in constraint satisfaction, indicating a shift towards more exploratory behavior that sacrifices adherence to constraints. However, the number of violations reveals that the FA performs well in terms of constraint satisfaction for most α values, with zero or a limited number of violations. However, an increase in α beyond a certain threshold (around 0.7) negatively affects the algorithm's ability to generate feasible solutions, leading to an increased number of violations.

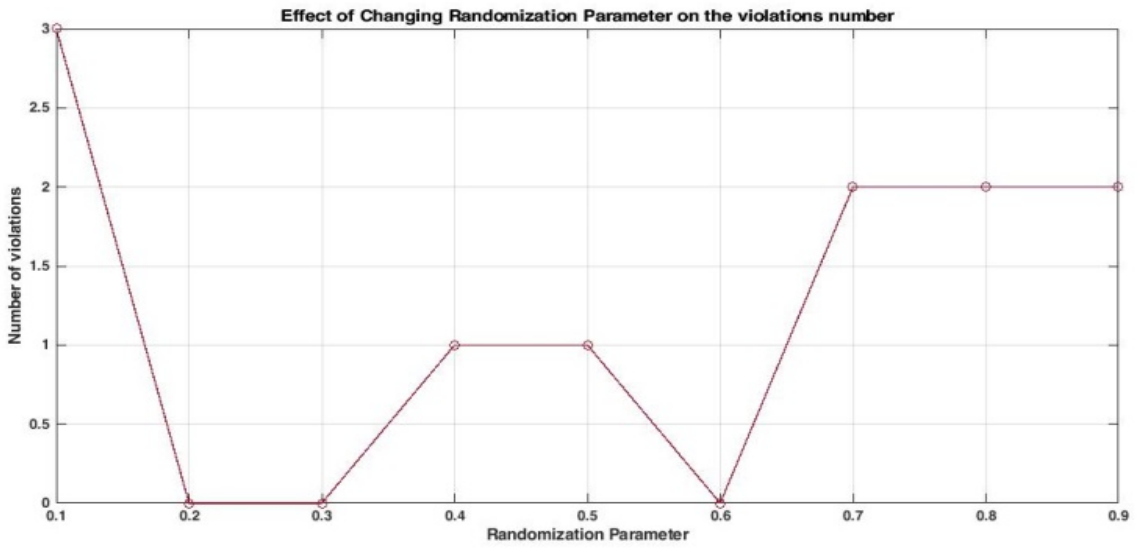

**Fig. 13:** Effect of changing randomization parameter on the violations number.

- The number of objective function evaluations measures the computational effort needed by the FA to search for optimal solutions for different values of α. **Fig. 14** provides a visual representation of the relationship between α and the number of objective function evaluations. Across all values of α, the number of objective function evaluations remains constant at 18,020. This indicates that the FA searches the solution space with the same number of evaluations regardless of the randomization parameter. Therefore, the number of objective function evaluations remains stable throughout the analysis, indicating that the computational effort required by the FA is independent on the randomization parameter.

