Application of a Novel Adaptive Chaos Butterfly Optimization Algorithm in Numerical Computation and Power Engineering

Kai He, *Member, IAENG,* Yong Zhang*, Ruilin Chang, Tao Qu

Abstract—**Butterfly optimization algorithm (BOA) is a meta-heuristic algorithm that has been widely applied in recent years. To alleviate the limitations of the traditional BOA, this paper proposes a novel adaptive chaos BOA, namely ACBOA. Firstly, with regard to the fragrance and switch probability in BOA, adaptive adjustments are employed to create more refined fragrance perception and search behaviors. Secondly, based on the randomness and ergodicity of Tent chaos mapping, a wider range of areas is provided for each butterfly to explore and exploit. Then, the proposed ACBOA and 9 comparison algorithms are subjected to a detailed numerical computation analysis using CEC2022 test suite. Finally, ACBOA is applied to 3 power engineering problems: economic emission dispatch, synchronous optimal pulse-width modulation (SOPWM) for 3-level inverters, and parameter extraction of photovoltaic model. Experiments demonstrate that the proposed algorithm outperforms the comparison algorithms and showcases strong potential for numerical computation and applications in power engineering.**

Index Terms—**Butterfly optimization algorithm, Adaptive, Chaos, Numerical computation, Power engineering**

I. Introduction

META-HEURISTIC search algorithms are a category of general optimization methods designed based on natural phenomena, biological behaviors, or human experiences [1]. These algorithms do not rely on the specific mathematical properties of the problems being dealt with. Instead, they possess remarkable characteristics, such as strong global search capabilities, excellent robustness, ease of implementation, and relatively low computational costs. Common types of meta-heuristic search algorithms include genetic algorithm (GA) [2], simulated annealing (SA) [3], and so on. Each of these algorithms simulates different natural or biological behaviors. For example, GA imitates the evolutionary processes of inheritance, crossover, and mutation in biological organisms, while the SA draws inspiration from the physical process of metal annealing. These algorithms have been widely applied in a broad range of fields, including engineering design [4], production scheduling [5], bioinformatics [6], and image processing [7]. The future development trends involve integrating with other methods or knowledge, making improvements to tackle complex problems, and achieving self-adaptation and intelligence by leveraging new technologies.

Butterfly optimization algorithm (BOA) is a typical population-based meta-heuristic algorithm proposed in 2019, which is designed based on the foraging behavior of butterflies attracted by the fragrance of flowers [8]. In BOA, the solution space corresponds to the search space of butterflies, and the quality of the solution corresponds to the concentration of the fragrance. The key elements include the fragrance concentration, perception distance, and switch probability, which are used to balance the global and local searches. This algorithm has been applied in various fields such as engineering optimization, image processing, and energy management. Although it has a simple principle, good global search ability, and strong adaptability, when dealing with large-scale, high-dimensional, and complex problems, it may encounter issues such as slow convergence and low precision [9]. Moreover, its performance is sensitive to parameter settings. In response to the deficiencies of BOA, researchers have carried out extensive studies, mainly focusing on [10]: (a) Heterogeneous integration; (b) Parameter adjustment; (c) Noise interference; (d) Movement behavior. The specific summary is as follows.

(a) Heterogeneous integration. BOA demonstrates excellent performance in addressing specific problems. However, it struggles to provide universal solutions for the complex and diverse real-world issues, which significantly restricts its application scope. In contrast, heterogeneous integration, which combines multiple approaches, offers greater flexibility in handling tasks across various fields, such as engineering optimization and data mining. Moreover, heterogeneous integration enhances the robustness of BOA. Through the collaborative operation of multiple algorithms, it reduces the risk of BOA failure due to its inherent limitations, thereby expanding its applicable range. For instance, Sushmita Sharma et al. [11] proposed an improvement to the search ability of BOA by leveraging the mutualistic phase of the symbiotic organisms search (SOS) algorithm. Additionally, Dana Marsetiya Utama et al. [12] utilized the classic tabu search (TS) algorithm, along with local flipping and swapping strategies, to enhance the performance of BOA, making it more suitable for vehicle routing problems.

(b) Parameter adjustment. The parameters of an optimization algorithm directly determine the performance of the algorithm. In the case of BOA, adjusting the fragrance parameter can influence the tendency of butterflies towards

Manuscript received April 21, 2025; revised June 19, 2025.

Kai He is a doctoral candidate of School of Electronic and Information Engineering, University of Science and Technology Liaoning, Anshan, Liaoning 114051, China (e-mail: kaihe@ustl.edu.cn).

Yong Zhang is a professor of School of Electronic and Information Engineering, University of Science and Technology Liaoning, Anshan, Liaoning 114051, China (* Corresponding author, phone: 8615841209991; e-mail: zy9091@163.com).

Ruilin Chang is an undergraduate of School of Electronic and Information Engineering, University of Science and Technology Liaoning, Anshan, Liaoning 114051, China (e-mail: 13904220098@163.com).

Tao Qu is an engineer of Power Branch Company of Gongchangling Mining Company Limited, Anshan Iron and Steel Group Corporation, Liaoyang, Liaoning 111007, China (e-mail: happy2002tao@163.com).

solutions, facilitating the screening of high-quality solutions. By regulating the switch probability, it is possible to control the switching frequency between global and local search modes. This helps to avoid premature convergence to local optima and improves the accuracy and efficiency of solution seeking. To address the limitation of BOA's relatively poor exploration ability, Fan YuQi et al. [13] modified the update strategy and fragrance coefficient. Lee Sen Tan et al. [14] first corrected the switching probability and sensory modality of BOA and then applied it to the training of wavelet neural networks. Sushmita Sharma et al. [15] employed Lagrangian interpolation, adaptive parameters, and Lévy flight to enhance the search ability of BOA.

(c) Noise interference. In the improvement of algorithms, the method of noise interference has been attracting increasing attention. It can enhance the diversity of the search. By introducing randomness into the position update rule of butterflies in BOA, it encourages the butterflies to explore more areas of the solution space, thus preventing premature convergence. Moreover, it helps the algorithm escape from local optima. The perturbation caused by noise can break the attraction of local optima. Aiming at the poor adaptability of the sensory modality and the blindness of the search in BOA, Kun Hu et al. [16] adopted the Weibull distribution and the normal distribution for improvement.

(d) Movement behavior. Movement behavior plays a crucial and multifaceted role in intelligent optimization algorithms, mainly manifested in guiding the search direction, balancing the search capabilities, and maintaining population diversity. In BOA, movement behavior is determined by factors such as the fragrance concentration of butterflies, the perception distance, and the switch probability. These factors dictate the way individual butterflies move and the search paths they take within the solution space. Through continuous movement, butterflies in BOA can explore new positions and seek better solutions. This movement behavior also helps to balance global search, which aims to extensively explore the solution space, and local search, which focuses on meticulously mining high-quality solution regions. As a result, it enables the effective optimization of problems. Yu Li et al. [17] introduced an opposition-based learning mechanism, adaptive elite mutation, and a piecewise adjustment factor to improve the search performance of BOA.

In order to further improve the performance of BOA and expand its application scope, inspired by previous studies, this paper proposes a novel adaptive chaos butterfly optimization algorithm (ACBOA). The main contributions of this paper are as follows.

(1) A novel variant of BOA is proposed to solve numerical computation problems and those in the field of power engineering.

(2) The proposed adaptive adjustment mechanism enhances the flexibility of the algorithm, enabling it to more effectively adapt to various optimization scenarios by balancing exploration and exploitation.

(3) Chaos mapping is used and integrated into a new movement pattern to traverse a wider solution space, thereby improving the search ability of the algorithm.

(4) The proposed ACBOA is subjected to a detailed numerical computation analysis on CEC2022 together with 9 comparison algorithms.

(5) ACBOA is applied to solve 3 application problems in the field of power engineering.

The subsequent parts of this paper are arranged as follows.
Section II describes the traditional BOA.
Section III details the proposed ACBOA.
Section IV demonstrates the evaluation on numerical computation.
Section V presents the application in power engineering.
Section VI discusses the conclusions and future work.

II. Butterfly optimization algorithm (BOA)

The process of BOA is as follows: After initializing the population, calculate the fragrance concentration to evaluate the solutions. Then, update the positions of the butterflies according to the rules based on the perceived distance and the switch probability to approach the optimal solution. This process continues until the termination condition is met.

A. Initialize

The initial positions of N butterflies are randomly generated within the domain $[lb, ub]$, as described by Eq.(1).

$$X_{i,j} = (ub_j - lb_j) * rand + lb_j, i \in [1, N], j \in [1, D] \quad (1)$$

Here, D represents the dimensionality, and $X_{i,j}$ stands for the j-th dimension of the i-th butterfly. $rand$ is a random number between [0,1].

