Evolutionary Intelligence
https://doi.org/10.1007/s12065-023-00892-6

RESEARCH PAPER

# Chaotic vegetation evolution: leveraging multiple seeding strategies and a mutation module for global optimization problems

**Rui Zhong**[1] · **Chao Zhang**[2] · **Jun Yu**[3]

Received: 3 October 2023 / Revised: 30 October 2023 / Accepted: 18 November 2023


**Abstract**
This paper focuses on improving the overall performance of the vegetation evolution (VEGE) algorithm and proposes a chaotic VEGE with multiple seeding strategies and a mutation module (CVEGE). While the original VEGE exhibits robust exploitation capabilities, it falls short in terms of exploration and overcoming local optima. Thus, we introduce the chaotic local search operators, multiple seed dispersion strategies, and a unique mutation module to address these mentioned limitations. Furthermore, we incorporate a simplified sigmoid transfer function into CVEGE and propose a binary variant known as binary chaotic vegetation evolution (BCVEGE). In numerical experiments, we evaluate CVEGE on 10-D, 30-D, 50-D, and 100-D CEC2020 benchmark functions, as well as four engineering optimization problems. Additionally, BCVEGE is subjected to testing on two combinatorial optimization problems: wrapper-based feature selection tasks and classic 0/1 knapsack problems. Here, we employ two classic algorithms (i.e. differential evolution and particle swarm optimization) and seven state-of-the-art competitor algorithms including the original VEGE as the competitor algorithms. The sufficient numerical experiments and statistical analysis practically show that our proposal: CVEGE and BCVEGE, are competitive with compared algorithms. Furthermore, the demonstrated performance and scalability of CVEGE and BCVEGE suggest their potential utility across a wide range of optimization tasks.

**Keywords** Vegetation evolution (VEGE) · Multiple seeding strategies · Mutation module · Chaotic local search · Feature selection · 0/1 knapsack problem

## 1 Introduction

Evolutionary computation (EC) has achieved unprecedented development in the past decades, numerous evolutionary algorithms (EAs) spring up like mushrooms to tackle with continuous problems [1–3], address real-world complex optimization challenges [4–7], feature selection [8–10] and neural architecture search [11–13]). As a kind of stochastic optimization methodology designed for global optimization, EAs iteratively apply search operators, often inspired by natural phenomena or biological behaviors, to explore potential solutions and converge towards acceptable ones. Here, we list a collection of representative EAs in Table 1 and classify them into six categories: swarm-inspired, evolutionary-based, human-inspired, math-inspired, physics-inspired, and chemistry-inspired.

As one of the most recent EAs, vegetation evolution (VEGE) draws inspiration from the simplified growth and reproduction mechanisms observed in real plants. Specifically, VEGE simulates the dual phases of the plant life cycle: the growth period and the maturity period, forming the foundation of optimization strategy. During the growth period, the plant attempts to extend its roots with minor perturbations to absorb additional nutrients, and VEGE employs a similar strategy to realize the exploitation operator, where the algorithm concentrates its efforts on searching around promising areas. During the maturity period, the matured

✉ Jun Yu
yujun@ie.niigata-u.ac.jp

Rui Zhong
rui.zhong.u5@elms.hokudai.ac.jp

Chao Zhang
zhang@u-fukui.ac.jp

[1] Graduate School of Information Science and Technology, Hokkaido University, Sapporo, Japan

[2] Department of Engineering, University of Fukui, Fukui, Japan

[3] Institute of Science and Technology, Niigata University, Niigata, Japan

Published online: 14 January 2024

**Table 1** Representative EAs in literature

| Category | Algorithm | Inspiration |
|---|---|---|
| Swarm-inspired | Particle swarm optimization (PSO) [14] | Movement of organisms in a bird flock |
| | Grey wolf optimizer (GWO) [1] | Leadership hierarchy and hunting mechanism of grey wolves |
| | Whale optimization algorithm (WOA) [2] | Social behavior of humpback whales |
| | Slime mould algorithm (SMA) [15] | Diffusion and foraging conduct of slime mould |
| Evolutionary-based | Genetic algorithm (GA) [16] | Evolutionary concepts |
| | Differential evolution (DE) [17] | Darwin's theory of evolution |
| | Genetic programming (GP) [18] | Biological evolution |
| | Evolutionary programming (EP) [19] | Finite state machine |
| Human-inspired | Human-inspired algorithm (HIA) [20] | Intelligent search strategies of mountain climbers |
| | Teaching-learning-based optimization (TLBO) [21] | Effect of influence of a teacher on learners |
| | Multi leader optimizer (MLO) [22] | Process of advancing members of the population |
| | Sewing training-based optimization (STBO) [23] | Teaching the process of sewing to beginner tailors |
| Math-inspired | Sine cosine algorithm (SCA) [24] | Proprieties of sine and cosine functions |
| | Chaos game optimization (CGO) [25] | Principles of chaos theory |
| | Pareto-like sequential sampling (PSS) [26] | Pareto's principle |
| | Runge Kutta optimization (RUN) [27] | Runge Kutta method |
| Physics-inspired | Gravitational search algorithm (GSA) [28] | Law of gravity and mass interactions |
| | Water cycle algorithm (WCA) [29] | Water cycle process in nature |
| | Spring search algorithm (SSA) [30] | The tensile force of spring and Hooke's law |
| | Momentum search algorithm (MSA) [31] | Newton's second law of motion |
| Chemistry-inspired | Chemical reaction optimization (CRO) [32] | Molecules in a chemical reaction |
| | Artificial chemical reaction optimization algorithm (ACROA) [33] | Types and occurring of chemical reactions |
| | Material generation algorithm (MGA) [34] | Chemical compounds and chemical reactions |

plants generate seeds and disperse them over a wide area. This process is designed to represent the exploration phase, where the algorithm is inclined to search unknown areas. The fundamental operational sequence of VEGE alternates between these two phases—growth and maturity—to facilitate convergence, and the detailed elaboration of VEGE is provided in Sect. 2.1.

The original VEGE has demonstrated its superiority when compared to well-established algorithms such as differential evolution (DE) [17], particle swarm optimization (PSO) [14], and the enhanced fireworks algorithm (EFWA) [35] in solving CEC2013 benchmark functions [3]. As a result, it has garnered widespread attention and has led to the development of several enhanced versions: Yu et al. [36] designed a mutation strategy and introduced an elite-based growth strategy to enhance the performance of standard VEGE. Yu et al. [37] accelerate the convergence of VEGE by simulating the sexual and the asexual reproduction of plants. Furthermore, Zhong et al. [38] proposed a dynamic seeding resource allocation mechanism and unique mutation module to enhance the optimization capacity of the conventional VEGE. Despite pieces of work that highlight VEGE's effectiveness across various dimensions, certain challenges remain unresolved. These include limitations in its exploration capacity, susceptibility to local optima, and the potential for premature convergence. Furthermore, its applicability is also constrained in specific scenarios, and concerns exist regarding its scalability when dealing with complex real-world optimization problems.

Addressing these issues is crucial for enhancing the overall effectiveness and scalability of VEGE. Therefore, this research paper has been devised with two primary objectives. The first objective is to introduce the chaotic local search technique, multiple seed dispersion strategies, and a unique mutation module to improve the overall optimization performance of VEGE, and we name our proposal CVEGE briefly. To evaluate the practical performance of CVEGE, we conduct comprehensive numerical experiments on 10-D, 30-D, 50-D, and 100-D CEC2020 benchmark functions and compare with nine classic or state-of-the-art EAs to analyze the performance of CVEGE when facing the problem with various scales. Ablation experiments are also conducted to investigate the contribution of the proposed three strategies independently. Moreover, the numerical experiments on four engineering optimization problems serve to reflect CVEGE's capability in addressing real-world challenges.

The second objective of this study extends the capabilities of CVEGE to encompass discrete optimization problems.

Our focus is on two specific tasks: the wrapper-based feature selection task and the 0/1 knapsack problems. To tackle these tasks, the simplified sigmoid transfer function [39] is embedded into CVEGE. As a result, a binary variant of CVEGE, referred to as BCVEGE, is proposed. In numerical experiments, BCVEGE undergoes rigorous testing on eight popular classification datasets and ten classic knapsack benchmark problems. To ensure a fair comparison, the compared EAs are equipped with the same simplified sigmoid transfer function. Thus, the main contributions of this paper are summarized as follows:

- We propose CVEGE as a solution for continuous optimization. During the growth period, we replace the random seed generator in the simple local search with chaotic maps, a modification that aligns better with practical applications of EAs [40]. In the maturity period, we incorporate an extra seed dispersion strategy and a unique mutation module into CVEGE. These two enhancements are aimed at bolstering the exploration capacity of CVEGE and equipping it with the ability to escape from local optima.
- We integrate the simplified sigmoid transfer function into CVEGE, enabling CVEGE to efficiently address optimization problems in discrete space.
- We conduct comprehensive numerical experiments to evaluate the performance of our proposed CVEGE and BCVEGE. We compare them against nine classic or advanced EAs as competitors. CEC2020 benchmark functions and four engineer optimization are employed to evaluate the CVEGE, while eight classification datasets and ten 0/1 knapsack benchmark problems are adopted to evaluate the BCVEGE. The experimental results, supported by statistical analyses, demonstrate that our proposals are competitive in solving both continuous and discrete optimization problems.

The remainder of this paper is organized as follows: Sect. 2 discusses the related works. Section 3 provides a detailed introduction to our proposal: CVEGE and BCVEGE. Section 4 covers numerical experiments and statistical results of the optimization. Section 5 analyzes the performance of CVEGE and BCVEGE and highlights some open topics for future research. Finally, Sect. 6 concludes this paper.

## 2 Related works

### 2.1 Vegetation evolution (VEGE)

The original VEGE simulates the simplified life cycle observed in natural plants to realize optimization. During the growth period, although modeling the precise growth mechanism of plants can be challenging, VEGE adopts two parameters, namely growth radius (*GR*) and growth direction (*GD*), to formulate the local search operator. This operator is designed to represent the growth behaviors of plant individuals as they extend their roots and branches to absorb more nutrients for survival, and it is expressed in Eq. (1).

$$X_{i,j}^{t+1} = X_{i,j}^{t} + GR \cdot GD \tag{1}$$

where $i$, $j$, $t$ in $X_{i,j}^{t}$ denote the $i$th individual, the $j$th dimension, and the $t$th iteration. $GR$ is a constant, and $GD$ is a random number in $[-1, 1]$. Within the growth cycle ($GC$) of the growth period, VEGE iteratively applies Eq. (1) to facilitate the exploitation. When a superior offspring is found, the new solution typically replaces the parent and survives to the next generation.

In nature, matured plants generate hundreds or even thousands of seeds, rather than a few, to increase the chances of offspring survival. Inspired by this natural phenomenon, VEGE adopts a one-to-many strategy in collaboration with the DE/cur/1-like mutation operator to facilitate the exploration. Equation (2) presents the formulation of this behavior.

$$X_{i}^{t+1} = X_{i}^{t} + MS \cdot (X_{r_1}^{t} - X_{r_2}^{t}) \tag{2}$$

$X_{r_1}^{t}$ and $X_{r_2}^{t}$ are two randomly sampled individual which are mutually different with $X_{i}^{t}$. $MS$ is a random moving factor to describe the uncertainty in seed dispersion such as water flow, wind, animals, and other factors. After the whole seeds are evaluated, VEGE adopts the top-$k$ selection scheme to select the best $k$ individuals to survive to the next generation.

