Journal of Network Intelligence ©2021 ISSN 2414-8105 (Online)
Taiwan Ubiquitous Information Volume **6**, Number **1**, February **2021**

Multi-cluster Based Equilibrium Optimizer Algorithm with Compact Approach for Power System Network

Xing-Wei Xu

College of Computer Science and Engineering
Shandong University of Science and Technology
Qingdao 266590, China
2480886053@qq.com

Tien-Szu Pan

Department of Electronic Engineering
National Kaohsiung University of Science and Technology
Kaohsiung 807618, Taiwan
tpan@nkust.edu.tw

Pei-Cheng Song

College of Computer Science and Engineering
Shandong University of Science and Technology
Qingdao 266590, China
spacewe@outlook.com

Chia-Cheng Hu

College of Artificial Intelligence
Yango University
Fuzhou 350015, China
jjhwu@ksts.seed.net.tw

Shu-Chuan Chu*

College of Computer Science and Engineering
Shandong University of Science and Technology
Qingdao 266590, China
College of Science and Engineering
Flinders University
Adelaide, Australia
*Corresponding author: scchu0803@gmail.com

Received October 2020; revised January 2021

Abstract. *Optimal allocation of distributed generation (DG) focuses on optimal location and sizing of DGs for promoting energy conversion efficiency and quality in the power distribution system. Nowadays, the loss minimization goal has received significant attention since it gains enormous benefits between economic and environmental fields. In this paper, the sensitivity factor method is used to determine the optimal DG positions, which reduces the search space by finding the best locations. This paper first introduces a compact technology based on the equilibrium optimizer (EO) algorithm. The compact equilibrium optimizer (cEO) algorithm has a considerable advantage in reducing the memory space of the potential bus selection. Then this paper implements and optimizes the cEO algorithm by the method of update interval. The value of an optimal interval is selected to promote the maximum contribution of the equilibrium pool update. According to the characteristics of the equilibrium pool, two kinds of parallel compact algorithms, public parallel compact EO (public pcEO) and private parallel compact EO (private pcEO) algorithm with different structures, are proposed. Compared with other algorithms, public pcEO has achieved excellent performance with less memory space. The proposed algorithms are tested on CEC 2014 functions. By comparing with other basic algorithms, the experimental results showed that two parallel compact algorithms could obtain competitive results and avoid getting into the optimal local solution. Then two proposed parallel algorithms are tested compared with some general algorithms to gain the most suitable sizing of DGs in those selected potential locations and gained great results.*

Keywords: Optimal allocation of distributed generation, Equilibrium optimizer, Parallel communication strategy, Compact strategy, Loss sensitivity factor

1. Introduction.

The concept of optimization is widely used in the decision-making, allocation, and production planning of practical problems. While the key is whether these problems could achieve the optimal solution, the research on the optimization methods is of great value both in theory and in practice. Use the optimization method to determine the value of some optional variables under constraints for achieving the optimal value of the selected objective function. Aiming at different problems, diverse optimization method, they can be simplified and classified into two types (mathematical optimization and the heuristic searching technique), was developed.

Mathematical optimization methods can generally be divided into two categories: deterministic and stochastic. Linear and non-linear optimization problems [1] are both belong to deterministic optimization problems. Only when the constraint condition and objective function are linear, this problem has linear characteristics. If the objective function or constraint conditions is non-linear, the problem is defined as non-linear optimization [2]. We hope to find the global optimal solution to the problem, and the global optimum is from the local optimum. There is no doubt that finding a global optimum is harder than finding the local optimum. When some mathematical optimization problems have non-convex characteristics, it is hard to find the global optimum in limited computation cost and converges to the local optimum smoothly. When the optimization problem is convex, the local optimal solution can be considered equivalent to the global optimal solution.

Compared with the traditional deterministic mathematical optimization, the meta-heuristic algorithm improves the heuristic algorithm [3, 4], which combines the characteristics of the local search algorithm and random algorithm. One of its main characteristics is stochastic. Meta-heuristics is an iterative generation process, which utilizes specific methods in heuristic algorithms to find feasible solutions in the problem space. Meta-heuristic algorithm increases the probability of finding the global optimum by improving randomness and doesn't depend on a particular problem. Biological behaviors and physical phenomena inspire many meta-heuristic algorithms. In recent years, many classic meta-heuristic algorithms have appeared such as Genetic Algorithm (GA) [5], Particle Swarm Optimization (PSO) [6] and Ant Colony Optimization (ACO) [7]. Genetic algorithm uses three processes of selection, crossover, and mutation operations to update individuals in each generation, promoting the population to evolve better and better approach the optimal solution. In a given environment, this process may promote populations to be more adaptable than previous generations. Cai and Lei [8] proposes a new evolutionary algorithm for many-objective optimization. PSO algorithm is a classic swarm intelligence algorithm and comes from studying the predation behavior of birds. Through the cooperation and information sharing among individuals in a group, to find the optimal solution. PSO algorithm has great development potential and value. It can be used in many fields and plays an important role. Similar swarm intelligence algorithms include Grey Wolf Optimizer (GWO) [9,10], QUasi-Affine TRansformation Evolutionary (QUATRE) [11,12], Cat Swarm Optimization (CSO) [13,14], Cuckoo Search (CS) [15,16], Fish Migration Optimization (FMO) [17–19], Butterfly Optimization Algorithm (BOA) [20] and Phasmatodea Population Evolution Algorithm (PPE) [21].

Compared with meta-heuristic algorithms based on biological behavior, some meta-heuristic algorithms come from physical models, just like the improved algorithm in this paper. The equilibrium Optimizer (EO) algorithm is based on a mass conservation model, representing the change process of the mass in the solution under both dynamic and static equilibrium state. The EO algorithm's mathematical basis is the ordinary differential equation of time and mass, which follows the rules of conservation of mass. In EO, each particle's concentration is used as a search and update tool to find the optimal solution to the problem. The four best particles and their average particles in the search process constitute the equilibrium pool, it plays an important role in balancing algorithm exploitation and exploration capabilities [22]. These particles are considered equilibrium candidates. Randomly select a particle from the equilibrium candidates, and use the selected particle to update each particle in the solution according to specific rules, enhancing the algorithm's global search capabilities. The EO algorithm has been proposed recently which although can achieve good results the application of the EO algorithm is less, such as identifying the parameters of solar photovoltaic [23] and solving economic dispatch problem [24]. There are few related kinds of research on the optimal allocation of distributed generation, so this article attempts to use the improved EO algorithm to solve the optimal allocation of distributed generation problem.

A power system is composed of generation, transmission, and distribution systems, and it is one of the most critical and complicated engineering systems in modern society. Unlike traditional centralized power generation, distributed generation refers to the system composed of small generation equipment close to the end-users and delivers power to customers. There are many advantages of distributed generation. First, as a result of distributed generation is closed to the power supply area, it can reduce the loss of long-distance power transmission. Unlike energy produced by coal-burning power stations, the wind and energy produced by the sun are generally considered zero-pollution. Distributed generation systems are gradually playing a pivotal role in the development of modern power systems. Nowadays, enormous benefits can be gained in economic and environmental fields because of the characteristics of distributed generators [25,26]. Proper planning of the location and size of distributed generators can benefit the entire power system's operation. Therefore, distributed generation has been widely concerned recently. There are three main objectives for the optimal allocation of distributed generators that are loss minimization, voltage stability, and hybrid objective [27]. Many meta-heuristic algorithms show good efficiency in solving such problems. Suresh and Belwin [28] proposed the dragonfly algorithm to solve the problem of the greatest benefit of distributed generators in the distribution system. Abu-Mouti and El-Hawary [29] uses an artificial bee colony algorithm to optimize the distributed generators positions and sizing in a power distribution system.

However, large-scale problems may limit the performance of the algorithm. Sun and Jin (eds.) [30] proposed the surrogate-assisted cooperative swarm to solve high-dimensional optimization problems and achieves good results. For big data optimization, a hybrid multi-objective firefly algorithm is proposed [31]. In large power distribution systems, few meta-heuristic algorithms consider the memory space of potential buses. This paper improves the EO algorithm by two different parallel strategy and compact method. In order to reduce the memory space of potential buses in the power system, compact technology is considered in the improvement of the EO algorithm. The idea of compact technology established a certain population probability model to simulate the distribution of all individuals in the population with less space, so the operation of the probability model can represent approximately the original population [32–34]. As the considerable memory savings, the algorithm's performance improved by using this technique may be limited due to less memory space. Therefore, this paper attempts to improve the performance of the algorithm by using interval update method and parallel technique. The interval update method is considered a catalyst for the combination of compact technology and the EO algorithm, which can give full play to the equilibrium pool's renewal role. Simultaneously, taking into account the differences in the equilibrium pool, two different parallel communication strategies are proposed for improving the algorithm's ability. Parallel technology divides the overall population into small part groups, emphasizing the diversity of solutions in the search problem. Each group has its own individuals but they are not independent of each other. Two different parallel communication strategies have different equilibrium pools, so they with varying spaces of memory. The concrete analysis for memory spaces of the equilibrium pool is illustrated in a later section.

Recently, there are some compact algorithms proposed. Harik, Lobo and Goldberg proposed the compact genetic algorithm (cGA), a probability distribution model is introduced based on a genetic algorithm (GA) to describe the distribution of individuals. Mininno and Neri (eds.) proposed the compact differential evolution (cDE) algorithm, cDE is similar to cGA algorithms. The use of compact technology in many algorithms is roughly the same, including compact particle swarm optimization (cPSO) [35], compact bat algorithm (cBA) [36], compact firefly algorithm (CFA) [37]. Compact technology makes a useful contribution for saving memory space, but the algorithm's performance needs to be tested. Parallel technology has made a meaningful contribution to improving the performance and efficiency of the algorithm [38–41]. It can achieve faster convergence and get better solutions by effectively exchange information between groups. This paper proposed two parallel compact EO algorithms (pcEO) and utilizes them to optimize distributed generation.

According to the above, the main contributions of this paper are listed as follows:

1) Improve and implement compact EO algorithms. For promoting the maximum contribution of the equilibrium pool update, the method of update interval is proposed. For four categories of test functions, experiments are tested and analyzed to find the value of the optimal interval.
2) After receiving the optimal interval value, the improved compact EO algorithm is tested and compared with other compact standard algorithms.
3) Propose two parallel EO algorithms with different structures of equilibrium pool. Private pEO algorithm means that each group of the entire population has an equilibrium pool. These equilibrium pools can be improved and updated by parallel

communication strategy and then achieve efficient communication, but the memory space of the algorithm is large. Next, the public pcEO algorithm with a collective equilibrium pool is proposed. It achieved excellent performance with less memory space.

4) Two improved pcEO algorithms are applied in the optimal allocation of distributed generation to determine the optimal sizing of DGs, then compared with other classic algorithms in the 30 buses system.

The structure of this paper is organized as follows. Firstly, the EO algorithm is introduced in Sect. 2. The basis of mathematical and evaluation criteria for optimal allocation of distributed generation is presented. Subsequently, the process of a compact EO algorithm and two different compact parallel algorithms are described in Sect. 3. The comparison results of different algorithms are evaluated in Sect. 4 and the application of the proposed algorithm in the optimal allocation of distributed generation is presented in Sect. 4. Finally, the conclusion of this article is given in Sect. 6.

2. **Related work and basic theory.**

2.1. **Equilibrium optimizer algorithm.**

The EO algorithm uses a mass conservation model to define and describe the particle concentration changes in the solution. The particle in the solution is in equilibrium or a dynamically changing state. According to the mass of leaving, entering, and being generated in solution, the ordinary differential equation of time and mass can be established, which follows the rules of conservation of mass, is expressed as:

$$v\frac{dc}{dt} = qc_e + g - qc \tag{1}$$

In Eq.(1), c and v indicated the concentration and volume respectively in this control volume, so $v\frac{dc}{dt}$ represents the change rate of mass over time. q represents the flow into and out of the system. c_e indicates the concentration of the solution in the equilibrium state with a production rate of 0. And g is the mass production rate of the system.