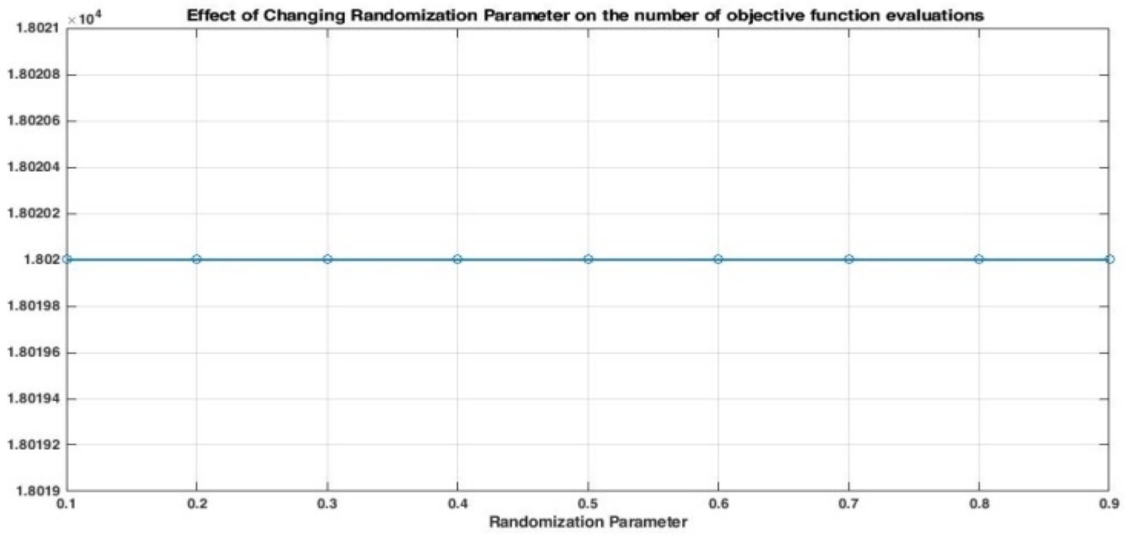

**Fig. 14:** Effect of changing randomization parameter on the number of objective function evaluations.

- The computation time represents the amount of time required by the FA to complete its execution for different values of the α. **Fig. 15** provides a visual representation of the relationship between α and computation time. From the results, it can be observed that the computation time increases gradually as the value of α increases. This indicates that higher values of α require more computational resources and time for the FA to converge to a solution. However, the computation time of the FA gradually increases with higher values of the randomization parameter α. Therefore, when selecting α, it is important to consider the balance between solution quality and computational efficiency, as higher α values may lead to longer computation times.

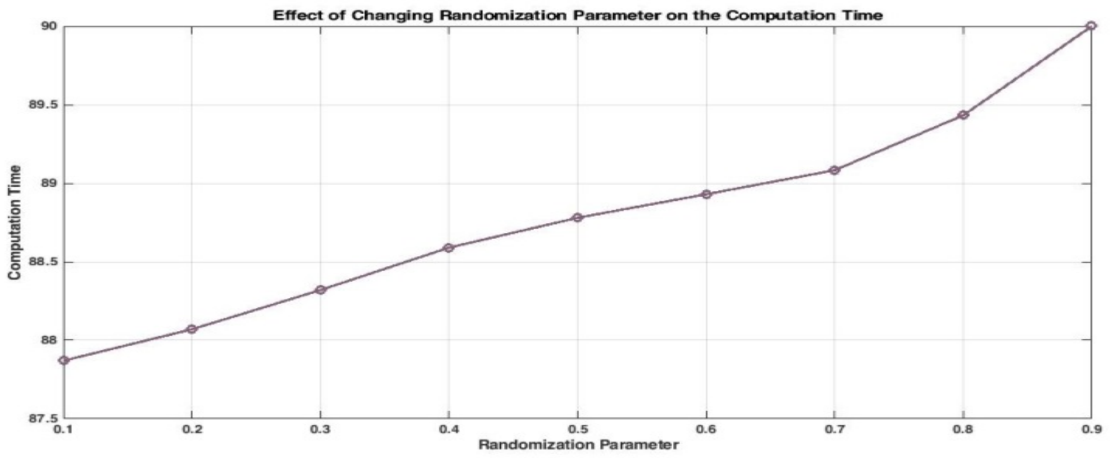

**Fig. 15:** Effect of changing Randomization Parameter on the computation time.

According to the analysis of the effects of changing the randomization parameter on the Firefly Algorithm's performance, setting the randomization parameter to 0.3 yields the best results in terms of objective function value and feasibility (number of violations). This shows that a moderate level of randomness strikes a balance between exploration and exploitation, allowing the algorithm to progress toward better solutions while meeting the set of constraints. Across a range of values, the quantity of objective function evaluations and computation time remain largely constant.

## *4.4 Effect of Changing the Absorption Coefficient (γ)*

This section investigates the impact of modifying the absorption factor on the performance of the FA. The objective is to evaluate how the algorithm performs under different γ values. To analyze this effect, a range of γ values from 0.0001 to 100, covering multiple orders of magnitude, is examined. The remaining parameters of the algorithm are kept constant, with the number of generations set to 900, the number of fireflies set to 20, and the randomization parameter set to 0.3 based on the previous analysis. The results obtained for various performance criteria at different γ values are summarized in Table 6. These criteria provide insights into the overall performance of the algorithm, its handling of constraints, the efficiency of its computations, and the quality of its solutions.