### 2.2 Chaotic maps and applications

Chaos, originating from non-linear dynamic systems, is considered of deterministic random method [41]. Chaos theory states that beneath the apparent randomness of chaotic complex systems, there exist underlying patterns, interconnection, constant feedback loops, repetition, self-similarity, fractals, and self-organization [42]. In recent advances, chaotic maps, as alternative techniques, have been widely applied to replace the random number generator for tasks like population initialization and search operators. For instance, Kohli et al. [41] adopted ten chaotic maps to fine-tune the crucial parameter $\alpha$ in the grey wolf optimizer (GWO). Instead of the random number generator, the introduction of chaotic maps can significantly accelerate the convergence of optimization processes, particularly in constrained problems and engineering problems. Similarly, Kaur et al. [43] extended the chaotic techniques to determine the probability parameter $p$ in the whale optimization algorithm (WOA) and proposed a chaotic WOA (CWOA). Yuzgec et al. [44] proposed a chaotic-based

differential evolution (CDE), where chaotic maps were employed to initialize the population. CDE exhibited competitive performance on CEC benchmark functions and a real-world optimization problem concerning the baker's yeast producer in Turkey. Li et al. [45] proposed a whale optimization algorithm with chaos strategy and weight factor (WOACW), incorporating a chaotic map for initial population generation and further enhancing the diversity. Gao et al. [40] introduced multiple chaotic maps into the local search operator independently, randomly, in parallel, and memory-selectively. This novel local search technique was embedded into adaptive differential evolution with optional external archive (JADE) [46] and performed competitiveness with JADE and other state-of-the-art optimization algorithms. Besides, some meta-heuristic algorithms have drawn inspiration from the characteristics of chaos, such as the chaos optimization algorithm (COA) [47] and the chaotic evolution (CE) [48].

### 2.3 Transfer functions

Many approaches can transfer the continuous optimization algorithm into discrete space, such as the permutation matrix approach [49], relative position indexing [50], and transfer functions [39, 51]. Among these, the transfer function is one of the easiest to implement and simplest methods for mapping real-number encoding to binary encoding. Numerous functions can serve as transfer functions, including S-shaped transfer functions [52], V-shaped transfer functions [53], X-shaped transfer functions [54], U-shaped transfer functions [55], Z-shaped transfer functions [56], and New V-shaped transfer functions [57]. First, the mechanism of transfer functions is described in Eq. (3).

$$bX_i^j = \begin{cases} 0, \ if \ F(X_i^j) < r \\ 1, \ otherwise \end{cases} \tag{3}$$

where $X_i^j$ and $bX_i^j$ are the real-number encoding and binary encoding of the $j^{th}$ dimension of the $i^{th}$ individual, respectively, $F(\cdot)$ is a transfer function, and $r$ is a random number in the range of (0, 1). Next, we will provide detailed introductions to various transfer functions.

*S-shaped transfer functions*: Eq. (4) lists four common S-shaped transfer functions.

$$\begin{aligned} S1(x) &= \frac{1}{1+e^{-2x}} \\ S2(x) &= \frac{1}{1+e^{-x}} \\ S3(x) &= \frac{1}{1+e^{-x/2}} \\ S4(x) &= \frac{1}{1+e^{-x/3}} \end{aligned} \tag{4}$$

Figure 1a visualizes these transfer functions.

*V-shaped transfer functions*: Eq. (5) lists four common V-shaped transfer functions.

$$\begin{aligned} V1(x) &= |\tanh(x)| \\ V2(x) &= \left| erf(\frac{\sqrt{\pi}}{2}x) \right| = \left| \frac{\sqrt{2}}{\pi} \int_0^{\frac{\sqrt{\pi}}{2}x} e^{-t^2} dt \right| \\ V3(x) &= \left| \frac{x}{\sqrt{1+x^2}} \right| \\ V4(x) &= \left| \frac{2}{\pi} \arctan(\frac{\pi}{2}x) \right| \end{aligned} \tag{5}$$

Figure 1b visualizes V-shaped transfer functions.

*X-shaped transfer functions*: Eq. (6) lists two common X-shaped transfer functions.

$$\begin{aligned} X1(x) &= \frac{1}{1+e^{x}} \\ X2(x) &= \frac{1}{1+e^{-x}} \end{aligned} \tag{6}$$

Figure 1c visualizes X-shaped transfer functions.

*U-shaped transfer functions*: Eq. (7) lists three common U-shaped transfer functions.

$$\begin{aligned} U1(x) &= \alpha|x^{\beta}|, \ \alpha = 1, \ \beta = 2 \\ U2(x) &= \alpha|x^{\beta}|, \ \alpha = 1, \ \beta = 3 \\ U3(x) &= \alpha|x^{\beta}|, \ \alpha = 1, \ \beta = 4 \end{aligned} \tag{7}$$

Figure 1d visualizes U-shaped transfer functions.

*Z-shaped transfer functions*: Eq. (8) lists four common Z-shaped transfer functions.

$$\begin{aligned} Z1(x) &= \begin{cases} \sqrt{1-2^x}, \ if \ x < 0 \\ 0, \ otherwise \end{cases} \\ Z2(x) &= \begin{cases} \sqrt{1-5^x}, \ if \ x < 0 \\ 0, \ otherwise \end{cases} \\ Z3(x) &= \begin{cases} \sqrt{1-8^x}, \ if \ x < 0 \\ 0, \ otherwise \end{cases} \\ Z4(x) &= \begin{cases} \sqrt{1-20^x}, \ if \ x < 0 \\ 0, \ otherwise \end{cases} \end{aligned} \tag{8}$$

Figure 1e visualizes Z-shaped transfer functions.

*New V-shaped transfer functions*: Eq. (9) lists four common New V-shaped transfer functions.

$$\begin{aligned} New \ V1(x) &= \sqrt{1-2^{-|x|}} \\ New \ V2(x) &= \sqrt{1-5^{-|x|}} \\ New \ V3(x) &= \sqrt{1-8^{-|x|}} \\ New \ V4(x) &= \sqrt{1-20^{-|x|}} \end{aligned} \tag{9}$$

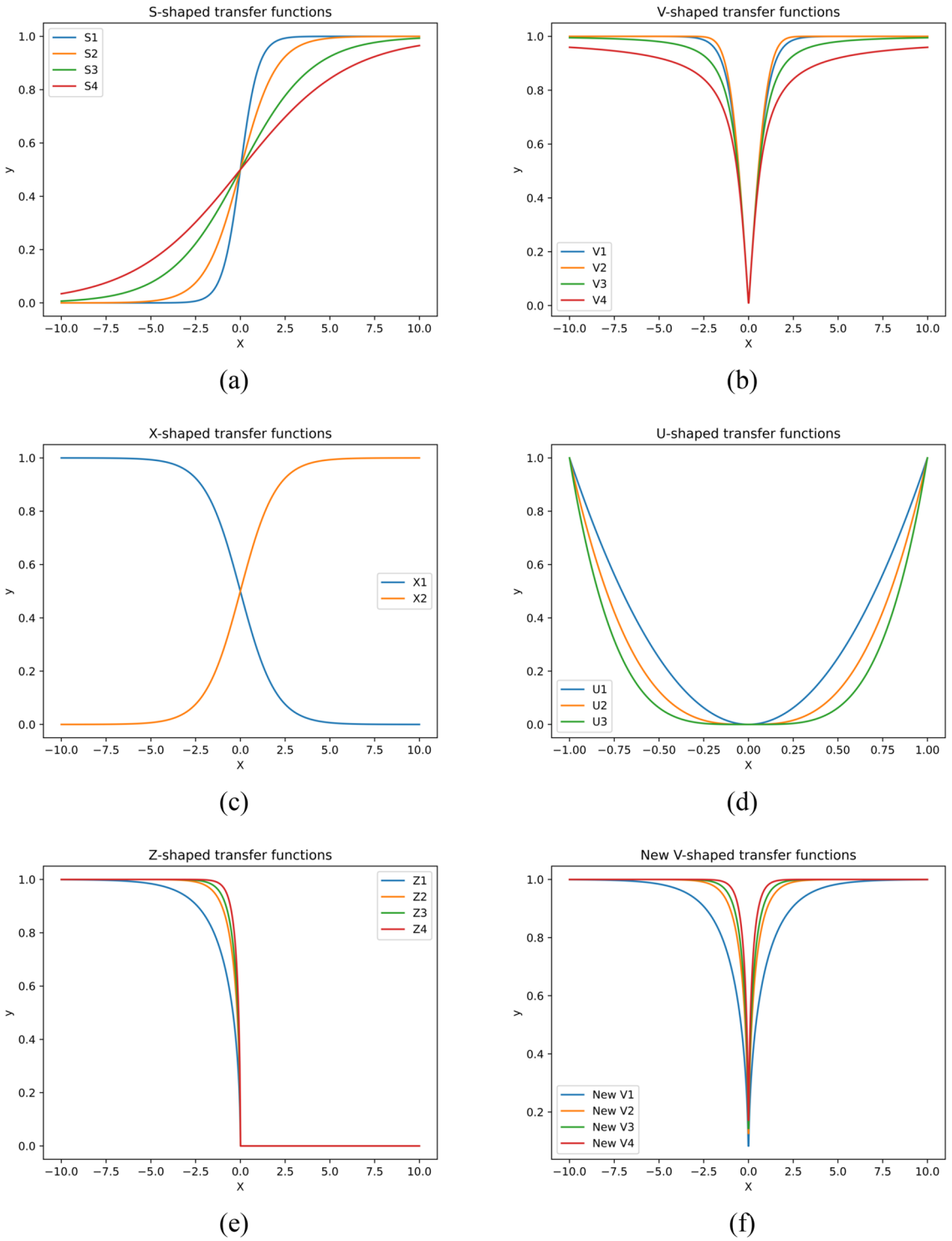


**Fig. 1** Various transfer functions and corresponding curves. **a** S-shaped transfer functions. **b** V-shaped transfer functions. **c** X-shaped transfer functions. **d** U-shaped transfer functions (**e**) Z-shaped transfer functions (**f**). New V-shaped transfer functions

Figure 1f visualizes Z-shaped transfer functions.

## 2.4 Wrapper-based feature selection

Feature selection holds significant importance in the fields of pattern recognition and machine learning [51, 58]. Feature

selection methodologies can be broadly categorized into three main categories: filter methods, embedded methods, and wrapper-based methods [59]. In this research, our primary focus is on wrapper-based methods, and the main framework is demonstrated in Fig. 2.

The essence of the wrapper-based feature selection task is the combinatorial optimization problem, where the goal is to select the most suitable subset for the specific task and the model construction. Supposing the number of features is $N$, and the possible cases of feature combination are $2^N - 1$. Because performing an exhaustive search of all possible combinations is an NP-hard problem, the meta-heuristic algorithm has become a popular approach for tackling the wrapper-based feature selection task.

