processes

MDPI

Article

DC Microgrid Enhancement via Chaos Game Optimization Algorithm

Abdelrahman S. Heikal [1], Ibrahim Mohamed Diaaeldin [1,*], Niveen M. Badra [1], Mahmoud A. Attia [2,*], Ahmed O. Badr [2], Othman A. M. Omar [1], Ahmed H. EL-Ebiary [2] and Hyun-Soo Kang [3,*]

[1] Engineering Physics and Mathematics Department, Faculty of Engineering, Ain Shams University, Cairo 11517, Egypt; 2100977@eng.asu.edu.eg (A.S.H.); niveen_badra@eng.asu.edu.eg (N.M.B.); osman_ahmed@eng.asu.edu.eg (O.A.M.O.)
[2] Electrical Power and Machines Department, Faculty of Engineering, Ain Shams University, Cairo 11517, Egypt; ahmed.badr@eng.asu.edu.eg (A.O.B.); ahmed.haitham@eng.asu.edu.eg (A.H.E.-E.)
[3] School of Information and Communication Engineering, College of Electrical and Computer Engineering, Chungbuk National University, Cheongju 28644, Republic of Korea
* Correspondence: ibrahimmohamed@eng.asu.edu.eg (I.M.D.); mahmoud.abdullah@eng.asu.edu.eg (M.A.A.); hskang@chungbuk.ac.kr (H.-S.K.); Tel.: +82-010-5167-8189 (H.-S.K.)

Abstract

Microgrids are increasingly being adopted as alternatives to traditional power transmission networks, necessitating improved performance strategies. Various mathematical optimization techniques are used to determine optimal controller parameters for these systems. These optimization methods can generally be categorized into natural, biological, and engineering-based approaches. In this research, the authors evaluated and compared several optimization techniques to enhance the secondary controller of DC microgrids, focusing on reducing operating time and minimizing error rates. Optimization tools were utilized to identify the optimal gain control parameters, aiming to achieve the best possible system performance. The enhanced controller response enables quicker recovery to steady-state conditions during sudden disturbances. The root-mean-square error (RMSE) served as a performance metric, with the proposed approach achieving a 15% reduction in RMSE compared to previous models. This improvement contributes to faster response times and lower energy consumption in microgrid operation.

Keywords: DC microgrids; smart grid; optimization; grey wolf optimization; chaos game optimization

check for updates

Academic Editor: Zineb Cabrane

Received: 12 May 2025
Revised: 14 June 2025
Accepted: 25 June 2025
Published: 27 June 2025

Citation: Heikal, A.S.; Diaaeldin, I.M.; Badra, N.M.; Attia, M.A.; Badr, A.O.; Omar, O.A.M.; EL-Ebiary, A.H.; Kang, H.-S. DC Microgrid Enhancement via Chaos Game Optimization Algorithm. *Processes* **2025**, *13*, 2042. https://doi.org/10.3390/pr13072042

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

The traditional power grid is no longer viewed as the most effective solution for power system networking due to various limitations, including inefficiencies in power delivery and distribution, high delivery costs, inadequate energy storage, and substantial carbon emissions. Additional challenges include chronic blackouts, expensive infrastructure, and operational constraints [1,2]. In contrast, microgrids offer significant advantages over traditional grids, including being two to three times more efficient, yielding 20 to 40% cost savings, reducing outages by up to 90%, and cutting emissions by 50 to 100%. Although microgrids require higher initial investment, their long-term savings and enhanced resilience make them a worthwhile investment, particularly for critical infrastructure and renewable energy integration [3]. Numerous incidents underscore the urgent need to improve traditional power systems [4]. For example, in February 2020, Storm Ciara caused

power outages for approximately 130,000 homes in France [5]. In the same month, Storm Sabine left around 60,000 homes in Bavaria without power [4]. Earlier, in March 2016, a major blackout in Turkey impacted at least 70 million people. These occurrences highlight the vulnerabilities of conventional grids and support the transition to more resilient alternatives, such as microgrids [2]. A microgrid is a localized energy system that integrates distributed energy resources such as solar panels, wind turbines, fuel cells, and energy storage units [5,6]. It can operate independently or in conjunction with the main grid. Compared to traditional power networks, smart grids offer numerous benefits, including reduced costs, cleaner energy, increased efficiency, lower emissions, and enhanced controllability [2]. However, they also encounter technical challenges, such as resynchronization with the main grid, which can lead to instability due to network inconsistencies [7]. To address these challenges, the smart grid has emerged as a comprehensive solution [2,5]. It incorporates intelligent systems for energy production and distribution, including smart meters, smart appliances, renewable energy resources, and energy-efficient technologies [6]. By leveraging information and communication technologies (ICTs), smart grids enable a two-way flow of electricity and information, thus transforming the energy infrastructure in terms of scalability, efficiency, reliability, and interoperability [5]. This integration of the electricity distribution system with ICT has given rise to a new energy paradigm where both energy and information flow bidirectionally [4,5,8]. The transition from traditional power systems to smart grids is revolutionizing the energy sector by enhancing reliability, performance, and control through real-time monitoring, automated operations, and advanced communication technologies [9]. According to the National Institute of Standards and Technology (NIST), the smart grid encompasses seven key domains: Markets, Service Providers, Operations, Bulk Generation, Transmission, Distribution, and Customers [9]. These domains are illustrated in the NIST conceptual model shown in Figure 1.

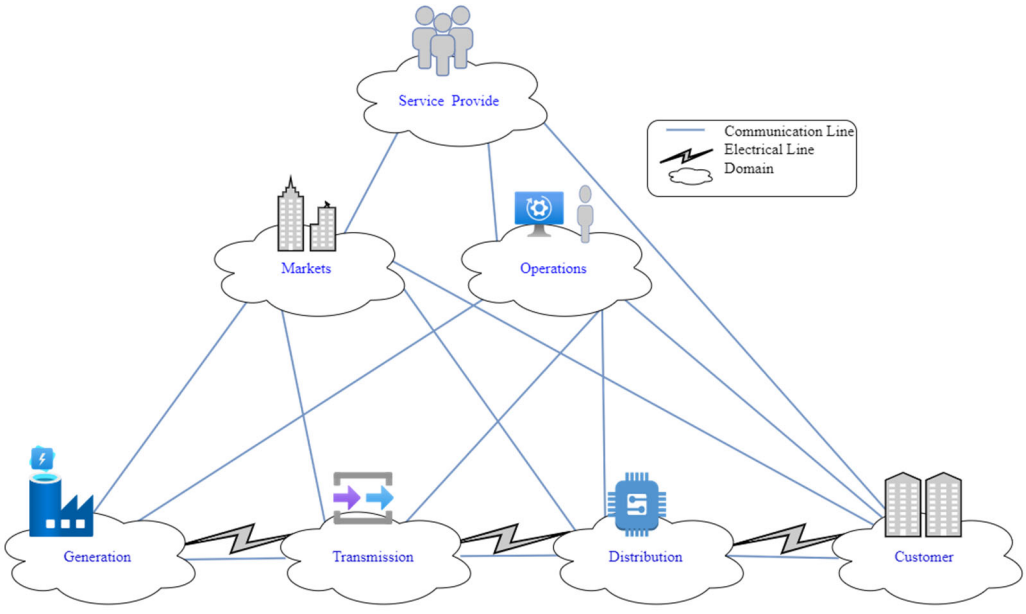


Figure 1. The NIST model of the smart grid.

Key aspects of DC microgrid operation include DC bus voltage regulation, power management, effective power sharing among energy storage devices (ESDs), and state of charge (SoC) restoration. However, maintaining stable DC bus voltage and managing power flows are complex tasks, primarily because DC microgrids interconnect multiple distributed generators (DGs), loads, utility grids, and energy storage devices (ESDs) through power electronic converters. To ensure safe and reliable operation, DC microgrids employ

various control strategies, including centralized, decentralized, distributed, multi-level, and hierarchical control systems [1]. Among these, centralized, decentralized, and distributed control architectures are commonly used [1,10]. Figure 2 illustrates a decentralized control configuration within a DC microgrid based on source-side management. In a decentralized system, control and ownership are distributed among multiple independent entities, each managing its segment of the network. These systems often rely on several central controllers, some of which store replicated service information accessible to users. Furthermore, DC microgrids can regulate the output of multiple parallel converters to enhance load sharing and improve operational stability [1,11].

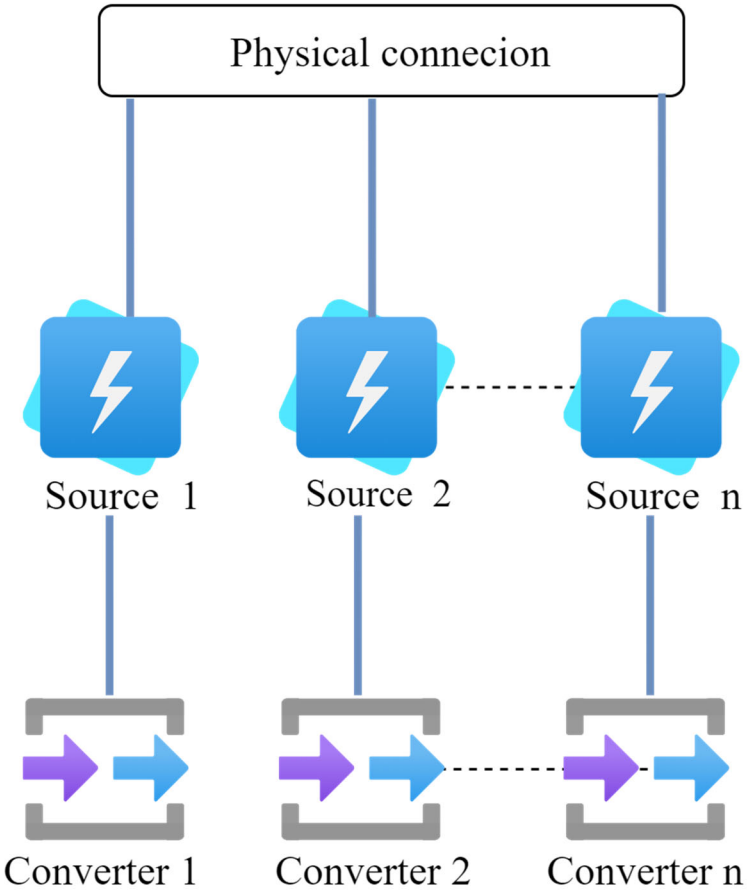


Figure 2. Decentralized control implementation.

Distributed control can be implemented in a source-based DC microgrid with communication capabilities, as illustrated in Figure 3. In a distributed system, control and data ownership are shared among multiple agents, enabling localized decision making and enhancing system scalability. Users are typically granted access to both hardware and software resources, which can improve system performance under certain operating conditions [1,9,10].

Centralized control can be implemented in a source-based DC microgrid, as shown in Figure 4. In a centralized architecture, all users connect to a central network controller, also referred to as a "host", which oversees system operations. The central controller retains all system data and user information, serving as the sole point of coordination accessible to all entities within the network [1,11].

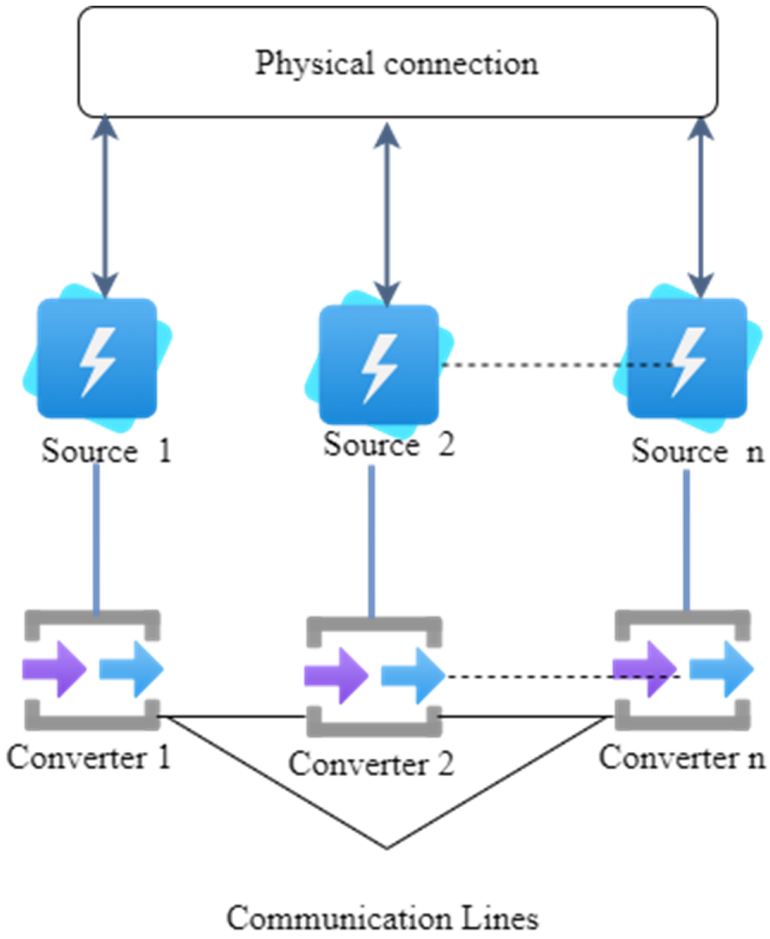


Figure 3. Distributed system implementation.

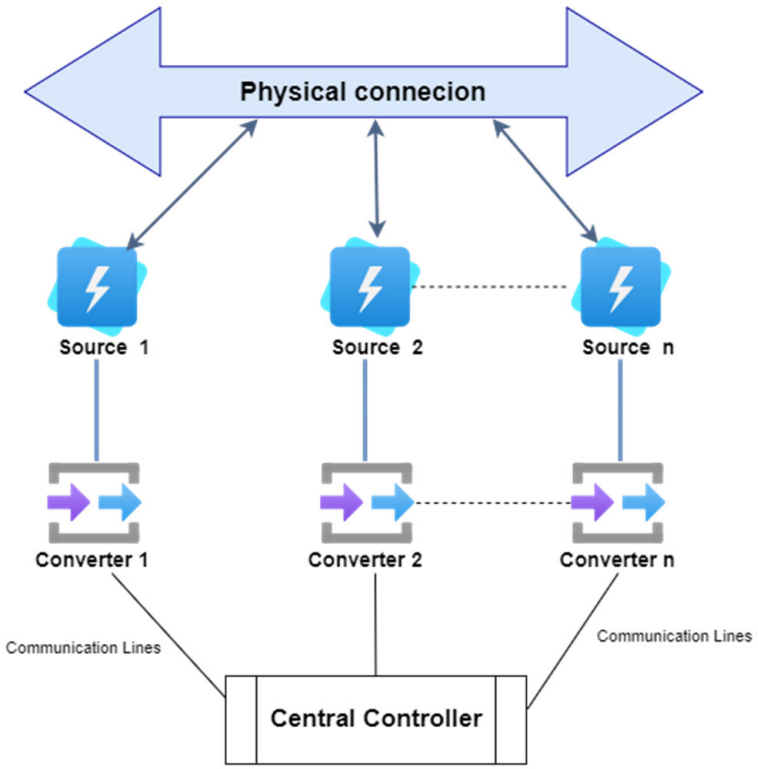


Figure 4. Centralized system implementation.