B. Fragrance

In BOA, fragrance serves as a guiding perceptual mechanism for the movement of butterflies, and its mathematical expression is given as Eq.(2).

$$f_i = I^a * c \quad (2)$$

Here, c represents the sensory modality, which has an impact on both the convergence speed and accuracy. I denotes the stimulus intensity, and its value is determined by the fitness value. The power exponent a can be calculated based on the current iteration t and the maximum iteration T, specifically, $a = 0.2 * (t/T) + 0.1$.

C. Search

Each butterfly in BOA generates a random number $rand$ between [0,1] and compares it with the switch probability p to determine whether to use global search or local search, which is expressed as Eq.(3).

$$X_i^{t+1} = \begin{cases} X_i^t + (r^2 * X_g^t - X_i^t) * f_i & \text{if } rand < p \\ X_i^t + (r^2 * X_l^t - X_k^t) * f_i & \text{otherwise} \end{cases} \quad (3)$$

Among them, X_g^t is the global optimal solution up to the t-th iteration, and X_l^t and X_k^t are the l-th and k-th butterflies randomly selected from the population. r denotes a random number between [0, 1].

III. The proposed adaptive chaos butterfly optimization algorithm (ACBOA)

In this section, we will introduce the adaptive adjustment mechanism and incorporate the chaos mapping into the position update process of BOA, so as to improve its search performance.

A. Adaptive adjustments

In BOA, fragrance is defined by the fitness function of the optimization problem. Due to the significant differences in the optimal solutions of different optimization problems, the fragrance value experiences substantial fluctuations. Such fluctuations have a negative impact on the butterfly position-updating process, thereby reducing the convergence efficiency and stability of the algorithm. [10] presents a new adaptive fragrance method, which has been proven to be effective. Therefore, this paper introduces the adaptive fragrance to replace Eq.(2), as shown in Eq.(4).

$$f_i = [fitness_i/(fitness_g + eps)]^a * (1 - 0.6 * \sqrt{t/T}) \quad (4)$$

$fitness_i$ and $fitness_g$ represent the fitness values of the i-th butterfly and X_g so far, respectively. eps is a floating-point relative precision number with a value of 2.2204*10^{-16}.

In addition, emphasizing exploration in the early stage and focusing on exploitation in the later stage is more conducive to enhancing the search performance of the algorithm. However, the switch probability p in BOA is fixed at 0.8, which may lead to an imbalance between exploration and exploitation [18]. Therefore, this paper employs a new adaptive switch probability p_a, as shown in Eq.(5).

$$p_a = 0.8 - 0.5 * t/T \quad (5)$$

These two adaptive adjustments to BOA help it create more refined fragrance perception and search behavior.

B. Chaos mapping

As can be seen from Eq.(3), the movement of each butterfly depends on the random number $rand$. Although this random number can provide equiprobable randomness, the blindness of the random number combined with the weakening effect of $rand * rand$ may lead to low search efficiency. Chaos is a common nonlinear phenomenon. By using a mapping relationship, a chaotic sequence within the range of [0, 1] can be generated, which features randomness and ergodicity [19]. In this paper, the Tent chaos mapping shown in Eq.(6) is introduced.

$$z_i = rand/(N * D) + (2 * rand)mod1 \quad (6)$$

The random numbers generated by the Tent chaos mapping are introduced into the solution space of the optimization problem, thereby generating noise disturbances to the movement of individuals. Specifically, we designed a butterfly position update behavior as shown in Eq.(7) to replace Eq.(3).

$$X_i^{t+1} = \begin{cases} X_i^t + z_i * (X_g^t - X_i^t) * f_i & \text{if } rand < p_a \\ X_i^t + z_i * (X_l^t - X_k^t) * f_i & \text{otherwise} \end{cases} \quad (7)$$

The chaos mapping interferes with the movement of individual butterflies and generates a broader area for exploration and exploitation, which is of great significance for discovering more promising candidate solutions.

C. Structure of the proposed ACBOA

The pseudo-code and flowchart of the proposed ACBOA can vividly illustrate its structure, as depicted in Algorithm 1 and Fig. 1, separately.

Algorithm 1: The pseudo-code of ACBOA

Input: Initialize parameters.
Output: Optimal solution X_g.
Initialize population by Eq.(1);
Evaluate fitnesses of X, get X_g and FEs;
$t = 0$, $T = \lceil FEs_m/N \rceil$;
while $t \le T$ && $FEs \le FEs_m$ **do**
  $t = t + 1$;
  Compute power exponent a;
  Get adaptive switch probabilit p_a by Eq.(5);
  for $i = 1$ ***to*** N **do**
    Calculate adaptive fragrance f_i by Eq.(4);
    Obtain chaos mapping z_i by Eq.(6);
    if $rand < p_a$ **then**
      $X_i^{t+1} = X_i^t + z_i * (X_g^t - X_i^t) * f_i$;
    else
      $X_i^{t+1} = X_i^t + z_i * (X_l^t - X_k^t) * f_i$;
    end
    Check boundaries and evaluate X_i^{t+1};
    Update X_i^{t+1}, X_g, and FEs;
  end
end

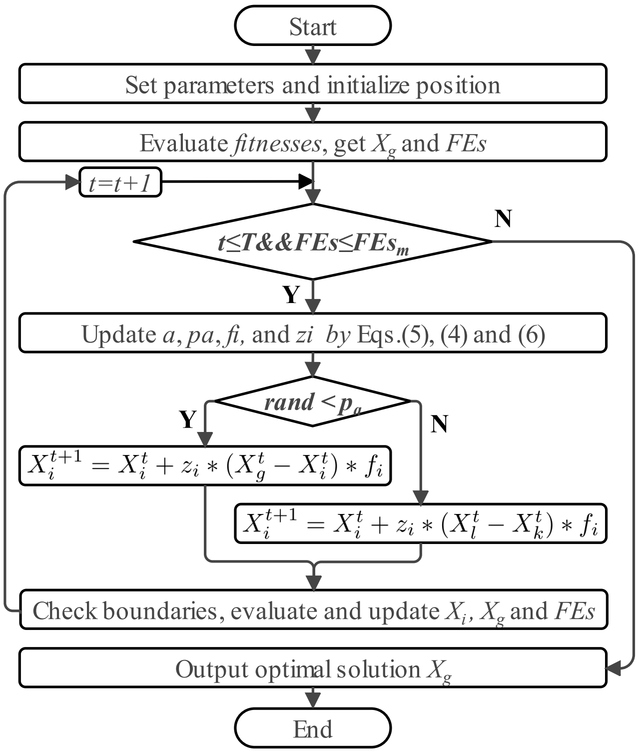


Fig. 1: Structure of ACBOA.

IV. Evaluation on Numerical Computation

In this section, we conduct a comprehensive numerical computation analysis of the proposed algorithm based on the well-known CEC2022 test suite.

A. Description of benchmark functions

CEC2022, namely the Congress on Evolutionary Computation 2022 [20], is a highly authoritative benchmark testing platform in the field of evolutionary computation.

This test suite has designed 12 benchmark test functions, covering various types such as unimodal, multimodal, hybrid, and composite functions. By flexibly adjusting dimensional parameters and adding noise interference, it constructs a diverse range of complex optimization environments. These test functions can systematically evaluate the global search ability of optimization algorithms, the efficiency of escaping from local optima, the convergence speed, and the stability performance. As a universal standard evaluation tool in the industry, CEC2022 provides a unified quantitative evaluation framework for researchers and has become an important reference system for verifying the performance superiority of new optimization algorithms.

B. Comparison algorithm and parameter setting

To fully verify the performance of ACBOA, we have selected 9 representative comparative algorithms, including BOA variants, classic algorithms, newly proposed algorithms, and highly cited algorithms. The detailed information and parameter settings of these algorithms are shown in Table I.

TABLE I
COMPARISON ALGORITHMS AND PARAMETERS SETTING.

Algorithm	Year	Ref.	Parameters setting
BOA	2019	[8]	p=0.8, c=0.01
BKA	2024	[21]	p=0.9, r=rand
CBOA	2022	[22]	I=(1+rand)
GWO	2014	[23]	A=[0,2], C=2*rand, a=[0,2]
MABOA	2021	[16]	p=0.8, c=c_t+0.025*c_t/T, N=220, λ=2
mMBOA	2020	[24]	p=0.8, c=0.01
PPSO	2019	[25]	v_{max}=0.5*dx, dx=ub-lb
SABOA	2020	[13]	p=0.8, c=0.01
WOA	2016	[26]	a=[0,2], b=1 A=[0,2], l=[-1,1], C=2*rand
ACBOA	Present	Present	c=0.01

All experiments were executed in the MATLAB software, operating on a Windows 11 based personal computer equipped with 64.0G of memory and a 3.40G Hz CPU. A population size of 30 individuals was specified for each algorithm run, and the maximum number of evaluations (FEs_m) for the 10-dimensional function was set to 2×10^5.