### 2.5 0/1 knapsack problems

The 0/1 knapsack problem is a classic combinatorial optimization problem that has been formally proven to be an NP-complete problem [61]. This problem can be described as follows: There are $N$ items, with each item represented by $x_i$, having a weight ($w_i$) and a price ($p_i$). The objective of this problem is to select a subset of these items while adhering to the constraint of not exceeding the capacity of knapsack ($C$), in order to maximize the total value of the selected items [62]. Equation (10) describes the mathematical model of the knapsack problem.

$$\begin{aligned} &\text{maximize} \sum_{i=1}^{N} p_i \cdot x_i \\ &\text{subject to} \sum_{i=1}^{N} w_i \cdot x_i \leq C \\ &\text{where } x_i \in \{0, 1\} \end{aligned} \tag{10}$$

where $x$ is a $N$-dimensional vector, which represents a trial solution for the 0/1 Knapsack problem. Besides, many real-world applications can be regarded as variants of knapsack problems, such as the selection of investments and portfolios [63], water resource engineering and flood management [64], generating keys for the Merkle-Hellman [63], among others.

## 3 Chaotic VEGE and its binary variant

This section provides a detailed introduction to our proposed CVEGE and BCVEGE. Section 3.1 offers an overview of CVEGE, and Sect. 3.2 elaborates on the techniques integrated into CVEGE. Finally, Sect. 3.3 delves into the adaptations made to BCVEGE for discrete optimization problems.

### 3.1 The overview of CVEGE

Figure 3 shows the main structure of CVEGE. We build upon the framework of VEGE, which optimizes by simulating the growth period and maturity period iteratively. The enhancements in CVEGE, including chaotic local search, multiple seeding strategies, and a unique mutation module, will be introduced in the subsequent sections.

### 3.2 Novel techniques in CVEGE

#### 3.2.1 Chaotic local search in CVEGE

In VEGE, the local search operator, as defined in Eq. (1), simulates the growth of plant individuals during the growth period. To enhance the exploitation capacity and expedite the convergence of optimization, this operator can be replaced with a chaotic local search. Here, we employ eight chaos maps to generate perturbations, and these maps are listed in Table 2.

These chaos maps have an equal probability of being selected to generate the chaotic sequence, and we can express the chaotic local search in CVEGE using Eq. (11).

$$X_{i,j}^{t+1} = X_{i,j}^{t} + GR \cdot \delta^{j} \tag{11}$$

where $\delta$ is a perturbation vector generated by a randomly selected chaos map and has an identical dimension size with the solution $X_i^t$.

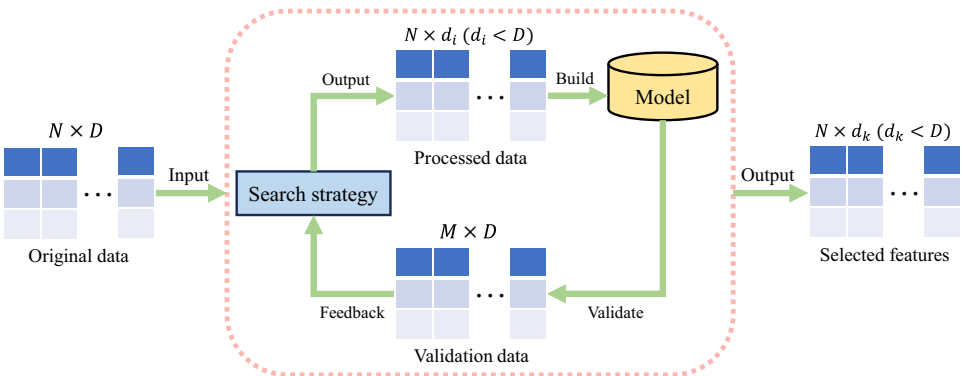


**Fig. 2** The framework of wrapper-based methods for feature selection [60]

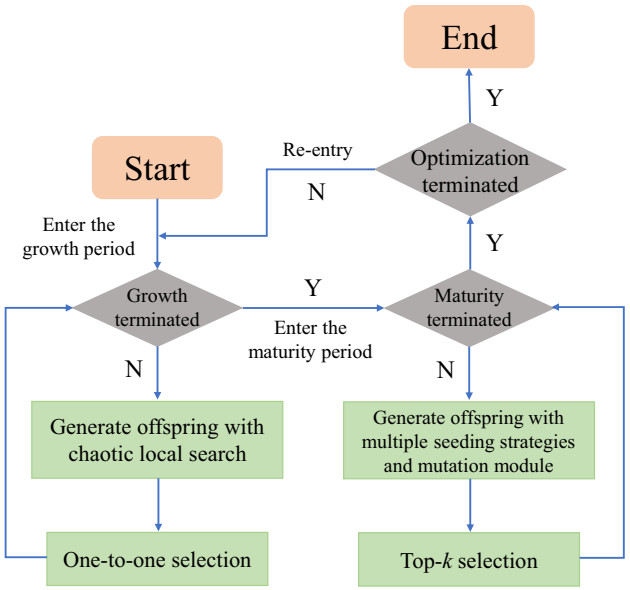


**Fig. 3** The framework of CVEGE

#### 3.2.2 Multiple seeding strategies in CVEGE

Exploration is an important aspect of optimization algorithm design, and a deficiency in exploration capability can restrict the algorithm from venturing into unknown areas. Thus, achieving a balanced trade-off between exploration and exploitation during the optimization process is vital. Since the original VEGE excels in exploitation but falls short in the exploration aspect, thus, we introduce an additional potent seeding strategy in the maturity period. This strategy aims to bolster exploration by enhancing the diversity of the search approach. Equation (12) outlines the additional seeding strategy in CVEGE.

$$X_{i,j}^{t+1} = \text{Gaussian}(X_{i,j}^{t}, \tau) + MS \cdot (X_{best,j} - X_{i,j}^{t}) \tag{12}$$

Gaussian($\mu$, $\sigma$) represents a random number sampled from a Gaussian distribution with an expectation is $\mu$ and the standard deviation is $\sigma$. The parameter $\tau = \log(t) \cdot (X_{i,j}^{t} - X_{best,j})/t$ is an adaptive parameter, where $t$ is the iteration time. The variable $MS$ remains the same as in the original VEGE and is used to describe the uncertainty in the seeding process. This search strategy draws inspiration from the seeding behaviors of dandelions, which is shown in Fig. 4.

Figure 4a illustrates the natural seeding behaviors of dandelions, which we abstract and represent in Fig. 4b. Each larger point in Fig. 4b represents a maternal dandelion plant, while the smaller points are the seeds generated by these dandelions. In the seeding behaviors of dandelions, the seeds are widely dispersed a the center of the maternal plant. To model this phenomenon, we adopt the Gaussian distribution, which is represented by the term Gaussian($X_{i,j}^{t}$, $\tau$) in the Eq. (12). Additionally, other environmental factors like strong wind, rain, and animal behaviors are captured by the $MS \cdot (X_{best,j} - X_{i,j}^{t})$, which introduces uncertainty into the seeding process.

#### 3.2.3 Mutation module in CVEGE

Another challenge stemming from the unbalanced exploration and exploitation abilities is the difficulty in escaping local optima once the optimization process becomes trapped in them. To address this and equip VEGE with the capability to escape local optima, we introduce a unique mutation module into CVEGE. This module combines three representative mutation strategies, and the mathematical models of these mutation operators are depicted in Eq. (13).

$$X_{i,j}^{t+1} = \begin{cases} X_{i,j}^{t} + \gamma \cdot N(0,1) \cdot (UB^{j} - LB^{j}), & r_1 < Mr_1 \\ X_{i,j}^{t}, & otherwise \end{cases} \tag{13a}$$

**Table 2** Eight chaos maps adopted in CVEGE

| Chaos maps | Equation | Parameters |
|---|---|---|
| Chebyshev map | $x_{n+1} = \cos(n/\cos(x_n))$ | |
| Circle map | $x_{n+1} = (x_n + b - a/2\pi \sin(2\pi x_n))mod(1)$ | $a$=0.5 and $b$=2 |
| Gaussian map | $x_{n+1} = \begin{cases} 0, & x_n = 0 \\ 1/(x_n mod(1)), & otherwise \end{cases}$ | |
| Iterative map | $x_{n+1} = \sin(a\pi/x_n)$ | $a \in (0,1)$ |
| Logistic map | $x_{n+1} = ax_n(1-x_n)$ | $a$=4 and $x_0 \neq \{0.25, 0.5, 0.75\}$ |
| Sawtooth map | $x_{n+1} = 2x_n mod(1)$ | |
| Sine map | $x_{n+1} = a/4 \sin(\pi x_n)$ | $a \in (0,4]$ |
| Tent map | $x_{n+1} = \begin{cases} x_n/0.7, & x_n < 0.7 \\ 10(1-x_n)/3, & x_n \geq 0.7 \end{cases}$ | |

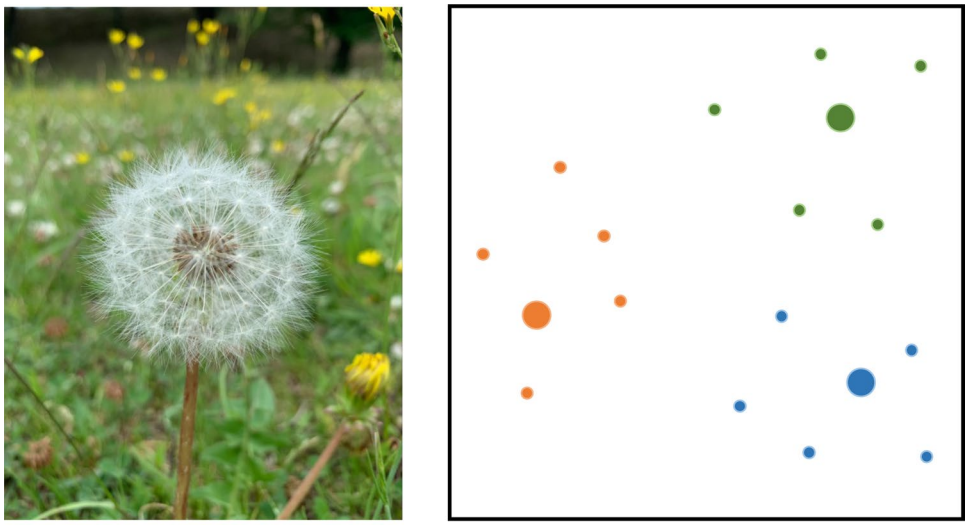


**Fig. 4** The seeding behaviors of dandelions. **a** In nature. **b** In abstract

$$X_{i,j}^{t+1} = \begin{cases} X_{i,j}^{t}, & r_1 < Mr_2 \\ X_{i,j}^{t}, & otherwise \end{cases} \tag{13b}$$

$$X_{i,j}^{t+1} = \begin{cases} X_{i,j}^{t} + (-1)^{r_2} + \textbf{Lévy}(), & r_1 < Mr_3 \\ X_{i,j}^{t}, & otherwise \end{cases} \tag{13c}$$

where $\gamma$ is a scaling factor, $N(0, 1)$ is a random number generated from a standard normal distribution, $UB$ and $LB$ are the upper bound and the lower bound of the search space respectively, $Mr_i$ is the mutation rate for the corresponding strategy, $r_1$ is a random number in (0, 1), while $r_2$ is randomly sampled from {0, 1}, and **Lévy**() represents the Lévy flight function. Algorithm 1 provides the pseudocode for this hybrid mutation strategy.