The control structure in DC microgrids typically consists of three layers: tertiary, secondary, and primary control [1,12]. Control signals and measurement data flow bidirectionally between these layers, with tertiary control offering higher-level coordination and optimization while primary control manages rapid local responses. The electrical grid primarily oversees current and voltage at the system level. Secondary control plays a more active role than primary control in voltage regulation and enhancing current sharing among distributed energy resources (DERs) [1,13,14]. In a hierarchical control topology, the primary control layer operates independently of communication between subsystems. It ensures current sharing among microgrid units while regulating the DC bus voltage through droop-based control mechanisms. When voltage deviations occur, the secondary control layer restores stability using low-bandwidth communication and droop control techniques. In cases of significant voltage deviations, centralized control may become inadequate in managing power flow. Under such conditions, decentralized controllers are utilized to address these limitations and maintain system performance [1,14,15]. The hybrid-triggered secondary control strategy addresses the limitations of existing event-triggered and self-triggered approaches in DC microgrids by minimizing communication, sampling, and computational requirements [16]. The multi-power small ship networks (MESMs) offer a robust two-stage joint planning approach that considers system uncertainty and thermal inertia, demonstrating effectiveness through detailed simulations [17]. Although more stable controllers are available, challenges persist in managing non-linear loads, variable operating points, and inherent time delays. To address these issues, several optimization techniques have been proposed for tuning controller parameters. Among these, artificial intelligence (AI)-based methods, such as neural networks, fuzzy logic, and fuzzy-neural systems, have been explored to enhance control performance [18]. However, these approaches often suffer from drawbacks like slow convergence and complex implementation. Recently, metaheuristic optimization algorithms have gained popularity due to their simplicity, ease of implementation, and independence from gradient information. These algorithms are generally categorized into five main groups: biological, swarm-based, physics-based, geometric, and human-based algorithms [19–21]. Examples of biological algorithms include Harris Hawks Optimization (HHO), the Whale Optimization Algorithm (WOA), and grey wolf optimization (GWO) [20,22]. Geometric-based algorithms include the Circle Search Algorithm (CSA) and Sine Cosine Algorithm (SCA). At the same time, swarm-based methods consist of the Salp Swarm Algorithm (SSA) [23], Particle Swarm Optimization (PSO), Transient Search Optimization (TSO), chaos game optimization (CGO) [21], and Dung Beetle Optimizer (DBO) [24]. Mathematical optimization generally involves selecting the most effective solution from a set of alternatives while considering defined objectives and constraints. In the context of smart grids, optimization seeks to strike the optimal balance between reliability, availability, efficiency, and cost. The key benefits of grid optimization include better utilization of existing infrastructure, deferring investments in new generation and distribution assets, lowering overall power delivery costs, improving grid reliability, and reducing resource consumption and greenhouse gas emissions [11,25,26]. In this paper, the proposed control strategy demonstrates improved performance compared to prior studies [26,27]. The enhanced DC microgrid model exhibited faster responses to sudden changes, reduced energy losses, improved system stability, and a lower error rate. Achieving optimal performance, reducing the RSM error, ensuring rapid responses, and swiftly attaining a steady state are quite challenging. To tackle this, we use YALMIP, a toolbox mentioned in [26], to solve linear matrix inequality (LMI) problems and determine the gains for the second control loop. Furthermore, the Whale Optimization Algorithm (WOA), as detailed in [28], is employed to refine the controller parameters.

2. Materials and Methods

2.1. DGUS System

This section presents the dynamic modeling of Islanded Microgrids (ImGs). The DC microgrid system depicted in Figure 5 was modeled using the MATLAB/SIMULINK software version R2024 b package and employs a decentralized control strategy. We will begin by describing an ImG that includes two parallel distributed generation units (DGUs). This model can be expanded to incorporate ImGs consisting of N DGUs. Consider the configuration shown in Figure 5, where two DGUs labeled i and j are connected through a DC line characterized by Rij > 0 and Lij > 0. Each DGU includes a DC voltage source representing a renewable resource and contains a buck converter to supply local DC loads linked to the point of common coupling (PCC) via a series LC filters [27,29–31].

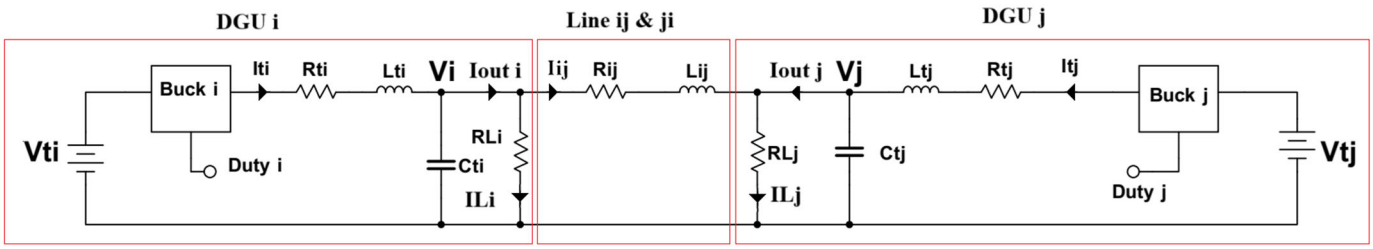


Figure 5. The electrical scheme of a DC ImG is composed of two radially connected DGUs with unmodeled loads.

Applying Kirchhoff's voltage law and Kirchhoff's current law to the electrical schematic in Figure 5 results in the following:

At DGU I, at node vi

$$\begin{aligned} \mathrm{Iti} &= \mathrm{Ici} + \mathrm{Iouti} \\ \mathrm{Iti} &= \mathrm{Ici} + \mathrm{Ili} + \mathrm{Iij} \\ \mathrm{Iti} &= \mathrm{Cti}\frac{\mathrm{dvi}}{\mathrm{dt}} + \mathrm{ILi} + \frac{\mathrm{Vi} - \mathrm{Vj}}{\mathrm{Rij}} \\ \frac{\mathrm{dvi}}{\mathrm{dt}} &= \frac{\mathrm{Iti}}{\mathrm{Cti}} + \left[\frac{\mathrm{Vj}}{\mathrm{CtixRij}} - \frac{\mathrm{Vi}}{\mathrm{CtixRij}}\right] - \frac{\mathrm{Ili}}{\mathrm{Cti}} \end{aligned} \quad (1)$$

At Loop DGU i

$$\begin{aligned} \mathrm{Vti} - \mathrm{Vi} &= \mathrm{ItixRti} + \mathrm{Lti}\frac{\mathrm{dIti}}{\mathrm{dt}} \\ \frac{\mathrm{dIti}}{\mathrm{dt}} &= -\mathrm{Iti}\frac{\mathrm{Rti}}{\mathrm{Lti}} - \frac{\mathrm{Vi}}{\mathrm{Lti}} + \frac{\mathrm{Vti}}{\mathrm{Lti}} \end{aligned} \quad (2)$$

Likewise, DGU j

$$\frac{\mathrm{dvj}}{\mathrm{dt}} = \frac{\mathrm{Itj}}{\mathrm{Ctj}} + \left[\frac{\mathrm{Vi}}{\mathrm{CtjxRji}} - \frac{\mathrm{Vj}}{\mathrm{CtjxRji}}\right] - \frac{\mathrm{Ilj}}{\mathrm{Ctj}} \quad (3)$$

$$\frac{\mathrm{dItj}}{\mathrm{dt}} = -\mathrm{Itj}\frac{\mathrm{Rtj}}{\mathrm{Ltj}} - \frac{\mathrm{Vj}}{\mathrm{Ltj}} + \frac{\mathrm{Vtj}}{\mathrm{Ltj}} \quad (4)$$

The DC microgrid system used to test the proposed control strategy is shown in Figure 6.

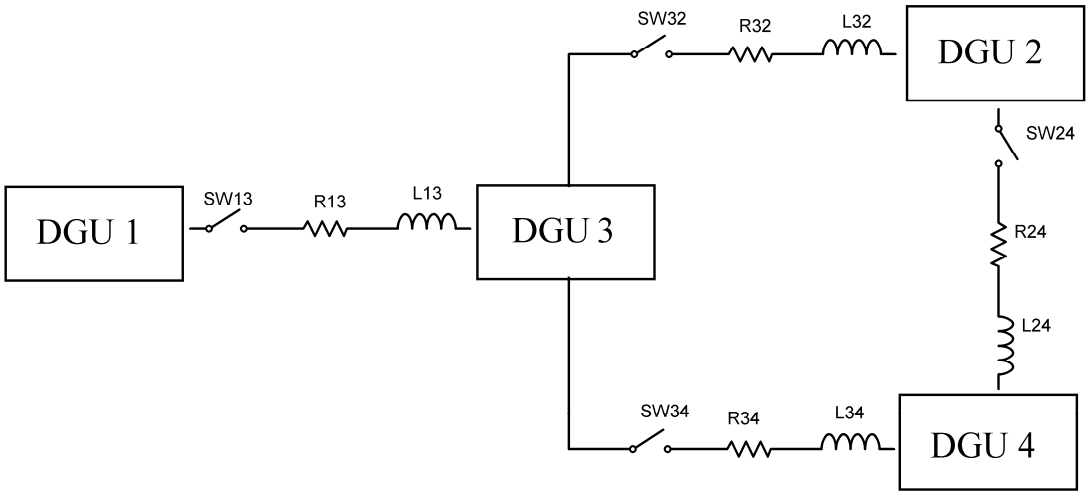


Figure 6. DC microgrid system model.

The system consists of four DGUs, each represented by a DC source and connected in a meshed structure through transmission lines associated with DC circuit breakers. The plug-and-play capability of the distributed control strategy has been previously tested in reference [32]. It is proven that this control strategy is scalable and stable, provided that a semi-definite optimization problem yielding the gains of the primary controller is solvable.

The equations describing each DGU in the system can be easily deduced using Kirchhoff's laws, taking into consideration all transmission lines, as follows [26,27]:

$$\frac{dVi}{dt} = \frac{Iti}{Cti} + \sum_{i=1}^{4}\left[\frac{Vi}{CtixRij} - \frac{Vj}{CtixRij}\right] - \frac{Ilj}{Ctj} \tag{5}$$

$$\frac{dIti}{dt} = -\frac{Vi}{Lti} - \frac{Rti}{Lti}Iti + \frac{Vti}{Lti} \tag{6}$$

A state-space model is presented for each distributed generation unit, accounting for the dynamics of the operating units, as referenced in [33] and illustrated in Equations (7) and (9). The voltage of generating unit i is considered the state *vi*, and the active coil current is denoted as *Iti*. The system signal *u*[*i*] is regarded as the junction voltage of the operating generating unit *vti*, while the local load current is represented by the disturbance signal d[*i*]. The dynamics of the operating units are represented by $\zeta[i]$, as shown in Equation (8) in the following state-space model of the complex PV network [26,27].

$$\begin{pmatrix} dVi/dt \\ dIti/dt \\ dvi/dt \end{pmatrix} = \begin{pmatrix} \frac{-1}{CtixRij} & \frac{1}{Cti} & 0 \\ \frac{-1}{Lti} & \frac{-Rti}{Lti} & 0 \\ -1 & 0 & 0 \end{pmatrix} \begin{pmatrix} Vi \\ Iti \\ vi \end{pmatrix} + \begin{pmatrix} 0 \\ \frac{1}{Lti} \\ 0 \end{pmatrix} Vti + \begin{pmatrix} \frac{-1}{Cti} \\ 0 \\ 0 \end{pmatrix} Ili + \text{error} \tag{7}$$

$$\frac{dVi}{dt} = \frac{-1}{CtixRij}Vi + \frac{1}{Cti}Iti - \frac{1}{Cti}Ili \tag{7a}$$

$$\frac{dIti}{dt} = \frac{-1}{Lti}Vi - \frac{Rti}{Lti}Iti + \frac{1}{Lti}Vti \tag{7b}$$

$$\frac{dvi}{dt} = -Vi \tag{7c}$$

Compare Equations (5) and (7a):

$$\text{Error}(0) = \frac{Vj}{CtixRij}Vi \tag{7d}$$

Compare Equations (6) and (7b):

$$\text{Error}(1) = 0 \tag{7e}$$

From (7d) and (7e) write the following error part:

$$\xi[i](t) = \sum_{i=1}^{4} \text{AijXj} \tag{8}$$

where Aij Error matrix $\text{Aij} = \begin{pmatrix} \frac{\text{Vj}}{\text{CtixRij}} \\ 0 \end{pmatrix} \begin{pmatrix} \text{Vi} & \text{Iti} \end{pmatrix}$.

Put Equation (7) in the state space model:

$$\dot{X}[i] = \text{Aij.X}[i] + \text{Bi.u}[i] + \text{Mi.d}[i] + \xi[i] \tag{9}$$

where $\text{X}[i] = \begin{pmatrix} \text{Vi} & \text{Iti} & \text{vi} \end{pmatrix}^{\text{T}}$.