**Table 6:** Simulation results for different values of absorption coefficient.

| **Criteria/Value of absorption coefficient** | **0,0001** | **0.001** | **0.01** | **0.1** | **1** | **10** | **100** |
|---|---|---|---|---|---|---|---|
| Objective function value | 1.4731 | 1.5103 | 1.7787 | 1.6834 | 1.4274 | 2.1706 | 2.1706 |
| Number of violations | 2 | 0 | 2 | 2 | 0 | 2 | 2 |
| Number of objection function evaluations | 18020 | 18020 | 18020 | 18020 | 18020 | 18020 | 18020 |
| Computation time | 95.24 | 95.48 | 95.68 | 95.84 | 96.45 | 96.69 | 96.92 |

- The objective function value shows variation with different γ values. When using lower γ values (0.0001 and 0.001), the objective function value increases, indicating poorer solution quality. However, as γ increases, the objective function value decreases and reaches its minimum at γ = 1 (1.4274). Subsequently, for higher γ values (10 and 100), the objective function value starts to increase again. A visual representation of the relationship between γ and the objective function values can be observed in **Fig. 16**. This indicates the existence of an optimal absorption factor range that leads to better solution quality, while extreme γ values may negatively impact the algorithm's performance. Based on the results, the FA performs well in terms of optimizing the objective function for this problem when the absorption factor is set to 1, resulting in the lowest objective function value. However, the optimal absorption factor may vary for different problems, as the objective function value is specific to each problem.
- The number of violations indicates the extent to which the solutions generated by the FA satisfy the problem's constraints for different values of the γ. The number of violations varies with different γ values. **Fig. 17** visually illustrates the effect of changing γ on the number of violations. At γ = 0.0001 and γ = 0.01, there are two violations,

indicating that the FA is not able to produce feasible solutions that satisfy all the problem constraints. This indicates that very small values of γ may impact the algorithm's ability to maintain constraint satisfaction. However, at γ = 0.001, γ = 0.1, and γ = 1, there are zero violations, indicating that the algorithm successfully generates feasible solutions without violating any constraints. This indicates that these values of γ allow the algorithm to effectively explore the solution space while maintaining constraint satisfaction. As the absorption factor increases further (γ = 10 and γ = 100), the number of violations increases again to two. This indicates that higher values of γ may lead to a decrease in constraint satisfaction, potentially indicating a shift towards more exploratory behavior in the algorithm that sacrifices constraint adherence. It can be concluded that γ values in the range of 0.001 to 1 result in feasible solutions with zero violations, indicating better constraint satisfaction. Extreme values of γ (very small or very large) may lead to an increased number of violations.