According to the mass balance equation of concentration in a control volume, the update equation of the EO algorithm is described as follows:

$$\vec{c} = \vec{c_e} + (\vec{c} - \vec{c_e}) \cdot \vec{F} + \frac{\vec{g}}{\vec{\lambda} v} \cdot (1 - \vec{F}) \tag{2}$$

$\vec{\lambda}$ is a set of random vectors from 0 to 1. An accurate definition of exponential term (F) can achieve the algorithm's optimal coordination in the exploration and development capabilities. The exponential term (F) is defined as:

$$\vec{F} = msign(\vec{a} - 0.5) \cdot (e^{-\vec{\lambda} t} - 1) \tag{3}$$

$\vec{a}$ is a random vector between 0 and 1, and m is a fixed value used to control the search ability of the algorithm. The value of m is proportional to the exploration ability of the algorithm, which means that when the value of m is more considerable, the algorithm has a stronger exploration performance. $sgin$ stands for the sign function. The time (t) is defined as a function of the number of iterations ($iter$), the predefined maximum number of iterations (max_iter), and a_2:

$$t = (1 - \frac{iter}{max\_iter})^{a_2 \frac{iter}{max\_iter}} \tag{4}$$

where a_2 is a constant representing the exploitation capability of the algorithm.

The generation rate ($\vec{g}$) impacts the exploitation phase by providing the exact solution. It is described as follow:

$$\vec{g} = \vec{g}_0 \cdot \vec{F} \tag{5}$$

where $\vec{g}_0$ is the initial generation rate:

$$\vec{g}_0 = g\vec{c}p \cdot (\vec{c_e} - \vec{c} \cdot \vec{\lambda}) \tag{6}$$

$$g\vec{c}p = \begin{cases} 0.5a_1 & a_2 \geq gp \\ 0 & a_2 < gp \end{cases} \tag{7}$$

In order to stabilize the exploration and exploitation capabilities of the algorithm, gp is set to 0.5. a_1 and a_2 are random numbers from 0 to 1. $g\vec{c}p$ is a set of vectors, each value in the vector is equal to the comparison result of gp and a_2.

The four best particles (E_1, E_2, E_3, E_4) and their average particles (E_{ave}) in the search process constitute the equilibrium pool. It plays an essential role in balancing algorithm exploitation and exploration capabilities. Randomly select a particle from the equilibrium candidates, and use the single particle selected in the equilibrium pool to update each particle in the solution according to specific rules. This paper uses $fitness(c_i)$ to represent a value of vector c_i under a specific objective function. As mentioned earlier, the equilibrium optimizer algorithm is described as follows:

Algorithm 1 EO algorithm

```
1: Assign parameters: $v = 1$, $gp = 0.5$, $m = 2$
2: Initializing the concentration of particles
3: Initializing the concentration of equilibrium candidates:$E_1, E_2, E_3, E_4, E_{ave}$ and its $fitness$
4: while $iter < max\_iter$ do
5:     for $i = 1$: number of particles do
6:         Calculate fitness of $i^{th}$ particle
7:         if $fitness(c_i) < fitness(E_1)$ then $E_1 = c_i, fitness(E_1) = fitness(c_i)$
8:         else if $fitness(c_i) < fitness(E_2)$ then $E_2 = c_i, fitness(E_2) = fitness(c_i)$
9:         else if $fitness(c_i) < fitness(E_3)$ then $E_3 = c_i, fitness(E_3) = fitness(c_i)$
10:        else if $fitness(c_i) < fitness(E_4)$ then $E_4 = c_i, fitness(E_4) = fitness(c_i)$
11:        end if
12:    end for
13:    Calculate $E_{ave}$
14:    Construct the equilibrium pool: $E_{pool} = \{E_1, E_2, E_3, E_4, E_{ave}\}$
15:    for $i = 1$: number of particles do
16:        Update concentrations $c_i$ by Eq.(2)
17:    end for
18:    $iter = iter + 1$
19: end while
```

2.2. **The basis for optimal allocation of distributed generation.** The optimal allocation of distributed generation can be divided into two stages: determining the optimal location of DGs and determining the optimal sizing of DGs. The objective function of the first stage is given in Sect. 2.2.2. And the objective function of the second stage is the minimum active power loss, which is introduced in Sect. 5.

2.2.1. *Power flow calculation in distribution network.* The analysis and calculation of load flow are significant and fundamental in any power system, an essential tool to study various planning and operation power systems. Newton-Raphson method is a classic algorithm based on the iterative process, which is widely used to calculate and analyze load flow. Through power flow calculation and analysis, each node's voltage value and voltage phase angle can be obtained. For the problem of non-linear equations, iteration is a powerful solution tool. In order to solve a set of simultaneous non-linear equations with a specific number of unknowns, an iterative method is used to approximate the solution of the nonlinear equations. A set of simultaneous equations are given by utilizing the power balance equation to define the active and reactive power of the network with N nodes:

$$\begin{cases} \Delta P_i = P_i - U_i \sum_{j=1}^{N} U_j (G_{ij} cos\delta_{ij} + B_{ij} sin\delta_{ij}) = 0 \\ \Delta Q_i = Q_i - U_i \sum_{j=1}^{N} U_j (G_{ij} sin\delta_{ij} - B_{ij} cos\delta_{ij}) = 0 \end{cases} \tag{8}$$

Here, P_i and Q_i represent the injected active power and reactive power of node i. The voltage of bus i is U_i, and U_j represents the voltage of the node j connected to node i. The conductance and susceptance between two nodes are defined as G_{ij} and B_{ij}. And the δ_{ij} is the difference of voltage phase angle.

The mathematical basis of the Newton-Raphson method is the Taylor series shown in Eq.(9):

$$\begin{aligned} & f\left(x_1 + \Delta x_{1,} x_2 + \Delta x_2, ..., x_n + \Delta x_n\right) - f\left(x_1, x_2, ..., x_n\right) = \frac{\partial f}{\partial x_1} \Delta x_1 \\ & + \frac{\partial f}{\partial x_2} \Delta x_2 + ... + \frac{\partial f}{\partial x_n} \Delta x_n \end{aligned} \tag{9}$$

According to the nodal power balance equations in Eq.(8) and Taylor series in Eq.(9), a set of simultaneous non-linear equations about voltage and voltage phase angle ($\Delta|V|$ and $\Delta\delta$) can be described as:

$$\begin{bmatrix} \Delta P \\ \Delta Q \end{bmatrix} = \begin{bmatrix} J_1 & J_2 \\ J_3 & J_4 \end{bmatrix} \begin{bmatrix} \Delta\delta \\ \frac{\Delta|V|}{|V|} \end{bmatrix} \tag{10}$$

In a network with n nodes, there is a slack bus. Assuming that there are no PV nodes, J_i is a $(n-1)\times(n-1)$ matrix. In each iteration, the values of $\Delta\delta_i$ and $\Delta|V|_i$ obtained from the equation is used to change δ_i and $|V|_i$ until the resulting value meets the condition of convergence.

$$\begin{cases} \delta_i^{t+1} = \delta_i^t + \Delta\delta_i^t \\ |V|_i^{t+1} = |V|_i^t + \Delta|V|_i^t \end{cases} \tag{11}$$

After the power flow calculation in the system, accurate information of each bus is obtained, which is conducive to planning the power system and estimating various indexes.

2.2.2. *Loss sensitivity factor.* The sensitivity factor method is determining the best potential positions of nodes according to power losses of each node. This method was initially commonly used to solve the problem of optimal allocation of capacitors. The sensitivity factor selects the potential optimal position from many nodes to reduce the solution space and operational complexity of the algorithm. It is advantageous to use the proposed algorithm to determine the suitable sizing of DGs. The real power loss (P_l) is described as follow:

$$P_l = \sum_{i=1}^{n}\sum_{j=1}^{n}\left[\alpha_{ij}\left(P_iP_j + Q_iQ_j\right) + \beta_{ij}\left(Q_iP_j - P_iQ_j\right)\right] \tag{12}$$

where:

$$\alpha_{ij} = \frac{R_{ij}}{U_iU_j}\cos\delta_{ij} \tag{13}$$

$$\beta_{ij} = \frac{R_{ij}}{U_iU_j}\sin\delta_{ij} \tag{14}$$

where R_{ij} represents the resistance between two nodes. The sensitivity factor corresponding to the node i (S_i) can be obtained from real power loss and nodal real injection power:

$$S_i = \frac{\partial P_l}{\partial P_i} = 2\sum_{i=1}^{n}\left(\alpha_{ij}P_j - \beta_{ij}Q_j\right) \tag{15}$$

The nodal sensitivity factors are sorted in descending order to select a certain number of potential optimal nodes based on priority. Then the distributed generators are arranged on these selected nodes.

3. **Compact equilibrium optimizer algorithm and two different parallel strategy.** There are two main parts to this section. The first part introduces how to use compact technology to improve the equilibrium optimizer algorithm. Next, the second part presents the different equilibrium pool structures of two parallel algorithms, then the specific steps of the two parallel algorithms are described in detail.

3.1. **Compact EO.** The basis of compact technology is the probability model which represents the distribution of individuals of the entire population. Chuang and Chen [42] estimated the likely structure of the feasible solutions to guide the algorithm to gradually find the possible optimal solution by explicitly building a probabilistic model. All algorithms can be abstractly simplified into a combined search and selection process for promising solutions within the search space. In the equilibrium optimizer algorithm, the search and selection operations are performed for the concentration of N particles. Due to the massive scale of the actual problem, the problem's procedures may take up a lot of memory space. The compact method establishes a certain population probability model to simulate the distribution of all individuals in the population with less space, which means that the operations of the probabilistic model represent the entire population. Therefore, the operation of the probability model is the operation of the entire population.

Suppose the individuals in the population obey a Gaussian probability distribution function (PDF) with a mathematical expectation of μ and a variance of σ^2. The probability vector (PV) to simulate the distribution of all individuals in the population [43,44]. PV is a $2 \times n$ matrix and can be defined as:

$$PV^t = \left[\mu^t, \sigma^t\right]^t \tag{16}$$

In order to keep the area of the PDF function equal to 1, the truncated normal distribution limits the value range of the variable and is defined as:

$$PDF = \frac{\sqrt{\frac{2}{\pi}}e^{-\frac{(x-\mu)^2}{2\sigma^2}}}{\delta\left(erf\left(\frac{\mu+1}{\sqrt{2}\delta}\right) - erf\left(\frac{\mu-1}{\sqrt{2}\delta}\right)\right)}, \tag{17}$$

Algorithm 2 Compact EO with interval

```
1: Assign parameters: $v = 1$, $gp = 0.5$, $m = 2$
2: Initializing the concentration of particles
3: Initializing the concentration of equilibrium candidates:$E_1, E_2, E_3, E_4, E_{ave}$ and its $fitness$
4: Initializing the $\mu$ and $\sigma$
5: while $iter < max\_iter$ do
6:     if ($iter$ is a multiple of $val$)
7:         Get $c$ via the inverse function of Eq.(18)
8:     end if
9:     $c_{old} = c$
10:    Calculate $E_{ave}$
11:    Construct the equilibrium pool: $E_{pool} = \{E_1, E_2, E_3, E_4, E_{ave}\}$
12:    Update concentrations $c$ by Eq.(2)
13:    Calculate fitness of $c$
14:    if $fitness(c) < fitness(E_1)$ then $E_1 = c, fitness(E_1) = fitness(c)$
15:    else if $fitness(c) < fitness(E_2)$ then $E_2 = c, fitness(E_2) = fitness(c)$
16:    else if $fitness(c) < fitness(E_3)$ then $E_3 = c, fitness(E_3) = fitness(c)$
17:    else if $fitness(c) < fitness(E_4)$ then $E_4 = c, fitness(E_4) = fitness(c)$
18:    end if
19:    Compare $fitness(c)$ and $fitness(c_{old})$, get $winner$ and $loser$
20:    Use $winner$ and $loser$ update $PV$ by Eq.(19)(20)
21:    $iter = iter + 1$
22: end while
```

CDF is calculated as follows:

$$CDF = \int_{-1}^{x} PDFdx = \frac{erf\left(\frac{\mu+1}{\sqrt{2}\delta}\right) + erf\left(\frac{x-\mu}{\sqrt{2}\delta}\right)}{erf\left(\frac{\mu+1}{\sqrt{2}\delta}\right) - erf\left(\frac{\mu-1}{\sqrt{2}\delta}\right)}, \tag{18}$$

where erf represents the Gauss error function.