2.1.1. Primary Control Layer

Each DGU has a primary controller designed to manage the duty cycle (D) of the unidirectional buck converter, which connects the DGU to the microgrid and regulates the DGU's output voltage. As shown in Figure 7, a state feedback controller is used for the first layer. Equation (11) presents the controller equation.

$$\text{Vti} = \begin{pmatrix} \text{Vi} & \text{Iti} & \text{vi} \end{pmatrix} \begin{pmatrix} \text{Ki1} \\ \text{Ki2} \\ \text{Ki3} \end{pmatrix} \tag{10a}$$

where

$$\mathbf{vi} = \int_{-\infty}^{\infty} \mathbf{Vref, i} - \mathbf{Vi} \tag{10b}$$

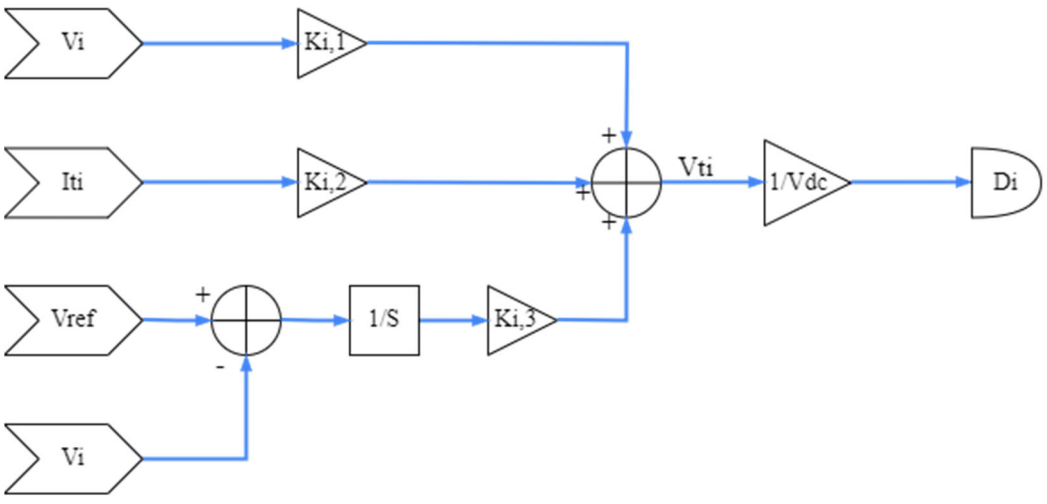


Figure 7. Primary control layer for DGUi.

From (10a) and (10b), we obtain the following:

$$\mathbf{u}[\mathbf{i}] = \mathbf{Ki.X}[\mathbf{i}] \tag{11}$$

where *ki* ∈ R1 × 3 vector, and ki = [k1i, k2i, k3i] and the control.

The action ui depends on the state of each DGU only.

$X[i] = \begin{pmatrix} Vi & Iti & vi \end{pmatrix}^T x$ is the system state vector after considering the integrator dynamics v*i* (12), which is required to eliminate the steady-state error.

From (10b), we obtain the following:

$$vi = \int_{-\infty}^{\infty} yref,[i] - V[i] \tag{12}$$

$$Duty\ i = \frac{Vti}{100} \quad \{0 \sim 100\}\%$$

To obtain a vector *ki*, the problem is formulated and solved using linear matrix inequality (LMI) [29] through YALMIP [34], a toolbox for solving LMI problems integrated with MATLAB R2024 b software [34,35]. Table 1 displays the controller's parameters.

Table 1. Primary layer controller's parameters.

Primary Control Gain Ki	
DGU i	Ki
DGU 1	K1 = [−2.31, −0.16, 13.55]
DGU 2	K2 = [−0.87, −0.05, 48.28]
DGU 3	K3 = [−0.48, −0.108, 30.67]
DGU 4	K4 = [−7, −0.175, 102.96]

Table 1 shows the gain parameters for the primary control layer [33].

2.1.2. Secondary Control Layer

Secondary control loops were added to the enhanced reference voltage of the primary loop Vref i by introducing a correction term Δ*vi* to the reference voltage, as illustrated in Figure 8. This adjustment aims to reduce and eliminate deviations caused by the primary controller and ensure proper current sharing among all DGUs. The equation representing the secondary control unit is given by Equation (13).

$$\dot{\Delta vi} = API * \sum_{i=1,j\neq i}^{N} (Ioutj - Iouti) \tag{13}$$

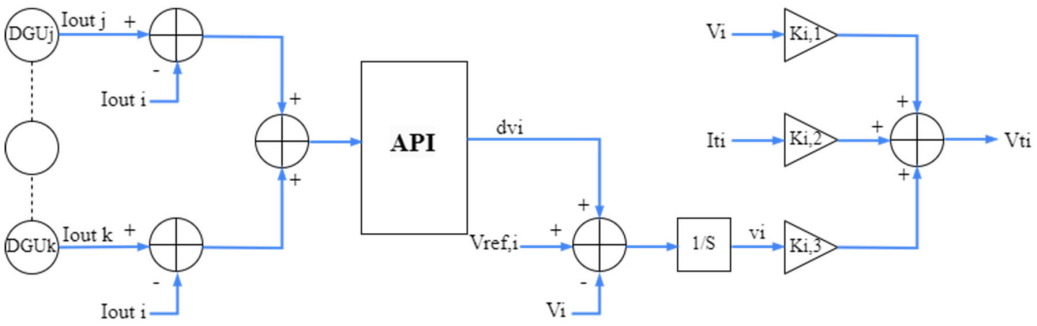


Figure 8. Secondary control layers for DGU i.

Where $\dot{\Delta vi}$ the API adaptive PI controller output.

This specific form of the correction term added by the secondary control layer is based on the consensus algorithm [36]. The idea is based on adjusting the voltage reference for each DGU to achieve equal current sharing between the DGUs, where the correction term is directly proportional to the error in the current sharing between each neighboring DGUs.

Iouti is the output current of DGUi, and *Ioutj* represents the output current of the neighboring DGUs. An adaptive PI controller is utilized for the secondary loop to obtain the correction voltage Δvi, which is expressed by the following:

$$\Delta \text{vi} = \text{Ki3}\left[(\text{Kp(t).e(t)}) + \int_0^t \text{Ki(t).e(t)dt}\right] \quad (14)$$

where KP and Ki are the proportional and integral gains of the PI controller, respectively. The adaptive controller utilizes the input error signal e(t) to update the proportional (Kp) and integral (Ki) gains online, as shown in Equations (15) and (16).

$$\text{Kp(t)} = \text{e}^2\text{(t)} + \text{Ki1}\int_0^t \text{e}^2\text{(t)dt} \quad (15)$$

$$\text{Ki(t)} = \text{Ki2}\int_0^t \text{e}^2\text{(t)dt} \quad (16)$$

K_1, K_2, and K_3 are arbitrary constants selected using optimization algorithms. The output of the API controller is described by Equation (14) [26], and the corresponding scheme is illustrated in Figure 9.

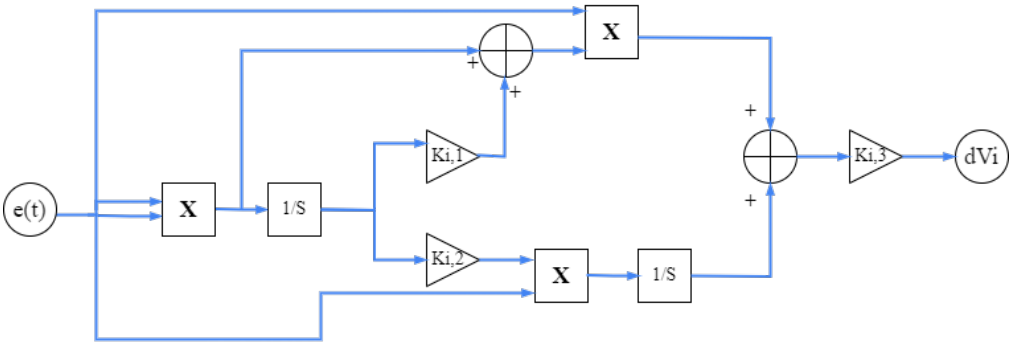


Figure 9. Adaptive PI control block diagram.

These adaptive controllers are utilized to enhance the dynamic performance of the current-sharing loop across different loading conditions and to improve microgrid stability [26,37].

2.2. Optimization Tools

Metaheuristic algorithms are classified into three main categories based on their inspiration: swarm-based, physics-based, evolutionary, and human-related algorithms. Swarm-based algorithms mimic the behaviors of organisms in reproduction, evolution, swarming, and foraging, such as the Salp Swarm Algorithm (SSA) and grey wolf optimization (GWO), which mimics the hierarchical order of a grey wolf's pack to hunt prey [22,23]. The Dung Beetle Optimizer (DBO), based on the SI technique, is developed to solve complex optimization problems [24]. The Whale Optimization Algorithm (WOA) imitates the spiral hunting technique of humpback whales [20,38], and the Harris Hawks Optimizer (HHO) reflects the collaborative hunting strategies of hawks [38,39]. Physics-based algorithms draw inspiration from well-known physical theories and phenomena [38,40]. The Transient Search Optimizer (TSO) simulates the transient response of electrical circuits that include energy storage devices [41]. Other physics-based metaheuristic algorithms include chaos game optimization (CGO) [21]. Geometric-based metaheuristic algorithms focus on the characteristics of figures in space, including dimensions, relative positions, distances, forms, and sizes. The Sine Cosine Algorithm (SCA) and Circle Search Algorithm (CSA) are examples of geometry-based methods [38]. Grey wolves typically prefer to live in

packs, which average between five to twelve members [22]. Noteworthy is their strict social hierarchy, which consists of Alpha, Beta, Delta, and Omega ranks [22]. Alphas hold the highest dominance in the hierarchy; they are the leaders responsible for decision making regarding the pack and its hunting activities. Betas are the second level in the grey wolf hierarchy [22,42]. They are subordinate wolves that assist the Alphas in making decisions and handling pack-related tasks. Deltas are dominant wolves that must report to Alphas and Betas, who are composed of scouts, rangers, elders, hunters, and caretakers. The lowest-ranking grey wolves are the Omegas [22], who act as scapegoats. Omegas must always submit to all other dominant wolves and are the last to eat [43]. To mathematically simulate the hunting behavior of grey wolves, consider that the Alpha (best candidate solution), Beta, and Delta possess better knowledge about the potential locations of prey. As a result, the first three best solutions obtained so far are saved, and the other search agents (including Omegas) are obliged to update their positions based on the location of the best search agent. The following formulas are proposed in this regard [22]:

$$\vec{D}_\alpha = \left|\vec{C}_1.\vec{X}_\alpha - \vec{X}\right|, \vec{D}_\beta = \left|\vec{C}_2.\vec{X}_\beta - \vec{X}\right|, \vec{D}_\delta = \left|\vec{C}_3.\vec{X}_\delta - \vec{X}\right| \tag{17}$$

$$\vec{X}_1 = \vec{X}_\alpha - \vec{A}_1.\vec{D}_\alpha, \vec{X}_2 = \vec{X}_\beta - \vec{A}_2.\vec{D}_\beta, \vec{X}_3 = \vec{X}_\delta - \vec{A}_3.\vec{D}_\delta \begin{cases} |A| < 1 & exploitation \\ |A| > 1 & \text{exploration} \end{cases} \tag{18}$$

$$\vec{X}(t+1) = \frac{\vec{X}_1 + \vec{X}_2 + \vec{X}_3}{3} \tag{19}$$

The vectors $\vec{C}_1$, $\vec{C}_2$, $\vec{C}_3$ contain random values in [0, 2].

The flow chart of the GWO algorithm is presented in Figure 10.

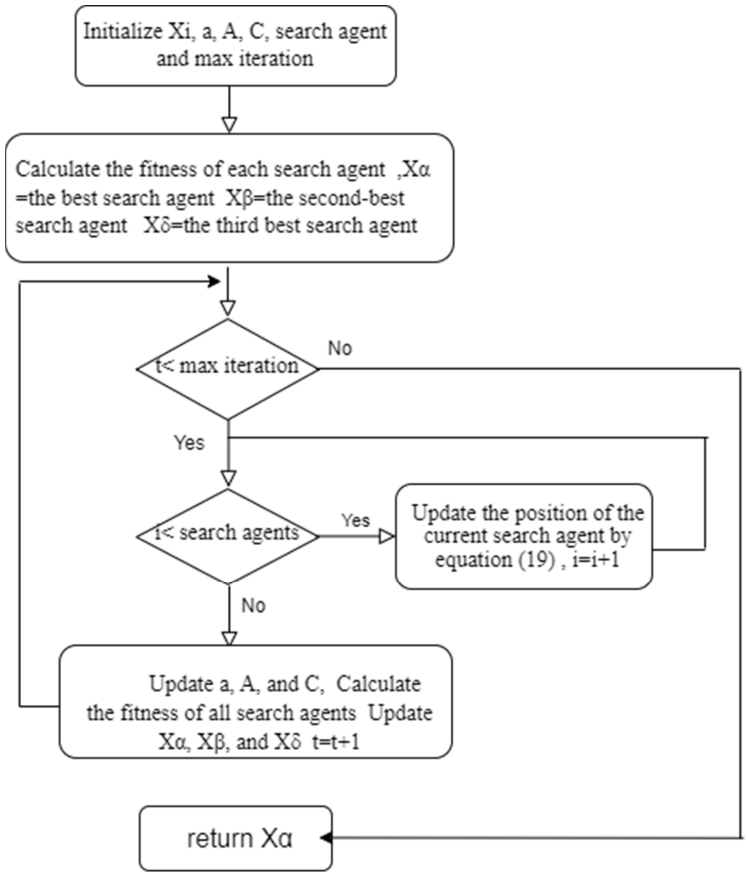


Figure 10. Flow chart of the GWO algorithm.

Chaos theory is a branch of mathematics that examines the specific characteristics of dynamical systems, which are extremely sensitive to initial conditions [21]. The mathematical model of the CGO algorithm is based on the fundamental concepts of fractals and chaotic games. The CGO algorithm considers several candidate solutions (X). Many natural evolutionary algorithms maintain a set of solutions that evolve through random modifications and selection, representing eligible seeds within the Sierpinski triangle. In this algorithm, each candidate solution (Xi) consists of decision variables (xi, j) that indicate the locations of these eligible seeds within the Sierpinski triangle [21,44]. The Sierpinski triangle serves as the search space for candidate solutions in the optimization algorithm.

The mathematical equation to show of these aspects is Equation (20):

$$X = \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_i \\ \vdots \\ X_n \end{bmatrix} = \begin{bmatrix} X_1^1 & X_1^2 & \dots & X_1^j & \dots & X_1^d \\ X_2^1 & X_2^2 & \dots & X_2^j & \dots & X_2^d \\ \vdots & \vdots & \dots & \vdots & \dots & \vdots \\ X_i^1 & X_i^2 & \dots & X_i^j & \dots & X_i^d \\ \vdots & \vdots & \dots & \vdots & \dots & \vdots \\ X_n^1 & X_n^2 & \dots & X_n^j & \dots & X_n^d \end{bmatrix}, \begin{cases} i = 1,2,3 & , \dots, n \\ j = 1,2,3 & , \dots, d \end{cases} \quad (20)$$

where n is the number of solution candidates inside the Sierpinski triangle, and d is the dimension of these seeds.

The initial positions of these eligible seeds are randomly determined in the search space, as shown in Equation (21):

$$X_j^i(0) = X_{i,min}^j + rand.\left(X_{i,max}^j - X_{i,min}^j\right), \quad \begin{cases} i = 1,2,3 & , \dots, n \\ j = 1,2,3 & , \dots, d \end{cases} \quad (21)$$

where $X_j^i(0)$ is the initial position of the eligible seed, and $X_{i,max}^j$ and $X_{i,min}^j$ are the minimum and maximum allowable values for the jth decision variable of the ith solution candidate; *rand* is a random number in the interval of [0, 1].