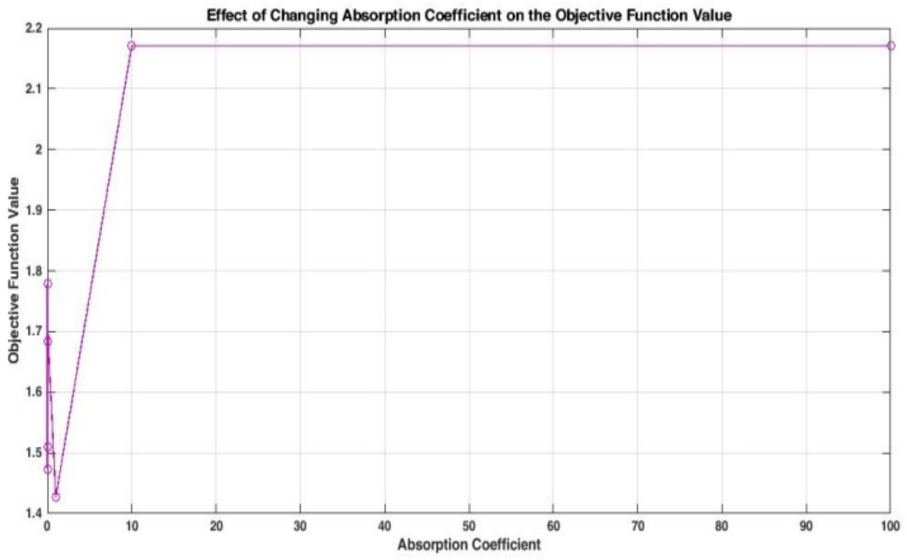

**Fig. 16:** Effect of changing absorption coefficient on the objective function value.

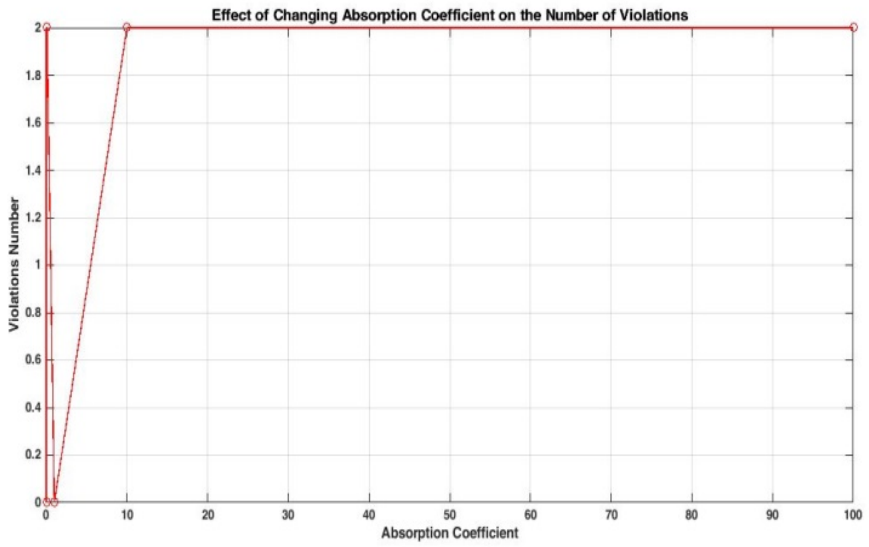

**Fig. 17:** Effect of changing absorption coefficient on the violations number.

- The number of objective function evaluations in the FA remains constant at 18020 for all values of the absorption factor (γ). This shows that the algorithm requires the same number of objective function evaluations regardless of the specific absorption factor chosen. This observation is visually represented in **Fig. 18**, which demonstrates the consistent relationship between γ and the number of objective function evaluations. The constant number of evaluations implies that the absorption factor does not directly influence the computational effort involved in evaluating the objective function.

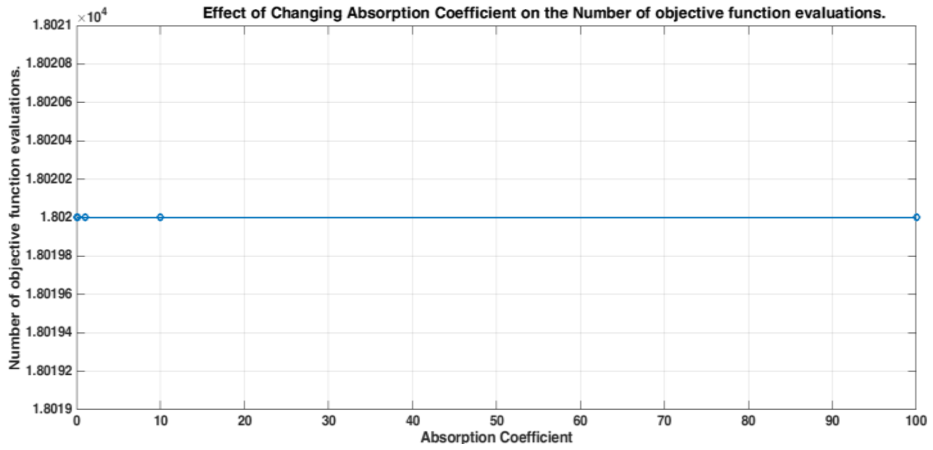

**Fig. 18:** Effect of changing absorption coefficient on the number of objective function evaluations.

- For various values of γ, the computation time—which shows the algorithm's computational effectiveness—was investigated. The results indicated that the computation time somewhat increases when the absorption factor increases from 0.0001 to 1. The rise is still modest and fits within a constrained range. The computation time is very consistent above an absorption value of 1, indicating that further raising the absorption factor has little effect on the amount of computation required to evaluate the objective function. **Fig. 19** provides a visual representation of the relationship between the γ and the computation time.