At the beginning of the compact method, in each dimension i, a randomly generated number r ranging from 0 to 1 as an input of the inverse function of CDF, where the inverse function of CDF is composed of parameters μ_i and δ_i in PV, then a solution is generated. Repeat this process to get a set of vectors c, and use vector c to represent the concentration of N particles in the EO algorithm. This is the beginning of the application of compact technology to the EO algorithm.

In order to facilitate the subsequent update and modification of the PV status, the initial values of $\mu[i]$ and $\sigma[i]$ are respectively defined as 0 and 10. Each individual is recorded and updated, compares the generated new individual with the original individual to determine the *winner* and *loser*, then utilizes the *winner* and *loser* to update the PV [45]. The updated rule of μ and σ values is respectively given as follows:

$$\mu^{t+1}[i] = \mu^{t}[i] + \frac{1}{N_p}\left(winner[i] - loser[i]\right) \tag{19}$$

$$\left(\sigma^{t+1}[i]\right)^2 = \left(\sigma^{t}[i]\right)^2 + \left(\mu^{t}[i]\right)^2 - \left(\mu^{t+1}[i]\right)^2 + \frac{1}{N_p}\left(winner[i]^2 - loser[i]^2\right) \tag{20}$$

In order to take advantage of the renewal role of the equilibrium pool, the idea of interval update is proposed. In the traditional compact method, it should be noted that the concentration of particles generated by PV has happened in the process of each iteration, which may weaken the search performance of particles. Thereby the method of update interval is proposed, which combines the excellent performance of the EO algorithm with the compact technology to reduce memory. And experiments were carried out for different intervals. Finally, the most suitable interval *val* for the algorithm is selected. The compact EO algorithm with interval is described in Algorithm 2.

Algorithm 3 Private pcEO algorithm

```
 1: $g$ is the number of groups, $G(i)$ represents the $i$-th group
 2: Assign parameters: $v = 1$, $gp = 0.5$, $m = 2$
 3: Initializing the concentration of particles of each group
 4: Initializing the concentration of equilibrium candidates:$E_1, E_2, E_3, E_4, E_{ave}$ and their $fitness$ of each group
 5: Initializing the $\mu$ and $\sigma$ of each group
 6: while $iter < max\_iter$ do
 7:     if ($iter$ is a multiple of $val$)
 8:         Get $c$ of each group via the inverse function of Eq.(18)
 9:     end
10:     $c_{old} = c$
11:     for $i = 1 : g$ do
12:         Construct the equilibrium pool of each group: $G(i).E_{pool} = \{G(i).E_1, G(i).E_2, G(i).E_3, G(i).E_4, G(i).E_{ave}\}$
13:         Update concentrations $G(i).c$ by Eq.(2)
14:         Calculate fitness of $G(i).c$
15:         if $fitness(G(i).c) < fitness(G(i).E_1)$ then $G(i).E_1 = G(i).c, fitness(G(i).E_1) = fitness(G(i).c)$
16:         else if $fitness(G(i).c) < fitness(G(i).E_2)$ then $G(i).E_2 = G(i).c, fitness(G(i).E_2) = fitness(G(i).c)$
17:         else if $fitness(G(i).c) < fitness(G(i).E_3)$ then $G(i).E_3 = G(i).c, fitness(G(i).E_3) = fitness(G(i).c)$
18:         else if $fitness(G(i).c) < fitness(G(i).E_4)$ then $G(i).E_4 = G(i).c, fitness(G(i).E_4) = fitness(G(i).c)$
19:         end if
20:     end for
21:     for $i = 1 : g$ do
22:         Compare $fitness(G(i).c)$ and $fitness(G(i).c_{old})$, get $winner$ and $loser$ to update $PV$ by Eq.(19)(20)
23:     end for
24:     for $i = 1 : g$ do
25:         Randomly select a group $k$ ($k \neq i$)
26:         if($fitness(G(i).E_1) > fitness(G(k).E_1)$)
27:             Get $new$ and $fitness(new)$ from $G(i).E_1$ and $G(k).E_1$ by crossover operation, update $G(i).E_1$ and $fitness(G(i).E_1)$
28:             Update $G(i).PV$ by crossover operation
29:         else Disturb and update $G(i).E_1$
30:         end if
31:     end for
32:     $iter = iter + 1$
33: end while
```

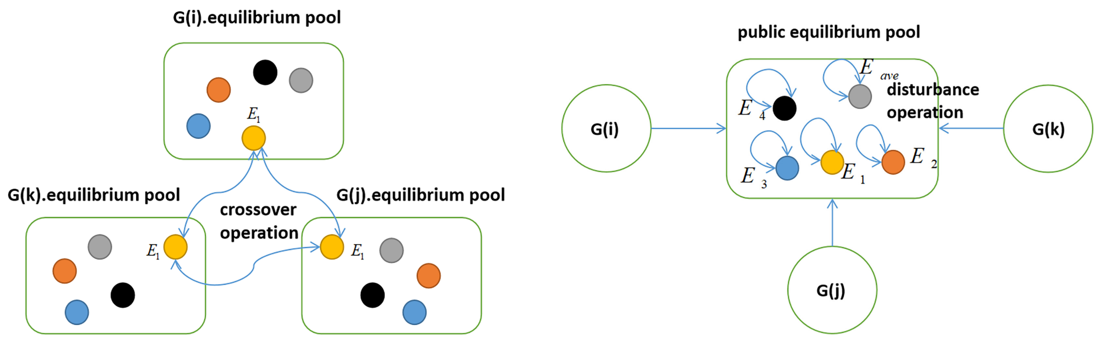
Algorithm 4 Public pcEO algorithm

```
1: $g$ is the number of groups, $G(i)$ represents the $i$-th group
2: Assign parameters: $v = 1$, $gp = 0.5$, $m = 2$
3: Initializing the concentration of particles of each group
4: Initializing the concentration of equilibrium candidates:$E_1, E_2, E_3, E_4, E_{ave}$
5: Initializing the $\mu$ and $\sigma$ of each group
6: while $iter < max\_iter$ do
7:     if ($iter$ is a multiple of $val$)
8:         Get $c$ of each group via the inverse function of Eq.(18)
9:     end if
10:    $c_{old} = c$
11:    Construct the equilibrium pool: $E_{pool} = \{E_1, E_2, E_3, E_4, E_{ave}\}$
12:    Update concentrations $G(i).c$ by Eq.(2)
13:    Calculate fitness of $G(i).c$
14:    if $fitness(G(i).c) < fitness(E_1)$ then $E_1 = G(i).c, fitness(E_1) = fitness(G(i).c)$
15:    else if $fitness(G(i).c) < fitness(E_2)$ then $E_2 = G(i).c, fitness(E_2) = fitness(G(i).c)$
16:    else if $fitness(G(i).c) < fitness(E_3)$ then $E_3 = G(i).c, fitness(E_3) = fitness(G(i).c)$
17:    else if $fitness(G(i).c) < fitness(E_4)$ then $E_4 = G(i).c, fitness(E_4) = fitness(G(i).c)$
18:    end if
19:    for $i = 1 : g$ do
20:        Compare $fitness(G(i).c)$ and $fitness(G(i).c_{old})$, get $winner$ and $loser$ to update $PV$ by Eq.(19)(20)
21:    end for
22:    if ($iter$ is a multiple of $val$)
23:        Sort $fitness$ of each group $G(i).c$
24:        Update $PV$ of the worst group by crossover operation
25:    end if
26:    Disturb $E_i$ randomly selected from equilibrium pool, then update $E_1$
27:    $iter = iter + 1$
28: end while
```

3.2. **Two kinds of parallel algorithms.** Through the previous contents, the compact EO algorithm is improved and proposed. In the case of less memory space, the performance of the compact algorithm can be achieve positive results. In order to further enhance the performance of the proposed algorithm, there are two parallel algorithms with different structures are proposed. The parallel communication method can be improve the ability and speed of algorithm to find the global optimal solution [46]. Parallel technology is used to divide the overall population into several independent groups to emphasize the diversity of solutions. The valuable information of each group is constantly exchanged by its own and other groups. Because of the various characteristics of the algorithm, parallel communication strategies are adopted, which means that parallel communication methods are flexible and diverse.

Aiming at the differences in equilibrium pools, the methods of public and private equilibrium pools are proposed in this article. A private equilibrium pool means that each group has its equilibrium pool and utilizes the pool to update the individuals in their group. The process does not interfere operations performed on the other groups in any way. On the contrary, the equilibrium pool is public and collective for each group in the public pcEO algorithm, which means that each group's particles are updated by one mutual equilibrium pool. This paper mainly uses crossover and disturbance operator as the communication strategy between different groups. As a result of the different structures of the equilibrium pool, the cross and disturbance operators for two parallel communication strategies are also different. The specific process is shown in the following algorithm 3,4.

There are some differences between the two parallel communication operations. In the private pcEO algorithm, the crossover operation is executed for E_1 which belongs to two groups. Compared to the private pcEO algorithm, the crossover operation of two different equilibrium pools cannot be carried out due to there is no second equilibrium pool in the public pcEO algorithm. The public pcEO algorithm only performs random disturbance operations on the particles in the collective equilibrium pool. These differences can be seen in Fig. 1. However, the two parallel algorithms have the same perturbation operation on PV. Meanwhile, memory space should not be ignored. Due to the different structures of the equilibrium pool, the memory space occupied by the equilibrium pool remains different. This article provides a detailed analysis of memory space in the following experiments.



(a) private parallel strategy (b) public parallel strategy

Figure 1. two parallel strategies

4. Experimental analysis.

4.1. Experimental benchmark functions and algorithm parameters information. In this paper, CEC 2014 is used as the standard function for testing and evaluating the algorithm's performance, and CEC 2014 consists of four categories of functions, which include unimodal functions, multimodal functions, hybrid functions, and composition functions. The unimodal functions (F1-F3) have a single optimum solution, which shows the performance of algorithm in finding a single optimal value. The practical problems often have more than one optimal solution. In the field of optimization, the extremum problem of the multimodal function is worth study to research in problematic issues. The multimodal functions (F4-F16) require the algorithm to have the ability of a global search ability to avoid falling into the local optimum. The hybrid functions (F17-F22) and composition functions (F23-F30) challenge the ability of algorithms to solve complex optimization problems [47].

D is defined as the dimension of the test function and the dimensions of the algorithms tested in this section are all 10. For convenience, all test algorithms maintain the same parameter settings as above. The settings of these parameters are shown in Table 1. Similarly, to provide a fair comparison, it is noted the total number of function calls of different algorithms on each benchmark function remains the same. In Table 1, P_n represented the population size, and there may be differences in the population size of different algorithms.