As mentioned earlier, the presence of prototypical patterns in the behavior of dynamical systems is a principle of chaos theory. Qualified seeds can serve as prototypical patterns of dynamical systems, as described by chaos theory. The suitability of these seeds as prototypical (self-similar) can be modeled using candidate solutions (X) to an optimization problem. The candidate solutions with the highest and lowest eligibility levels correspond to the best and worst fitness values, respectively [21].

The primary concept of this mathematical model is to generate different eligible seeds within the search space to complete the overall shape of a Sierpinski triangle. In this regard, the methodology of creating new seeds within a Sierpinski triangle is also utilized. For each of the eligible seeds in the search space (Xi), a temporary triangle is drawn with three seeds as follows:

- The position of the so far found global best (*GB*).
- The position of the mean group (*MGi*).
- The position of the *i*th solution candidate (*Xi*) as the selected seed.

The best candidate solution found so far, with the highest qualification, is denoted as GB. In contrast, the average of the values of some randomly selected qualified seeds with equal probability is denoted MGi, which includes the current initial qualified seed (Xi). The vertices of the Sierpinski triangle, consisting of GB, MGi, and the selected qualified seed (Xi), are temporary triangles for each of the initial qualified seeds in the search space. This triangle is intended to generate some new seeds within the search space that can

be considered new qualified seeds to complete the Sierpinski triangle. Figure 11a shows a schematic of the generated temporary triangles, while Figure 11b provides a detailed schematic description of this aspect [21].

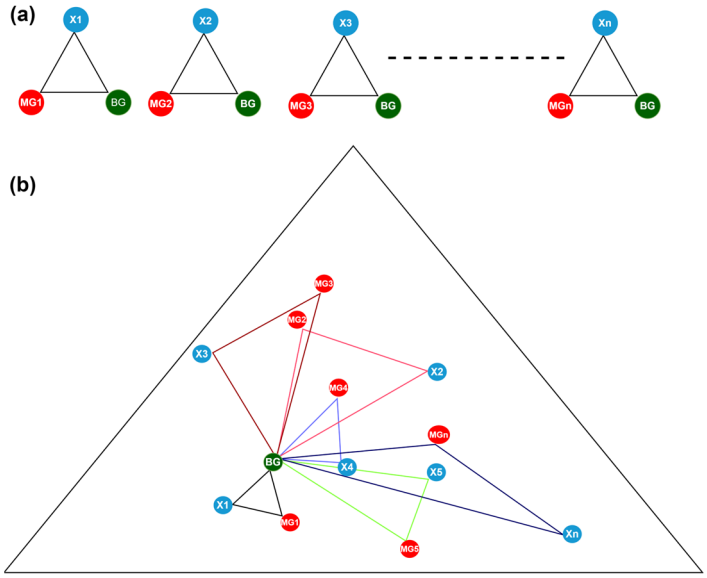


Figure 11. (**a**) Schematic view of creating temporary triangles; (**b**) temporary triangles in the search space.

Generating new qualified seeds in the search space is the primary purpose of constructing temporary triangles. Four methods have been developed to achieve this goal. The ith temporary triangle (the *i*th iteration) contains the n available qualified seeds obtained in the previous iteration and three vertices of the Sierpinski triangle [GB (green seed), MGi (red seed), and Xi (blue seed)]. Three seeds in this temporary triangle are used to generate new seeds according to the chaos game methodology. The first seed is placed at Xi, the second at GB, and the third at MGi. For the first seed, a die with three green faces and three red faces is used. The die is thrown and, depending on the color (green or red) shown, the seed at Xi is moved toward GB (green face) or MGi (red face). To select the green or red faces, a second random integer generator with a value of 1 or 0 is used. If the green face appears, the seed at Xi moves toward GB, while if the red face appears, the seed at Xi moves toward MGi. Although the probability of both green and red faces appearing in the game is equal, the possibility of generating two identical random integers for GB and MGi is also considered, as the seed moves in Xi towards a point on the connected lines between GB and MGi. Since the seed's movement in the search space must be limited due to the chaos game methodology, some randomly generated operators are used to control this aspect. Figure 12a shows a schematic representation of the process described for the first seed, while the mathematical representation of this process is shown in Equation (22) [21]:

$$\mathrm{Seed}_{\mathrm{i}}^{1} = \mathrm{X_i} + \alpha_{\mathrm{i}} \times (\beta_{\mathrm{i}} \times \mathrm{GB} - \gamma_{\mathrm{i}} \times \mathrm{MG_i}), \mathrm{i} = 1, 2, \ldots, \mathrm{n}. \quad (22)$$

where $\mathbf{X_i}$ is the ith solution candidate, **GB** is the global best solution found so far, and $\mathbf{MG_i}$ is the mean value of some selected eligible seeds. $\boldsymbol{\alpha_i}$ is the randomly generated factorial for modeling the movement limitations of the seeds. At the same time, each of the $\boldsymbol{\beta_i}$ and $\boldsymbol{\gamma_i}$ are generated by a random integer of 0 or 1 for modeling the possibility of rolling a die.

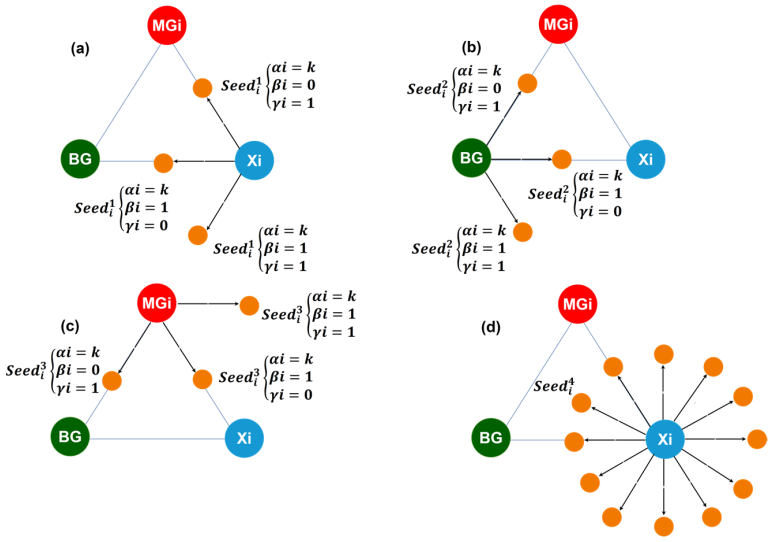


Figure 12. Schematic view of position updates for the (**a**) first, (**b**) second, (**c**) third, and (**d**) fourth seed in the search space.

For the second seed (**GB**), a die with three blue and three red sides is used. The die is thrown, and, depending on the color (blue or red) shown, the seed in **GB** is moved towards $\mathbf{X_i}$ (blue side) or $\mathbf{MG_i}$ (red side). This side is designed as shown for the first seed. If the blue side appears, the seed moves towards $\mathbf{X_i}$; if the red side appears, the seed moves towards $\mathbf{MG_i}$. As in the process of moving the first seed, the second seed can move towards a point on the line connecting $\mathbf{X_i}$ and $\mathbf{MG_i}$. This movement is determined by using some randomly generated operators. Figure 12b provides a schematic representation of the process described for the second seed, while the mathematical representation of this process is shown in Equation (23) [21]:

$$Seed_i^2 = GB + \alpha_i \times (\beta_i \times X_i - \gamma_i \times MG_i), i = 1, 2, \ldots, n. \quad (23)$$

where $\boldsymbol{\alpha_i}$ is the randomly generated factorial for modeling the movement limitations of the seeds while each of the $\boldsymbol{\beta_i}$ and $\boldsymbol{\gamma_i}$ selected by a random integer of 0 or 1 for modeling the possibility of rolling a die.

The third seed, called MGi, has two sides: blue and green. It uses a random integer generator function that generates only two integers, 0 and 1. When a zero appears, the seed moves to Xi (the blue side), or when a one appears, it moves to GB (the green side). It is worth noting that the seed can also move along the lines connecting Xi and GB. The mathematical equation in Equation (24) explains this process:

$$Seed_i^3 = MG_i + \alpha_i \times (\beta_i \times X_i - \gamma_i \times GB), i = 1, 2, \ldots, n. \quad (24)$$

The specific process for the third seed is described in Figure 12c. To generate the fourth seed, a different method is used to perform the mutation phase on the eligible seed locations in the search space. Random modifications in randomly selected decision variables determine the locations of this seed. Figure 12d shows a schematic representation of the process described for the fourth seed, where this aspect is represented mathematically in Equation (25) [21]:

$$Seed_i^4 = X_i\left(X_i^k = X_i^k + R\right), k = [1, 2, \ldots, d]. \quad (25)$$

where k is a random integer in the interval of $[1, d]$, and R is a uniformly distributed random number in the interval of $[0, 1]$.

To control and adjust the exploration and exploitation rate of the CGO algorithm, four different formulations are presented for $\boldsymbol{\alpha_i}$, which controls the movement limitations of the seeds using Equation (26):

$$\boldsymbol{\alpha_i} = \begin{cases} Rand \\ 2xRand \\ (\delta xRand) + 1 \\ (\varepsilon xRand) + (\sim \varepsilon) \end{cases} \quad (26)$$

where *Rand* is a uniformly distributed random number in the interval of [0, 1], while δ and ε are random integers in the interval of [0, 1].

To determine whether a new seed should be included among the eligible seeds in the search space, the eligibility of the available seeds and the new seeds should be examined together to avoid self-similarity problems in fractals. The quality of the new candidate solutions is compared with the available ones; the best ones are retained, and the seeds with the worst fitness values, which correspond to the worst self-similarity levels, are excluded. To reduce the complexity of the mathematical model, the substitution process is applied in the mathematical approach. All eligible seeds found so far in the search space are used to complete the general form of the Sierpinski triangle. To deal with the solution variables (X_j^i) violating the boundary conditions of the variables, a mathematical flag is defined, which for the X_j^i is outside the variable's range, and the flag orders a boundary change for the violating variables. The flow chart of the algorithm is presented in Figure 13.

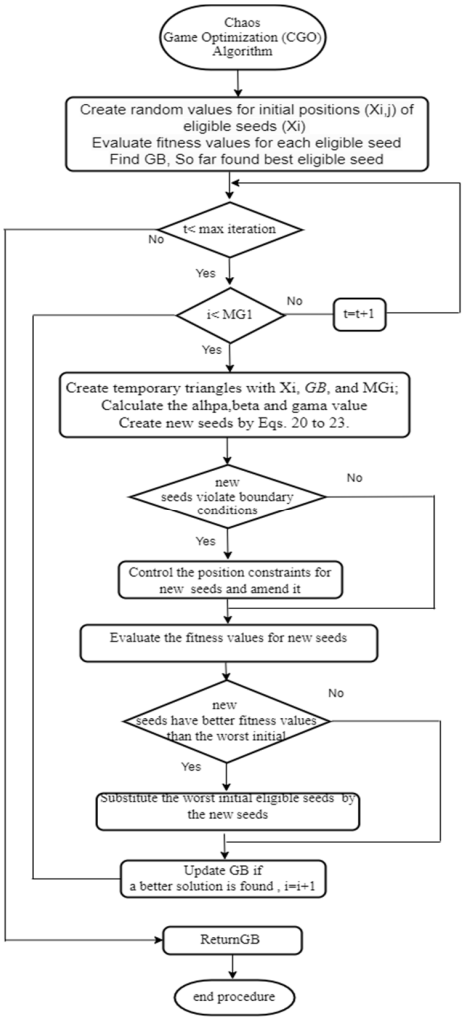


Figure 13. The flow chart of the CGO algorithm.

3. Results

The simulation of the DC microgrid system model shown in Figure 9 was carried out using MATLAB/Simulink software. Ten optimization tools (CGO, GWO, CSA, HHO, WOA, SSA, SCA, PSO, TSO, and DBO) were utilized to find the optimal values of the adaptive PI controller gains [Ki1, Ki2, Ki3]. The simulation period was established at 15 s, and a sudden local load was introduced between 10 and 12 s. The sudden local load values are provided in Table 2.

Table 2. Sudden local load.

Sudden Local Load RL [Ohm]	
DGU 1	RL = 20 ohms
DGU 2	RL = 18 ohms
DGU 3	RL = 16 ohms
DGU 4	RL = 14 ohms

The system's performance was evaluated based on the percentage error in current sharing, which indicates the deviation from the ideal equal current distribution among all DGUs in the mesh. The percentage error in the current sharing and the value of equal current sharing are calculated using Equations (27) and (28).

$$e_i\% = \frac{I_{equalshare} - I_{outi}}{I_{equalshare}} \times 100 \quad (27)$$

$$I_{equalshare} = \frac{1}{n}\sum_{k=1}^{n}(I_{outi}) \quad (28)$$

The performance of gains is measured and drawn by minimum root-mean-square error (RMS) for the average percentage of absolute error in the current line by the Equations (29) and (30).

$$\overline{E_i} = \frac{1}{n}\sum_{l=1}^{n} I_{equalshare} \quad (29)$$

$$RMS\ Error = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\overline{E_i}^2} \quad (30)$$

All optimization tools use a search value of 25 and a maximum of 10 iterations, with a boundary limit of [0, 1]. Table 3 presents the results of all optimization tools, which calculate gains based on the root-mean-square (RMS) error. To ensure that the results were not random or occurred by chance, the experiment was repeated 10 times, and the best results were chosen. The mean, standard deviation, and *t*-test were calculated for each sample in Table 4. The minimum RMS error achieved by the CGO optimizer is lower by 0.38%, 1.01%, 1.26%, 1.32%, 1.71%, 3.09%, 3.32%, 5.25%, and 5.15% compared to the GWO, CSA, HHO, WOA, SSA, SCA, PSO, TSO, and DBO optimizers, respectively. The minimum runtime achieved by the GWO optimizer is shorter by 76.85%, 16.09%, 56.55%, 0.59%, 57.17%, 58.03%, 9.72%, 4.84%, and 4.35% compared to CGO, CSA, HHO, WOA, SSA, SCA, PSO, TSO, and DBO, respectively. The proposed DC microgrid system is tested under two events: meshing at t = 1.5 s and a sudden local load change for each distributed generation unit (DGU) during the period of t = [10 s, 12 s], when it is required to supply 1.5 times its previous output current. This test is conducted to evaluate the current-sharing capability of the DGUs.

Table 3. All the optimization tool results.