In conclusion, setting the absorption coefficient to γ = 1 in the FA leads to improved solution quality and constraint satisfaction. The algorithm exhibits consistent computational efficiency across different absorption factor values, enabling reliable comparisons and practical implementation. However, it is important to consider the specific problem domain and adjust the absorption factor accordingly for optimal performance.

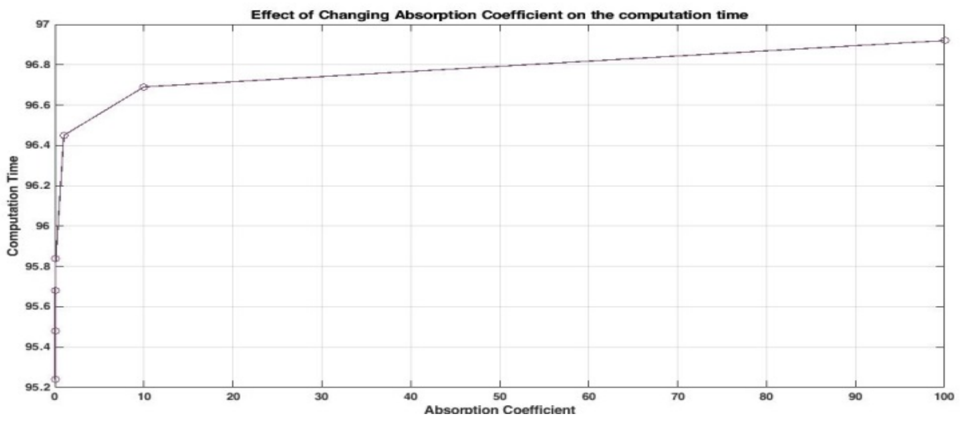

**Fig. 19:** Effect of changing absorption coefficient on the computation time.

## 5 Conclusions and Future Works

In this study, authors applied the Firefly Algorithm to address the coordination problem of the IEEE 3-bus network. The objective of this study was to analyze the effect of key parameters, namely the number of generations, population size, absorption coefficient, and randomization parameter, on the algorithm's performance. Through extensive experimentation and performance evaluation, the study demonstrates the effects of these parameters on solution quality, feasibility, computational effort, and efficiency. The analysis of the number of generations revealed that increasing this parameter led to improved solution quality, as evidenced by a decrease in the objective function value. However, beyond a certain threshold (around 1600 generations), further increasing the number of generations did not significantly enhance the solution quality. Feasibility of solutions also improved with higher numbers of generations, with zero constraint violations observed for larger generation values. Nonetheless, a slight increase in violations at very high generation values suggests a balance between solution quality and feasibility. Moreover, the number of objective function evaluations and computation time exhibited a linear relationship with the number of generations, highlighting the need for increased computational resources as the iterations progressed. Regarding the population size, the results indicated that larger populations generally resulted in better solution quality, as reflected by a decrease in the objective function value. Feasibility also improved with larger population sizes, accompanied by a reduction in constraint violations. However, a slight increase in violations was seen at very large population sizes, indicating a possible balance between solution quality and feasibility. The number of objective function evaluations increased proportionally with population size, as did computation time, underscoring the demand for increased computational resources with larger populations. The randomization parameter exerted a noticeable but less significant influence on the algorithm's performance compared to the number of generations and population size. Modulating α did not yield significant changes in solution quality, feasibility, or computation time. However, slight variations were observed, including a decrease in the objective function value and an increase in constraint violations at extreme α values. These results show that a moderate α value within the explored range is preferable. Finally, the absorption coefficient emerged as an important factor impacting the algorithm's convergence and solution quality. Lower γ values expedited convergence but could lead to suboptimal solutions, while higher γ values fostered exploration but demanded increased computational effort. Fine-tuning γ represents a potential avenue for enhancing the algorithm's performance. This study comprehensively investigated the effects of key parameters in the FA for solving the coordination problem of the IEEE 3-bus network. Further research could focus on refining these parameters and exploring adaptive techniques to achieve optimal performance in coordination problems for power systems.

## Conflict of interest

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

## References

[1] X. Yang, and X. He, Firefly algorithm: recent advances and applications, *International Journal of Swarm Intelligence*, **1** (1), 36-50 (2013).