**Algorithm 1 Mutation strategies**

**Require:** Individual: $X$, Dimension: $D$
**Ensure:** Offspring: $O$
$r \leftarrow$ **rand**()
**if** $r < 1/3$ **then**
    Mutate the individual $X$ with Eq. (13a)
**else if** $r < 2/3$ **then**
    Mutate the individual $X$ with Eq. (13b)
**else**
    Mutate the individual $X$ with Eq. (13c)
**end if**
**return** $O$

This mixed mutation module significantly enhances the ability to escape from local optima and prevent premature convergence. To provide a comprehensive overview, the pseudocode of the complete CVEGE is shown in Algorithm 2.

**Algorithm 2 CVEGE**

**Require:** Population size: $N$, Dimension: $D$, Maximum iteration: $T$
**Ensure:** Offspring: $O$
$P \leftarrow$ **initialPop**$(N, D)$
$O \leftarrow$ **best**$(P)$
$t \leftarrow 0$
**while** $t < T$ **do**
    **while** Growth period not terminated **do** # Growth period
        Update each individual by Eq. (11)
        One-to-one selection
        $O \leftarrow$ **best**$(P)$
    **end while**
    **while** Maturity period not terminated **do** # Maturity period
        **if rand**() $< 0.5$ **then**
            Generate seeds by Eq. (2)
        **else**
            Generate seeds by Eq. (12)
        **end if**
        Mutate seeds by Algorithm 1
        Top-$k$ selection
        $O \leftarrow$ **best**$(P)$
    **end while**
**end while**
**return** $O$

### 3.3 Binary CVEGE (BCVEGE)

This section provides a detailed introduction to the binary variant of CVEGE, known as BCVEGE. From the explanation of transfer functions in Sect. 2.3, randomness exists in the encoding schemes. This means that an identical solution transformed by a specific transfer function might result in different binary solutions. To address this, we simplify the mapping principle of the transfer function to ensure the uniqueness of the mapping, and the simplified sigmoid transfer function is defined in Eq. (14).

$$bX_{i,j}^{t} = \begin{cases} 0, \ if \ \frac{1}{1+e^{-X_{i,j}^{t}}} < \varepsilon \\ 1, \ otherwise \end{cases} \tag{14}$$

where $\varepsilon$ represents the truncation threshold, which can be used to control the sparsity of binary solutions. Here, we incorporate the simplified transfer function into CVEGE to create its binary counterpart, which we refer to as BCVEGE.

The application of BCVEGE to both the feature selection task and the 0/1 knapsack problem is explained in the following sections. In wrapper-based feature selection, defining the objective function can be challenging because it requires balancing the accuracy of the predictive model and the scale of the feature subset simultaneously. Thus, some researchers regard the wrapper-based feature selection task as a multi-objective optimization problem [65, 66]. However, in this paper, our focus is on the single-objective feature selection task, and the objective function defined in [67] is employed, which is formulated in Eq. (15)

$$\text{Fit} = \alpha\gamma_M(D) + \beta\frac{|D|}{|C|} \tag{15}$$

where $\gamma_M(D)$ is the classification error rate given model $M$ and the selected feature subset $D$. $|D|$ is the scale of the feature subset, while $|C|$ is the scale of all features. $\alpha$ and $\beta$ are two hyper-parameters to balance the importance of each component, which are typically set to 0.99 and 0.01.

In the wrapper-based feature selection task, the choice of a classification model is crucial. For this purpose, we employ the K-Nearest neighbor (KNN) [68], which is a straightforward and efficient machine learning method commonly used to assess the quality of the selected feature subset. Here, we use the KNN ($K$=5) as the classifier and $k$-fold cross-validation ($k$=10) is incorporated during the training of KNN to alleviate the randomness in the experiments.

The 0/1 knapsack problem is an easy-implemented task to evaluate the performance of binary optimization algorithms. The primary distinction in this task is that the 0/1 knapsack problem is a maximization problem. However, it can be conveniently transformed into a minimization problem using Eq. (16).

$$\max(f(x)) \Leftrightarrow \min(-f(x)) \tag{16}$$

## 4 Numerical experiments

This section implements comprehensive numerical experiments to evaluate the performance of our proposed CVEGE and BCVEGE. Section 4.1 provides an overview of the experimental settings, including details about the experimental environments, benchmark functions, and methods used for comparison, along with their respective parameters. Section 4.2 presents the experimental outcomes and statistical results.

### 4.1 Experiment settings

#### 4.1.1 Experimental environments and implementation

All the optimization techniques described in this research were implemented using Python 3.11. The testing and experimentation phases were conducted on a Lenovo Legion R9000P system, featuring Windows 11 as its operating system. The system is equipped with an AMD Ryzen 7 5800 H processor clocked at 3.20 GHz and 16GB of RAM. This robust hardware configuration ensures the fairness and reliability of computational experiments conducted in this study.

#### 4.1.2 Benchmark functions

We first introduce the continuous benchmark functions to evaluate the CVEGE. Two categories of problems are involved: (1). 10-D, 30-D, 50-D, and 100-D CEC2020 benchmark functions [69], which are summarized in Table 3. These benchmark functions are provided by OpFuNu library [70]. (2). Four real-world engineering optimization problems: tension/compression spring design, pressure vessel design, corrugated bulkhead design, and welded beam design. Here, we only list the definition of these problems, and the detailed information and visualization can be found in [71, 72].

**Tension/compression spring design (TCSD)**

$$\begin{aligned} &\text{minimize} \\ &f(X) = (x_3+2)x_2x_1^2 \\ &\text{subject to} \\ &g_1(X) = 1 - \frac{x_2^3x_3}{71785x_1^4} \le 0 \\ &g_2(X) = \frac{4x_2^2 - x_1x_2}{12566(x_2x_1^3 - x_1^4)} + \frac{1}{5108x_1^2} - 1 \le 0 \\ &g_3(X) = 1 - \frac{140.45x_1}{x_2^2x_3} \le 0 \\ &g_4(X) = \frac{x_1+x_2}{1.5} - 1 \le 0 \\ &\text{where} \\ &0.05 \le x_1 \le 2 \\ &0.25 \le x_2 \le 1.3 \\ &2 \le x_3 \le 15 \end{aligned} \tag{17}$$

**Table 3** Summary of the CEC2020 benchmark functions: Uni.= Unimodal function, Multi.= Multimodal function, Hybrid.= Hybrid function, Comp.= Composition function

| Fun | Description | Feature | Optimum |
|---|---|---|---|
| $f_1$ | Shifted and rotated bent cigar function | Uni | 100 |
| $f_2$ | Shifted and rotated Schwefel's function | Multi. | 1100 |
| $f_3$ | Shifted and rotated Lunacek bi-Rastrigin function | | 700 |
| $f_4$ | Expanded Rosenbrock's plus Griewangk's function | | 1900 |
| $f_5$ | Hybrid function 1 (N = 3) | Hybrid. | 1700 |
| $f_6$ | Hybrid function 2 (N = 4) | | 1600 |
| $f_7$ | Hybrid function 3 (N = 5) | | 2100 |
| $f_8$ | Composition function 1 (N = 3) | Comp. | 2200 |
| $f_9$ | Composition function 2 (N = 4) | | 2400 |
| $f_{10}$ | Composition function 3 (N = 5) | | 2500 |

**Pressure vessel design (PVD)**:

$$
\begin{aligned}
&\text{minimize}\\
&f(X) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3\\
&\text{subject to}\\
&g_1(X) = -x_1 + 0.0193x_3 \le 0\\
&g_2(X) = -x_2 + 0.00954x_3 \le 0\\
&g_3(X) = -\pi x_3^2x_4 - \frac{4}{3}\pi x_3^3 + 1296000 \le 0\\
&g_4(X) = x_4 - 240 \le 0\\
&\text{where}\\
&0 \le x_1 \le 99\\
&0 \le x_2 \le 99\\
&10 \le x_3 \le 200\\
&10 \le x_4 \le 200
\end{aligned}
\tag{18}
$$

**Corrugated bulkhead design (CBD)**:

$$
\begin{aligned}
&\text{minimize}\\
&f(X) = \frac{5.885x_4(x_1 + x_3)}{x_1 + \sqrt{|x_3^2 - x_2^2|}}\\
&\text{subject to}\\
&g_1(X) = -x_4x_2(0.4x_1 + \frac{x_3}{6}) + 8.94(x_1 + \sqrt{|x_3^2 - x_2^2|}) \le 0\\
&g_2(X) = -x_4x_2^2(0.2x_1 + \frac{x_3}{12}) + 2.2(8.94(x_1 + \sqrt{|x_3^2 - x_2^2|}))^{4/3} \le 0\\
&g_3(X) = -x_4 + 0.0156x_1 + 0.15 \le 0\\
&g_4(X) = -x_4 + 0.0156x_3 + 0.15 \le 0\\
&g_5(X) = -x_4 + 1.05 \le 0\\
&g_6(X) = -x_3 + x_2 \le 0\\
&\text{where}\\
&0 \le x_1, x_2, x_3 \le 100\\
&0 \le x_4 \le 5
\end{aligned}
\tag{19}
$$

*Welded beam design (WBD)*: Due to the limitation of space, definitions such as $\tau_{max}$, $\sigma_{max}$ are omitted, which can be found in [71].

minimize

$$
\begin{aligned}
&f(X) = 1.10471x_1^2 + 0.04811x_3x_4(14 + x_2)\\
&\text{subject to}\\
&g_1(X) = \tau(X) - \tau_{max} \le 0\\
&g_2(X) = \sigma(X) - \sigma_{max} \le 0\\
&g_3(X) = \theta(X) - \theta_{max} \le 0\\
&g_4(X) = x_1 - x_4 \le 0\\
&g_5(X) = P - P_c(X) \le 0\\
&g_6(X) = 0.125 - x_1 \le 0\\
&g_6(X) = 1.10471x_1^2 + 0.04811x_3x_4(14 + x_2) - 5 \le 0\\
&\text{where}\\
&0.1 \le x_1, x_4 \le 2\\
&0.1 \le x_2, x_3 \le 10
\end{aligned}
\tag{20}
$$

Since all engineering problems involve constraints, and the original EAs, including CVEGE, cannot deal with constraints, it is necessary to introduce a constraint-handling technique. For the sake of simplicity and fair evaluation, all techniques are equipped with the outer death penalty function [73], which assigns the worst fitness value to solutions that violate the constraints.

The second numerical experiment aims to evaluate the performance of BCVEGE in discrete optimization. Two kinds of datasets are contained in this experiment: (1). Eight open access classification datasets, which are downloaded from Kaggle [74]. Table 4 summarizes the brief information of eight datasets. (2). Standard ten 0/1 knapsack problems, with basic information summarized in Table 5. Further details about the problem definitions can be found in [75].