C. Evaluation results analysis

ACBOA was comprehensively compared with 9 algorithms, including BOA and its multiple variants, across multiple indicators. The results after 30 runs are presented in Table II, and the rank are based on the 'Mean' values. As shown in the table, ACBOA demonstrated the best performance on the 12 functions in numerical computations, with an overall ranking score of 18 and ultimately securing the first place. ACBOA found solutions closest to the optimal ones for functions f_1, f_2, f_6, f_8, f_9, f_{10}, f_{11}, and f_{12}, outperforming the comparative algorithms by a wide margin. For functions f_3, f_4, f_5, and f_7, although ACBOA did not rank first, it still ranked among the top. When ranked according to the average values of all the tested functions, there is no doubt that ACBOA is the most outstanding. Additionally, the running efficiency of ACBOA is also highly competitive.

D. Convergence performance analysis

In this part, the average convergence performance and result distribution of the algorithms were analyzed based on the unimodal function f_1, and the results are shown in Fig. 2. As can be seen from Fig. 2(a), ACBOA exhibits a much faster convergence speed and higher optimization accuracy for this function compared to the comparative algorithms. In contrast to the relatively poor convergence curve of BOA, this advantage of ACBOA is attributed to the adaptive chaos method proposed in this paper. Additionally, the box plot shown in Fig. 2(b) also indicates that ACBOA has excellent stability in solving the function.

E. Running time analysis

The running time of an algorithm is a core indicator for measuring its efficiency, which directly affects its practicality and scalability in scenarios such as real-time computing and big data processing, and is crucial to whether the algorithm can meet the performance requirements and resource constraints of practical applications.

We have plotted a three-dimensional bar chart as shown in Fig. 3 for the running times in Table II. It is not difficult to conclude that the running time of ACBOA is slightly longer than that of BOA. When compared with other algorithms, the running efficiency of ACBOA is extremely excellent, as it can provide outstanding search results in a relatively short period of time.

F. Friedman rank test

The Friedman rank test is a non-parametric statistical method used to test whether multiple related samples are from the same distribution, and it is often employed to compare the performance differences of multiple algorithms or treatment methods across multiple datasets or experimental conditions.

Based on this, Fig. 4 presents the results of the test using this method. The obtained p-value is 4.481521603E-14, which is much smaller than 0.05, indicating that there are significant performance differences among these algorithms. As can be seen from the figure, the bar of ACBOA is the shortest, with an average rank of 1.5, which is much smaller than that of the comparative algorithms. This confirms that ACBOA has the best search ability in numerical computations.

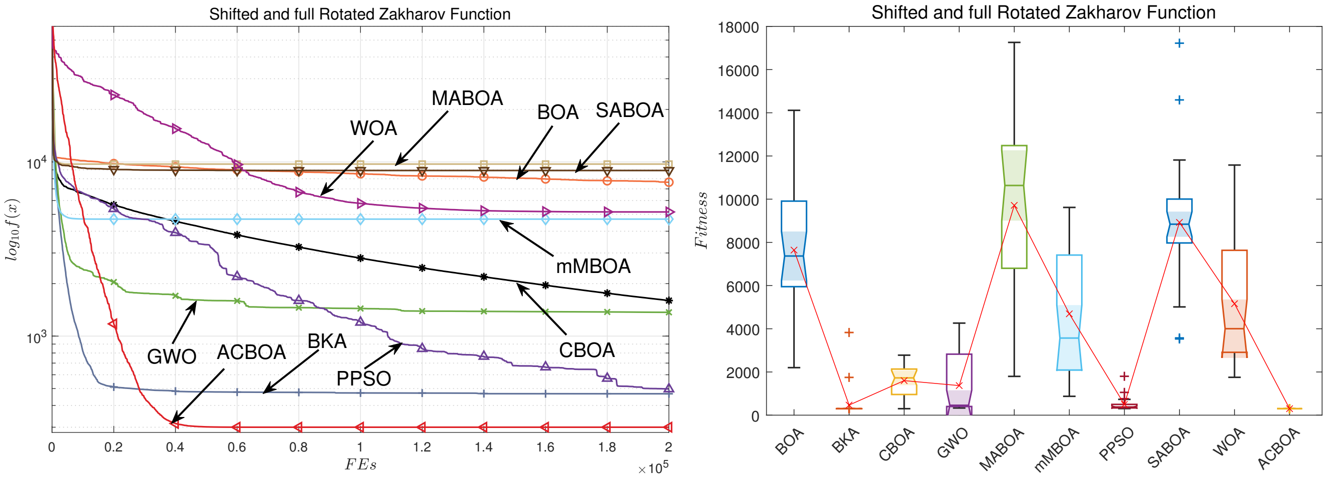
G. Wilcoxon signed rank test

Although the Friedman rank test results has confirmed the existence of significant performance differences between ACBOA and the comparative algorithms, the specific differences between ACBOA and each individual algorithm remain unknown. The Wilcoxon signed rank test, which is used to examine whether two related samples are from populations with the same median, is commonly employed to analyze the differences in paired data and is particularly suitable for the current testing scenario. Table III presents the results of the Wilcoxon signed rank test.

In this table, the symbols 'R-' and 'R+' denote the mean rank sums for instances where ACBOA surpasses and underperforms compared algorithms, respectively, across the 12

TABLE II
NUMERICAL COMPUTATION RESULTS ON CEC2022 BENCHMARK FUNCTION.