First of all, the influence of the interval on the EO algorithm is analyzed and evaluated through the experimental results of four categories of functions in this paper. Then the optimal interval is selected for the following experiments and improvement. Next, the compact EO algorithm is compared with other improved compact algorithms including cCS, cPSO, cBA, and cDE algorithms in this article. Subsequently, there are two improved parallel cEO algorithms are compared with the proposed cEO and the initial EO algorithm. Finally, the proposed private pcEO and public pcEO algorithms are compared with other common and classic algorithms, including PSO, GA, GWO, BOA, and gravitational search algorithm (GSA) [48].

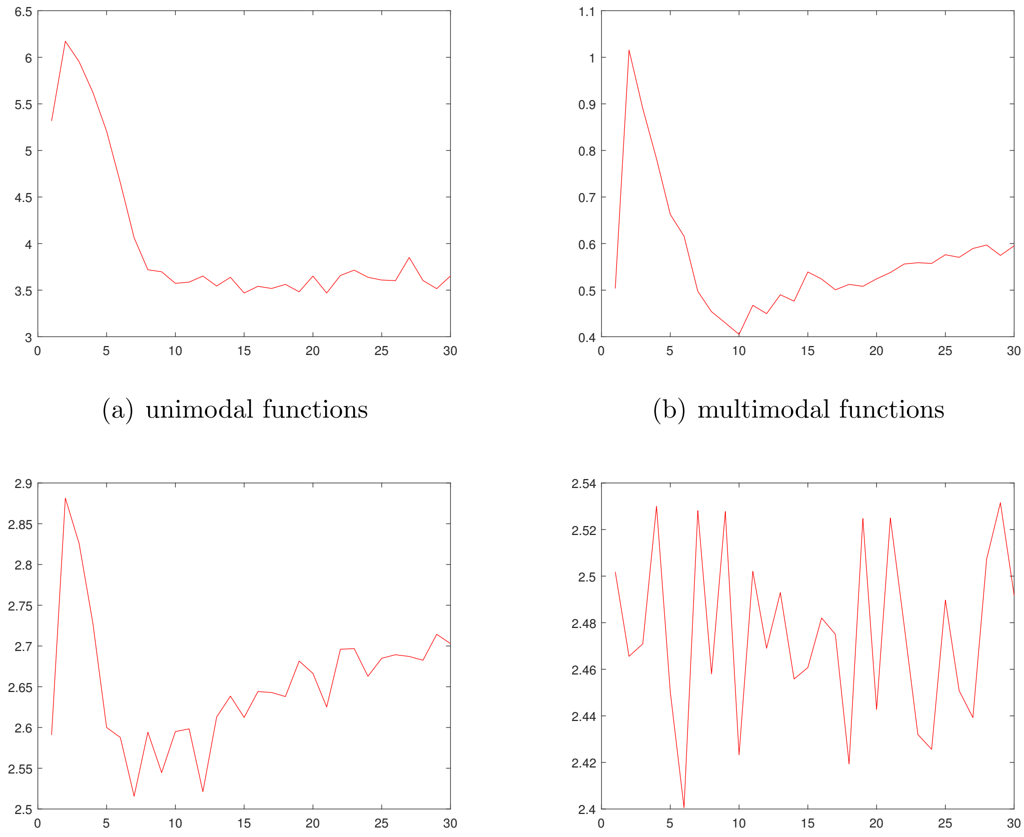
Table 1. The parameter settings of related algorithms

Name	Parameter
cCS	Virtual $P_n = 200,\ P_a = 0.25$
cPSO	Virtual $P_n = 200,\ phi_1 = -0.2,\ phi_2 = -0.07,\ phi_3 = 3.74,$ $\gamma_1 = 1,\ \gamma_2 = 1$
cBA	Virtual $P_n = 200$, $loudness = 0.5$, $pulserate = 0.5$, $f_{min} = 0,\ f_{max} = 2$
cDE	Virtual $P_n = 200,\ F = 0.5,\ Cr = 0.5$
PSO	$P_n = 30,\ c_1 = 2.0,\ c_2 = 2.0,\ w = 0.9$
GA	$P_n = 30$, $mutation\ rate = 0.5$, $crossover\ rate = 0.4$
BOA	$P_n = 30$, $probabibility\ switch = 0.8$, $power\ exponent = 0.1$, $sensory\ modality = 0.01$
GSA	$P_n = 30$, $alfa = 20$, $Rpower = 1$, $G_0 = 100$
GWO	$P_n = 30$, a decreases linearly from 2 to 0
EO	$v = 1,\ gp = 0.5,\ m = 2,\ P_n = 30$
cEO	Virtual $P_n = 200,\ v = 1,\ gp = 0.5,\ m = 2$
Private pcEO	Virtual $P_n = 200,\ v = 1,\ gp = 0.5,\ m = 2,\ groups = 4$
Public pcEO	Virtual $P_n = 200,\ v = 1,\ gp = 0.5,\ m = 2,\ groups = 4$

4.2. **Comparison of different interval on compact EO algorithm.** According to the four categories of functions on CEC 2014, the cEO algorithm with different intervals is tested on the four categories of functions that the purpose is to analyze the influence of interval on different categories of functions to find the most suitable interval. Although the interval is different, the total number of function calls for different algorithms on experimental benchmark functions is the same. In order to minimize the error, the number of evaluations for all functions is set to 50,000. The algorithm is tested 20 times on each benchmark function to get the average. This section uses f_i to highlight changes in results, f_i is the evaluation of the final convergence result of the algorithm and is defined as follows:

$$f_i = \log_{10}(F_i - F_i^*). \tag{21}$$

where F_i is the convergence results of the algorithm in the test function i, and F_i^* is the value of the optimal solution is defined in CEC 2014 functions. The smaller the value of f_i, the better the algorithm will perform. In Fig. 2, the y-axis is defined as the average of the f_i of all functions of the same category and the x-axis represents the value of the interval. The test results of the interval on four categories functions are shown in Fig. 2.



(a) unimodal functions (b) multimodal functions

(c) hybrid functions (d) composition functions

Figure 2. The results of the interval on four categories of functions

The chart of the simple multimodal and hybrid functions indicates that the best results would be obtained when the interval is close to 10. As the interval becomes bigger, the performance of the algorithm becomes worse in the multimodal functions. The algorithm performs well and steadily in the unimodal functions until the interval reaches a specific value in the first chart. Unlike the above three categories of functions above, the algorithm shows instability in the composition functions. It is illustrated that the optimal value of the interval may be different in the four categories of functions. In order to play the role of compact technology, the value of interval is as small as possible. When the value of interval is 7, good results are achieved on the four categories of functions, so the algorithm selected the 7 as the value of interval for subsequent comparison and improvement.

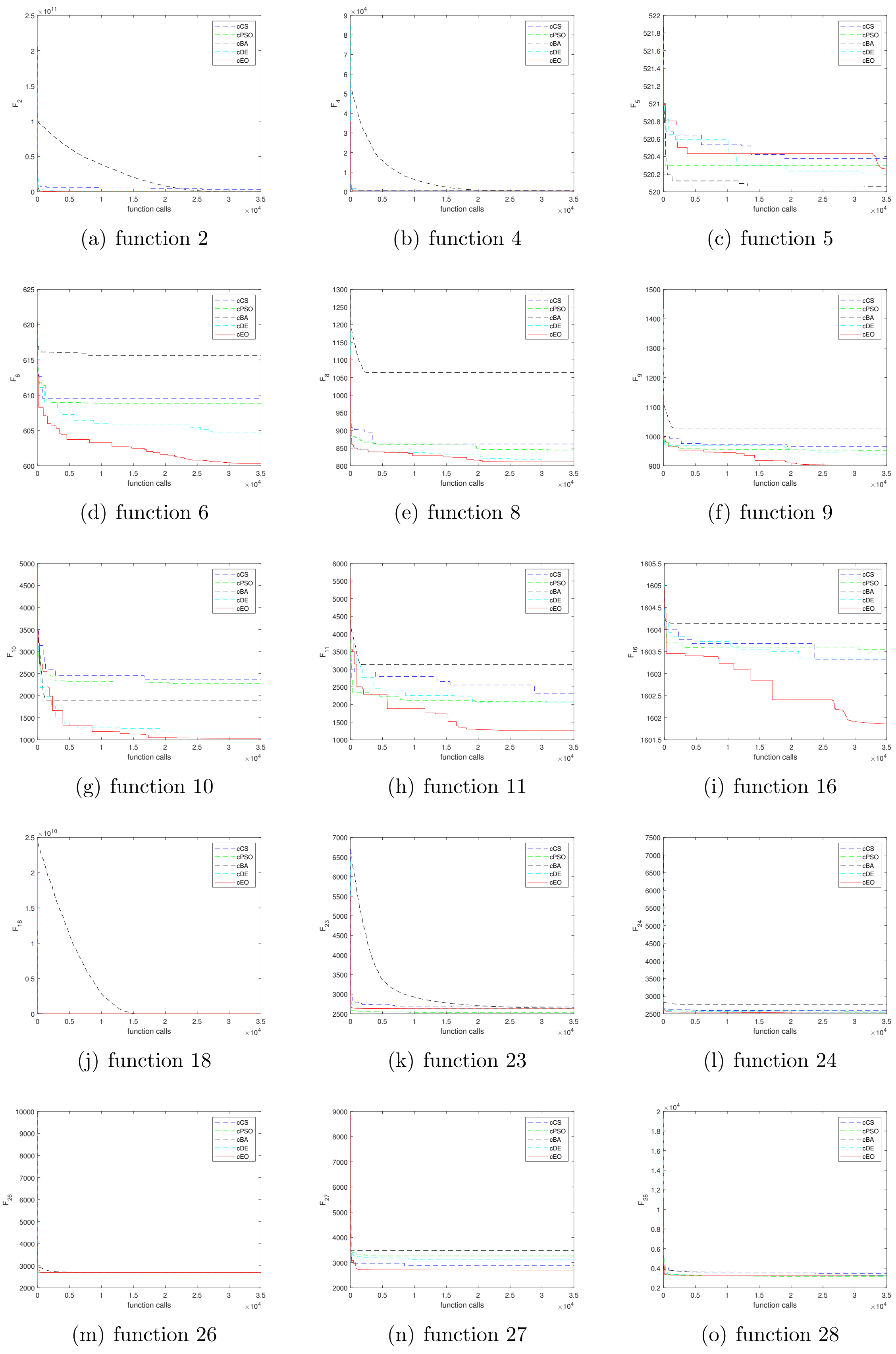
4.3. **cEO algorithm compared with other compact algorithms.** The main work of this section is to compare the cEO algorithm with other compact algorithms. These algorithms include cCS, cPSO, cBA and cDE algorithms. All algorithms were performed 20 times on each benchmark function and ensure that all evaluation times of algorithm are 35000. At the same time, means, variance, and the optimal value of functions are recorded. Unlike the experiment in the previous section, the F_i is used to evaluate the results of the algorithm in the following experiments. The various information about the algorithm is listed in Table 1.

In order to make the experimental results more obvious and accurate, this paper uses Wilcoxon's sign rank test to test each performance of algorithm at a significant level $\alpha = 0.05$. The symbol $(<)$ indicates that the algorithm has poor performance compared with the compact EO algorithm on the tested benchmark function. The meaning of symbol (>) is opposite to symbol (<) which means that the performance of this algorithm is better. And the symbol (=) illustrates that the final performance of two algorithms is the same. The comparison results are illustrated in Table 2.