[2] Á. Eiben, R. Hinterding, and Z. Michalewicz, Parameter control in evolutionary algorithms, *IEEE Trans. Evol. Comput.* **3**, 124–141 (1999).

[3] A. Saremi, T. ElMekkawy, and G. Wang, Tuning the parameters of a memetic algorithm to solve vehicle routing problem with backhauls using design of experiments, *Int. Journal Oper. Res.,* **4**, 206–219 (2007).

[4] R. Olympia, F. Stefka, and M. Paprzycki, *Influence of the population size on the genetic algorithm performance in case of cultivation process modelling, in: Fed*, In Proc. 2013 Federated Conference on Computer Science and Information Systems, 371–376 (2013).

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

[6] O. Verma, D. Aggarwal, and T. Patodi, Opposition and dimensional based modified firefly algorithm, *Expert Syst. Appl.*, **44**, 168–176 (2016).

[7] A. Gandomi, X. Yang, S. Talatahari, and A. Alavi, Firefly algorithm with chaos, *Commun. Nonlinear Sci. Numer. Simul.,* **18,** 89–98 (2013).

[8] W. Khan, N. Hamadneh, S. Tilahun, and J. Ngnotchouye, A review and comparative study of firefly algorithm and its modified versions, *Optim. Algorithms-Methods Appl*., **45**, 281–313 (2016).

[9] B. Amiri, L. Hossain, J. Crawford, and R. Wigand, Community detection in complex networks: Multi–objective enhanced firefly algorithm, *Knowledge-Based Syst.,* **46,** 1–11 (2013).

[10] L. Bagadi, G. Rao, and N. Kumar, Firefly, Teaching Learning Based Optimization and Kalman Filter Methods for GPS Receiver Position Estimation, *Procedia Comput. Sci.,* **143,** 892–898 (2018).

[11] K. Dhal, S. Sahoo, A. Das, and S. Das, *Effect of population size over parameter-less firefly algorithm*, in Appl. Firefly Algorithm Its Var., N. Dey, Case Stud. New Dev., Springer, 237–266 (2019).

[12] T. Foqha, S. Alsadi, S. Refaata, and K. Abdulmawjood, Experimental Validation of a Mitigation Method of Ferranti Effect in Transmission Line, *IEEE Access.* **11,** 15878–15895 (2023).

[13] X. Yang, *Nature-inspired optimization algorithms*, Academic Press, (2020).

[14] Y. Mo, Y. Ma, and Q. Zheng, *Optimal choice of parameters for firefly algorithm*, in 2013 Fourth Int. Conf. Digit.

Manuf. Autom., IEEE, 887–892 (2013).

[15] S. Arora, and S. Singh, The firefly optimization algorithm: convergence analysis and parameter selection, *Int. J. Comput. Appl.,* **69,** 48-52 (2013).

[16] P. Kumar, O. AlZaabi, K. Al Jaafari, P. Kumar, K. Al Hosani, and U. Muduli, *Comparative Assessment of Optimization Techniques Employed for Coordination of Directional Overcurrent Relays*, in 2023 IEEE IAS Glob. Conf. Renew. Energy Hydrog. Technol., IEEE, 1–6 (2023).

[17] X. Yang, Multiobjective firefly algorithm for continuous optimization, *Eng. Comput.*, **29**, 175–184 (2013).

[18] I. Fister, I. Fister Jr, X. Yang, and J. Brest, A comprehensive review of firefly algorithms, *Swarm Evol. Comput.* **13**, 34–46 (2013).

[19] S. Farahani, A. Abshouri, B. Nasiri, and M. Meybodi, A Gaussian firefly algorithm, *Int. J. Mach. Learn. Comput.,* **1**, 448 (2011).

[20] M. H. Hussain, I. Musirin, A.F. Abidin, S.R.A. Rahim, Multi-objective approach for solving directional overcurrent relay problem using modified firefly algorithm, Delta. 3 (2001) 21–26.

[21] M. Ravber, S. Liu, M. Mernik, and M. Črepinšek, Maximum number of generations as a stopping criterion considered harmful, *Appl. Soft Comput.,* **128**, 109478 (2022).