Fun	Index	BOA	BKA	CBOA	GWO	MABOA	mMBOA	PPSO	SABOA	WOA	ACBOA
f_1	Mean	7642.958687	466.730037	1599.695747	1369.589284	9714.094524	4692.596160	496.978698	8924.423867	5159.362284	300.000000
	Std	3058.520036	687.548553	682.324323	1431.552820	4007.594940	2901.043361	303.041918	2820.043967	2941.146078	0.000000
	Best	2200.738723	300.081511	300.372431	331.489398	1795.958801	872.952015	304.565064	3528.859552	1753.063362	300.000000
	Worst	14113.094894	3827.457330	2778.051014	4262.183842	17257.909503	9618.164715	1799.959087	17222.529953	11579.934150	300.000000
	Time	7.01212845	7.46803910	7.60066186	7.30032167	14.33408493	8.36841643	7.01794777	7.07006126	7.40254996	7.02374137
	Rank	8	2	5	4	10	6	3	9	7	1
f_2	Mean	2562.872228	431.855721	413.349595	417.276045	915.059619	482.120980	419.034444	525.246628	423.053283	400.199370
	Std	1134.615114	72.105371	30.748853	18.780959	326.920336	71.023192	26.478439	191.927689	30.622208	0.724451
	Best	728.947147	400.039084	400.000470	400.374161	507.303449	405.474332	400.289658	408.489785	400.053233	400.000104
	Worst	4527.979181	781.394687	539.539805	472.042367	1951.464500	725.538122	474.956731	1468.752212	492.890154	403.986681
	Time	6.24428548	6.57971441	6.37408064	6.42537233	12.02950943	7.30346384	6.16884685	6.22332508	6.40015231	6.14474640
	Rank	10	6	2	3	9	7	4	8	5	1
f_3	Mean	658.071236	626.787327	602.710545	600.584623	640.991506	626.808783	635.260157	640.136697	624.880181	606.693430
	Std	6.430912	11.067440	3.038749	0.901772	8.510138	11.102776	11.219831	9.753285	12.798787	3.629170
	Best	646.972477	607.402973	600.000020	600.005125	622.950684	605.700045	613.295192	619.014700	607.344251	602.830789
	Worst	670.521534	653.791429	611.889450	604.133929	656.664273	656.204004	657.545152	658.964890	650.532879	615.420695
	Time	7.15022623	7.56086585	7.45156808	7.46072240	13.04802369	8.39918838	7.18781808	7.32594258	7.40598570	7.20424351
	Rank	10	5	2	1	9	6	7	8	4	3
f_4	Mean	839.632904	824.082194	814.369664	815.117091	853.491184	828.348241	833.725911	844.615492	834.007940	817.740364
	Std	10.437733	11.677279	4.343991	5.095814	10.187130	9.502724	14.974639	13.869421	11.266451	4.399957
	Best	819.752937	808.053309	806.964713	807.319829	834.026727	814.454803	808.453672	821.992299	816.914397	806.998545
	Worst	869.109865	855.788726	824.873902	827.258178	874.871499	850.295094	877.712681	878.160950	865.667673	825.571929
	Time	6.39881764	6.71011322	6.60866575	6.57180693	12.24047339	7.52669333	6.31845427	6.43530144	6.50254299	6.37710494
	Rank	8	4	1	2	10	5	6	9	7	3
f_5	Mean	1838.790287	1081.950682	1044.501653	912.525922	1195.821707	1112.171096	1405.467783	1425.124661	1301.596409	917.570555
	Std	366.582637	113.767789	72.430878	31.664172	192.966048	138.069335	346.093530	258.500031	291.738066	12.891556
	Best	1197.425071	920.700396	913.898789	900.037278	920.997417	914.498997	1023.265223	1042.380883	917.305471	900.832321
	Worst	2559.472522	1346.003179	1209.030847	1030.920897	1831.586690	1463.144724	2845.106604	2187.261086	2164.735840	944.231485
	Time	6.54812772	6.81063988	6.71291428	6.71550321	12.44994135	7.65371653	6.41594077	6.54574604	6.61116067	6.50307091
	Rank	10	4	3	1	6	5	8	9	7	2
f_6	Mean	48347697.696670	2432.457174	1984.264660	6169.474581	8524607.106688	23414.132026	3943.383099	4063.120884	3603.615738	1800.467345
	Std	88628480.177338	1312.565239	273.710129	2339.316186	29419273.874233	32704.136914	2085.866089	2085.311430	1694.735717	0.117430
	Best	128472.177590	1840.697834	1823.125620	1946.130502	7322.857191	2855.919096	1866.719533	1893.616312	1831.476036	1800.203368
	Worst	434799058.846670	7619.737665	3037.020699	8217.328855	146627097.098612	156461.186352	8117.413376	10905.494511	7994.875018	1800.662666
	Time	6.34147050	6.76269828	6.55981089	6.52344172	12.19765091	7.47689677	6.30561051	6.39194600	6.47760205	6.35123161
	Rank	10	3	2	7	9	8	5	6	4	1
f_7	Mean	2098.208890	2043.642918	2017.430440	2030.638504	2086.949322	2057.312841	2092.082118	2112.777157	2050.081221	2026.284899
	Std	26.811171	16.706370	17.567410	11.373981	27.015191	20.805948	40.013463	32.469406	18.320468	2.545013
	Best	2028.308618	2020.211892	2000.003054	2009.074616	2044.055048	2025.619304	2046.879021	2066.090248	2023.028786	2017.694360
	Worst	2145.503706	2090.381686	2080.190506	2058.903851	2143.019955	2099.629274	2213.971715	2183.745807	2087.203260	2030.193086
	Time	7.99881241	8.27522267	8.15638841	8.19973023	13.86902707	9.07964484	7.82481087	8.01841566	8.10388600	7.93285217
	Rank	9	4	1	3	7	6	8	10	5	2
f_8	Mean	2761.908530	2229.352655	2223.480004	2221.241300	2265.237611	2259.497004	2263.660462	2283.789134	2230.316878	2211.128752
	Std	2081.831498	23.536164	22.433205	7.326659	59.287421	52.858402	59.867765	76.904775	5.554735	2.734134
	Best	2230.432931	2207.942141	2201.028057	2202.247931	2228.102607	2225.666942	2225.748772	2223.376370	2218.128864	2207.269781
	Worst	13726.124566	2348.325352	2339.022300	2230.047190	2467.127604	2365.048694	2467.919542	2537.840742	2243.647905	2216.596286
	Time	8.48419806	8.83303304	8.73723569	8.73810474	14.44167972	9.64410814	8.31641967	8.49510319	8.58396446	8.51720628
	Rank	10	4	3	2	8	6	7	9	5	1
f_9	Mean	2812.653242	2538.829039	2574.135778	2559.698152	2739.135748	2636.961387	2556.610421	2669.250559	2534.666202	2505.165333
	Std	76.985979	36.588827	28.259008	27.915715	54.916551	47.074076	44.834272	61.035477	26.750998	33.895567
	Best	2678.809415	2529.284391	2539.706602	2529.284472	2636.394700	2533.843685	2529.301827	2532.484968	2529.286499	2420.351096
	Worst	3015.714440	2689.849327	2676.216332	2625.696269	2878.145526	2711.318319	2699.270651	2865.472105	2676.218025	2529.284383
	Time	7.65366748	7.91283148	7.80943361	7.97428731	13.57057643	8.73076826	7.44219390	7.68411179	7.67891558	7.60249907
	Rank	10	3	6	5	9	7	4	8	2	1
f_{10}	Mean	2612.505837	2636.889018	2572.484260	2578.131269	2722.304412	2608.578700	2722.328042	2666.441725	2590.370828	2504.468965
	Std	233.990070	213.608302	55.364983	60.463991	311.422550	61.441566	365.857356	192.741817	192.214678	21.501177
	Best	2500.906562	2500.285612	2500.713014	2500.255718	2508.425210	2500.617717	2500.989540	2502.108291	2500.309139	2500.343892
	Worst	3467.637039	3419.532025	2631.812532	2736.124861	4191.644258	2671.077764	4045.191349	3545.891428	3501.029525	2618.308475
	Time	7.21734430	7.61146773	7.39907056	7.45494010	13.10425268	8.33703202	7.12295689	7.25338151	7.27398934	7.20659100
	Rank	6	7	2	3	9	5	10	8	4	1
f_{11}	Mean	3219.336064	2716.638556	2661.266430	2769.020630	3436.036112	2895.487238	2808.524336	3157.052443	2816.879227	2600.000252
	Std	268.879213	210.620788	124.134100	148.778293	481.397798	231.917878	156.478341	402.586065	126.731130	0.000671
	Best	2815.567761	2602.537830	2600.000184	2600.154119	2793.809762	2639.315970	2603.604303	2731.725727	2600.330446	2600.000003
	Worst	3986.715944	3200.331192	2930.790100	3189.569522	4469.048640	3384.151899	3200.939514	4214.853263	3184.756391	2600.003509
	Time	7.75714041	8.13706611	8.02722690	7.99682937	13.74520344	8.89933509	7.69290599	7.89510532	7.93651862	7.77125535
	Rank	9	3	2	4	10	7	5	8	6	1
f_{12}	Mean	2921.169671	2871.705488	2906.955745	2866.730110	2995.415549	2889.663992	2903.247167	2920.100555	2882.192152	2862.245087
	Std	48.803470	18.895953	26.711204	8.012627	97.308023	37.315112	51.934154	52.152661	30.295713	4.082418
	Best	2867.931577	2859.763583	2867.131087	2858.758112	2881.165063	2865.532848	2863.597818	2867.834839	2862.567377	2858.708351
	Worst	3042.998895	2948.285584	2968.381589	2893.600440	3345.645198	3039.746602	3065.166570	3050.310609	2985.399598	2881.290968
	Time	8.18591974	8.45279001	8.32136573	8.33440867	14.10965507	9.27562869	7.98436081	8.21657893	8.22123885	8.09179019
	Rank	9	3	7	2	10	5	6	8	4	1
Total rank		109	48	36	37	106	73	73	100	60	18
Final rank		10	4	2	3	9	6.5	6.5	8	5	1



(a) Convergence curve

(b) Box plot

Fig. 2: Convergence curve and box plot on f_4 of CEC2022.

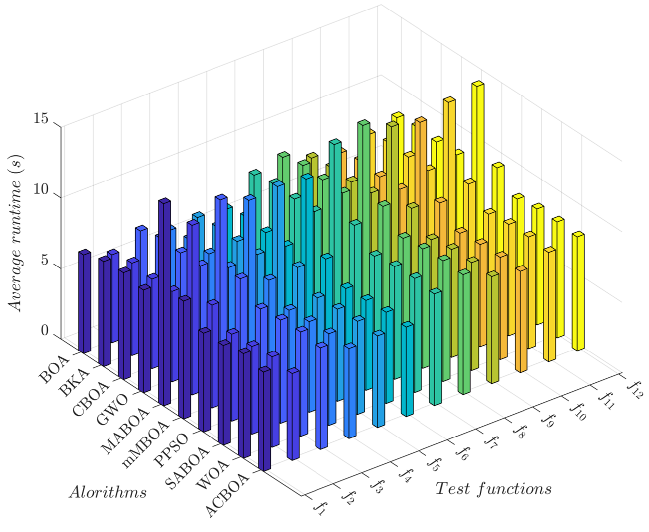


Fig. 3: Average running time of ACBOA with 9 comparison algorithms.

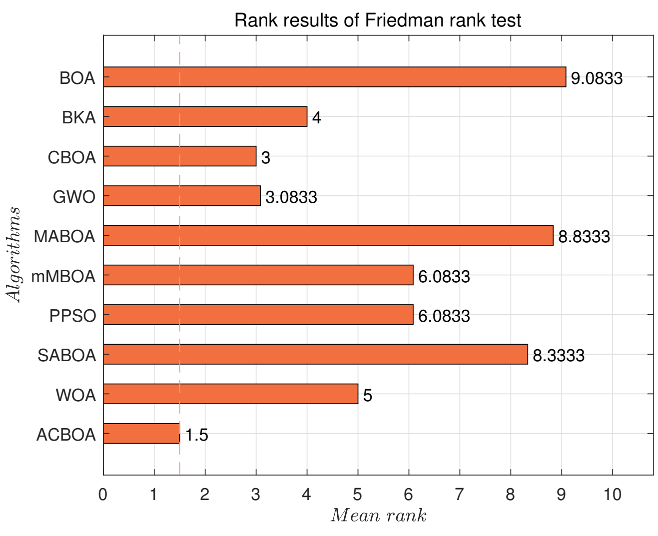


Fig. 4: Friedman rank test results.