Algorithm	K1	K2	k3	RMS Error	Searching Time (sec)
CGO	0.9714	0.0081	1.0000	1.8117	427,161
GWO	0.9288	0.0121	0.7139	1.8187	98,888
CSA	0.9501	0.0186	0.4384	1.8301	117,855
HHO	0.9295	0.0227	0.3997	1.8348	227,608
WOA	1.0000	0.0243	0.3497	1.8360	99,480
SSA	0.8979	0.0209	0.6652	1.8433	230,876
SCA	0.9705	0.0348	0.3962	1.8694	235,628
PSO	0.9972	0.0000	0.9825	1.8738	109,531
TSO	0.9949	0.0587	0.1705	1.9120	103,917
DBO	0.9037	0.0597	0.3933	2.3214	335,267
YALMIP	1	0.8	0.9	2.6386	

Table 4. Statistical analysis of the compared algorithms.

Algorithm	RMS Error Repeated Results (10 Times)					Std. Dev.	Minimum	Mean	*t*-Test
CGO	1.8772	1.8240	1.8135	1.8117	1.8117	0.020578	1.8117	1.819557	-
	1.8117	1.8117	1.8117	1.8117	1.8117				
GWO	1.9975	1.9803	1.8663	1.8248	1.8248	0.070156	1.8187	1.857706	0.052429
	1.8191	1.8191	1.8191	1.8189	1.8187				
CSA	1.8515	1.8433	1.8433	1.8373	1.8369	0.008196	1.8278	1.835744	8.08×10^{-7}
	1.8322	1.8297	1.8278	1.8278	1.8278				
HHO	1.9975	1.8510	1.8418	1.8369	1.8369	0.050524	1.8348	1.853767	2.33×10^{-11}
	1.8369	1.8364	1.8361	1.8353	1.8348				
WOA	1.9975	1.9975	1.8412	1.8412	1.8375	0.067455	1.836	1.868511	0.016809
	1.8364	1.8364	1.8360	1.8360	1.8360				
SSA	1.9975	1.9975	1.9104	1.8869	1.8675	0.061846	1.8433	1.887288	0.01394
	1.8480	1.8437	1.8437	1.8433	1.8433				
SCA	1.9975	1.8920	1.8920	1.8920	1.8920	0.037298	1.8694	1.895449	2.3×10^{-8}
	1.8920	1.8920	1.8694	1.8694	1.8694				
PSO	1.9066	1.8844	1.8844	1.8738	1.8738	0.010588	1.8738	1.879173	0.02222
	1.8738	1.8738	1.8738	1.8738	1.8738				
TSO	1.9975	1.9143	1.9143	1.9143	1.9143	0.026575	1.912	1.921768	4.41×10^{-13}
	1.9143	1.9143	1.9120	1.9120	1.9120				
DBO	2.7024	2.6145	2.4315	2.3718	2.3512	0.136239	2.3214	2.407784	2.24×10^{-7}
	2.3508	2.3245	2.3214	2.3214	2.3214				

The minimum RMS error achieved by the CGO optimizer is 31.33% lower than that of YALMIP [26] and 1.32% lower than that of WOA [28]. The CGO algorithm takes a long time during the search stage but achieves the quickest response to a steady state, reaching a value of 10.5% during the run stage.

Figure 14 demonstrates that the CGO has the lowest RMS error when compared to ten optimizers. Figure 15 presents the standard deviation for an experiment repeated over ten iterations.

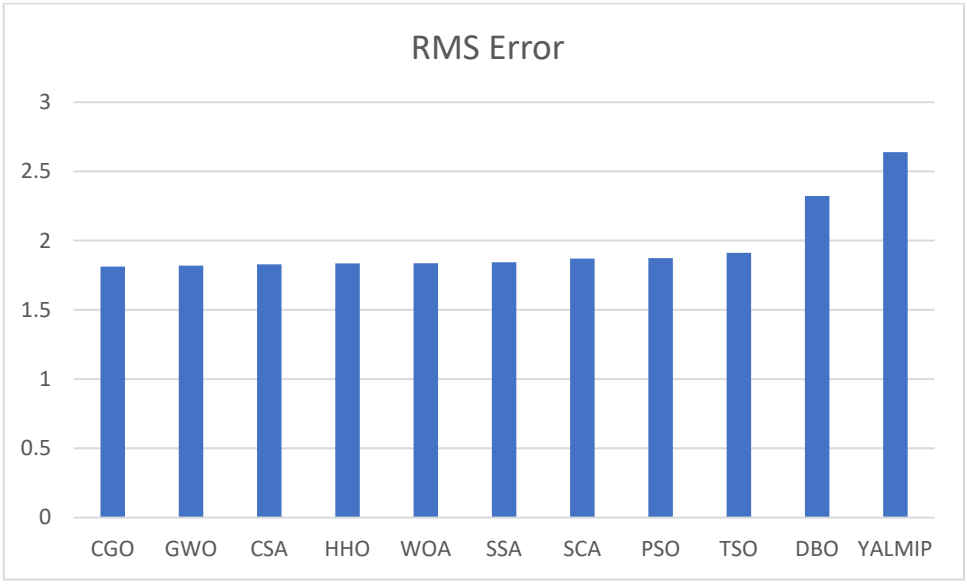


Figure 14. RMS error All _Algorithms.

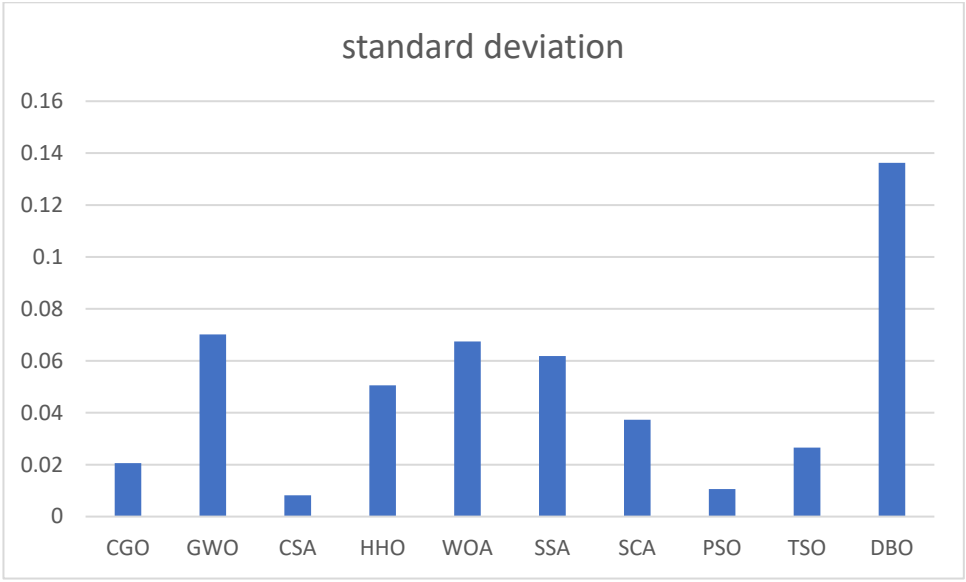


Figure 15. Standard deviation All _Algorithms.

The settings of the algorithms applied in this paper are displayed in Table 5.

These results are shown in Figure 5, where a reference voltage of 48 Vdc is applied to each DGU. The optimization methods mentioned above are utilized to achieve the best results during sudden load changes, particularly between 10 and 12 s.

Figures 16–31 show zoomed-in views of the positive and negative edges during sudden load changes. The first zoomed area focuses on the CGO, GWO, HHO, SCA, and PSO optimizers when the load is applied, while the second zoomed area highlights the CSA, WOA, SSA, and TSO optimizers under the same conditions. Similarly, the third zoomed area examines the CGO, GWO, HHO, SCA, and PSO optimizers when the load is released, and the fourth zoomed area concentrates on the CSA, WOA, SSA, TSO, and DBO optimizers during the load release phase.

Table 5. Specific settings of the compared algorithms.

Algorithms	Specific Parameters	Value
CGO	α	Random 1 to 4
	β	Random 1 to 2
	γ	Random 1 to 2
GWO	a1	Random 2 to 0
	r1	Random 0 to 1
	r2	Random 0 to 1
CSA	c	0.8
	p	0.991
HHO	r1, r2, r3, and r4	Random 0 to 1
	q	Random 0 to 1
	E1	Random 2 to 0
	E0	Random -2 to 0
WOA	r1 and r2	Random 0 to 1
	a	Random 0 to 2
	b	1
	p	Random 0 to 1
SSA	c1	1 to 1.054
	c2 and c3	Random 0 to 1
	s	1
SCA	r1	Random 2 to 0
	r2	Random 0 to 2π
	r3 and r4	Random 0 to 1
	t	2
	a	2
PSO	w1	0.5 to 0.3
	c1	2
	c2	2
TSO	r1, r2, and r3	Random 0 to 1
	k	1
DBO	p	0.2
	a	1
	r2	Random 0 to 1
	b1	0.3
	b2	0.1

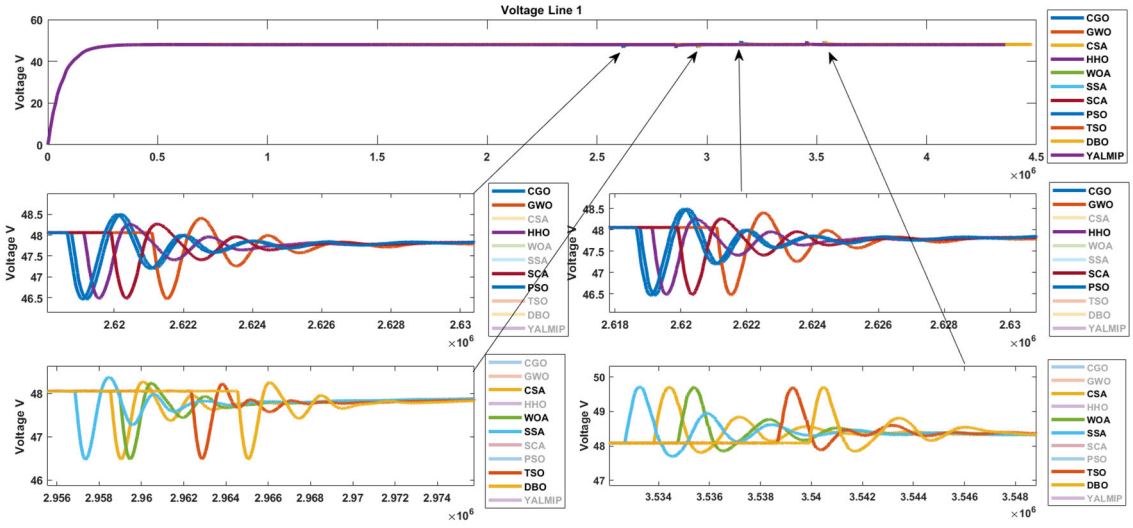


Figure 16. Voltage Line 1.

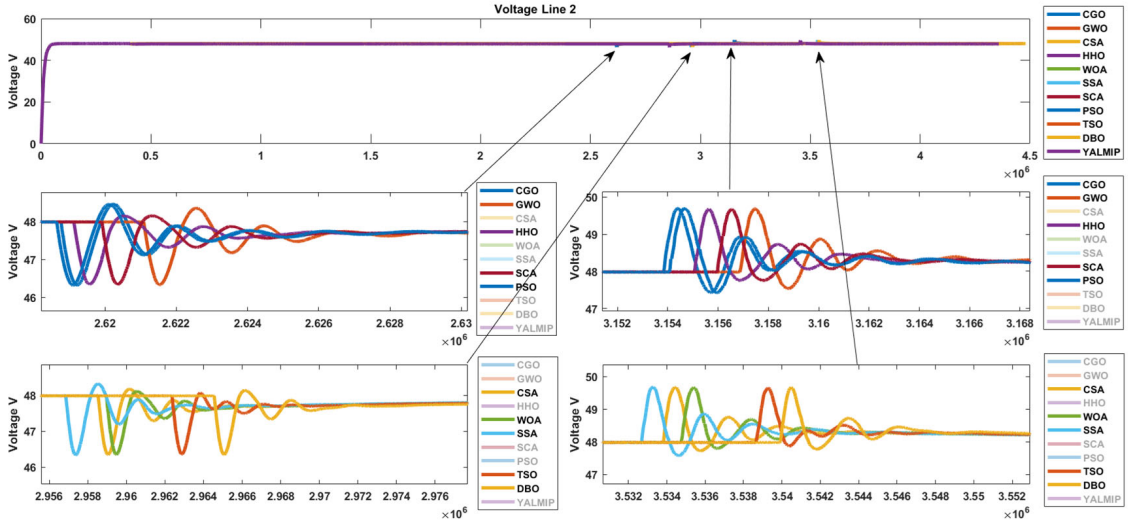


Figure 17. Voltage Line 2.

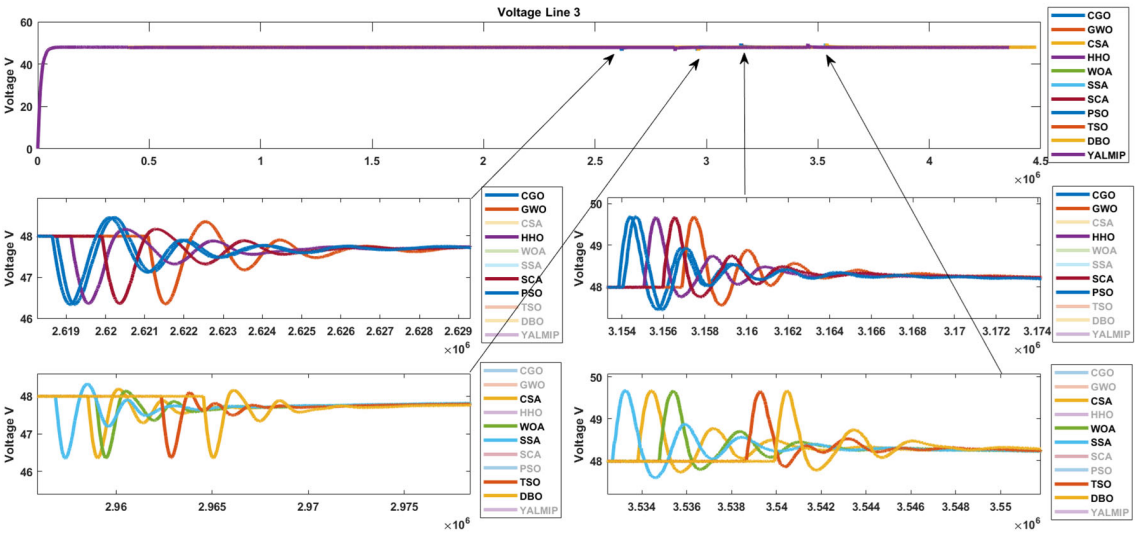


Figure 18. Voltage Line 3.