#### 4.1.3 Compared methods and parameters

To thoroughly evaluate the performance of our proposal, nine meta-heuristics algorithms including two classic and seven state-of-the-art are employed as the competitor algorithms, they are PSO [14], DE [76], gradient-based

**Table 4** Summary of eight classification datasets

| Datasets | # of instances | # of attributes | # of classes |
|---|---|---|---|
| Cervical cancer risk (CCR) | 858 | 35 | 2 |
| Fetal health (FH) | 2126 | 21 | 3 |
| Mobile price (MP) | 2000 | 20 | 8 |
| Mushrooms | 8124 | 22 | 2 |
| Water quality (WQ) | 7999 | 20 | 2 |
| Diabetes | 128 | 10 | 2 |
| Heart disease (HD) | 302 | 14 | 2 |
| Dry bean (DB) | 2500 | 16 | 8 |

**Table 5** Summary of ten 0/1 knapsack problems

| Func | Dimension | $x^*$ | $f(x^*)$ |
|---|---|---|---|
| $f_1$ | 10 | {0,1,1,1,0,0,0,1,1,1} | 295 |
| $f_2$ | 20 | {1,1,1,1,1,1,1,1,1,1,1,1,0,1,0,1,0,1,1} | 1024 |
| $f_3$ | 4 | {1,1,0,1} | 35 |
| $f_4$ | 4 | {0,1,0,1} | 23 |
| $f_5$ | 15 | {0,0,1,0,1,0,1,1,0,1,1,1,0,1,1} | 481.0694 |
| $f_6$ | 10 | {0,0,1,0,1,1,1,1,1,1} | 52 |
| $f_7$ | 7 | {1,0,0,1,0,0,0} | 107 |
| $f_8$ | 23 | {1,1,1,1,1,1,1,1,0,0,1,0,0,0,0,1,1,0,0,0,0,0,0} | 9767 |
| $f_9$ | 5 | {1,1,1,1,0} | 130 |
| $f_{10}$ | 20 | {1,1,1,1,1,1,1,1,1,0,1,1,1,1,0,1,0,1,1,1} | 1025 |

optimizer (GBO) [77], life choice-based optimizer (LCBO) [78], enhanced tug of war optimization (ETWO) [79], Archimedes optimization algorithm (AOA) [80], improved artificial ecosystem-based optimization (IAEO) [81], chaos game optimization (CGO) [82], and VEGE [3]. All the compared meta-heuristics algorithms, except for VEGE, are sourced from the MEALPY library [83]. Detailed parameter settings for continuous optimization problems are listed in Table 6.

To ensure a fair comparison among the algorithms, the following experimental settings have been established: the population size of all algorithms, except for VEGE and CVEGE, is set to 100, the maximum FEs for CEC2020 is set to 1000*D (D=dimension), and the maximum FEs for tension/compression spring design, pressure vessel design, corrugated bulkhead design, and welded beam design are set to 10,000, 15,000, 10,000, and 10,000 respectively which are suggested in [84]. To mitigate the effects of randomness, 30 independent trial runs are implemented for each experiment.

In the combinatorial optimization tasks, we have employed the primary parameter settings outlined in Table 6. Specifically, the population size for all algorithms has been fixed at 10. Additionally, all competitor algorithms equip the simplified sigmoid transfer function with $\epsilon = 0.5$ to solve the binary optimization problems. For the feature selection task, the maximum FEs have been set to 1000, while the maximum FEs in the 0/1 knapsack problem are set to 50*D. Similarly, each algorithm is executed for 30 trial runs.

## 4.2 Experimental and statistical results

This section presents the experimental results and performs statistical analysis. We have collected the optimal fitness values from 30 trial runs of each optimization algorithm, then, the Friedman test is employed to assess the significance of the results. If statistical significance exists, the Mann–Whitney U test is applied to calculate the $p$-value between every pair of algorithms, followed by correction using the Holm multiple comparison test [85] to identify statistical significance. The symbols +, ≈, and − indicate that our proposed CVEGE is significantly better, has no significant difference, or is significantly worse compared to the competitor algorithms. The best-performing algorithm is denoted in bold.

### 4.2.1 Optimization performance on CEC2020

Tables 7, 8, 9, and 10 present the results for 10-D, 30-D, 50-D, and 100-D CEC2020 benchmark functions respectively. Due to the limitation of space, we only provide the convergence curves of $f_1$ (unimodal), $f_2$ (multimodal), $f_5$ (hybrid), and $f_8$ (composition) in Fig. 5. Moreover, the ablation experiments on 30-D and 50-D CEC2020 benchmark functions are listed in Tables 11 and 12, respectively.

### 4.2.2 Optimization performance on engineering problems

Table 13 provides a summary of the mean, the std, the worst, and the best of all algorithms across four engineering problems, each with 30 independent trail runs. Since all engineering problems contain constraints, and if an algorithm fails to find a feasible solution in more than 1 out of 30 trial runs, the mean, the std, and the worst statistics are marked as *NaN*.

### 4.2.3 Optimization performance on wrapper-based feature selection tasks

In the wrapper-based feature selection task, we are interested in two key metrics: average fitness value and classification accuracy. The performance of these metrics on eight datasets is summarized in Table 14.

### 4.2.4 Optimization performance on 0/1 knapsack problems

Table 15 provides a detailed summary of the experimental results obtained from 30 trial runs on 0/1 knapsack problems.

**Table 6** Parameter settings of algorithms in continuous optimization tasks

| Alg | Parameters | Value |
|---|---|---|
| PSO (1995) | Inertia factor $w$ | 1 |
| | Acceleration coefficients $c_1$ and $c_2$ | 2.05 |
| | Max. and min. speed | 2, −2 |
| DE (1996) | Mutation strategy | DE/cur-to-rand/1/bin |
| | Scaling factor $F$ | 0.8 |
| | Crossover rate $Cr$ | 0.9 |
| GBO (2020) | Probability parameter $pr$ | 0.5 |
| | $\beta_{min}$ and $\beta_{max}$ | 0.2, 1.2 |
| LCBO (2020) | Coefficient factor $r_1$ | 2.35 |
| ETWO (2020) | Parameter-free | |
| AOA (2021) | Factor $c_1$, $c_2$, $c_3$, and $c_4$ | 2, 5, 2, and 0.5 |
| | Acceleration $acc_{min}$ and $acc_{max}$ | 0.1 and 0.9 |
| IAEO (2021) | Parameter-free | |
| CGO (2021) | Parameter-free | |
| VEGE (2022) | Population size | 10 |
| | Growth cycle $GC$ | 6 |
| | Growth radius $GR$ | 2 |
| | Growth direction $GD$ | A random number in [−1, 1] |
| | Total # of seeds | 60 |
| | Moving scaling $MS$ | A random number in [−2, 2] |
| CVEGE | Scaling factor $\gamma$ | 0.05 |
| | Mutation rate $Mr_1$, $Mr_2$, and $Mr_3$ | 0.1, 0.5, and 0.01 |

## 5 Discussion

In this section, we conduct both theoretical and practical analyses of CVEGE. Theoretical analysis focuses on the computational complexity of CVEGE, while practical analysis evaluates its performance across four distinct tasks: CEC2020 benchmark functions, four engineering problems, wrapper-based feature selection tasks, and 0/1 knapsack problems. Then, we discuss some potential open topics for further development of CVEGE.

### 5.1 Computational complexity of CVEGE

The computational complexity of CVEGE can be divided into two main stages: initialization and iteration, with the iteration phase further divided into the growth period and the maturity period. Supposing the dimension of the problem is $D$, the population size is $N$, the maximum iteration time is $T$, the growth cycle is $GC$ and the total seed number is $K$. First, the computational complexity of population initialization is $O(N \cdot D)$, then, CVEGE enters the growth period, and the time complexity for the offspring generation and selection of the growth period in one generation are $O(N \cdot D \cdot GC)$ and $O(N \cdot GC)$ respectively. In the maturity period, the computational complexity of the seeding operator is $O(K \cdot D)$, and the top-$k$ selection strategy first combines maternal plants (i.e. parents) and seeds (i.e. offspring), then selects the best $N$ solutions to survive. Thus the sort operator is involved in this process and the time complexity is $O((N + K) \cdot \log(K))$. In total, the Computational complexity of the maturity period is $O(K \cdot D + (N + K) \cdot \log(K))$.

In summary, the computational complexity of the entire CVEGE is

$$\begin{aligned} &O(N \cdot D + T \cdot ((N \cdot D \cdot GC + N \cdot GC) + (K \cdot D + (N + K) \cdot \log(K)))) \\ &:= O(N \cdot D + T \cdot (N \cdot D \cdot GC + K \cdot D + (N + K) \cdot \log(K))) \end{aligned} \tag{21}$$

### 5.2 Performance analysis on CEC2020 benchmark functions

The experimental and statistical results obtained from the evaluation of CVEGE on the CEC2020 benchmark functions at various scales provide substantial evidence of its effectiveness. Across a total of 40 test functions, our proposed CVEGE is significantly better than the compared algorithms in the majority of cases. Furthermore, we observed no significant deterioration in performance, demonstrating the superiority of our approach in the context of standard benchmark functions commonly used in practical applications.

Next, we will analyze the performance of CVEGE from three key aspects: exploitation, exploration, and its ability to overcome the local optima. To evaluate the exploitation capacity of optimization algorithms, we consider the