TABLE III
WILCOXON TEST (AVERAGE) RESULTS.

ACBOA vs.	p-value	R+	R-	+/=/-
BOA	0.00000178598	0.25000000000	464.75000000000	12/0/0
BKA	0.00226653414	21.00000000000	444.00000000000	12/0/0
CBOA	0.00139832031	109.33333333333	355.66666666667	9/0/3
GWO	0.01392925062	117.58333333333	347.41666666667	8/2/2
MABOA	0.00000173440	0.00000000000	465.00000000000	12/0/0
mMBOA	0.00000388993	2.91666666667	462.08333333333	12/0/0
PPSO	0.00000203555	1.08333333333	463.91666666667	12/0/0
SABOA	0.00000173440	0.00000000000	465.00000000000	12/0/0
WOA	0.00001012967	6.25000000000	458.75000000000	12/0/0

benchmark functions. The notations '+', '-' and '=' signify that ACBOA demonstrates superior performance, inferior performance, and no significant difference relative to the comparative algorithms when solving these 12 functions. From the table, ACBOA exhibits unparalleled advantages, and even achieves the result of '12/0/0' when compared with multiple algorithms, such as BOA, BKA, MABOA, mMBOA, PPSO, SABOA, and WOA.

V. APPLICATION IN POWER ENGINEERING

In this section, the proposed ACBOA is applied to solve 3 kinds of classical applications in power engineering.

A. Economic emission dispatch

Economic emission dispatch is a crucial power system operation strategy aiming to simultaneously minimize the total generation cost and pollutant emissions of power plants by optimally allocating generation power among units while satisfying various operational constraints, thus achieving a balance between economic benefits and environmental protection in power generation [29]. Taking the classic IEEE 14-bus system as an example, we demonstrate the capabilities of ACBOA and the comparative algorithms in dispatching this system. The total demand of this system is 259MW, and the schematic diagram and dataset can be obtained from [30, 31].

TABLE IV
ECONOMIC EMISSION DISPATCH RESULTS OF IEEE 14-BUS SYSTEM.

Generator	BOA	BKA	CBOA	GWO	MABOA	mMBOA	PPSO	SABOA	WOA	ACBOA
PG1 (MW)	164.32539583	173.56921422	123.54432791	154.82335146	131.25881091	131.71560883	174.69918984	173.95602505	203.94749288	173.62587860
PG2 (MW)	38.77336581	46.30829989	49.02195813	48.82007148	50.11297343	51.77904463	36.71541704	23.10226156	20.00000000	46.21477671
PG3 (MW)	22.25429728	19.12238589	26.79763391	16.74869965	32.07792544	23.10235560	19.15320607	23.14500919	15.05240712	19.15924469
PG4 (MW)	23.64680670	10.00000000	40.75633249	28.50044930	17.70191463	21.78062271	18.43208705	10.21354423	10.00000000	10.00000000
PG5 (MW)	10.00000000	10.00000000	18.87999975	10.10727503	27.85068960	30.62263431	10.00000000	28.58305997	10.00000000	10.00000000
Fuel cost ($/h)	691.29712724	686.48998842	715.71179604	693.23755272	718.24975979	712.53682887	689.00878513	708.02233726	700.92200234	686.48704641
Emission (ton/h)	0.16968754	0.18163645	0.14751034	0.16451556	0.14807151	0.14825457	0.17979395	0.18020527	0.21846232	0.18166861
EED ($/h)	705.79282798	701.63839820	729.53589940	707.48897366	752.73883500	726.56203940	704.00353112	723.05138740	719.14167505	701.63813866
Mean	761.94978973	706.78422931	767.98162290	776.74656339	936.35077699	888.63692810	846.87809785	840.66811853	974.33122215	701.63813866
Std	28.29858961	6.11376875	23.98472000	82.98082431	115.93823962	127.95420160	120.59442545	93.76667766	137.70010091	0.00000000
Worst	814.9110862	725.9507249	819.0891783	981.3119689	1253.8385762	1233.4350892	1170.4539780	1151.6202043	1199.1214718	701.6381387
Time (s)	17.15402660	16.31202990	17.74202720	17.77850720	20.80040750	17.81061850	16.68047750	17.52453090	16.68315500	17.02877280

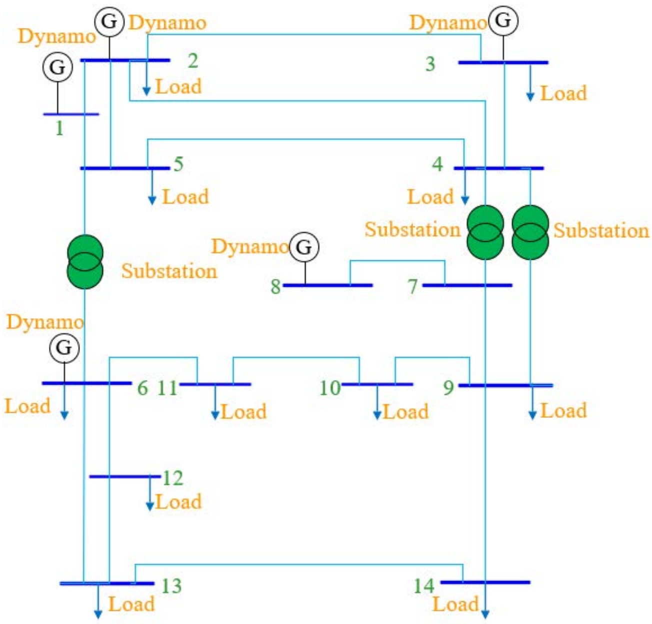


Fig. 5: IEEE 14-bus system [30].

Table IV shows the optimal dispatch solutions of the 5 generators in this system and the corresponding economic emission costs. From this, it can be concluded that when the 5 units output 173.62587860MW, 46.21477671MW, 19.15924469MW, 10.00000000MW, and 10.00000000MW respectively, ACBOA achieves the minimum total economic emission cost of 701.63813866$/h. In this state, the fuel cost obtained by ACBOA is 686.48704641$/h, and the corresponding emission is 0.18166861ton/h. The dispatching capabilities of other algorithms are all weaker than that of ACBOA. Especially when considering the Mean and Std values, ACBOA demonstrates unparalleled stability, and its time consumption also has an advantage. Fig. 6 also intuitively confirms the strong performance of ACBOA in terms of convergence accuracy and stability. Compared with BOA and its variants, the excellent performance of ACBOA once again illustrates that the improvements in this paper are very effective.

In particular, Fig. 7 more intuitively reflects the optimal result of economic emission dispatch. This result also shows that the economic emission problem requires not only considering the economic cost but also taking into account the control of pollutant emissions. Only by minimizing the total cost of economic emissions can a reasonable dispatching scheme be achieved, which also meets the requirements of the current society for environmental issues.

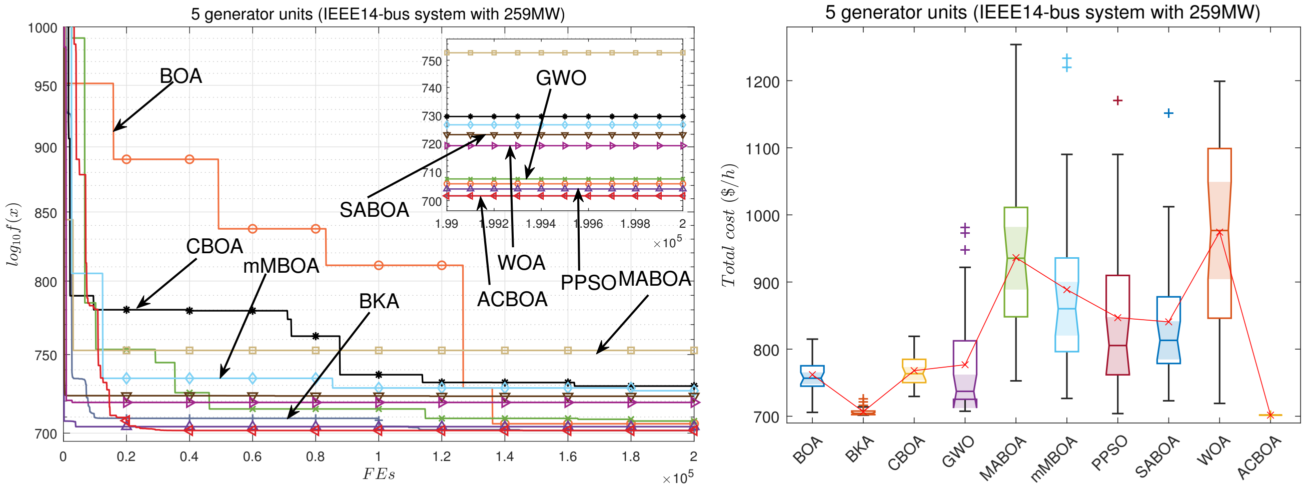
B. Synchronous optimal pulse-width modulation (SOPWM) for 3-level inverters

Synchronous optimal pulse-width modulation (SOPWM) for 3-level inverters is a technology that achieves synchronous operation while optimizing the pulse widths of 3-level inverters, so as to realize efficient power conversion, reduce harmonic distortion, improve the quality of the output voltage, control electrical parameters more optimally, and enhance the overall performance of the inverter system [28]. The SOPWM for 3-level inverters problem is a 25-dimensional constrained optimization problem, and its optimal value is 0.03073936. The topological diagram is shown in Fig. 8.