Table 2. Comparison results of the cEO with cCS, cPSO, cBA, and cDE under Wilcoxon's signed rank test at the significant level $\alpha = 0.05$

Function	cCS	cPSO	cBA	cDE	cEO
F_1	2.2165e+07 <	4.4544e+06 <	1.2827e+06 <	1.6240e+06 <	2.4728e+05
F_2	2.3738e+09 <	6.1044e+06 <	1.5812e+04 >	6.6887e+07 <	2.6152e+04
F_3	1.8622e+04 <	6.4851e+03 <	7.6821e+04 <	6.2305e+03 <	9.6989e+02
F_4	6.4214e+02 <	4.4164e+02 <	4.0515e+02 =	4.4657e+02 <	4.2370e+02
F_5	5.2034e+02 <	5.2047e+02 <	5.2006e+02 <	5.2017e+02 <	5.2002e+02
F_6	6.0878e+02 <	6.0634e+02 <	6.1573e+02 <	6.0509e+02 <	6.0071e+02
F_7	7.4529e+02 <	7.0122e+02 <	7.2539e+02 <	7.0171e+02 <	7.0013e+02
F_8	8.6085e+02 <	8.4262e+02 <	9.7228e+02 <	8.0895e+02 >	8.1060e+02
F_9	9.5826e+02 <	9.4744e+02 <	1.0909e+03 <	9.2567e+02 <	9.1155e+02
F_{10}	2.2902e+03 <	1.9814e+03 <	2.1076e+03 <	1.1085e+03 >	1.1729e+03
F_{11}	2.4826e+03 <	2.1241e+03 <	2.7229e+03 <	1.8709e+03 <	1.4255e+03
F_{12}	1.2012e+03 <	1.2010e+03 <	1.2024e+03 <	1.2006e+03 <	1.2002e+03
F_{13}	1.3018e+03 <	1.3006e+03 <	1.3009e+03 <	1.3006e+03 <	1.3002e+03
F_{14}	1.4091e+03 <	1.4007e+03 <	1.4004e+03 <	1.4006e+03 <	1.4002e+03
F_{15}	2.0729e+03 <	1.5041e+03 <	1.5761e+03 <	1.5056e+03 <	1.5017e+03
F_{16}	1.6036e+03 <	1.6031e+03 <	1.6047e+03 <	1.6031e+03 <	1.6023e+03
F_{17}	6.0677e+04 <	1.0603e+04 <	1.0442e+05 <	1.4619e+05 <	4.3188e+03
F_{18}	1.0681e+05 <	1.0949e+04 >	1.6366e+04 <	1.7172e+04 <	1.2080e+04
F_{19}	1.9085e+03 <	1.9048e+03 <	2.0493e+03 <	1.9023e+03 <	1.9022e+03
F_{20}	6.3931e+03 <	3.2777e+03 <	3.4913e+04 <	7.5439e+03 <	2.9434e+03
F_{21}	8.8056e+03 <	7.3161e+03 <	7.1409e+03 <	1.2765e+04 <	5.8578e+03
F_{22}	2.2889e+03 <	2.3672e+03 <	2.7809e+03 <	2.2224e+03 =	2.2460e+03
F_{23}	2.6668e+03 <	2.5182e+03 =	2.6296e+03 <	2.6317e+03 <	2.5259e+03
F_{24}	2.5825e+03 <	2.5606e+03 <	2.7273e+03 <	2.5402e+03 <	2.5239e+03
F_{25}	2.6873e+03 <	2.6834e+03 >	2.7048e+03 <	2.6861e+03 =	2.6866e+03
F_{26}	2.7015e+03 <	2.7006e+03 <	2.8441e+03 <	2.7005e+03 <	2.7001e+03
F_{27}	2.8855e+03 =	3.1702e+03 <	3.5401e+03 <	2.9637e+03 =	2.9784e+03
F_{28}	3.4899e+03 <	3.2195e+03 =	4.0794e+03 <	3.2814e+03 <	3.2426e+03
F_{29}	5.9112e+04 =	4.2958e+06 <	4.3426e+03 =	3.1152e+03 =	1.9202e+05
F_{30}	6.0647e+03 <	6.0166e+03 <	4.7499e+03 <	3.2882e+03 =	3.8377e+03
<=>	28/0/2	26/2/2	27/1/2	23/2/5	-

According to the experimental results in Table 2, the performance of the cEO algorithm is better than other compact algorithms. Compared with cCS, the cEO algorithm almost wins all benchmark functions. Besides, compared with the cPSO, only two results of benchmark functions are inferior to the cPSO. The results of cBA comparisons are similar to cPSO, and the proposed cEO algorithm wins on 27 benchmark functions. In the comparisons with cDE algorithm, the EO algorithm wins on 23 tested benchmark functions. The experimental results effectively prove that the performance of the cEO algorithm is better than other compact algorithms.



(a) function 2 (b) function 4 (c) function 5

(d) function 6 (e) function 8 (f) function 9

(g) function 10 (h) function 11 (i) function 16

(j) function 18 (k) function 23 (l) function 24

(m) function 26 (n) function 27 (o) function 28

FIGURE 3. Comparison of convergence curves between cEO and other compact algorithms

Table 3. Comparison of memory space, dimension and population size

Name	Population Size	Dimension	Memory Size
EO	30	10	$30 \times D + 5 \times D$
cEO	1	10	$1 \times D + 2 \times D + 5 \times D$
private pcEO	4	10	$4 \times D + 8 \times D + 20 \times D$
public pcEO	4	10	$4 \times D + 8 \times D + 5 \times D$

Table 4. Comparison results of the two proposed parallel algorithms with cEO and EO algorithm under Wilcoxon's signed rank test at the significant level $\alpha = 0.05$

Functions	EO		cEO		private pcEO	public pcEO
F_1	7.9558e+04 $>$	$>$	2.5873e+05 $=$	$=$	3.6095e+05 $<$	3.3412e+05 $=$
F_2	2.6555e+03 $=$	$=$	7.3492e+03 $<$	$<$	3.8920e+03 $<$	3.7685e+03 $=$
F_3	8.6962e+02 $>$	$=$	1.0066e+03 $=$	$=$	1.0881e+03 $<$	1.0103e+03 $=$
F_4	4.2815e+02 $<$	$<$	4.3287e+02 $<$	$<$	4.0689e+02 $=$	4.0732e+02 $<$
F_5	5.2040e+02 $<$	$<$	5.1869e+02 $>$	$=$	5.2007e+02 $<$	5.2002e+02 $>$
F_6	6.0096e+02 $=$	$=$	6.0164e+02 $<$	$<$	6.0114e+02 $=$	6.0117e+02 $<$
F_7	7.0004e+02 $>$	$>$	7.0009e+02 $>$	$=$	7.0019e+02 $<$	7.0010e+02 $>$
F_8	8.0523e+02 $>$	$>$	8.1145e+02 $=$	$=$	8.1473e+02 $<$	8.1209e+02 $=$
F_9	9.1218e+02 $>$	$=$	9.1423e+02 $=$	$<$	9.1871e+02 $<$	9.1271e+02 $>$
F_{10}	1.1464e+03 $>$	$>$	1.1773e+03 $=$	$=$	1.3001e+03 $<$	1.2419e+03 $=$
F_{11}	1.5121e+03 $=$	$<$	1.4314e+03 $>$	$<$	1.5987e+03 $<$	1.4203e+03 $>$
F_{12}	1.2010e+03 $<$	$<$	1.2002e+03 $=$	$=$	1.2003e+03 $<$	1.2002e+03 $=$
F_{13}	1.3002e+03 $<$	$<$	1.3001e+03 $=$	$=$	1.3001e+03 $=$	1.3001e+03 $<$
F_{14}	1.4003e+03 $<$	$<$	1.4002e+03 $<$	$<$	1.4001e+03 $<$	1.4001e+03 $=$
F_{15}	1.5016e+03 $<$	$<$	1.5011e+03 $<$	$<$	1.5011e+03 $=$	1.5011e+03 $<$
F_{16}	1.6025e+03 $<$	$=$	1.6022e+03 $=$	$=$	1.6023e+03 $=$	1.6025e+03 $<$
F_{17}	5.0406e+03 $<$	$<$	4.4342e+03 $<$	$<$	3.7736e+03 $=$	3.9639e+03 $<$
F_{18}	9.6672e+03 $<$	$<$	1.0277e+04 $<$	$<$	9.2188e+03 $=$	9.4171e+03 $<$
F_{19}	1.9019e+03 $=$	$=$	1.9025e+03 $<$	$<$	1.9019e+03 $=$	1.9021e+03 $<$
F_{20}	2.4490e+03 $=$	$=$	3.9137e+03 $<$	$<$	2.8775e+03 $<$	2.8766e+03 $=$
F_{21}	2.7188e+03 $>$	$>$	6.7739e+03 $<$	$<$	3.7411e+03 $>$	5.8156e+03 $<$
F_{22}	2.2447e+03 $<$	$=$	2.2573e+03 $<$	$<$	2.2297e+03 $=$	2.2447e+03 $<$
F_{23}	2.6295e+03 $<$	$<$	2.5518e+03 $<$	$>$	2.5000e+03 $>$	2.5949e+03 $<$
F_{24}	2.5256e+03 $<$	$=$	2.5407e+03 $<$	$<$	2.5255e+03 $=$	2.5317e+03 $<$
F_{25}	2.6874e+03 $=$	$<$	2.6893e+03 $=$	$<$	2.6950e+03 $<$	2.6812e+03 $>$
F_{26}	2.7002e+03 $<$	$<$	2.7001e+03 $=$	$<$	2.7001e+03 $<$	2.7001e+03 $>$
F_{27}	2.9907e+03 $<$	$=$	2.9711e+03 $<$	$=$	2.9233e+03 $=$	3.0292e+03 $<$
F_{28}	3.2373e+03 $<$	$<$	3.2137e+03 $<$	$<$	3.0443e+03 $=$	3.1736e+03 $<$
F_{29}	3.7358e+05 $<$	$<$	2.3344e+05 $<$	$<$	3.3492e+03 $>$	3.5995e+03 $<$
F_{30}	3.7336e+03 $=$	$=$	4.0323e+03 $<$	$<$	3.8392e+03 $<$	3.7469e+03 $=$
<=>	16/7/7		17/10/3		-	15/9/6
<=>	14/11/5		19/10/1		15/12/3	-

Table 5. Comparison results of the two proposed parallel algorithms with PSO, GA, BOA, GSA, and GWO algorithm under Wilcoxon's signed rank test at the significant level $\alpha = 0.05$