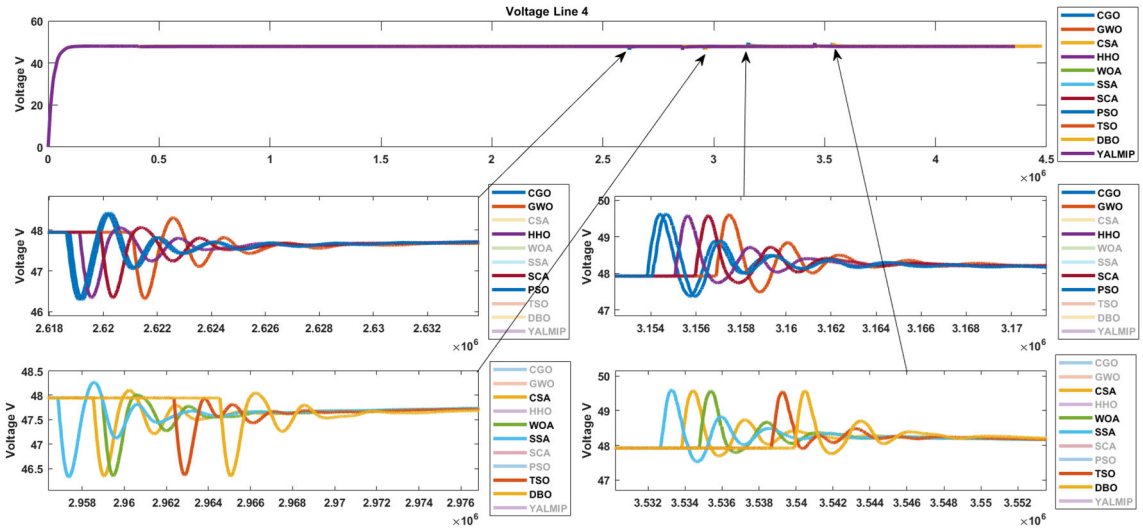


Figure 19. Voltage Line 4.

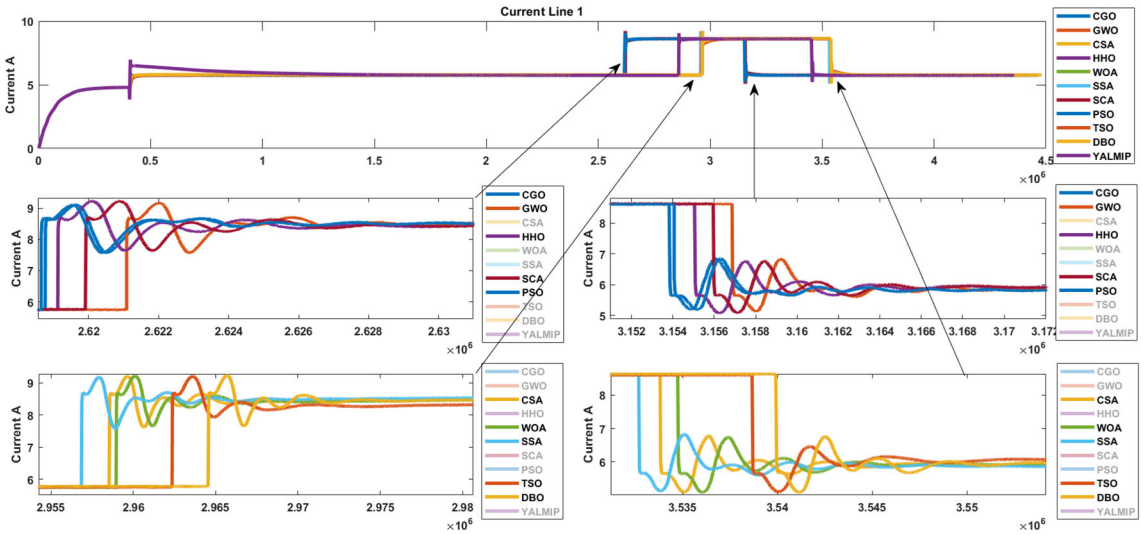


Figure 20. Current Line 1.

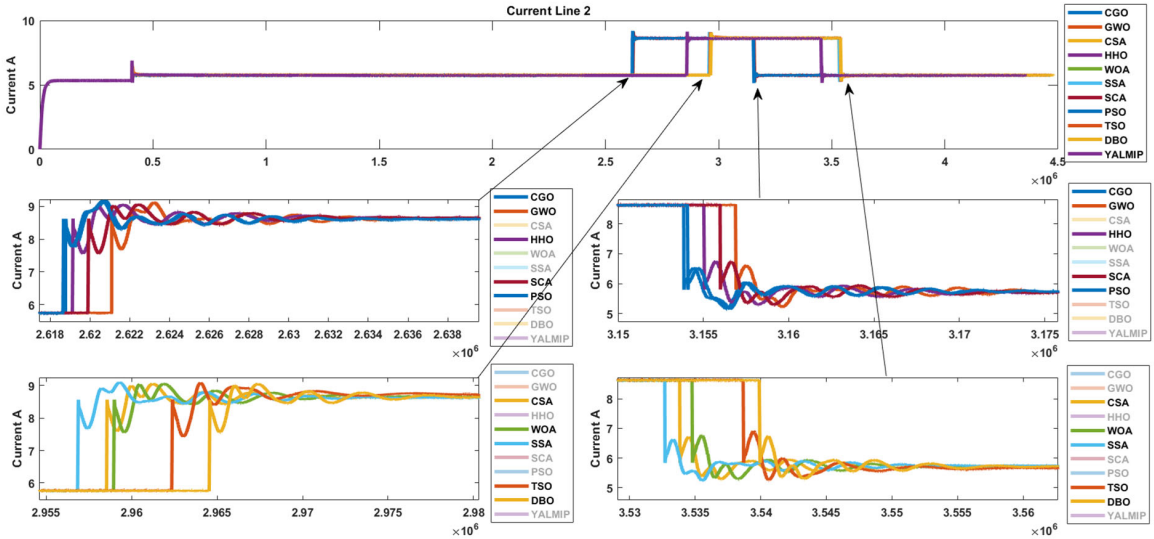


Figure 21. Current Line 2.

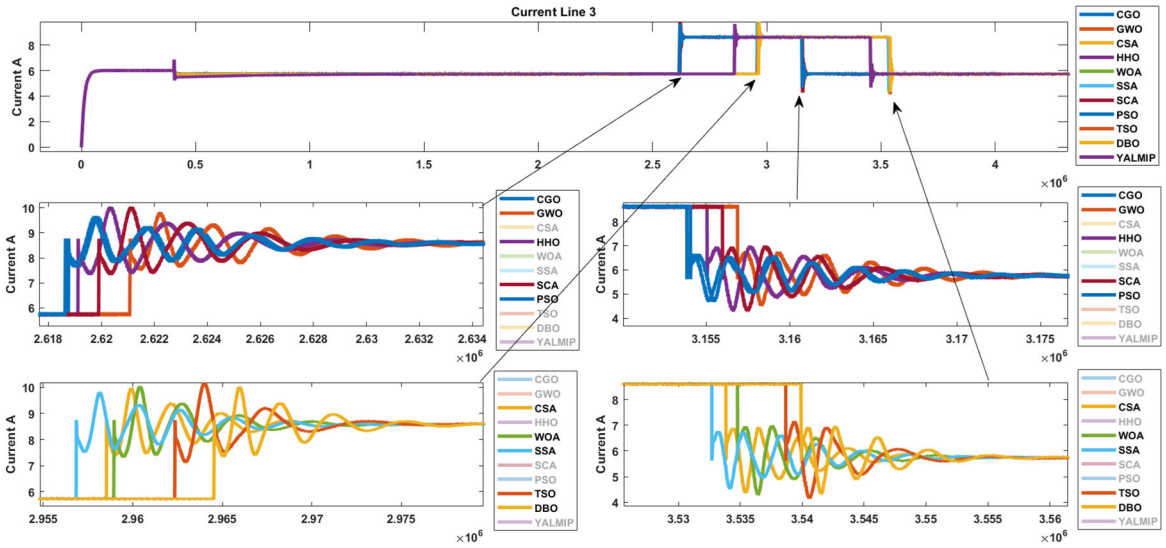


Figure 22. Current Line 3.

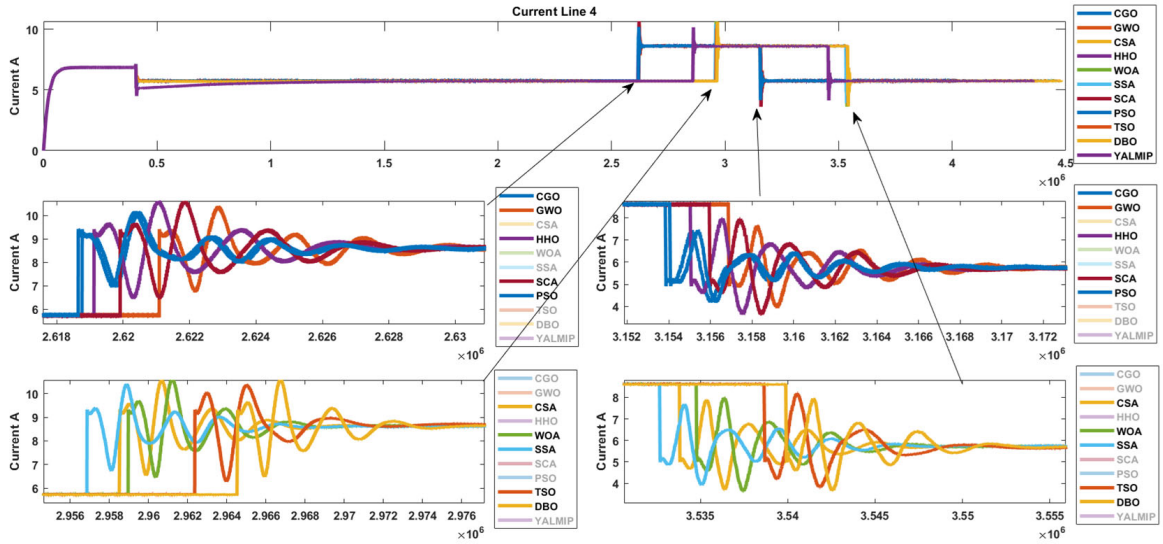


Figure 23. Current Line 4.

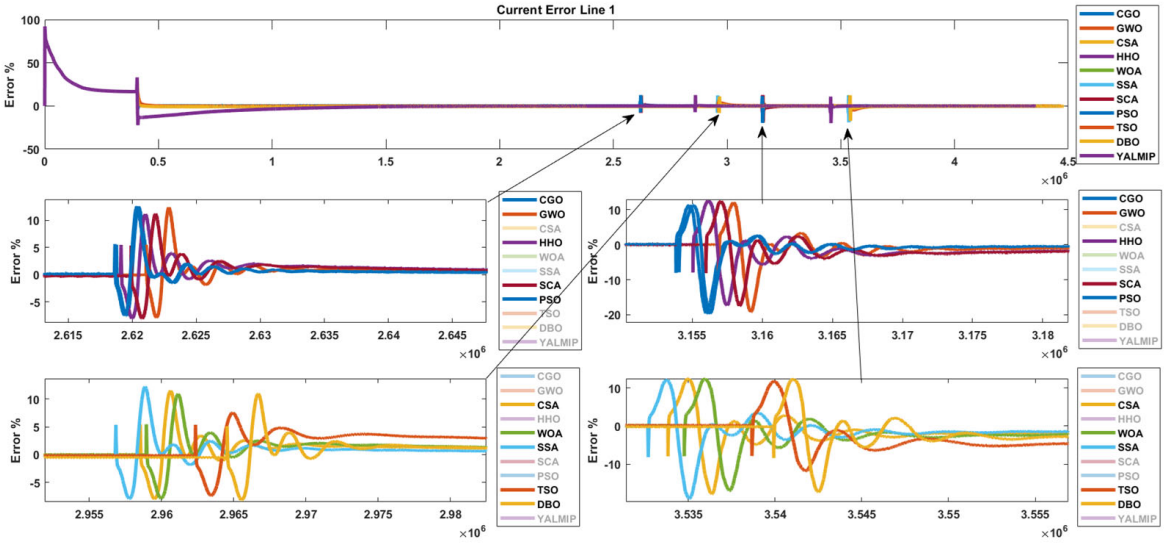


Figure 24. Current Error Line 1.

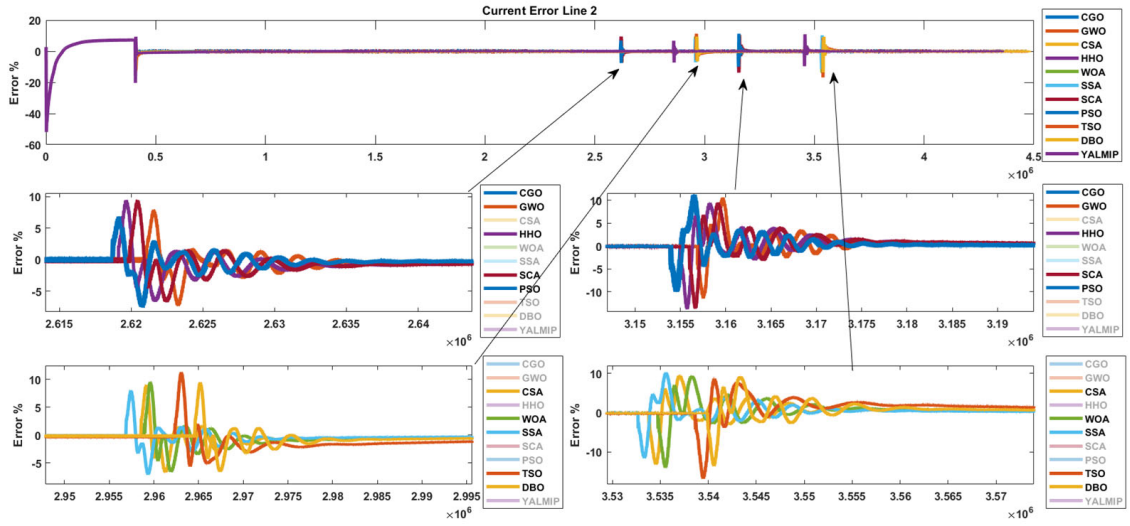


Figure 25. Current Error Line 2.