Table V and Fig. 9 present the results of the optimization of this power engineering application by various algorithms. Due to space limitations, only the performance indicators are provided here. From this table, the time consumption of each algorithm does not vary significantly, all falling within the range of 16s to 20s. However, in terms of the 'Best' value, ACBOA's value of 0.0319922000 is superior to WOA's 0.0319922534, SABOA's 0.0319924000, GWO's 0.0319922263, and CBOA's 0.0319922958. Similarly, the Std value of 0.1730051596 also demonstrates the remarkable stability performance of ACBOA, which corresponds to Fig. 9(b). The convergence shown in Fig. 9(a) indicates that the convergence accuracy of ACBOA is significantly improved after 0.2×10^5 function evaluations (FEs), and this advantage is maintained until the end, ultimately making ACBOA the algorithm closest to the theoretical optimal solution.

C. Parameter extraction of photovoltaic cell model

Parameter extraction of photovoltaic (PV) cell model is of crucial significance for accurately describing the electrical characteristics of PV cell, evaluating its performance, optimizing the design of PV system, predicting the power



(a) Convergence curve

(b) Box plot

Fig. 6: Convergence curve and box plot for applying on economic emission dispatch (IEEE 14-bus system).

TABLE V
COMPARISON RESULTS OF SOPWM FOR 3-LEVEL INVERTERS.

Index	BOA	BKA	CBOA	GWO	MABOA	mMBOA	PPSO	SABOA	WOA	ACBOA
Best	1.0679914755	0.0319923000	0.0319922958	0.0319922263	1.0679924000	0.2751466646	2.8770242657	0.0319924000	0.0319922534	0.0319922000
Mean	1.0679921595	0.0665257200	0.0319923818	0.1967289781	1.0679924000	0.8098678433	5.4806570940	0.4118590655	0.4463923589	0.1243100630
Std	0.0000001809	0.1891468590	0.0000000384	0.5174570890	0.0000000000	0.3454827729	1.5219507217	0.5077772869	0.5162107369	0.1730051596
Worst	1.0679923995	1.0679924000	0.0319924000	2.8680468119	1.0679924000	1.0679923933	10.0086856962	1.0679924000	1.0679924000	0.7470218271
Time (s)	16.0637473	17.4887749	19.5258866	16.4810255	20.1860846	17.9347172	18.8096759	19.1965126	17.8564688	18.0134182

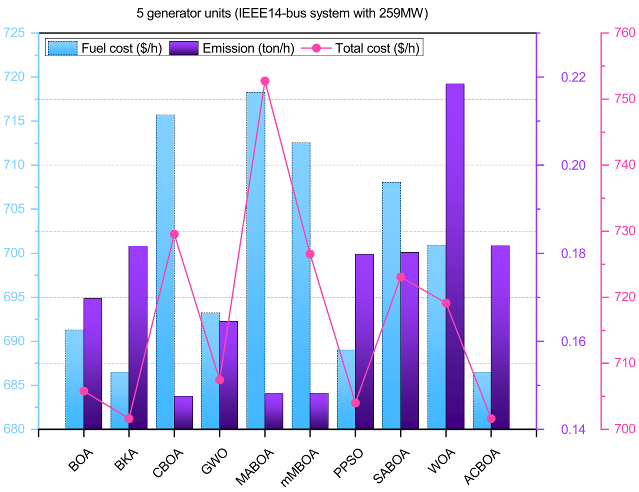


Fig. 7: Optimal economic emission dispatch results of IEEE 14-bus system.

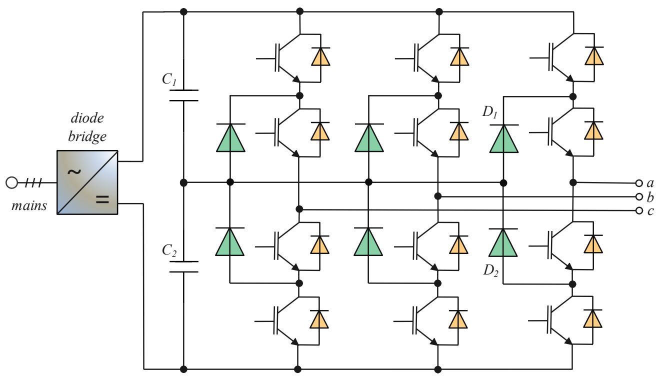
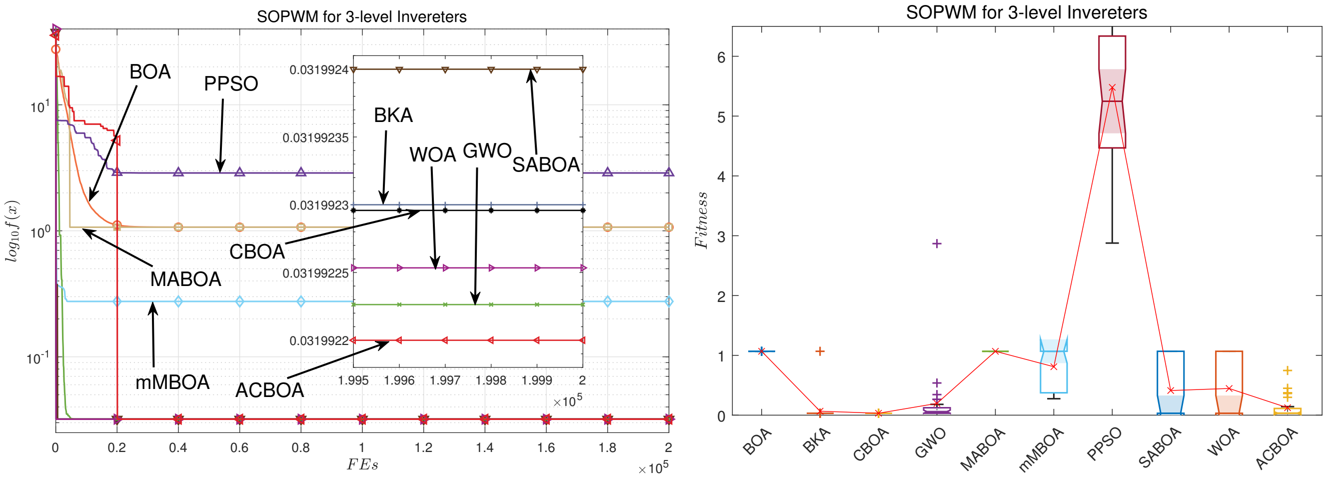


Fig. 8: SOPWM for 3-level inverters topology diagram.

generation efficiency, and gaining an in-depth understanding of the behavior of PV cell under different operating conditions. It serves as an important foundation for the efficient application and development of PV technology. This problem is a boundary-constrained optimization problem, aiming to minimize the root mean square error (RMSE) of the calculated current of the extracted model relative to the measured current, as detailed in [10]. The equivalent circuit diagram under three-diode modeling (TDM) is described in Fig. 10. The tested model is PVM 752 GaAs, and the dataset is obtained from [27]. The optimal parameter results achieved by various algorithms are shown in Table VI, and the convergence curve and box plot are illustrated in Fig. 11.

The results show that ACBOA found the best RMSE (0.000229451) within 4.7556196s, outperforming algorithms such as CBOA, BKA, mMBOA, and PPSO. Its stability performance is also the best, achieving an Std value of 0.0000122. The convergence curve indicates that the initial convergence speed of ACBOA is not satisfactory, but it demonstrates excellent convergence accuracy in the later stage. This suggests that these improvements in this paper can help the algorithm escape from the trap of local optimality. The box plot shown in Fig. 11(b) also illustrates the outstanding stability of ACBOA, which is significantly stronger than that of BOA and other BOA variants.