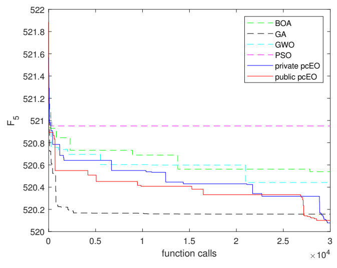
Function	PSO	GA	BOA	GSA	GWO	private pcEO	public pcEO
F_1	1.6820e+08 <<	9.2891e+07 <<	1.7198e+08 <<	4.1604e+08 <<	6.8036e+06 <<	3.6095e+05	3.3412e+05
F_2	1.6404e+10 <<	4.0420e+09 <<	4.2552e+09 <<	9.3971e+09 <<	1.0512e+08 <<	3.8920e+03	3.7685e+03
F_3	2.4940e+05 <<	3.0503e+05 <<	1.4363e+04 <<	6.0976e+04 <<	7.2798e+03 <<	1.0881e+03	1.0103e+03
F_4	3.6413e+03 <<	1.3053e+03 <<	2.3635e+03 <<	4.0797e+03 <<	4.4051e+02 <<	4.0689e+02	4.0732e+02
F_5	5.2084e+02 <<	5.2008e+02 <<	5.2049e+02 <<	5.2083e+02 <<	5.2042e+02 <<	5.2007e+02	5.2002e+02
F_6	6.1309e+02 <<	6.0968e+02 <<	6.0857e+02 <<	6.1249e+02 <<	6.0267e+02 <<	6.0114e+02	6.0117e+02
F_7	8.8639e+02 <<	7.8092e+02 <<	8.8953e+02 <<	9.2426e+02 <<	7.0131e+02 <<	7.0019e+02	7.0010e+02
F_8	9.3325e+02 <<	8.7100e+02 <<	8.6794e+02 <<	8.9647e+02 <<	8.1346e+02 =<	8.1473e+02	8.1209e+02
F_9	1.0263e+03 <<	9.6606e+02 <<	9.7105e+02 <<	9.9676e+02 <<	9.1668e+02 =<	9.1871e+02	9.1271e+02
F_{10}	2.5541e+03 <<	1.8815e+03 <<	2.7434e+03 <<	3.2119e+03 <<	1.3976e+03 <<	1.3001e+03	1.2419e+03
F_{11}	3.4158e+03 <<	2.9082e+03 <<	2.7421e+03 <<	3.3676e+03 <<	1.5446e+03 =<	1.5987e+03	1.4203e+03
F_{12}	1.2028e+03 <<	1.2013e+03 <<	1.2017e+03 <<	1.2030e+03 <<	1.2009e+03 <<	1.2003e+03	1.2002e+03
F_{13}	1.3049e+03 <<	1.3026e+03 <<	1.3045e+03 <<	1.3050e+03 <<	1.3002e+03 <<	1.3001e+03	1.3001e+03
F_{14}	1.4390e+03 <<	1.4148e+03 <<	1.4355e+03 <<	1.4460e+03 <<	1.4003e+03 <<	1.4001e+03	1.4001e+03
F_{15}	2.6527e+05 <<	7.8113e+03 <<	4.3794e+03 <<	3.9042e+04 <<	1.5018e+03 <<	1.5011e+03	1.5011e+03
F_{16}	1.6044e+03 <<	1.6039e+03 <<	1.6037e+03 <<	1.6042e+03 <<	1.6026e+03 <<	1.6023e+03	1.6025e+03
F_{17}	6.0062e+06 <<	3.9675e+06 <<	3.7368e+05 <<	4.7720e+06 <<	7.1402e+04 <<	3.7736e+03	3.9639e+03
F_{18}	4.4743e+07 <<	2.4335e+07 <<	5.7841e+05 <<	9.7701e+07 <<	9.5649e+03 <<	9.2188e+03	9.4171e+03
F_{19}	1.9443e+03 <<	1.9341e+03 <<	1.9472e+03 <<	1.9801e+03 <<	1.9031e+03 <<	1.9019e+03	1.9021e+03
F_{20}	4.1986e+05 <<	2.4321e+07 <<	9.7790e+03 <<	2.8477e+06 <<	7.0792e+03 <<	2.8775e+03	2.8766e+03
F_{21}	1.9704e+06 <<	2.1705e+06 <<	1.9491e+05 <<	1.9492e+06 <<	9.4017e+03 <<	3.7411e+03	5.8156e+03
F_{22}	2.6421e+03 <<	2.6135e+03 <<	2.4212e+03 <<	2.6940e+03 <<	2.2894e+03 <<	2.2297e+03	2.2447e+03
F_{23}	2.8748e+03 <<	2.5518e+03 <=	2.5000e+03 >>	2.9270e+03 <<	2.6339e+03 <<	2.5000e+03	2.5949e+03
F_{24}	2.6499e+03 <<	2.5888e+03 <<	2.5925e+03 <<	2.6290e+03 <<	2.5312e+03 <=	2.5255e+03	2.5317e+03
F_{25}	2.7151e+03 <<	2.7000e+03 <<	2.6979e+03 <<	2.7095e+03 <<	2.7000e+03 <<	2.6950e+03	2.6812e+03
F_{26}	2.7084e+03 <<	2.7091e+03 <<	2.7097e+03 <<	2.7177e+03 <<	2.7002e+03 <<	2.7001e+03	2.7001e+03
F_{27}	3.4287e+03 <<	3.2184e+03 <<	2.9434e+03 <=	3.5602e+03 <<	3.0425e+03 <<	2.9233e+03	3.0292e+03
F_{28}	4.5731e+03 <<	3.6885e+03 <<	3.4578e+03 <<	4.7881e+03 <<	3.2520e+03 <<	3.0443e+03	3.1736e+03
F_{29}	5.7196e+06 <<	5.9818e+06 <<	1.7537e+04 <<	7.5842e+07 <<	4.0480e+05 <<	3.3492e+03	3.5995e+03
F_{30}	7.0434e+04 <<	5.4395e+04 <<	1.3839e+04 <<	1.6629e+05 <<	4.2285e+03 <<	3.8392e+03	3.7469e+03
<=>	30/0/0	30/0/0	29/0/1	30/0/0	27/3/0	-	
<=>	30/0/0	29/1/0	28/1/1	30/0/0	29/1/0		-

In order to further show the experimental results of the performance of the algorithm, the convergence curve is used to visually illustrate the degree of convergence of the algorithm in Fig. 3. For four categories of functions, this paper selects some representative functions to analyze and display. It can be seen that many algorithms quickly find the optimal value in unimodal functions F2, F3. In the multimodal functions F6, F9, F10, F11 and F16, the convergence speed of the proposed algorithm cEO is obviously faster than other compact algorithms. In addition, the cEO also performs well in the hybrid functions and composition functions.

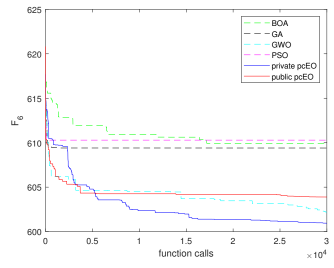
4.4. **Comparison of two parallel cEO algorithms.** In order to verify the improvement effect of parallel technology, this paper compares the two parallel compact EO algorithms with the initial EO algorithm and the cEO algorithm.

Before the comparison, we first analyze the memory space of four kinds of algorithms. As mentioned in Sect. 3.2, the structure of the public equilibrium pool is different from that of the private equilibrium pool. It can be predicted that the memory space of the private pcEO algorithm is larger than that of the public pcEO algorithm. The information of population size and differences in memory space about proposed algorithms are shown in Table 3.

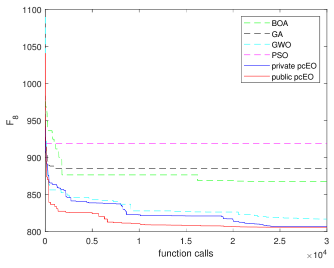
In the EO algorithm, the population size is set to 30, plus the five particles of the equilibrium pool, so the memory space size is $35 \times D$. In the two parallel compact EO algorithms, each group of PV represents the distribution of population and must be recorded, including μ and σ vector. As a result of each group has its independent



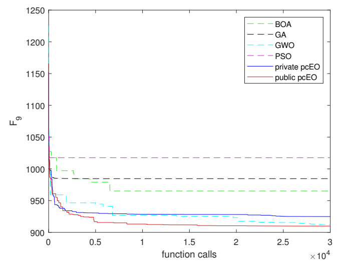
(a) function 5



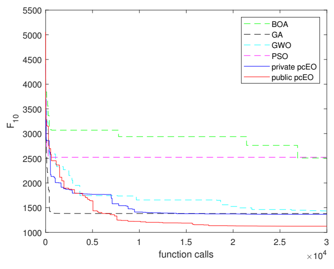
(b) function 6



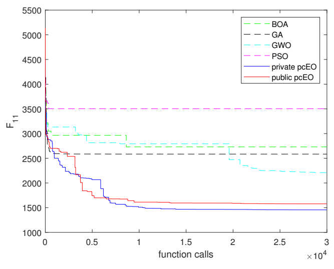
(c) function 8



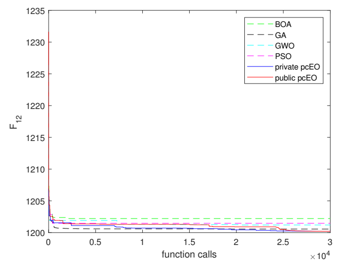
(d) function 9



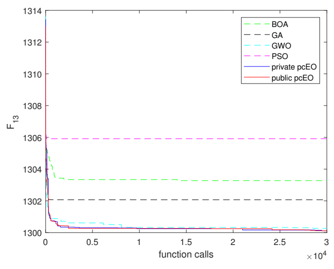
(e) function 10



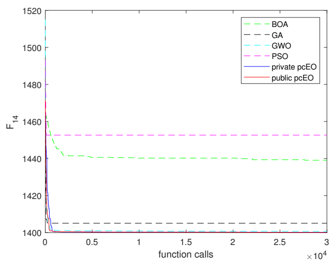
(f) function 11



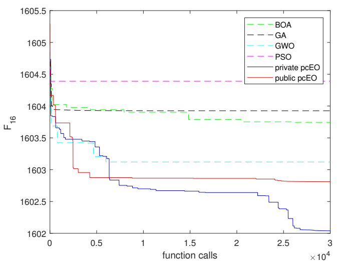
(g) function 12



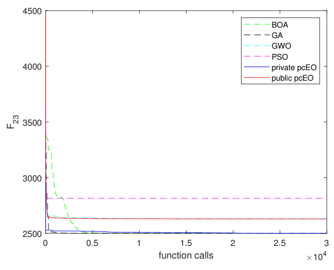
(h) function 13



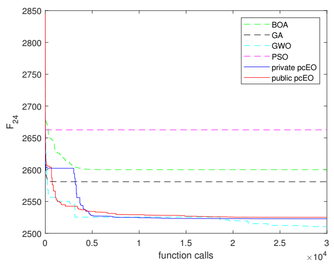
(i) function 14



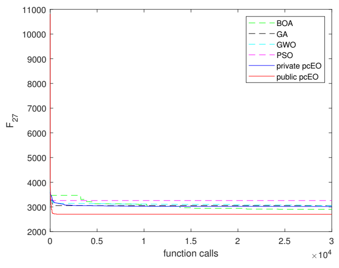
(j) function 16



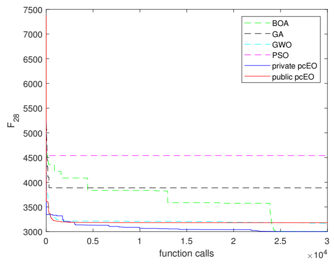
(k) function 23



(l) function 24



(m) function 27



(n) function 28

FIGURE 4. Comparison of convergence curves between two proposed pEO algorithms and other standard algorithms

equilibrium pool in the process of private pcEO algorithm, the memory space is more $(P_n - 1) \times 5 \times D$ than the public pcEO algorithm, where P_n is the population size.

In the experiments of Sect. 4.4 and 4.5, each algorithm was run 20 times on the benchmark function and evaluation times was 30000. Through the average result of the experiment in Table 4, compared with other improved EO algorithms, the public pcEO algorithm can obtain more competitive results.

4.5. **Comparison with other common and classic algorithms.** This section's main objective is to compare two parallel compact algorithms with other general algorithms, including the PSO, GA, GSA, GWO, and BOA algorithm. The detailed definition information of these algorithms can be found in Table 1.

The detailed results of the comparison are shown in Table 5. The penultimate row of the table represents the experimental results of the private pcEO algorithm compared with other algorithms. Compared with GSA, PSO, and GA algorithm, the private pcEO achieves better results on all benchmark functions as well as compared with the homologous algorithm GWO, the proposed private pcEO performs better on 27 tested benchmark functions. The comparison results in the last row are shown that the performance of public pcEO is the same as that of private pcEO.

The convergence curve is obtained in Fig. 4, and then some representative functions are selected for analysis from the experimental results. For multimodal and hybrid functions, two parallel cEO algorithm shows great advantages compared with other algorithms. The parallel cEO algorithms have a fast convergence speed and avoid falling into local optimum, especially in the F5, F10, F11, F16, F24 functions. The public pcEO uses less memory space however it also obtained better results than the private pcEO algorithm in the F1, F6, F9, F10, F12, F24 functions. The results further confirm the ability of convergence of the public pcEO algorithm.