[22] I. Fister Jr, M. Perc, S. Kamal, and I. Fister, A review of chaos-based firefly algorithms: perspectives and research challenges, *Appl. Math. Comput.,* **252,** 155–165 (2015).

[23] X. Yang, *Firefly algorithms for multimodal optimization*, in Stoch. Algorithms Found. Appl. 5th Int. Symp. SAGA 2009, Sapporo, Japan, Proc. 5, Springer, 169–178 (20090.

[24] X. Yang, *Cuckoo search and firefly algorithm*, in theory and applications, Springer, (2013).

[25] M. Sulaiman, S. Muhammad, and A. Khan, Improved solutions for the optimal coordination of docrs using firefly algorithm, *Complexity*, **2018** (2018).

[26] N. Cheung, X Ding, and H. Shen, Adaptive firefly algorithm: parameter analysis and its application, *PLoS One.*, **9 (**e112634**)** (2014).

[27] J. Brest, S. Greiner, B. Boskovic, M. Mernik, and V. Zumer, Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems, *IEEE Trans. Evol. Comput.* **10,** 646–657 (2006).

[28] A. Tjahjono, D. Anggriawan, A. Faizin, A. Priyadi, M. Pujiantara, T. Taufik, and M. Purnomo, Adaptive modified firefly algorithm for optimal coordination of overcurrent relays, *IET Gener. Transm. Distrib.*, **11,** 2575–2585 (2017).

[29] S. Alsadi, and Y. Nassar, A general expression for the shadow geometry for fixed mode horizontal, step-like structure and inclined solar fields, *Sol. Energy.*, **181**, 53–69 (2019).

[30] Y. Nassar, and S. Alsadi, Assessment of solar energy potential in Gaza Strip-Palestine, *Sustain. Energy Technol. Assessments.,* **31,** 318–328 (2019).

[31] I. Younas, F. Kamrani, C. Schulte, and R. Ayani, *Optimization of task assignment to collaborating agents*, in 2011 IEEE Symp. Comput. Intell. Sched., 17–24 (2011).

[32] Y. Muhammad, M. Raja, M. Shah, S. Awan, F. Ullah, N. Chaudhary, K. Cheema, A. Milyani, and C. Shu, Optimal coordination of directional overcurrent relays using hybrid fractional computing with gravitational search strategy, *Energy Reports.*, **7**, 7504–7519 (2021).

[33] P. Alaee, and T. Amraee, Optimal Coordination of Directional Overcurrent Relays in Meshed Active Distribution Network Using Imperialistic Competition Algorithm, *J. Mod. Power Syst. Clean Energy.,* **9**, 416–422 (2020).

[34] M. El-kordy, A. El-fergany, and A. Gawad, Various Metaheuristic-Based Algorithms for Optimal Relay Coordination: Review and Prospective, *Arch. Comput. Methods Eng.*, **28**, 3621–3629 (2021).

[35] M. Yang, and A. Liu, Applying hybrid PSO to optimize directional overcurrent relay coordination in variable network topologies, *J. Appl. Math.*, **2013** (2013).

[36] F. Sampaio, F. Tofoli, L. Melo, G. Barroso, R. Sampaio, and R. Leão, Smart Protection System for Microgrids with Grid-Connected and Islanded Capabilities Based on an Adaptive Algorithm, *Energies*, **16**, 2273 (2023).

[37] A. Al-Roomi, and M. El-Hawary, *Optimal coordination of directional overcurrent relays using hybrid BBO-LP algorithm with the best extracted time-current characteristic curve*, in 2017 IEEE 30th Can. Conf. Electr. Comput. Eng., IEEE, 1–6 (2017).

[38] T. Foqha, S. Alsadi, O. Omari, M. AL-Mousa, S. Aljazzar, and M. Kanan, J. Asad, A New Iterative Approach for Designing Passive Harmonic Filters for Variable Frequency Drives, *Appl. Math.*, **17**, 453–468 (2023).

[39] Y. Paithankar, and S. Bhide, Fundamentals of power system protection, *PHI Learning Pvt. Ltd.,* (2022).

[40] P. Bedekar, and P. Korde, *Determining optimum time multiplier setting of overcurrent relays using modified Jaya algorithm*, in 2017 Innov. Power Adv. Comput. Technol., IEEE, 1–6 (2017).