**Table 7** Experimental and statistical results on 10-D CEC2020 suite

| Func. | | PSO | DE | GBO | LCBO | ETWO | AOA | IAEO | CGO | VEGE | CVEGE |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $f_1$ | Mean | 4.72e+09 + | 7.64e+08 + | 2.26e+07 + | 8.93e+08 + | 5.67e+08 + | 8.01e+08 + | 1.19e+08 + | 4.64e+08 + | 2.66e+05 + | **2.47e+03** |
| | Std | 1.97e+09 | 1.61e+08 | 1.98e+07 | 7.48e+08 | 2.06e+08 | 6.87e+08 | 1.41e+08 | 4.50e+08 | 1.27e+05 | 2.89e+03 |
| $f_2$ | Mean | 3.52e+11 + | 7.50e+10 + | 8.75e+08 + | 5.23e+10 + | 5.26e+10 + | 1.08e+11 + | 9.04e+09 + | 5.87e+10 + | 2.93e+07 + | **3.62e+05** |
| | Std | 2.22e+11 | 2.17e+10 | 9.17e+08 | 4.56e+10 | 1.58e+10 | 8.57e+10 | 8.35e+09 | 6.06e+10 | 1.48e+07 | 2.55e+05 |
| $f_3$ | Mean | 1.25e+11 + | 3.49e+10 + | 2.62e+08 + | 1.31e+10 + | 2.13e+10 + | 2.63e+10 + | 3.59e+09 + | 2.18e+10 + | 1.13e+07 + | **2.11e+05** |
| | Std | 9.21e+10 | 9.08e+09 | 1.77e+08 | 9.24e+09 | 6.48e+09 | 3.68e+10 | 4.42e+09 | 3.02e+10 | 5.17e+06 | 3.49e+05 |
| $f_4$ | Mean | 3.31e+03 + | 1.93e+03 + | 1.90e+03 + | 1.96e+03 + | 1.91e+03 + | 2.32e+03 + | 1.91e+03 + | 1.93e+03 + | 1.91e+03 + | **1.90e+03** |
| | Std | 2.86e+03 | 1.66e+01 | 1.18e+00 | 1.01e+02 | 2.68e+00 | 1.16e+03 | 6.97e+00 | 5.61e+01 | 1.64e+00 | 1.05e+00 |
| $f_5$ | Mean | 7.77e+05 + | 1.23e+04 + | 2.74e+04 + | 2.60e+04 + | 1.59e+05 + | 1.35e+05 + | 1.68e+04 + | 1.35e+04 + | 1.15e+04 ≈ | **7.42e+03** |
| | Std | 1.05e+06 | 3.46e+03 | 1.29e+04 | 1.25e+04 | 1.10e+05 | 2.16e+05 | 7.92e+03 | 7.52e+03 | 7.31e+03 | 5.87e+03 |
| $f_6$ | Mean | 1.58e+05 + | 2.88e+03 + | 4.35e+03 + | 3.08e+03 + | 9.54e+03 + | 5.25e+03 + | 2.38e+03 + | 2.35e+03 + | 2.26e+03 + | **1.93e+03** |
| | Std | 2.61e+05 | 3.93e+02 | 2.44e+03 | 2.19e+03 | 6.03e+03 | 3.37e+03 | 5.16e+02 | 7.13e+02 | 5.58e+02 | 5.51e+02 |
| $f_7$ | Mean | 1.20e+06 + | 5.72e+04 + | 2.85e+04 + | 2.01e+04 + | 2.20e+05 + | 9.52e+04 + | 1.28e+04 ≈ | 1.18e+04 ≈ | **8.45e+03** ≈ | 9.89e+03 |
| | Std | 2.11e+06 | 2.27e+04 | 1.68e+04 | 1.66e+04 | 1.57e+05 | 1.31e+05 | 7.75e+03 | 6.70e+03 | 5.41e+03 | 7.47e+03 |
| $f_8$ | Mean | 2.33e+03 + | 2.32e+03 + | 2.31e+03 + | 2.32e+03 + | 2.32e+03 + | 2.32e+03 + | 2.31e+03 + | 2.31e+03 + | 2.31e+03 + | **2.31e+03** |
| | Std | 1.29e+01 | 1.48e+00 | 2.00e+00 | 5.65e+00 | 1.14e+00 | 1.05e+01 | 3.39e+00 | 5.75e+00 | 8.00e+00 | 2.03e+00 |
| $f_9$ | Mean | 5.19e+03 + | 3.82e+03 + | 2.76e+03 + | 3.81e+03 + | 3.68e+03 + | 3.90e+03 + | 3.04e+03 + | 3.36e+03 + | 2.63e+03 + | **2.59e+03** |
| | Std | 8.01e+02 | 1.50e+02 | 7.95e+01 | 6.13e+02 | 1.40e+02 | 6.67e+02 | 1.99e+02 | 5.34e+02 | 6.50e+01 | 5.35e+01 |
| $f_{10}$ | Mean | 3.13e+03 + | 3.09e+03 + | **3.00e+03** ≈ | 3.10e+03 + | 3.04e+03 + | 3.11e+03 + | 3.07e+03 + | 3.08e+03 + | 3.00e+03 ≈ | 3.00e+03 |
| | Std | 1.40e+02 | 3.37e+01 | 1.08e+01 | 4.50e+01 | 1.63e+01 | 5.88e+01 | 5.37e+01 | 6.34e+01 | 3.02e+01 | 2.48e+01 |
| +/≈/− summary: | | 10/0/0 | 10/0/0 | 9/1/0 | 10/0/0 | 10/0/0 | 10/0/0 | 9/1/0 | 9/1/0 | 7/3/0 | |

$f_1$: Unimodal function; $f_2 - f_4$: Multimodal functions; $f_5 - f_7$: Hybrid functions; $f8 - f_{10}$: Composition functions; mean and std: the mean and the standard deviation of 30 trial runs

**Table 8** Experimental and statistical results on 30-D CEC2020 suite

| Func. | | PSO | DE | GBO | LCBO | ETWO | AOA | IAEO | CGO | VEGE | CVEGE |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $f_1$ | Mean | 4.42e+10 + | 3.87e+10 + | 6.46e+08 + | 3.02e+10 + | 1.68e+09 + | 2.41e+10 + | 7.20e+09 + | 2.13e+10 + | 6.49e+06 + | **8.52e+04** |
| | Std | 1.53e+10 | 5.06e+09 | 3.33e+08 | 6.51e+09 | 2.07e+08 | 6.18e+09 | 2.16e+09 | 5.75e+09 | 1.19e+06 | 6.52e+04 |
| $f_2$ | Mean | 4.73e+12 + | 3.79e+12 + | 8.36e+10 + | 3.37e+12 + | 1.70e+11 + | 3.61e+12 + | 8.91e+11 + | 2.86e+12 + | 6.96e+08 + | **1.29e+07** |
| | Std | 1.89e+12 | 3.83e+11 | 4.28e+10 | 9.61e+11 | 2.19e+10 | 9.85e+11 | 3.63e+11 | 8.49e+11 | 1.54e+08 | 1.11e+07 |
| $f_3$ | Mean | 1.40e+12 + | 1.24e+12 + | 2.75e+10 + | 1.02e+12 + | 5.96e+10 + | 1.02e+12 + | 2.15e+11 + | 8.21e+11 + | 2.21e+08 + | **2.62e+06** |
| | Std | 6.43e+11 | 1.56e+11 | 1.12e+10 | 1.87e+11 | 9.29e+09 | 2.39e+11 | 6.58e+10 | 2.20e+11 | 5.10e+07 | 1.55e+06 |
| $f_4$ | Mean | 7.07e+05 + | 9.84e+04 + | 1.96e+03 + | 1.53e+05 + | 1.96e+03 + | 8.45e+04 + | 8.85e+03 + | 3.18e+04 + | 1.94e+03 + | **1.92e+03** |
| | Std | 7.91e+05 | 3.84e+04 | 2.31e+01 | 1.01e+05 | 1.02e+01 | 6.93e+04 | 1.00e+04 | 3.43e+04 | 6.90e+00 | 4.44e+00 |
| $f_5$ | Mean | 5.74e+07 + | 1.51e+07 + | 8.64e+05 + | 1.89e+07 + | 1.33e+06 + | 9.30e+06 + | 5.06e+06 + | 1.72e+06 + | 3.58e+05 + | **1.31e+05** |
| | Std | 7.35e+07 | 5.77e+06 | 5.35e+05 | 2.75e+07 | 4.45e+05 | 1.23e+07 | 5.57e+06 | 2.41e+06 | 1.61e+05 | 5.92e+04 |
| $f_6$ | Mean | 1.53e+08 + | 2.02e+05 + | 3.86e+04 + | 1.95e+06 + | 4.87e+04 + | 3.47e+05 + | 9.62e+04 + | 6.59e+04 + | 1.77e+04 + | **7.60e+03** |
| | Std | 3.05e+08 | 4.71e+05 | 2.85e+04 | 2.96e+06 | 1.99e+04 | 8.99e+05 | 1.16e+05 | 1.19e+05 | 1.33e+04 | 7.16e+03 |
| $f_7$ | Mean | 3.41e+08 + | 3.56e+07 + | 1.36e+06 + | 9.51e+07 + | 2.63e+06 + | 1.91e+07 + | 1.36e+07 + | 1.44e+07 + | 4.87e+05 ≈ | **4.15e+05** |
| | Std | 2.44e+08 | 1.06e+07 | 8.82e+05 | 1.25e+08 | 7.74e+05 | 3.13e+07 | 1.17e+07 | 1.69e+07 | 2.55e+05 | 2.82e+05 |
| $f_8$ | Mean | 3.56e+03 + | 2.65e+03 + | 2.44e+03 + | 3.40e+03 + | 2.44e+03 + | 3.01e+03 + | 2.91e+03 + | 3.10e+03 + | 2.99e+03 + | **2.37e+03** |
| | Std | 5.78e+02 | 3.37e+01 | 1.90e+01 | 5.02e+02 | 6.46e+00 | 3.00e+02 | 2.30e+02 | 3.32e+02 | 3.32e+02 | 3.66e+00 |
| $f_9$ | Mean | 2.68e+04 + | 1.30e+04 + | 4.56e+03 + | 2.35e+04 + | 5.59e+03 + | 2.25e+04 + | 8.46e+03 + | 1.72e+04 + | 2.86e+03 + | **2.62e+03** |
| | Std | 8.62e+03 | 6.30e+02 | 3.90e+02 | 3.41e+03 | 2.41e+02 | 4.80e+03 | 9.54e+02 | 3.44e+03 | 1.46e+02 | 8.46e+00 |
| $f_{10}$ | Mean | 6.58e+03 + | 5.11e+03 + | 3.06e+03 + | 4.69e+03 + | 3.09e+03 + | 4.49e+03 + | 3.53e+03 + | 3.84e+03 + | 2.94e+03 + | **2.93e+03** |
| | Std | 2.08e+03 | 4.66e+02 | 4.54e+01 | 5.35e+02 | 2.17e+01 | 4.09e+02 | 1.73e+02 | 3.83e+02 | 2.54e+01 | 1.00e+01 |
| +/≈/− summary: | | 10/0/0 | 10/0/0 | 10/0/0 | 10/0/0 | 10/0/0 | 10/0/0 | 10/0/0 | 10/0/0 | 9/1/0 | |