5. **Optimal placement of distributed generation.** The concept of Newton-Raphson power flow calculation and loss sensitivity factor is introduced in Sect. 2.2. This section aims to use proposed parallel cEO algorithms to plan the optimal placement of distributed generation.

Algorithm 5 The steps of using the method of loss sensitivity factors to find the optimal location.

```
1: After reading line and load data of system, using Newton-Raphson power flow
   method to calculate voltage angle and voltage magnitude by Eq.(8)(10)(11).
2: Calculate load flow and loss sensitivity factors by Eq.(12)(13)(14)(15).
3: Store the nodal sensitivity factors in the array in descending order.
4: Select the buses as the best suitable locations.
```

Accurately planning of the position and sizing of distributed generators is of great significance to power system optimization. The arrangement of suitable locations for distributed generator units can effectively decrease the system losses and costs. Therefore, it is essential to determine a practical possible scheme and process. Concurrently, the planning of location and sizing of DGs should be assessed and analyzed through a power flow program for distribution networks. First of all, an accurate and reliable mathematical model is necessary to evaluate various indicators in the power system.

The analysis and evaluation of the optimal locations for DGs in the power system are based on node power loss. According to the previously described loss sensitivity factors, the Newton-Raphson method must first be used to calculate the active and reactive power losses for all buses. The locations corresponding to the buses that have the highest loss are selected as the potential best locations of DGs placement. Then the sensitivity factors of the node are sorted in descending order to choose a certain number of potential optimal nodes based on priority by Eq.(15). The reason is that buses with high losses can reduce more loss than low. This approach aims to reduce the complexity of searching by identifying potential positions, which helps the subsequent algorithm determine the most suitable distributed generator size. The specific steps are shown in Algorithm 5.

TABLE 6. Initial Bus data of the system

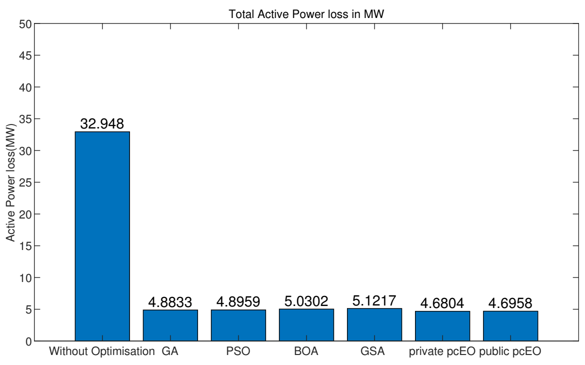
Bus	$Type$	Vsp	P_{Gi}	Q_{Gi}	P_{Li}	Q_{Li}	Q_{min}	Q_{max}
1	Slack Bus	1.04	0	0	0	0	0	0
2	PV Bus	1.023	50	40.0	27.7	52.7	-40	50
3	PQ Bus	1.09	0	0	2.2	1.6	0	0
4	PV Bus	1.06	0	0	7.6	1.6	0	0
5	PV Bus	1.01	0	37.0	94.2	19.0	-40	40
6	PQ Bus	1.0	0	0	0.0	0.0	0	0
7	PV Bus	1.0	0	0	22.8	10.9	0	0
8	PV Bus	1.06	0	37.3	30.0	30.0	-10	40
9	PQ Bus	1.05	0	0	0.0	0.0	0	0
10	PQ Bus	1.0	0	19.0	5.8	2.0	0	0
11	PV Bus	1.082	0	16.2	0.0	0.0	-6	24
12	PQ Bus	1.09	0	0	11.2	7.5	0	0
13	PV Bus	1.071	0	10.6	0.0	0.0	-6	24
14	PQ Bus	1.0	0	0	6.2	1.6	0	0
15	PQ Bus	1.053	0	0	8.2	2.5	0	0
16	PQ Bus	1.0	0	3.2	3.5	1.8	0	0
17	PV Bus	1.02	0	0	9.0	5.8	0	0
18	PQ Bus	1.0	0	2.2	3.2	0.9	0	0
19	PQ Bus	1.0	0	0	9.5	3.4	0	0
20	PQ Bus	1.01	0	0	2.2	0.7	0	0
21	PQ Bus	1.0	0	0	17.5	11.2	0	0
22	PQ Bus	1.05	0	2.2	0.0	0.0	0	0
23	PQ Bus	1.02	0	0	3.2	1.6	0	0
24	PQ Bus	1.03	0	4.3	8.7	6.7	0	0
25	PQ Bus	1.0	0	0	0.0	0.0	0	0
26	PQ Bus	1.0	0	0	3.5	2.3	0	0
27	PQ Bus	1.09	0	0	0.0	0.0	0	0
28	PQ Bus	1.08	0	0	0.0	0.0	0	0
29	PQ Bus	1.0	0	0	2.4	0.9	0	0
30	PQ Bus	1.04	0	0	10.6	1.9	0	0

Next, according to the buses with suitable locations, the proposed algorithm is used for finding optimal sizing of the DGs. There are several common and classic algorithms that have been used for optimal sizing of distributed generation. Two parallel cEO algorithms are compared with GA, PSO, GSA, and BOA algorithms in this section to determine which algorithm is best for application. The population size of PSO, GA, GSA, and BOA algorithms is all set to 30. Meanwhile, the number of groups for both parallel cEO algorithms is set to 4. Thereby, the memory space of other algorithms is larger than two parallel compact EO algorithms, especially for the public pcEO algorithm. The times

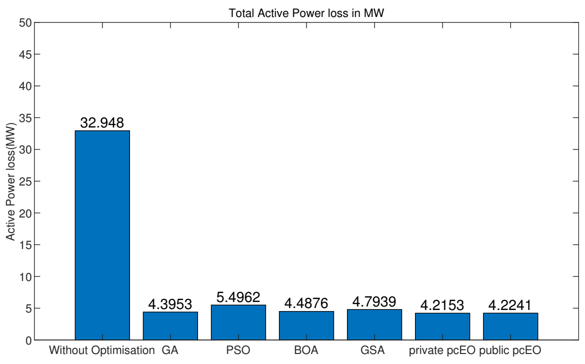
TABLE 7. Line data of the system

From	*To*	*R*	*X*	*B*/2
1	2	0.0472	0.0379	0.0045
1	3	0.0119	0.2560	0.0042
2	4	0.0460	0.2559	0.0085
3	4	0.0120	0.0379	0.0102
2	5	0.0472	0.1983	0.0209
2	6	0.0472	0.1304	0.0187
4	6	0.0119	0.2090	0.0045
5	7	0.0460	0.1160	0.0102
6	7	0.0267	0.0820	0.0085
6	8	0.0120	0.0420	0.0045
6	9	0.0	0.2080	0.0
6	10	0.0	0.5560	0.0
9	11	0.0	0.2080	0.0
9	10	0.0	0.1100	0.0
4	12	0.0	0.2560	0.0
12	13	0.0	0.1400	0.0
12	14	0.0636	0.2559	0.0
12	15	0.0662	0.1304	0.0
12	16	0.0945	0.1987	0.0
14	15	0.2210	0.1997	0.0
16	17	0.0824	0.1923	0.0
15	18	0.1073	0.2185	0.0
18	19	0.0639	0.1292	0.0
19	20	0.0340	0.0680	0.0
10	20	0.0936	0.2090	0.0
10	17	0.0324	0.0845	0.0
10	21	0.0348	0.0749	0.0
10	22	0.0727	0.1499	0.0
21	23	0.0116	0.0236	0.0
15	23	0.1000	0.2020	0.0
22	24	0.1150	0.1790	0.0
23	24	0.1320	0.2700	0.0
24	25	0.1885	0.3292	0.0
25	26	0.2544	0.3800	0.0
25	27	0.1093	0.2087	0.0
28	27	0.0169	0.3960	0.0
27	29	0.2198	0.4153	0.0
27	30	0.3202	0.6027	0.0
29	30	0.2399	0.4533	0.0
8	28	0.0636	0.2000	0.0214
6	28	0.0169	0.0599	0.065

of function evaluation for all algorithms are set to 1200, and other parameters are the same as the previous section. The objective of this paper is to minimize active power loss. Each algorithm needs to use the Newton-Raphson power flow solution to calculate the active power loss of the system during each search step. The initial bus data and line

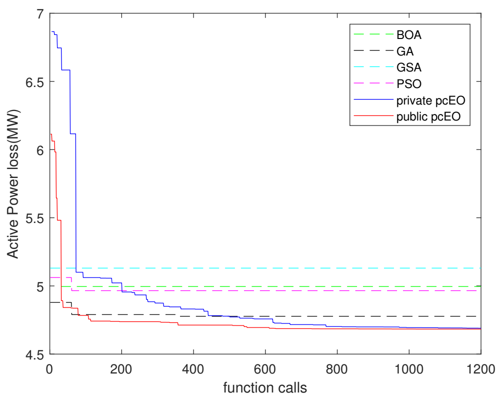


(a) $dim = 10$

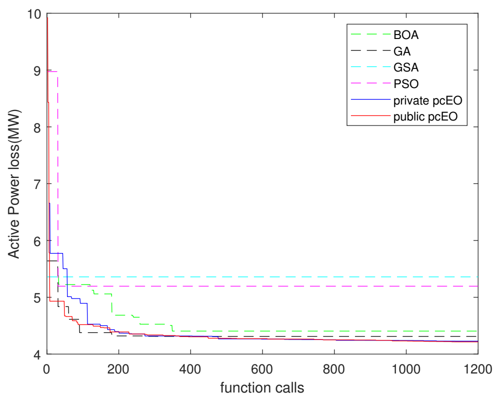


(b) $dim = 20$

FIGURE 5. Comparison results of different algorithms in the optimal allocation of DGs



(a) $dim = 10$



(b) $dim = 20$

FIGURE 6. Comparison results of convergence curve with the different algorithms.

data of the 30 bus system are respectively shown in Table 6 and Table 7 where P_{Gi} and Q_{Gi} respectively represented the active and reactive power emitted by the power supply.

Furthermore, P_{Li} and Q_{Li} respectively represent the active and reactive power consumed by the load. Vsp means the voltage of the bus and Q_{min} and Q_{max} limit the reactive power of the bus. In Table 7, R and X are indicated the impedance of the line and $B/2$ is the ground admittance.

In order to test and evaluate the performance of the proposed algorithms under different conditions, the influence of the number of potential locations is considered in the algorithm. Fig. 5 is shown the comparison results of different algorithms in the optimal sizing of DGs while the number of potential sites is set to 10. Similarly, while the number of potential locations is set to 20, results are illustrated in Fig. 5 where dim represented the number of potential sites.

Compared with other common algorithms, two parallel cEO algorithms can achieve the minimum total active power in both cases. In contrast, PSO and GSA algorithm are showed instability in the various number of potential locations. Next, the convergence ability of algorithms is evaluated in the article in both cases, and the results are shown in Fig. 6. Compared to the convergence curves of different algorithms, the proposed parallel cEO algorithms focus on implementing the ability of global search and avoids falling into the local optimum.