(a) Convergence curve (b) Box plot

Fig. 9: Convergence curve and box plot for applying on SOPWM for 3-level inverters.

TABLE VI
PARAMETER EXTRACTION RESULTS OF PV CELL MODEL (PVM 752 GAAS UNDER TDM).

Parameter	BOA	BKA	CBOA	GWO	MABOA	mMBOA	PPSO	SABOA	WOA	ACBOA
I_{ph} (A)	0.0888325331	0.1001996409	0.1001877664	0.1002978134	0.0627364177	0.0999378913	0.0999571969	0.1053547413	0.1001307692	0.1001369526
I_{sd1} (A)	1.000000E-12	1.020936E-12	1.000000E-12	1.287949E-11	2.872776E-07	1.898039E-12	1.326421E-11	1.000000E-12	2.286951E-12	1.000000E-12
I_{sd2} (A)	1.000000E-12	9.130281E-12	1.000781E-12	3.163330E-11	1.209796E-07	1.001172E-12	1.000000E-12	1.000000E-12	1.489150E-11	5.834344E-11
I_{sd3} (A)	1.000000E-12	1.659926E-11	1.000000E-12	6.660607E-12	5.654730E-08	2.276784E-12	1.001940E-12	1.000000E-12	3.546900E-12	1.000000E-12
R_s (Ω)	0.2108371660	0.6635028575	0.6666897070	0.6249428422	0.7234554337	0.6576940782	0.6281873344	0.6021766815	0.6320248375	0.6598853781
R_p (Ω)	211.93344611	503.76101185	518.36536083	501.25391605	652.59011771	798.40988693	790.83555725	69.54562505	778.04110741	641.49064876
n_1	1.325638589	1.543800560	1.575188309	1.960278443	2.000000000	1.644364475	1.728989852	1.605666953	1.741638020	1.544807489
n_2	1.423450804	1.999983344	1.600650152	2.000000000	2.000000000	1.682315429	2.000000000	1.605666953	1.740750714	1.972328012
n_3	1.342288120	1.881057077	1.643449811	1.665062486	2.000000000	1.619492049	1.613924789	1.605666953	1.730711959	1.951292234
RMSE	3.672459393	0.000236838	0.000235401	0.000344021	56.418052514	0.000243329	0.000289129	0.003036292	0.000443621	0.000229451
Mean	875.7103923	3.6094097	28.3727755	0.0012862	145.9560745	3.5951887	18.5506200	75.0967131	0.0221125	0.0002501
Std	781.3560002	19.6105184	28.3656492	0.0018067	38.4376611	19.6131925	42.0912541	84.9355225	0.0344604	0.0000122
Worst	2886.283323	107.440059	82.906251	0.006623	212.704153	107.440059	124.173854	322.810855	0.083251	0.000282
Time (s)	4.4597961	4.6557713	4.8381308	5.4795505	7.4413578	5.6720555	5.3977920	5.3184402	5.3324986	4.7556196

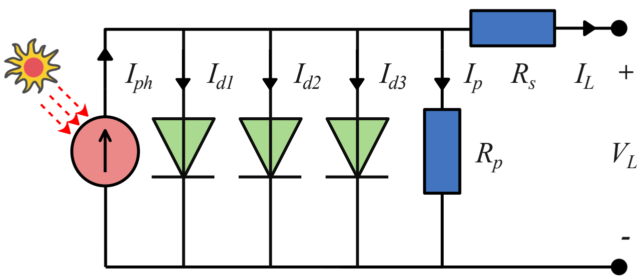
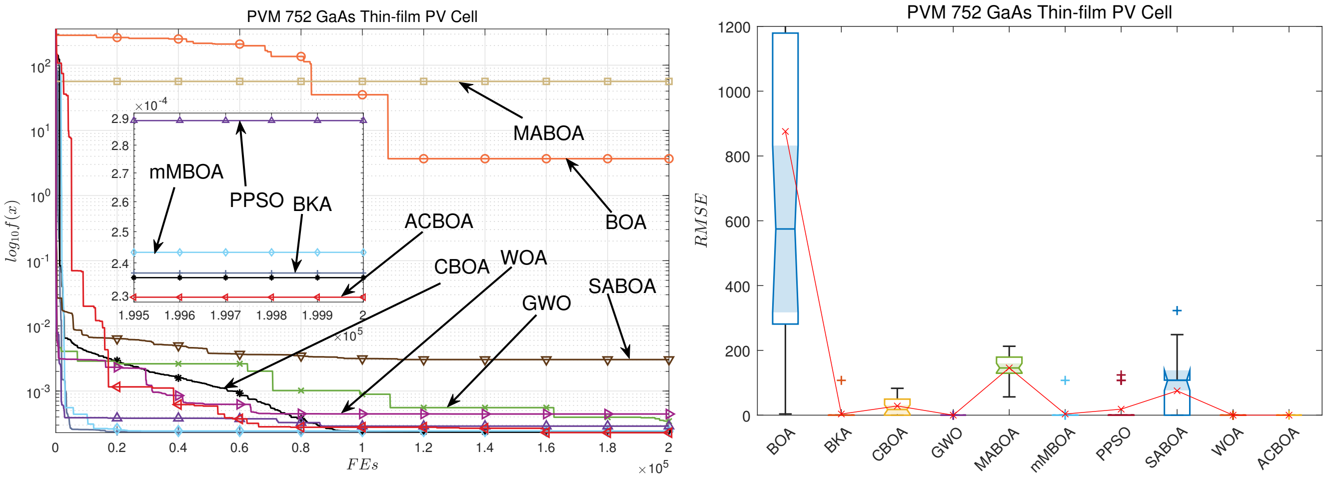


Fig. 10: Equivalent circuit diagram of PV cell model built with TDM.

D. Results analysis and discussion

The aforementioned 3 power engineering problems are common and classical cases, with unknown search spaces and complex constraint environments. This places extremely high demands on the search capabilities of algorithms. Algorithms are required to possess powerful exploration abilities to find feasible solutions that are close to the optimal ones. This is highly challenging and can effectively verify the applicability of algorithms in solving practical engineering problems. The comprehensive results fully demonstrate the superiority of ACBOA.

In terms of parameter extraction of PV cell model, ACBOA achieved the lowest RMSE of 0.000229451 within a short time of 4.7556196s, outperforming several other competitive algorithms. Its excellent stability, reflected by the Std value of 0.0000122, combined with the unique convergence pattern—slow initial convergence but high convergence accuracy in the later stage—highlights the effectiveness of the proposed improvements in escaping from local optimal solutions. Regarding the SOPWM for 3-level inverters, although the time consumption of different algorithms was comparable, ACBOA obtained the best 'Best' value of 0.0319922000, which is closer to the theoretical optimal value of 0.03073936 than those of other algorithms. The Std value of 0.1730051596 and the convergence behavior further confirm its superior performance and stability. In the economic emission dispatch of the IEEE 14-bus system, ACBOA achieved the lowest total economic emission cost of 701.63813866$/h, while also achieving the optimal power allocation for the 5 generators. Whether in terms of economic cost or emission control, as well as in terms of the excellent



(a) Convergence curve

(b) Box plot

Fig. 11: Convergence curve and box plot for applying on PV cell model.

stability indicators, it demonstrated advantages under different evaluation criteria.

These results collectively demonstrate that ACBOA effectively balances the capabilities of exploration and exploitation, making it highly suitable for solving complex, nonlinear, and multimodal optimization problems in power engineering. The performance of this algorithm not only validates the significance of the proposed adaptive and chaos mechanisms but also provides a reliable solution approach for similar power system optimization tasks.

VI. Conclusion

In conclusion, the traditional butterfly optimization algorithm (BOA) has certain limitations in practical applications, which motivated the development of the adaptive chaos BOA (ACBOA) proposed in this study. By adaptively adjusting the fragrance and switch probability in BOA, more precise fragrance perception and search behaviors are achieved. Meanwhile, the introduction of Tent chaos mapping leverages its randomness and ergodicity to expand the exploration and exploitation scope for each butterfly. The extensive numerical experiments conducted using CEC2022 test suite revealed that ACBOA exhibits superior performance compared to 9 other benchmark algorithms, demonstrating its enhanced search efficiency and convergence accuracy. Furthermore, the successful application of ACBOA to 3 distinct power engineering problems, namely the parameter extraction of photovoltaic cell model, synchronous optimal pulse-width modulation (SOPWM) for 3-level inverters, and economic emission dispatch, further validates its effectiveness. In these real-world applications, ACBOA consistently outperformed the comparative algorithms, achieving more optimal solutions and showcasing remarkable stability and adaptability.