**Table 9** Experimental and statistical results on 50-D CEC2020 suite

| Func. | | PSO | DE | GBO | LCBO | ETWO | AOA | IAEO | CGO | VEGE | CVEGE |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $f_1$ | Mean | 1.06e+11 + | 1.07e+11 + | 2.71e+09 + | 7.66e+10 + | 1.71e+09 + | 8.07e+10 + | 2.24e+10 + | 6.41e+10 + | 1.99e+07 + | **3.91e+05** |
| | Std | 2.51e+10 | 1.04e+10 | 7.57e+08 | 7.93e+09 | 2.19e+08 | 1.32e+10 | 4.87e+09 | 5.44e+09 | 3.65e+06 | 1.98e+05 |
| $f_2$ | Mean | 1.14e+13 + | 1.15e+13 + | 2.98e+11 + | 8.18e+12 + | 1.94e+11 + | 9.40e+12 + | 2.48e+12 + | 6.79e+12 + | 2.71e+09 + | **5.85e+07** |
| | Std | 2.81e+12 | 1.39e+12 | 8.81e+10 | 1.06e+12 | 2.02e+10 | 1.12e+12 | 5.73e+11 | 1.04e+12 | 5.08e+08 | 3.70e+07 |
| $f_3$ | Mean | 4.31e+12 + | 4.13e+12 + | 9.52e+10 + | 2.77e+12 + | 6.65e+10 + | 2.95e+12 + | 8.26e+11 + | 2.12e+12 + | 8.24e+08 + | **1.58e+07** |
| | Std | 1.16e+12 | 4.60e+11 | 2.75e+10 | 3.85e+11 | 5.78e+09 | 4.78e+11 | 1.99e+11 | 2.88e+11 | 1.34e+08 | 7.53e+06 |
| $f_4$ | Mean | 1.94e+06 + | 1.40e+06 + | 2.29e+03 + | 7.56e+05 + | 2.46e+03 + | 7.63e+05 + | 5.59e+04 + | 3.07e+05 + | 1.97e+03 + | **1.95e+03** |
| | Std | 1.26e+06 | 6.32e+05 | 4.82e+02 | 3.38e+05 | 1.44e+02 | 5.04e+05 | 5.80e+04 | 1.29e+05 | 1.17e+01 | 1.38e+01 |
| $f_5$ | Mean | 2.09e+08 + | 6.23e+07 + | 5.66e+06 + | 7.31e+07 + | 2.71e+06 + | 3.50e+07 + | 2.18e+07 + | 1.19e+07 + | 1.12e+06 + | **7.80e+05** |
| | Std | 1.27e+08 | 2.18e+07 | 2.72e+06 | 4.36e+07 | 7.17e+05 | 5.08e+07 | 1.13e+07 | 5.38e+06 | 3.62e+05 | 3.21e+05 |
| $f_6$ | Mean | 4.30e+09 + | 9.22e+07 + | 2.34e+05 + | 7.07e+08 + | 1.21e+05 + | 9.89e+08 + | 7.34e+06 + | 9.66e+06 + | 2.20e+04 + | **8.59e+03** |
| | Std | 3.52e+09 | 3.22e+07 | 2.63e+05 | 5.69e+08 | 6.74e+04 | 2.36e+09 | 8.34e+06 | 1.11e+07 | 6.18e+03 | 4.31e+03 |
| $f_7$ | Mean | 3.61e+09 + | 7.53e+08 + | 1.28e+07 + | 2.45e+09 + | 1.14e+07 + | 1.04e+09 + | 1.46e+08 + | 5.05e+08 + | 1.52e+06 ≈ | **1.36e+06** |
| | Std | 2.59e+09 | 1.81e+08 | 6.87e+06 | 1.89e+09 | 2.31e+06 | 1.09e+09 | 1.10e+08 | 6.86e+08 | 6.74e+05 | 9.73e+05 |
| $f_8$ | Mean | 7.70e+03 + | 3.19e+03 + | 2.62e+03 + | 9.69e+03 + | 2.52e+03 + | 6.57e+03 + | 6.30e+03 + | 7.49e+03 + | 6.26e+03 + | **2.44e+03** |
| | Std | 3.20e+03 | 1.06e+02 | 4.89e+01 | 2.13e+03 | 1.16e+01 | 1.55e+03 | 2.33e+03 | 1.87e+03 | 1.59e+03 | 1.28e+02 |
| $f_9$ | Mean | 5.29e+04 + | 2.74e+04 + | 6.53e+03 + | 5.44e+04 + | 6.02e+03 + | 5.65e+04 + | 1.85e+04 + | 4.89e+04 + | 3.16e+03 + | **2.78e+03** |
| | Std | 1.41e+04 | 1.91e+03 | 1.07e+03 | 4.67e+03 | 2.52e+02 | 6.35e+03 | 5.36e+03 | 6.34e+03 | 2.44e+02 | 2.55e+02 |
| $f_{10}$ | Mean | 1.93e+04 + | 1.23e+04 + | 4.37e+03 + | 1.53e+04 + | 4.00e+03 + | 1.57e+04 + | 7.03e+03 + | 8.98e+03 + | 3.54e+03 + | **3.35e+03** |
| | Std | 7.12e+03 | 1.78e+03 | 3.35e+02 | 2.94e+03 | 1.02e+02 | 3.03e+03 | 7.70e+02 | 1.17e+03 | 1.66e+02 | 1.22e+02 |
| +/≈/− summary: | | 10/0/0 | 10/0/0 | 10/0/0 | 10/0/0 | 10/0/0 | 10/0/0 | 10/0/0 | 10/0/0 | 9/1/0 | |

**Table 10** Experimental and statistical results on 100-D CEC2020 suite

| Func. | | PSO | DE | GBO | LCBO | ETWO | AOA | IAEO | CGO | VEGE | CVEGE |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $f_1$ | Mean | 3.28e+11 + | 3.33e+11 + | 1.27e+10 + | 2.11e+11 + | 9.03e+09 + | 2.32e+11 + | 6.96e+10 + | 1.49e+11 + | 9.46e+07 + | **1.23e+06** |
| | Std | 4.54e+10 | 2.41e+10 | 2.16e+09 | 1.51e+10 | 2.11e+09 | 1.94e+10 | 1.35e+10 | 5.77e+09 | 1.22e+07 | 3.22e+05 |
| $f_2$ | Mean | 3.19e+13 + | 3.08e+13 + | 1.30e+12 + | 2.36e+13 + | 8.94e+11 + | 2.67e+13 + | 6.93e+12 + | 1.67e+13 + | 9.06e+09 + | **1.50e+08** |
| | Std | 3.98e+12 | 2.17e+12 | 2.94e+11 | 1.79e+12 | 1.51e+11 | 2.25e+12 | 1.36e+12 | 7.59e+11 | 9.17e+08 | 4.61e+07 |
| $f_3$ | Mean | 1.20e+13 + | 1.17e+13 + | 4.50e+11 + | 8.53e+12 + | 3.14e+11 + | 9.48e+12 + | 2.36e+12 + | 5.47e+12 + | 3.41e+09 + | **4.96e+07** |
| | Std | 1.71e+12 | 8.57e+11 | 9.99e+10 | 5.30e+11 | 5.58e+10 | 5.48e+11 | 4.02e+11 | 2.32e+11 | 4.65e+08 | 1.25e+07 |
| $f_4$ | Mean | 9.32e+06 + | 2.10e+07 + | 5.12e+03 + | 3.11e+06 + | 3.53e+04 + | 4.15e+06 + | 1.41e+05 + | 4.22e+05 + | 2.09e+03 + | **2.05e+03** |
| | Std | 4.21e+06 | 6.64e+06 | 2.22e+03 | 1.10e+06 | 7.55e+03 | 1.54e+06 | 6.70e+04 | 6.45e+04 | 2.51e+01 | 2.31e+01 |
| $f_5$ | Mean | 1.18e+09 + | 7.70e+08 + | 4.46e+07 + | 6.06e+08 + | 4.33e+07 + | 3.79e+08 + | 1.76e+08 + | 6.41e+07 + | 7.72e+06 ≈ | **6.33e+06** |
| | Std | 4.45e+08 | 1.04e+08 | 1.75e+07 | 2.67e+08 | 7.98e+06 | 1.86e+08 | 4.42e+07 | 2.37e+07 | 2.61e+06 | 2.12e+06 |
| $f_6$ | Mean | 2.77e+10 + | 3.43e+09 + | 1.01e+06 + | 1.18e+10 + | 3.83e+05 + | 1.41e+10 + | 8.17e+07 + | 8.61e+09 + | 1.27e+05 + | **1.07e+04** |
| | Std | 2.00e+10 | 1.24e+09 | 8.54e+05 | 6.44e+09 | 1.20e+05 | 8.39e+09 | 8.80e+07 | 6.41e+09 | 3.81e+04 | 5.72e+03 |
| $f_7$ | Mean | 2.11e+10 + | 5.21e+09 + | 8.18e+07 + | 1.39e+10 + | 1.49e+08 + | 1.20e+10 + | 8.66e+08 + | 4.29e+09 + | 4.94e+06 + | **1.99e+06** |
| | Std | 1.32e+10 | 1.48e+09 | 4.43e+07 | 4.67e+09 | 2.54e+07 | 6.45e+09 | 3.07e+08 | 2.16e+09 | 1.83e+06 | 7.86e+05 |
| $f_8$ | Mean | 2.54e+04 + | 4.51e+03 + | 2.84e+03 + | 2.47e+04 + | 2.73e+03 + | 1.80e+04 + | 1.86e+04 + | 1.96e+04 + | 1.48e+04 + | **2.47e+03** |
| | Std | 8.22e+03 | 3.77e+02 | 8.64e+01 | 2.36e+03 | 3.15e+01 | 3.30e+03 | 5.17e+03 | 2.44e+03 | 2.92e+03 | 2.40e+02 |
| $f_9$ | Mean | 2.28e+05 + | 1.16e+05 + | 2.55e+04 + | 1.62e+05 + | 2.36e+04 + | 1.74e+05 + | 9.41e+04 + | 1.30e+05 + | 4.45e+03 + | **2.92e+03** |
| | Std | 4.26e+04 | 1.27e+04 | 4.83e+03 | 6.01e+03 | 1.18e+03 | 8.93e+03 | 1.98e+04 | 6.61e+03 | 2.52e+02 | 3.69e+02 |
| $f_{10}$ | Mean | 4.39e+04 + | 4.19e+04 + | 5.05e+03 + | 2.78e+04 + | 5.35e+03 + | 3.20e+04 + | 9.26e+03 + | 2.01e+04 + | 3.56e+03 ≈ | **3.54e+03** |
| | Std | 8.89e+03 | 7.87e+03 | 2.38e+02 | 3.11e+03 | 1.23e+02 | 6.36e+03 | 1.05e+03 | 2.07e+03 | 7.62e+01 | **8.50e+01** |
| +/≈/− summary: | | 10/0/0 | 10/0/0 | 10/0/0 | 10/0/0 | 10/0/0 | 10/0/0 | 10/0/0 | 10/0/0 | 8/2/0 | |