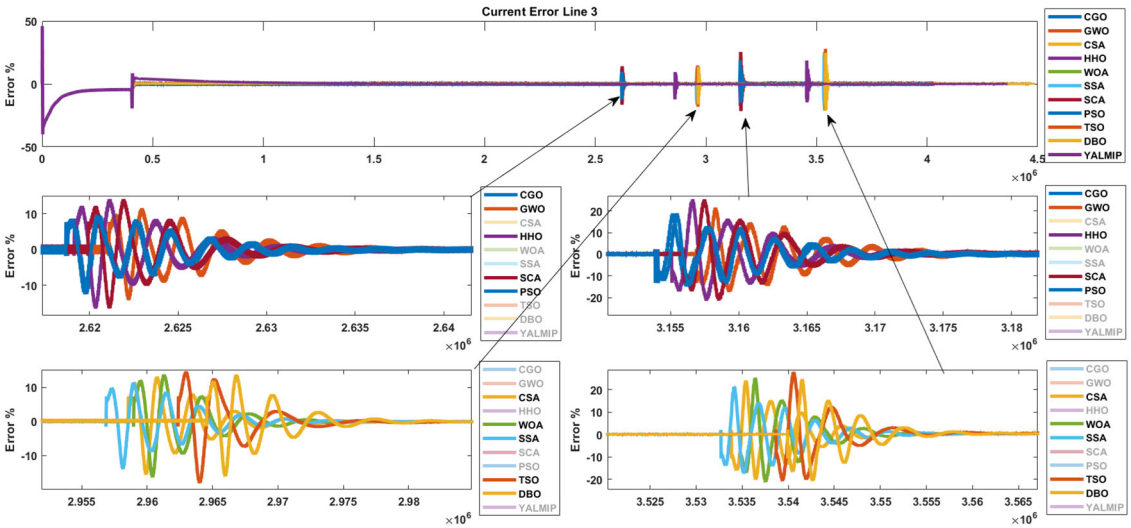


Figure 26. Current Error Line 3.

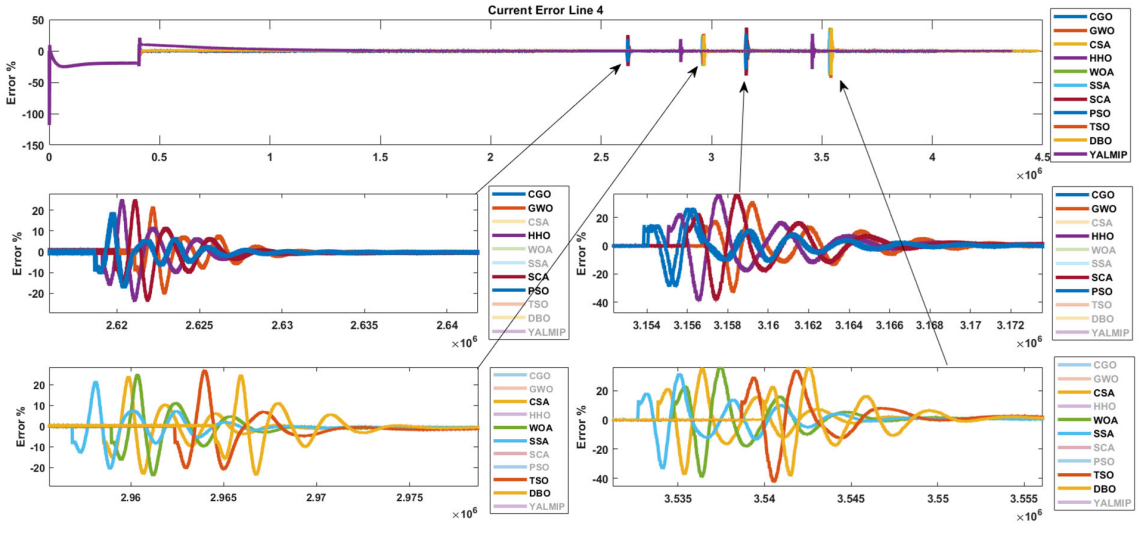


Figure 27. Current Error Line 4.

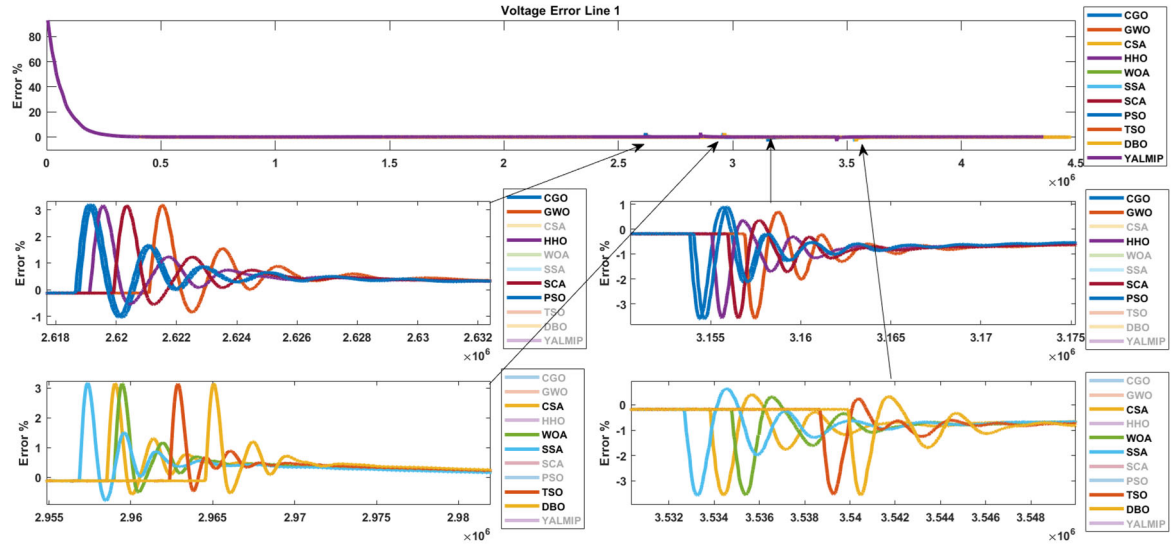


Figure 28. Voltage Error Line 1.

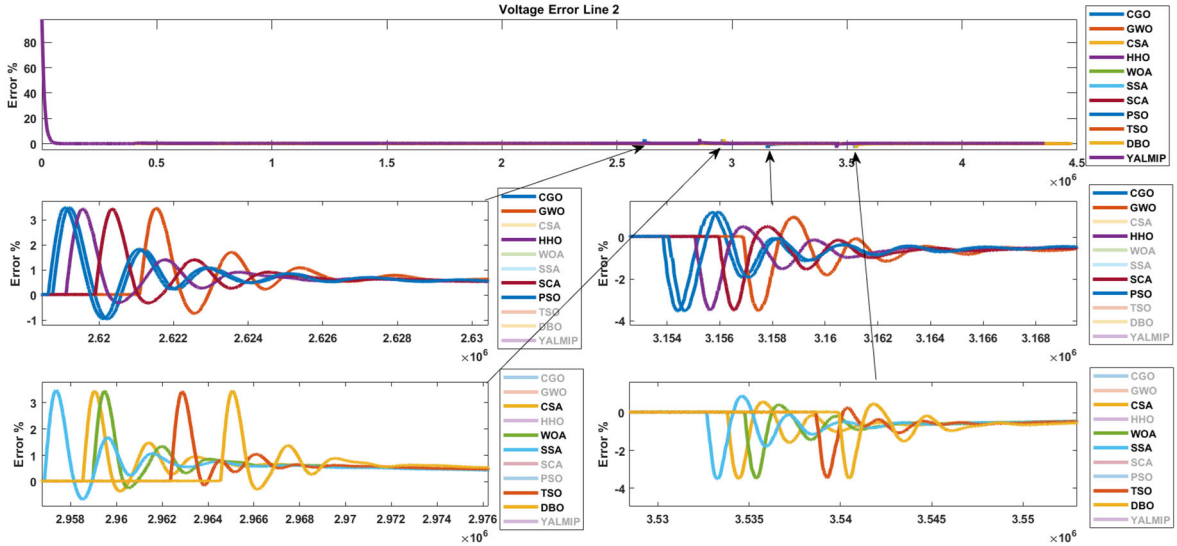


Figure 29. Voltage Error Line 2.

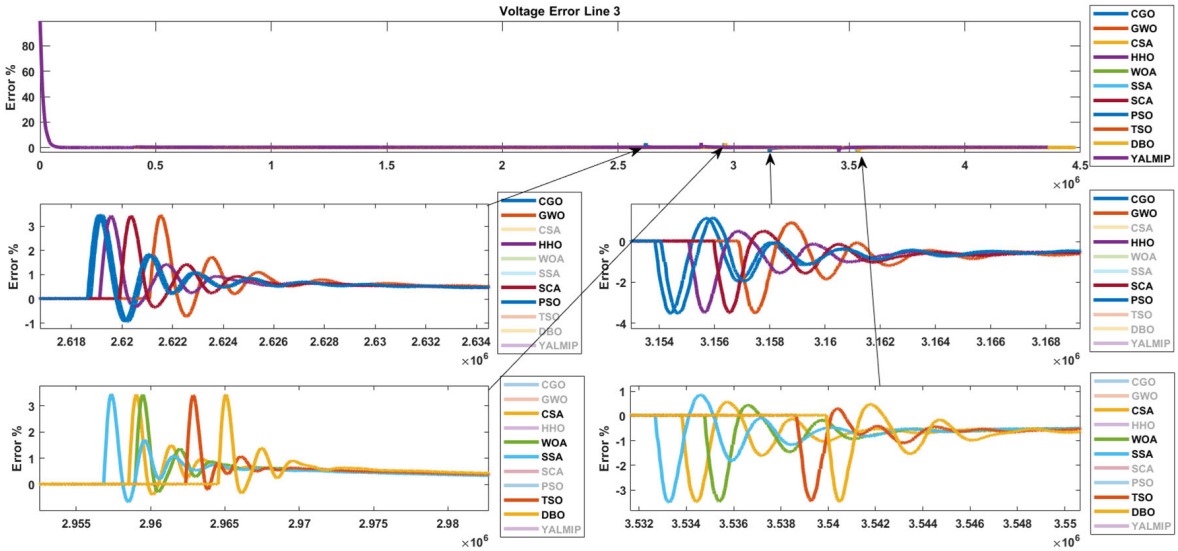


Figure 30. Voltage Error Line 3.

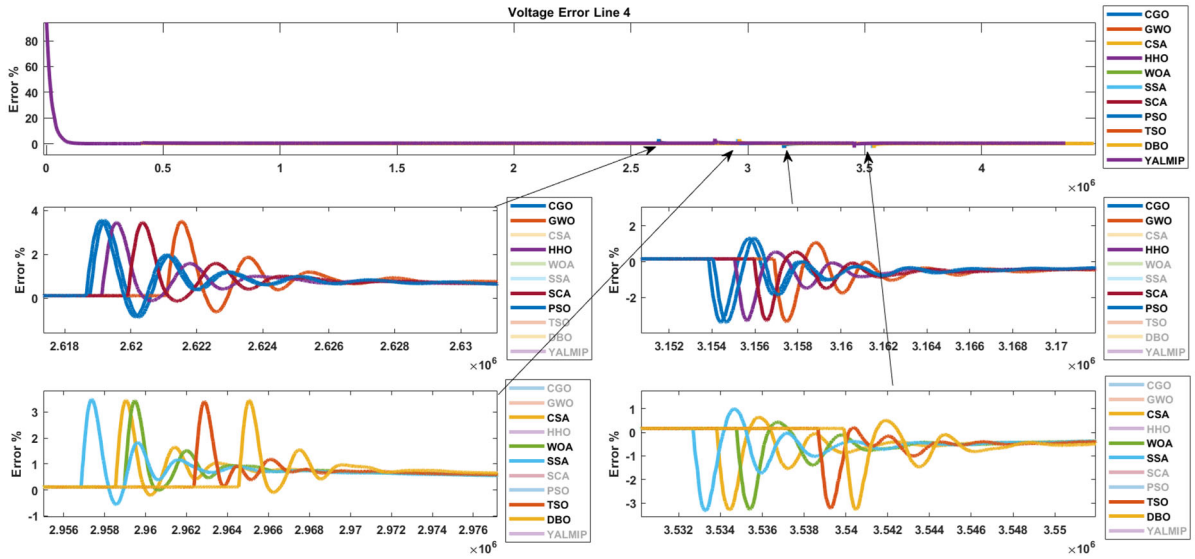


Figure 31. Voltage Error Line 4.

Figure 16 illustrates Voltage Line 1, depicting oscillations around the reference voltage. The RMS error is employed to evaluate performance, with a maximum overshoot of ±1.6 V DC. The TSO optimizer demonstrates the least oscillation, while the CGO optimizer achieves the fastest response to a steady state, registering a value of 10.5%.

Figure 17 illustrates Voltage Line 2, showing oscillations around the reference voltage. The RMS error, calculated to assess performance, resulted in a maximum overshoot of 1.5 V DC. The CGO optimizer demonstrates the least oscillation and the fastest response to steady state, with a value of 10.5%.

Figure 18 illustrates Voltage Line 3, displaying oscillations around the reference voltage, with a maximum overshoot of ±1.4 V DC. The CGO optimizer exhibits minimal oscillation and responds the fastest to steady state, with a value of 10.5%.

Figure 19 illustrates Voltage Line 4, showing oscillations around the reference voltage, with a maximum overshoot of ±1.5 V DC. The CGO and GWO optimizers exhibit the least oscillation and the fastest response to a steady state, both at a value of 10.5%.

Figure 20 illustrates Current Line 1, where a sudden load change causes overshoots and peaks, with a maximum overshoot of ±8% of the normal current. The CGO optimizer performs optimally, exhibiting the lowest oscillation and the fastest response to steady state.

Figure 21 illustrates Current Line 2, where a sudden change in load causes overshoots and peaks, resulting in a maximum overshoot of ±10% of the normal current. The CGO optimizer performs best, demonstrating the lowest oscillation and the fastest response to steady state.

Figure 22 illustrates Current Line 3, where a sudden load change causes overshoots and peaks, resulting in a maximum overshoot of ±30% of the normal current. The CGO optimizer demonstrates superior performance with minimal oscillation and a faster response to steady state.

Figure 23 illustrates Current Line 4, where a sudden load change causes overshoots and peaks, with a maximum overshoot of ±33% of the normal current. The CGO optimizer demonstrates the most effective performance, exhibiting the lowest oscillation and the quickest response to a steady state.

Figure 24 illustrates the current error at Line 1, showing a maximum deviation of ±10%. The CGO optimizer achieves the lowest RMS error, followed by GWO.