6. Conclusion.

Two improved EO algorithms based on compact and parallel communication technology are proposed to determine the optimal allocation of the DG units in this paper. The basis of the EO algorithm is the establishment of a mass conservation model, which represents the change process of the mass in the solution under both dynamic and static equilibrium state. The purpose of this paper is to minimize the active power loss in the distribution power system. Compact technology can express the distributed characteristics of distributed generators in large power systems and take up less memory space. The interval update method can take advantage of the exploration ability of the equilibrium algorithm and advance better balance the EO algorithm and a compact scheme. Next, two parallel cEO algorithms with different structures of the equilibrium pool are proposed. Parallel communication strategy is different as a result of the different structures of the equilibrium pool of both two algorithms. Compared with other common algorithms the public pcEO algorithm takes less memory space and has good convergence performance. The objective of Sect. 5 is to find optimal sizing of the DGs, and introduce that compared with other algorithms, two parallel cEO algorithms can obtain better results and faster convergence speed. The proposed approach by adopting some advanced schemes [49–52].

REFERENCES

[1] D. G. Luenberger, Y. Ye, *Linear and nonlinear programming*, Springer, Switzerland, 1984.

[2] I. Griva, S. G. Nash, A. Sofer, *Linear and nonlinear optimization*, Siam, Philadephia, the United States, 2009.

[3] X. Xue, J. S. Pan, An overview on evolutionary algorithm based ontology matching, *Journal of Information Hiding and Multimedia Signal Processing*, vol. 9, no. 1, pp. 75–88, 2018.

[4] I. BoussaïD, J. Lepagnot, P. Siarry, A survey on optimization metaheuristics, *Information Sciences*, vol. 237, pp. 82–117, 2013.

[5] J. H. Holland, *Adaptation in natural and artificial systems:an introductory analysis with applications to biology, control, and artificial intelligence*, MIT Press, London, England, 1992.

[6] R. Eberhart, J. Kennedy, A new optimizer using particle swarm theory, in *Proceedings of the Sixth International Symposiumon Micro Machine and Human Science*, pp. 39–43, 1995.

[7] M. Dorigo, M. Birattari, T. Stutzle, Ant colony optimization, *IEEE Computational Intelligence Magazine*, vol. 1, no. 4, pp. 28–39, 2006.

[8] D. Cai, X. Lei, A New Evolutionary Algorithm Based on Uniform and Contraction for Many-objective Optimization, *Journal of Network Intelligence*, vol. 2, no. 1, pp. 171–185, 2017.

[9] S. Mirjalili, S. M. Mirjalili, A. Lewis, Grey wolf optimizer, *Advances in Engineering Software*, vol. 69, pp. 46–61, 2014.

[10] J. S. Pan, P. Hu, S. C. Chu, A. Lewis, Novel parallel heterogeneous meta-heuristic and its communication strategies for the prediction of wind power, *Processes*, vol. 7, no. 11, 845, 2019.

[11] Z. Y. Meng, J. S. Pan, QUasi-Affine TRansformation Evolution with External ARchive (QUATRE-EAR): An enhanced structure for Differential Evolution, *Knowledge-Based Systems*, vol. 155, pp. 35-52, 2018.

[12] B. Q. Jiang, J. S. Pan, A parallel quasi-affine transformation evolution algorithm for global optimization, *Journal of Network Intelligence*, vol. 4, no. 2, pp. 30-46, 2019.

[13] S. C. Chu, P. W. Tsai, J. S. Pan, Cat Swarm Optimization, in *2008 International Conference on Machine Learning and Cybernetics*, pp. 854–858, 2006.

[14] J. Li, M. Gao, J. S. Pan, S. C. Chu, A parallel compact cat swarm optimization and its application in dv-hop node localization for wireless sensor network, *Wireless Networks*, pp. 1-21, DOI: 10.1007/s11276-021-02563-9

[15] A. H. Gandomi, X. S. Yang, A. H. Alavi, Cuckoo search algorithm: a metaheuristic approach to solve structural optimization problems, *Engineering with Computers*, vol. 29, no. 1, pp. 17-35, 2013.

[16] J. S. Pan, P. C. Song, S. C. Chu, Y. J. Peng, Improved compact cuckoo search algorithm applied to location of drone logistics hub, *Mathematics*, vol. 8, no. 3, 333, 2020.

[17] J. S. Pan, P. W. Tsai, Y. B. Liao, Fish Migration Optimization Based on the Fishy Biology, in *2010 Fourth International Conference on Genetic and Evolutionary Computing*, pp. 783–786, 2010.

[18] Q. W. Chai, S. C. Chu, J. S. Pan, W. M. Zheng, Applying adaptive and self assessment fish migration optimization on localization of wireless sensor network on 3-dterrain, *Journal of Information Hiding and Multimedia Signal Processing*, vol. 11, no. 2, pp. 90-102, 2020.

[19] B. Guo, Z. Zhuang, J. S. Pan, S. C. Chu, Optimal Design and Simulation for PID Controller Using Fractional-Order Fish Migration Optimization Algorithm, *IEEE Access*, vol. 9, pp. 8808–8819, 2021.

[20] S. Arora, S. Singh, Butterfly optimization algorithm: a novel approach for global optimization, *Soft Computing*, vol. 23, no. 3, pp. 715–734, 2019.

[21] P. C. Song, S. C. Chu, J. S. Pan, H. Yang, Phasmatodea population evolution algorithm and its application in length-changeable incremental extreme learning machine, in *2020 2nd International Conference on Industrial Artificial Intelligence*, pp. 1–5, 2020.

[22] A. Faramarzi, M. Heidarinejad, B. Stephens, S. Mirjalili, Equilibrium optimizer: A novel optimization algorithm, *Knowledge-Based Systems*, vol. 191, 105190, 2020.

[23] M. Abdel-Basset, R. Mohamed, S. Mirjalili, R. K. Chakrabortty, M. J. Ryan, Solar photovoltaic parameter estimation using an improved equilibrium optimizer, *Solar Energy*, vol. 209, pp. 694–708, 2020.

[24] S. Agnihotri, A. Atre, H. Verma, Equilibrium optimizer for solving economic dispatch problem, in *2020 IEEE 9th Power India International Conference*, pp. 1–5, 2020.

[25] W. El-Khattam, M. M. Salama, Distributed generation technologies, definitions and benefits, *Electric Power Systems Research*, vol. 71, no. 2, pp. 119–128, 2004.

[26] P. Chiradeja, R. Ramakumar, An approach to quantify the technical benefits of distributed generation, *IEEE Transactions on Energy Conversion*, vol. 19, no. 4, pp. 764–773, 2004.

[27] U. Sultana, A. B. Khairuddin, M. Aman, A. Mokhtar, N. Zareen, A review of optimum DG placement based on minimization of power losses and voltage stability enhancement of distribution system, *Renewable and Sustainable Energy Reviews*, vol. 63, pp. 363–378, 2016.

[28] M. Suresh, E. J. Belwin, Optimal DG placement for benefit maximization in distribution networks by using Dragonfly algorithm, *Renewables: Wind, Water, and Solar*, vol. 5, no. 1, pp. 1-8, 2018.

[29] F. S. Abu-Mouti, M. E. El-Hawary, Optimal Distributed Generation Allocation and Sizing in Distribution Systems via Artificial Bee Colony Algorithm, *IEEE Transactions on Power Delivery*, vol. 26, no. 4, pp. 2090–2101, 2011.

[30] C. Sun, Y. Jin, R. Cheng, J. L. Ding, J. C. Zeng, Surrogate-assisted cooperative swarm optimization of high-dimensional expensive problems, *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 4, pp. 644–660, 2017.

[31] H. Wang, W. J. Wang, L. Z. Cui, H. Sun (eds.), A hybrid multi-objective firefly algorithm for big data optimization, *Applied Soft Computing*, vol. 69, pp. 806–815, 2018.

[32] G. R. Harik, F. G. Lobo, D. E. Goldberg, The compact genetic algorithm, *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 4, pp. 287–297, 1999.

[33] E. Mininno, F. Neri, F. Cupertino, D. Naso, Compact Differential Evolution, *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 1, pp. 32–54, 2010.

[34] T. T. Nguyen, J. S. Pan, T. K. Dao, An improved flower pollination algorithm for optimizing layouts of nodes in wireless sensor network, *IEEE Access*, vol. 7, pp. 75985–75998, 2019.

[35] F. Neri, E. Mininno, G. Iacca, Compact particle swarm optimization, *Information Sciences*, vol. 239, pp. 96–121, 2013.

[36] J. S. Pan, T. K. Dao, T. T. Nguyen, A Compact Bat Algorithm for Unequal Clustering in Wireless Sensor Networks, *Applied Sciences*, vol. 9, no. 10, 1973, 2019.

[37] X. Xue, A compact firefly algorithm for matching biomedical ontologies, *Knowledge and Information Systems*, vol. 62, pp. 2855–2871, 2020.

[38] D. Abramson, J. Abela, A Parallel Genetic Algorithm for Solving the School Timetabling Problem, *https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.55.2679&rep=rep1&type=pdf*

[39] M. Randall, A. Lewis, A parallel implementation of ant colony optimization, *Journal of Parallel and Distributed Computing*, vol. 62, no. 9, pp. 1421–1432, 2002.

[40] P. W. Tsai, J. S. Pan, S. M. Chen, B. Y. Liao, S. P. Hao, Parallel cat swarm optimization, in *2008 International Conference on Machine Learning and Cybernetics*, pp. 3328–3333, 2008.

[41] V. Roberge, M. Tarbouchi, G. Labonte, Comparison of Parallel Genetic Algorithm and Particle Swarm Optimization for Real-Time UAV Path Planning, *IEEE Transactions on Industrial Informatics*, vol. 9, no. 1, pp. 132–141, 2013.

[42] C. Y. Chuang, Y. P. Chen, On the effectiveness of distributions estimated by probabilistic model building, in *Proceedings of the 10th annual conference on Genetic and evolutionary computation*, pp. 391–398, 2008.

[43] E. Mininno, F. Neri, F. Cupertino, D. Naso, Compact differential evolution, *IEEE Transactions on Evolutionary Computation*, vol. 15, no. 1, pp. 32–54, 2011.

[44] P. C. Song, J. S. Pan, S. C. Chu, A parallel compact cuckoo search algorithm for three-dimensional path planning, *Applied Soft Computing*, vol. 94, 106443, 2020.

[45] E. Mininno, F. Cupertino, D. Naso, Real-valued compact genetic algorithms for embedded microcontroller optimization, *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 2, pp. 203–219, 2008.

[46] J. F. Chang, R. JohnFrancis, J. S. Pan, S. C. Chu, A parallel particle swarm optimization algorithm with communication strategies, *Journal of Information Science and Engineering*, vol. 21, pp. 809–818, 2005.

[47] J. J. Liang, B. Y. Qu, P. N. Suganthan, Problem definitions and evaluation criteria for the CEC 2014 special session and competition on single objective real-parameter numerical optimization, *https://bee22.com/resources/Liang%20CEC2014.pdf.*

[48] E. Rashedi, H. Nezamabadi-Pour, S. Saryazdi, GSA: a gravitational search algorithm, *Information Sciences*, vol. 179, no. 13, pp. 2232–2248, 2009.

[49] Z. Meng, J. S. Pan, L. Kong, Parameters with adaptive learning mechanism (PALM) for the enhancement of differential evolution, *Knowledge-Based Systems*, vol. 141, pp. 92–112, 2008.

[50] J. S. Pan, Z. Meng, H. Xu, X. Li, A matrix-based implementation of de algorithm: the compensation and deficiency, in *30th International Conference on Industrial Engineering and Other Applications of Applied Intelligent Systems*, vol. 10350, pp. 72–81, 2008.

[51] J. S. Pan, X. Wang, S. C. Chu, T. Nguyen, A multi-group grasshopper optimisation algorithm for application in capacitated vehicle routing problem, *Data Science and Pattern Recognition*, vol. 4, no. 1, pp. 41–56, 2020.

[52] X. Xue, H. Yang, J. Zhang, J. Zhang, D. Chen, An Automatic Biomedical Ontology Meta-matching Technique, *Journal of Network Intelligence*, vol. 4, no. 3, pp. 109–113, 2019.