Overall, the proposed ACBOA not only overcomes the shortcomings of the traditional BOA, but also demonstrates strong potential in solving complex numerical computation problems and practical engineering applications in the field of power system, providing a promising approach for future research and development in this area. However, despite some limitations existing in this study, the proposed algorithm mainly focuses on static optimization scenarios, and its effectiveness in dynamic and real-time optimization problems remains to be verified. Additionally, when dealing with large-scale and high-dimensional problems in power engineering, the computational complexity of ACBOA may increase significantly.

In future work, we will make efforts to improve the adaptability of ACBOA to dynamic environments. This includes developing real-time optimization strategies and incorporating time-varying factors into the algorithm framework. To address the issue of computational complexity, we will explore advanced parallel computing techniques and lightweight optimization models. Moreover, expanding the application scope of ACBOA to more complex power engineering systems, such as smart grids with a high proportion of renewable energy access, integrated energy systems, and power distribution network reconfiguration, will also be important research directions.

References

[1] A. E. Ezugwu, A. K. Shukla, R. Nath, A. A. Akinyelu, J. O. Agushaka, H. Chiroma, and P. K. Muhuri, "Metaheuristics: A comprehensive overview and classification along with bibliometric analysis," *Artificial Intelligence Review*, vol. 54, no. 6, pp. 4237–4316, 2021.

[2] L. Chen, "Parameter tuning of a PID controller based on the cellular genetic algorithm," *Engineering Letters*, vol. 32, no. 4, pp. 828–834, 2024.

[3] Z. Yu, X. Zhang, and X. Shen, "An improved differential evolutionary algorithm based on simulatedannealing and levy flights mechanism," *Engineering Letters*, vol. 29, no. 2, pp. 697–703, 2021.

[4] X. Ji, Q. Qu, Y.-L. Ren, J.-X. Lian, and T.-R. Jiang, "Improved DBO algorithm incorporating disorienting behavior and dynamic population strategy for engineering problem solving," *Engineering Letters*, vol. 33, no. 1, pp. 90–103, 2025.

[5] G. Xie, M. Zhang, M. Yang, and D. Wang, "Economic dispatch of isolated microgrids based on enhanced sparrow search algorithm," *Engineering Letters*, vol. 32, no. 4, pp. 753–760, 2024.

[6] R. Zhou, Y. Zhang, and K. He, "A novel hybrid binary whale optimization algorithm with chameleon hunting mechanism for wrapper feature selection in QSAR classification model:A drug-induced liver injury case study," *Expert Systems with Applications*, vol. 234, p. 121015, 2023.

[7] Y. Tao, W. Cui, J. Zhao, W. Zhang, and Z. Zhang, “A snake encryption algorithm for image with multiple chaos fusion,” *Engineering Letters*, vol. 30, no. 3, pp. 1034–1043, 2022.

[8] S. Arora and S. Singh, “Butterfly optimization algorithm: A novel approach for global optimization,” *Soft Computing*, vol. 23, no. 3, pp. 715–734, 2019.

[9] S. N. Makhadmeh, M. A. Al-Betar, A. K. Abasi, M. A. Awadallah, I. A. Doush, Z. A. A. Alyasseri, and O. A. Alomari, “Recent advances in butterfly optimization algorithm, its versions and applications,” *Archives of Computational Methods in Engineering*, vol. 30, no. 2, pp. 1399–1420, 2023.

[10] K. He, Y. Zhang, Y.-K. Wang, R.-H. Zhou, and H.-Z. Zhang, “EABOA: Enhanced adaptive butterfly optimization algorithm for numerical optimization and engineering design problems,” *Alexandria Engineering Journal*, vol. 87, pp. 543–573, 2024.

[11] S. Sharma, A. K. Saha, and S. Nama, “An enhanced butterfly optimization algorithm for function optimization,” in *Advances in Intelligent Systems and Computing*, ser. Advances in Intelligent Systems and Computing, 2020, pp. 593–603.

[12] D. M. Utama, D. S. Widodo, M. F. Ibrahim, and S. K. Dewi, “A new hybrid butterfly optimization algorithm for green vehicle routing problem,” *Journal of Advanced Transportation*, vol. 2020, pp. 1–14, 2020.

[13] Y. Fan, J. Shao, G. Sun, and X. Shao, “A self-adaption butterfly optimization algorithm for numerical optimization problems,” *IEEE Access*, vol. 8, pp. 88 026–88 041, 2020.

[14] L. S. Tan, Z. Zainuddin, and P. Ong, “Wavelet neural networks based solutions for elliptic partial differential equations with improved butterfly optimization algorithm training,” *Applied Soft Computing*, vol. 95, p. 106518, 2020.

[15] S. Sharma, S. Chakraborty, A. K. Saha, S. Nama, and S. K. Sahoo, “mLBOA: A modified butterfly optimization algorithm with lagrange interpolation for global optimization,” *Journal of Bionic Engineering*, vol. 19, no. 4, pp. 1161–1176, 2022.

[16] K. Hu, H. Jiang, C.-G. Ji, and Z. Pan, “A modified butterfly optimization algorithm: An adaptive algorithm for global optimization and the support vector machine,” *Expert Systems*, vol. 38, no. 3, p. e12642, 2021.

[17] Y. Li, X. Yu, and J. Liu, “An opposition-based butterfly optimization algorithm with adaptive elite mutation in solving complex high-dimensional optimization problems,” *Mathematics and Computers in Simulation*, vol. 204, pp. 498–528, 2023.

[18] S. Sharma, A. K. Saha, S. Roy, S. Mirjalili, and S. Nama, “A mixed sine cosine butterfly optimization algorithm for global optimization and its application,” *Cluster Computing*, vol. 25, no. 6, pp. 4573–4600, 2022.

[19] L. Yao, P. Yuan, C.-Y. Tsai, T. Zhang, Y. Lu, and S. Ding, “ESO: An enhanced snake optimizer for real-world engineering problems,” *Expert Systems with Applications*, vol. 230, p. 120594, 2023.

[20] A. Kumar, K. V. Price, A. W. Mohamed, and A. A. Hadi, “Problem definitions and evaluation criteria for the CEC 2022 special session and competition on single objective bound constrained numerical optimization,” IEEE Congress on Evolutionary Computation (CEC), Nanyang Technological University, Singapore, Tech. Rep., 2021.

[21] J. Wang, W.-c. Wang, X.-x. Hu, L. Qiu, and H.-f. Zang, “Black-winged kite algorithm: A nature-inspired meta-heuristic for solving benchmark functions and engineering problems,” *Artificial Intelligence Review*, vol. 57, no. 4, p. 98, 2024.

[22] E. Trojovská and M. Dehghani, “A new human-based metahurestic optimization method based on mimicking cooking training,” *Scientific Reports*, vol. 12, no. 1, p. 14861, 2022.

[23] S. Mirjalili, S. M. Mirjalili, and A. Lewis, “Grey wolf optimizer,” *Advances in Engineering Software*, vol. 69, no. 3, p. 46–61, 2014.

[24] S. Sharma and A. K. Saha, “m-MBOA: A novel butterfly optimization algorithm enhanced with mutualism scheme,” *Soft Computing*, vol. 24, no. 7, pp. 4809–4827, 2020.

[25] M. Ghasemi, E. Akbari, A. Rahimnejad, S. E. Razavi, S. Ghavidel, and L. Li, “Phasor particle swarm optimization: A simple and efficient variant of PSO,” *Soft Computing*, vol. 23, no. 19, pp. 9701–9718, 2019.

[26] Mirjalili, Seyedali, Lewis, and Andrew, “The whale optimization algorithm,” *Advances in engineering software*, vol. 95, pp. 51–67, 2016.

[27] L. Muthuramalingam, K. Chandrasekaran, and F. J. Xavier, “Electrical parameter computation of various photovoltaic models using an enhanced jumping spider optimization with chaotic drifts,” *Journal of Computational Electronics*, vol. 21, pp. 905–941, 2022.

[28] A. K. Rathore, J. Holtz, and T. Boller, “Synchronous optimal pulsewidth modulation for low-switching-frequency control of medium-voltage multilevel inverters,” *IEEE Transactions on Industrial Electronics*, vol. 57, no. 7, pp. 2374–2381, 2010.

[29] K. He, Y. Zhang, Y. Wang, and R. Zhou, “Solving power system economic emission dispatch problem under complex constraints via dimension differential learn butterfly optimization algorithm with FDC-based,” *Computers & Industrial Engineering*, vol. 197, p. 110568, 2024.

[30] X. Liu and T. Zhang, “Analysis of IEEE 14 bus power flow calculation characteristics based on newton-raphson method,” in *2022 IEEE 5th International Conference on Automation, Electronics and Electrical Engineering (AUTEEE)*, 2022, pp. 1092–1097.

[31] S. P. Karthikeyan, K. Palanisamy, C. Rani, I. J. Raglend, and D. P. Kothari, “Security constrained unit commitment problem with operational, power flow and environmental constraints,” *WSEAS Transactions on Power Systems*, vol. 4, no. 2, pp. 53–66, 2009.