Figure 25 illustrates the current error at Line 2, indicating a maximum deviation of ±8%. The CGO optimizer achieves the lowest RMS error, followed by GWO.

Figure 26 illustrates the current error at Line 3, showing a maximum deviation of ±18%. The CGO and GWO optimizers present the lowest RMS error.

Figure 27 illustrates the existing error at Line 4, showing a maximum deviation of ±20%. The CGO optimizer achieves the lowest RMS error.

Figure 28 illustrates the voltage error in Line 1, showing a maximum deviation of ±3%. The CGO optimizer demonstrates the lowest RMS error, followed by GWO.

Figure 29 illustrates the voltage error in Line 2, showing a maximum deviation of ±3.5%. The CGO optimizer exhibits the lowest RMS error, followed by GWO.

Figure 30 illustrates the voltage error in Line 3, showing a maximum deviation of ±3.2%. The CGO optimization results in the lowest RMS error, followed by GWO.

Figure 31 illustrates the voltage error in Line 4, showing a maximum deviation of ±3.1%. The CGO optimization results in the lowest RMS error, followed by GWO.

4. Discussion

The performance of the optimization methods can be ranked from the lowest to the highest oscillation in the following order: TSO, DBO, CGO, GWO, WOA, CSA, SCA, SSA, HHO, and PSO. The simulation run times can be ranked from fastest to slowest based on the response achieved when using the optimization tools: CGO, GWO, PSO, HHO, SCA, SSA, CSA, WOA, TSO, and DBO.

5. Conclusions

To address DC microgrid control issues, adaptive PI control serves as a secondary control layer for the DGU, enhanced by ten types of optimization tools to boost the performance of the DGU mesh model. RMS error determines the optimal adaptive PI control gains. The CGO tool achieved the lowest RMS error, with a ratio of 5.25% of the maximum error, and the current error did not exceed ±15%. The simulation run time is 75% faster when using selective gains based on optimization tools. As a result of reducing the error rate and accelerating the model's running time, wasted energy consumption was minimized, achieving a state of independence. Steady state is completed 50% faster when sudden changes in model current, resulting from incorrect data or unrealistic loads, are addressed. The effect of low current oscillations on energy storage losses has been considered in the modified version of the manuscript.

Author Contributions: A.S.H.: Methodology, Conceptualization, Writing—original draft.; I.M.D.: Data curation, Conceptualization, Writing—review & editing; N.M.B.: Supervision, Writing—review & editing, Formal analysis; M.A.A.: Project administration, Conceptualization, review & editing, A.O.B.: review & editing, Supervision, Formal analysis; O.A.M.O.: review & editing, Supervision, Formal analysis; A.H.E.-E.: review & editing, Supervision, Formal analysis; H.-S.K.: Funding acquisition, review & editing, Conceptualization. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korean government (Ministry of Science and ICT) (RS-2023-NR076833, 50%), and partly by the Innovative Human Resource Development for Local Intellectualization program through the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (Ministry of Science and ICT (MSIT)) (IITP-2025-RS-2020-II201462, 50%).

Data Availability Statement: The datasets used and/or analyzed during the current study are available from the corresponding author.

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

References

1. Ashok Kumar, A.; Amutha Prabha, N. A comprehensive review of DC microgrid in market segments and control technique. *Heliyon* **2022**, *8*, e11694. [CrossRef] [PubMed]
2. Al-Kahtani, M.S.; Karim, L. A Survey on Attacks and Defense Mechanisms in Smart Grids. *Int. J. Comput. Eng. Inf. Technol.* **2019**, *11*, 94–100.
3. Banerji, A.; Sen, D.; Bera, A.K.; Ray, D.; Paul, D.; Bhakat, A.; Biswas, S.K. Microgrid: A Review. In Proceedings of the 2013 IEEE Global Humanitarian Technology Conference: South Asia Satellite (GHTC-SAS), Trivandrum, India, 23–24 August 2013; pp. 27–35. [CrossRef]
4. Zhang, Z.; Gong, S.; Dimitrovski, A.D.; Li, H. Time Synchronization Attack in Smart Grid: Impact and Analysis. *IEEE Trans. Smart Grid* **2013**, *4*, 87–98. [CrossRef]
5. Khoei, T.T.; Ould Slimane, H.; Kaabouch, N. A Comprehensive Survey on the Cyber-Security of Smart Grids: Cyber-Attacks, Detection, Countermeasure Techniques, and Future Directions. *arXiv* **2022**, arXiv:2207.07738.
6. Yakout, A.H.; AboRas, K.M.; Kotb, H.; Alharbi, M.; Shouran, M.; Abdul Samad, B. A Novel Ultra Local Based-Fuzzy PIDF Controller for Frequency Regulation of a Hybrid Microgrid System with High Renewable Energy Penetration and Storage Devices. *Processes* **2023**, *11*, 1093. [CrossRef]
7. Yakout, A.H.; Hasanien, H.M.; Turky, R.A.; Abu-Elanien, A.E.B. Improved reinforcement learning strategy of energy storage units for frequency control of hybrid power systems. *J. Energy Storage* **2023**, *72*, 108248. [CrossRef]
8. Llaria, A.; Dos Santos, J.; Terrasson, G.; Boussaada, Z.; Merlo, C.; Curea, O. Intelligent Buildings in Smart Grids: A Survey on Security and Privacy Issues Related to Energy Management. *Energies* **2021**, *14*, 2733. [CrossRef]
9. Danda, B.R.; Chandra, B. *Cyber Security for Smart Grid Systems: Status, Challenges and Perspectives*; Institute of Electrical and Electronics Engineers: New York, NY, USA, 2015. [CrossRef]
10. Al-Ismail, F.S. DC Microgrid Planning, Operation, and Control: A Comprehensive Review. *IEEE Access* **2021**, *9*, 36154–36172. [CrossRef]
11. Li, D.; Ho, C.N.M. A Module-Based Plug-n-Play DC Microgrid with Fully Decentralized Control for IEEE Empower a Billion Lives Competition. *IEEE Trans. Power Electron.* **2021**, *36*, 1764–1776. [CrossRef]
12. Pullaguram, D.; Mishra, S.; Senroy, N. Event-Triggered Communication Based Distributed Control Scheme for DC Microgrid. *IEEE Trans. Power Syst.* **2018**, *33*, 5583–5593. [CrossRef]
13. Kumar, J.; Agarwal, A.; Agarwal, V. A review on overall control of DC microgrids. *J. Energy Storage* **2019**, *21*, 113–138. [CrossRef]
14. Yamashita, D.Y.; Vechiu, I.; Gaubert, J.-P. A review of hierarchical control for building microgrids. *Renew. Sustain. Energy Rev.* **2020**, *118*, 109523. [CrossRef]
15. Olivares, D.E.; Mehrizi-Sani, A.; Etemadi, A.H.; Cañizares, C.A.; Iravani, R.; Kazerani, M.; Hajimiragha, A.H.; Gomis-Bellmunt, O.; Saeedifard, M.; Palma-Behnke, R.; et al. Trends in Microgrid Control. *IEEE Trans. Smart Grid* **2014**, *5*, 1905–1919. [CrossRef]
16. Li, X.; Hu, C.; Luo, S.; Lu, H.; Piao, Z.; Jing, L. Distributed Hybrid-Triggered Observer-Based Secondary Control of Multi-Bus DC Microgrids Over Directed Networks. *IEEE Trans. Circuits Syst. I Regul. Pap.* **2025**, *72*, 2467–2480. [CrossRef]
17. Yang, N.; Xu, G.; Fei, Z.; Li, Z.; Du, L.; Guerrero, J.M.; Huang, Y.; Yan, J.; Xing, C.; Li, Z. Two-Stage Coordinated Robust Planning of Multi-Energy Ship Microgrids Considering Thermal Inertia and Ship Navigation. *IEEE Trans. Smart Grid* **2025**, *16*, 1100–1111. [CrossRef]
18. Bidram, A.; Davoudi, A. Hierarchical Structure of Microgrids Control System. *IEEE Trans. Smart Grid* **2012**, *3*, 1963–1976. [CrossRef]
19. Mirjalili, S. SCA: A Sine Cosine Algorithm for solving optimization problems. *Knowl. Based Syst.* **2016**, *96*, 120–133. [CrossRef]
20. Mirjalili, S.; Lewis, A. The Whale Optimization Algorithm. *Adv. Eng. Softw.* **2016**, *95*, 51–67. [CrossRef]
21. Talatahari, S.; Azizi, M. Chaos Game Optimization: A novel metaheuristic algorithm. *Artif. Intell. Rev.* **2021**, *54*, 917–1004. [CrossRef]
22. Mirjalili, S.; Mirjalili, S.M.; Lewis, A. Grey Wolf Optimizer. *Adv. Eng. Softw.* **2014**, *69*, 46–61. [CrossRef]
23. Mirjalili, S.; Gandomi, A.H.; Mirjalili, S.Z.; Saremi, S.; Faris, H.; Mirjalili, S.M. Salp Swarm Algorithm: A bio-inspired optimizer for engineering design problems. *Adv. Eng. Softw.* **2017**, *114*, 163–191. [CrossRef]
24. Xue, J.; Shen, B. Dung beetle optimizer: A new meta-heuristic algorithm for global optimization. *J. Supercomput.* **2023**, *79*, 7305–7336. [CrossRef]
25. Riverso, S.; Sarzo, F.; Ferrari-Trecate, G. Plug-and-Play Voltage and Frequency Control of Islanded Microgrids with Meshed Topology. *IEEE Trans. Smart Grid* **2015**, *6*, 1176–1184. [CrossRef]
26. El-Ebiary, A.H.; Mokhtar, M.; Attia, M.A.; Marei, M.I. A Distributed Adaptive Control Strategy for Meshed DC Microgrids. In Proceedings of the 2023 IEEE Conference on Power Electronics and Renewable Energy (CPERE), Luxor, Egypt, 19–21 February 2023; pp. 1–6. [CrossRef]

27. Khader, A.H.; Yakout, A.H.; El-Sharkawy, M. Damping Interarea Oscillations Using PID Power System Stabilizer with Grey Wolf Algorithm and Particle Swarm Algorithm. In Proceedings of the 2019 21st International Middle East Power Systems Conference (MEPCON), Cairo, Egypt, 17–19 December 2019; pp. 330–336. [CrossRef]
28. Mosaad, A.M.; Attia, M.A.; Abdelaziz, A.Y. Whale optimization algorithm to tune PID and PIDA controllers on AVR system. *Ain Shams Eng. J.* **2019**, *10*, 755–767. [CrossRef]
29. Tucci, M.; Riverso, S.; Vasquez, J.C.; Guerrero, J.M.; Ferrari-Trecate, G. A Decentralized Scalable Approach to Voltage Control of DC Islanded Microgrids. *IEEE Trans. Control Syst. Technol.* **2016**, *24*, 1965–1979. [CrossRef]
30. Riverso, S.; Farina, M.; Ferrari-Trecate, G. Plug-and-play model predictive control based on robust control invariant sets. *Automatica* **2014**, *50*, 2179–2186. [CrossRef]
31. Yuan, M.; Fu, Y.; Mi, Y.; Li, Z.; Wang, C. Hierarchical control of DC microgrid with dynamical load power sharing. *Appl. Energy* **2019**, *239*, 1–11. [CrossRef]
32. El-Ebiary, A.H.; Attia, M.A.; Awad, F.H.; Marei, M.I.; Mokhtar, M. Kalman Filters Based Distributed Cyber-Attack Mitigation Layers for DC Microgrids. *IEEE Trans. Circuits Syst. I Regul. Pap.* **2024**, *71*, 1358–1370. [CrossRef]
33. EL-Ebiary, A.H.; Mokhtar, M.; Mansour, A.M.; Awad, F.H.; Marei, M.I.; Attia, M.A. Distributed Mitigation Layers for Voltages and Currents Cyber-Attacks on DC Microgrids Interfacing Converters. *Energies* **2022**, *15*, 9426. [CrossRef]
34. Lofberg, J. YALMIP: A Toolbox for Modeling and Optimization in MATLAB. In Proceedings of the 2004 IEEE International Conference on Robotics and Automation (IEEE Cat. No.04CH37508), Taipei, Taiwan, 2–4 September 2004; pp. 284–289. [CrossRef]
35. Choi, K.; Liu, H. Matlab and Simulink Basics. In *Problem-Based Learning in Communication Systems Using Matlab and Simulink*; Wiley: Hoboken, NJ, USA, 2016; pp. 1–15. [CrossRef]
36. Olfati-Saber, R.; Murray, R.M. Consensus Problems in Networks of Agents with Switching Topology and Time-Delays. *IEEE Trans. Autom. Contr.* **2004**, *49*, 1520–1533. [CrossRef]
37. Mokhtar, M.; Marei, M.I.; El-Sattar, A.A. Improved Current Sharing Techniques for DC Microgrids. *Electr. Power Compon. Syst.* **2018**, *46*, 757–767. [CrossRef]
38. Qais, M.H.; Hasanien, H.M.; Turky, R.A.; Alghuwainem, S.; Tostado-Véliz, M.; Jurado, F. Circle Search Algorithm: A Geometry-Based Metaheuristic Optimization Algorithm. *Mathematics* **2022**, *10*, 1626. [CrossRef]
39. Heidari, A.A.; Mirjalili, S.; Faris, H.; Aljarah, I.; Mafarja, M.; Chen, H. Harris hawks optimization: Algorithm and applications. *Future Gener. Comput. Syst.* **2019**, *97*, 849–872. [CrossRef]
40. Qais, M.H.; Hasanien, H.M.; Alghuwainem, S. Output power smoothing of grid-connected permanent-magnet synchronous generator driven directly by variable speed wind turbine: A review. *J. Eng.* **2017**, *2017*, 1755–1759. [CrossRef]
41. Qais, M.H.; Hasanien, H.M.; Alghuwainem, S. Transient search optimization: A new meta-heuristic optimization algorithm. *Appl. Intell.* **2020**, *50*, 3926–3941. [CrossRef]
42. Mech, L.D. Alpha Status, Dominance, and Division of Labor in Wolf Packs. 1999. Available online: https://digitalcommons.unl.edu/usgsnpwrc/353 (accessed on 24 June 2025).
43. Muro, C.; Escobedo, R.; Spector, L.; Coppinger, R.P. Wolf-pack (*Canis lupus*) hunting strategies emerge from simple rules in computational simulations. *Behav. Process.* **2011**, *88*, 192–197. [CrossRef]
44. Dahou, A.; Chelloug, S.A.; Alduailij, M.; Elaziz, M.A. Improved Feature Selection Based on Chaos Game Optimization for Social Internet of Things with a Novel Deep Learning Model. *Mathematics* **2023**, *11*, 1032. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.