[41] M. Ghanbari, M. Gandomkar, and J. Nikoukar, Protection Coordination of Bidirectional Overcurrent Relays Using Developed Particle Swarm Optimization Approach Considering Distribution Generation Penetration and Fault Current Limiter Placement, *IEEE Can. J. Electr. Comput. Eng.*, **44**, 143–155 (2021).

[42] H. Zeineldin, E. El-Saadany, and M. Salama, Optimal coordination of overcurrent relays using a modified particle swarm optimization, *Electr. Power Syst. Res.*, **76**, 988–995 (2006).

[43] A. Mahari, and H. Seyedi, An analytic approach for optimal coordination of overcurrent relays, *IET Gener. Transm. Distrib.,* **7**, 674–680 (2013).

[44] A. ElFergany, Optimal directional digital overcurrent relays coordination and arc flash hazard assessments in meshed networks, *Int. Trans. Electr. Energy Syst.*, **26,** 134–154 (2016).

[45] I. Trivedi, S. Purani, and P. Jangir, *Optimized over-current relay coordination using Flower Pollination Algorithm*, in 2015 IEEE Int. Adv. Comput. Conf., IEEE, 72–77 (2015).

[46] S. ElSayed, and E. Elattar, Hybrid Harris hawks optimization with sequential quadratic programming for optimal coordination of directional overcurrent relays incorporating distributed generation, *Alexandria Eng. J.,* **60,** 2421–2433 (2021).

[47] K. Habib, X. Lai, A. Wadood, S. Khan, Y. Wang, and S. Xu, Hybridization of PSO for the Optimal Coordination of Directional Overcurrent Protection Relays, *Electronics*, **11**, 180 (2022).

[48] Y. Damchi, M. Dolatabadi, H. Mashhadi, and J. Sadeh, MILP approach for optimal coordination of directional overcurrent relays in interconnected power systems, *Electr. Power Syst. Res.*, **158**, 267–274 (2018).

[49] M. Alam, B. Das, and V. Pant, An interior point method based protection coordination scheme for directional overcurrent relays in meshed networks, *Int. J. Electr. Power Energy Syst.*, **81**, 153–164 (2016).

[50] A. R. Al-Roomi, *Optimal Coordination of Power Protective Devices with Illustrative Examples*, John Wiley & Sons, (2021).

[51] E. Sorrentino, and J. Rodríguez, Effects of fault type and pre-fault load flow on optimal coordination of directional overcurrent protections, *Electr. Power Syst. Res.*, **213**, 108685 (2022).

[52] S. Alsadi, and T. Foqha, Mass flow rate optimization in solar heating systems based on a flat-plate solar collector: A case study, *World Journal of Advanced Research and Reviews*, **12** (3), 61-71 (2021).

[53] F. Albasri, A. Alroomi, and J. Talaq, Optimal coordination of directional overcurrent relays using biogeography-based optimization algorithms, *IEEE Trans. Power Deliv.*, **30**, 1810–1820 (2015).

[54] V. Rajput, and K. Pandya, Coordination of directional overcurrent relays in the interconnected power systems using effective tuning of harmony search algorithm, *Sustain. Comput. Informatics Syst.*, **15**, 1–15 (2017).

[55] J. Radosavljević, and M. Jevtić, Hybrid GSA-SQP algorithm for optimal coordination of directional overcurrent relays, *IET Gener. Transm. Distrib.*, **10**, 1928–1937 (2016).

[56] A. Al-Roomi, and M. El-Hawary, *Is It Enough to just Rely on Near-End, Middle, and Far-End Points to get Feasible Relay Coordination?*, in 2019 IEEE Can. Conf. Electr. Comput. Eng., IEEE, 1–5 (2019).

[57] S. Ramli, M. Usama, H. Mokhlis, W. Wong, M. Hussain, M. Muhammad, and N. Mansor, Optimal directional overcurrent relay coordination based on computational intelligence technique: a review, *Turkish J. Electr. Eng. Comput. Sci.*, **29**, 1284–1307 (2021).

[58] J. Moirangthem, K. KR, S. Dash, and R. Ramaswami, Adaptive differential evolution algorithm for solving nonlinear coordination problem of directional overcurrent relays, *IET Gener. Transm. Distrib.*, **7** 329–336 (2013).