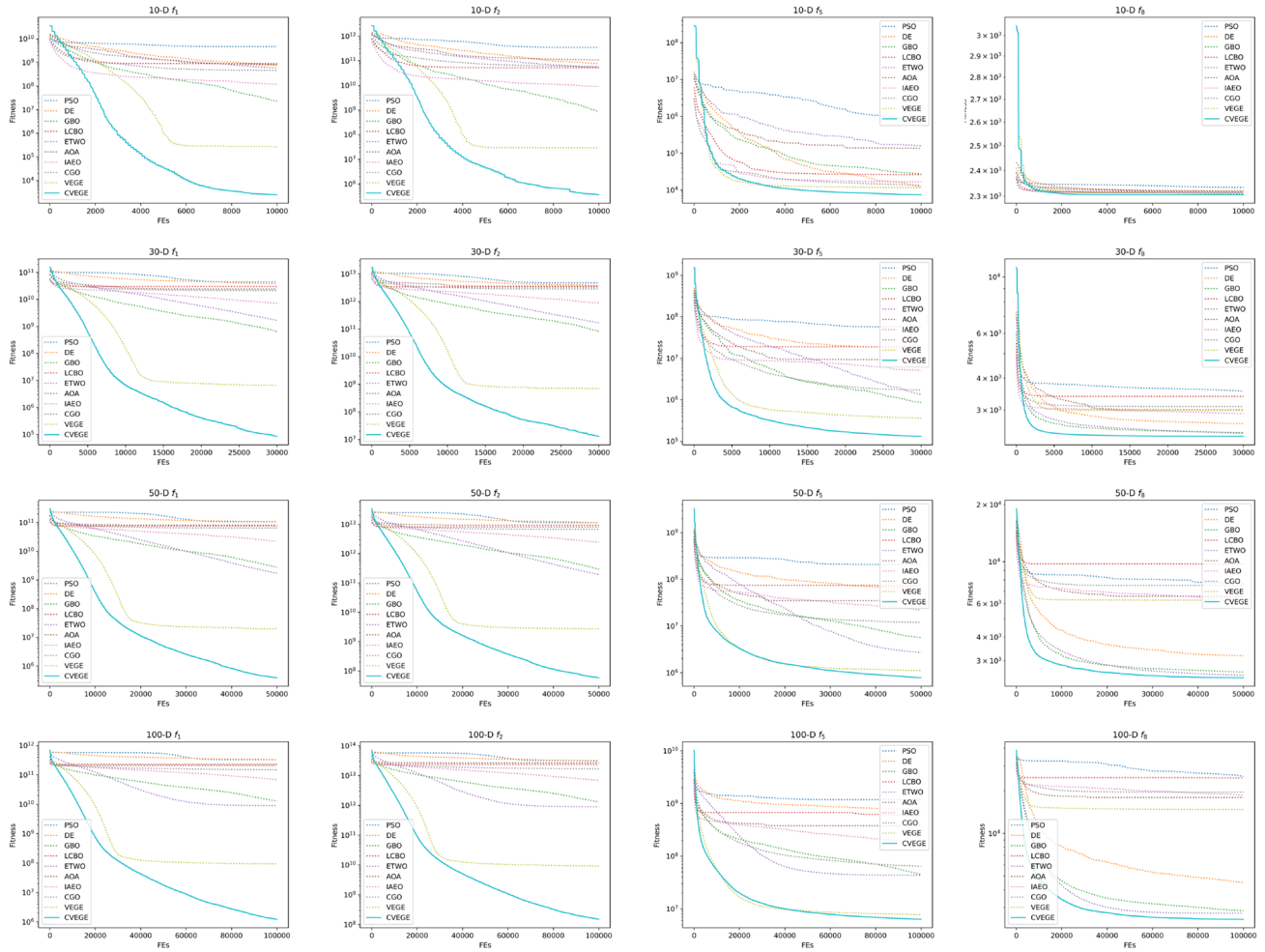


**Fig. 5** Convergence curves of ten EAs on representative functions

unimodal function, denoted as $f_1$. From the experimental results presented in Tables 7, 8, 9, and 10, CVEGE inherits the powerful exploitation capacity from VEGE and further achieves the improvements. Besides, we closely examine the convergence curve illustrated in Fig. 5. During the initial stage of optimization, CVEGE has a fast convergence speed and approaches acceptable solutions, and at the late stage of optimization, the convergence speed of other optimization algorithms including VEGE decelerates, while CVEGE maintains its efficient search behaviors. This observation can reflect the superiority of CVEGE on the exploitation aspect adequately.

$f_2$ to $f_{10}$ encompass multimodal, hybrid, and composition functions, rendering them suitable for evaluating the exploration ability of algorithms and their capacity to escape local optima. From the experimental and statistical results, our proposed CVEGE is significantly better than the compared algorithms in the majority of cases. Remarkably, as the dimension of the problem increases, the domination of CVEGE over the other competitor algorithms remains unscathed by the so-called 'curse of dimensionality'. This phenomenon supports the high efficiency of our proposal in addressing relative high-dimensional optimization problems. Meanwhile, we notice that, in the case of $f_5$ and $f_{10}$, CVEGE is significantly better than VEGE in 30-D and 50-D, while its performance is approximately on par with VEGE in 10-D and 100-D. We conjecture that the techniques integrated into CVEGE are highly adaptive to the median-scale problems but may not be optimized for the low- and high-dimensional problems.

Furthermore, we re-emphasize the high convergence speed of CVEGE. Since the initial population size of CVEGE is set to 10 while the other optimization algorithms except for VEGE are set to 100, thus the initial convergence of CVEGE and VEGE in the convergence curve may appear relatively disadvantaged. However, the outstanding convergence speed accelerates CVEGE

**Table 11** Ablation experiments on 30-D CEC2020 suite

| Func. | | VEGE | VEGE-1 | VEGE-2 | VEGE-3 | CVEGE |
|---|---|---|---|---|---|---|
| $f_1$ | Mean | 6.49e+06 + | 9.18e+06 + | 9.35e+05 + | 1.20e+05 + | **8.52e+04** |
| | Std | 1.19e+06 | 1.69e+06 | 7.91e+05 | 5.96e+04 | 6.52e+04 |
| $f_2$ | Mean | 6.96e+08 + | 1.01e+09 + | 1.70e+08 + | 1.79e+07 + | **1.29e+07** |
| | Std | 1.54e+08 | 2.72e+08 | 1.58e+08 | 8.26e+06 | 1.11e+07 |
| $f_3$ | Mean | 2.21e+08 + | 2.93e+08 + | 4.80e+07 + | 6.02e+06 + | **2.62e+06** |
| | Std | 5.10e+07 | 6.91e+07 | 4.02e+07 | 3.75e+06 | 1.55e+06 |
| $f_4$ | Mean | 1.94e+03 + | 1.94e+03 + | 1.93e+03 + | 1.92e+03 ≈ | **1.92e+03** |
| | Std | 6.90e+00 | 7.23e+00 | 9.03e+00 | 3.89e+00 | 4.44e+00 |
| $f_5$ | Mean | 3.58e+05 + | 4.43e+05 + | 3.05e+05 + | **1.28e+05** + | 1.31e+05 |
| | Std | 1.61e+05 | 1.90e+05 | 1.95e+05 | 5.17e+04 | 5.92e+04 |
| $f_6$ | Mean | 1.77e+04 + | 2.45e+04 + | 1.10e+04 ≈ | 9.55e+03 ≈ | **7.60e+03** |
| | Std | 1.33e+04 | 1.59e+04 | 1.14e+04 | 8.06e+03 | 7.16e+03 |
| $f_7$ | Mean | 4.87e+05 ≈ | 5.85e+05 ≈ | 6.16e+05 ≈ | 4.16e+05 ≈ | **4.15e+05** |
| | Std | 2.55e+05 | 2.44e+05 | 3.91e+05 | 1.99e+05 | 2.82e+05 |
| $f_8$ | Mean | 2.99e+03 + | 2.88e+03 + | 2.54e+03 + | 2.37e+03 ≈ | **2.37e+03** |
| | Std | 3.32e+02 | 2.43e+02 | 9.07e+01 | 4.75e+00 | 3.66e+00 |
| $f_9$ | Mean | 2.86e+03 + | 2.82e+03 + | 2.71e+03 + | 2.64e+03 + | **2.62e+03** |
| | Std | 1.46e+02 | 4.33e+01 | 1.37e+02 | 7.62e+01 | 8.46e+00 |
| $f_10$ | Mean | 2.94e+03 + | 2.95e+03 + | 2.94e+03 ≈ | **2.93e+03** ≈ | 2.93e+03 |
| | Std | 2.54e+01 | 3.35e+01 | 3.05e+01 | 7.40e+00 | 1.00e+01 |
| +/≈/− summary: | | 9/1/0 | 9/1/0 | 7/3/0 | 5/5/0 | |

(VEGE-1: VEGE+chaotic local search; VEGE-2: VEGE+multiple seeding strategy; VEGE-3: VEGE+mutation module)

**Table 12** Ablation experiments on 50-D CEC2020 suite

| Func. | | VEGE | VEGE-1 | VEGE-2 | VEGE-3 | CVEGE |
|---|---|---|---|---|---|---|
| $f_1$ | Mean | 1.99e+07 + | 2.73e+07 + | 5.25e+06 + | 5.17e+05 + | **3.91e+05** |
| | Std | 3.65e+06 | 4.05e+06 | 2.61e+06 | 2.46e+05 | 1.98e+05 |
| $f_2$ | Mean | 2.71e+09 + | 3.11e+09 + | 7.34e+08 + | 6.37e+07 ≈ | **5.85e+07** |
| | Std | 5.08e+08 | 4.94e+08 | 5.95e+08 | 2.74e+07 | 3.70e+07 |
| $f_3$ | Mean | 8.24e+08 + | 1.14e+09 + | 2.44e+08 + | 1.98e+07 + | **1.58e+07** |
| | Std | 1.34e+08 | 2.10e+08 | 1.98e+08 | 7.65e+06 | 7.53e+06 |
| $f_4$ | Mean | 1.97e+03 + | 1.98e+03 + | 1.97e+03 + | 1.95e+03 ≈ | **1.95e+03** |
| | Std | 1.17e+01 | 1.10e+01 | 1.49e+01 | 8.91e+00 | 1.38e+01 |
| $f_5$ | Mean | 1.12e+06 + | 1.18e+06 + | 1.09e+06 ≈ | **6.19e+05** ≈ | 7.80e+05 |
| | Std | 3.62e+05 | 3.35e+05 | 5.01e+05 | 2.48e+05 | 3.21e+05 |
| $f_6$ | Mean | 2.20e+04 + | 2.96e+04 + | 1.10e+04 ≈ | **7.57e+03** ≈ | 8.59e+03 |
| | Std | 6.18e+03 | 9.48e+03 | 4.16e+03 | 2.99e+03 | 4.31e+03 |
| $f_7$ | Mean | 1.52e+06 ≈ | 2.26e+06 + | 1.85e+06 ≈ | **1.13e+06** ≈ | 1.36e+06 |
| | Std | 6.74e+05 | 9.76e+05 | 7.65e+05 | 7.72e+05 | 9.73e+05 |
| $f_8$ | Mean | 6.26e+03 + | 6.26e+03 + | 3.97e+03 + | **2.42e+03** ≈ | 2.44e+03 |
| | Std | 1.59e+03 | 1.71e+03 | 2.00e+03 | 4.27e+01 | 1.28e+02 |
| $f_9$ | Mean | 3.16e+03 + | 3.31e+03 + | 2.97e+03 + | **2.75e+03** ≈ | 2.78e+03 |
| | Std | 2.44e+02 | 2.71e+02 | 2.94e+02 | 2.16e+02 | 2.55e+02 |
| $f_10$ | Mean | 3.54e+03 + | 3.51e+03 + | 3.41e+03 ≈ | 3.40e+03 ≈ | **3.35e+03** |
| | Std | 1.66e+02 | 1.74e+02 | 1.60e+02 | 1.11e+02 | 1.22e+02 |
| +/≈/− summary: | | 9/1/0 | 10/0/0 | 6/4/0 | 2/8/0 | |

**Table 13** Experimental and statistical results on four engineering problems

| Func. | | PSO | DE | GBO | LCBO | ETWO | AOA | IAEO | CGO | VEGE | CVEGE |
|---|---|---|---|---|---|---|---|---|---|---|---|
| TCSD | Mean | 1.6305e−02 + | 1.4257e−02 + | 1.5742e−02 + | 1.5129e−02 + | *NaN* + | *NaN* + | 1.3959e−02 ≈ | 1.5628e−02 + | 1.3220e−02 ≈ | **1.3065e −02** |
| | Std | 3.9606e−03 | 1.8493e−03 | 3.0763e−03 | 3.4619e−03 | *NaN* | *NaN* | 1.5370e−03 | 6.5760e−03 | 7.9896e−04 | **7.0931e −04** |
| | Worst | 2.6981e−02 | 1.9217e−02 | 2.3899e−02 | 2.9488e−02 | *NaN* | *NaN* | 1.7774e−02 | 4.5636e−02 | **1.6280e −02** | **1.6280e −02** |
| | Best | 1.2845e−02 | 1.2760e−02 | 1.2775e−02 | 1.2684e−02 | 1.2863e−02 | 1.2820e−02 | 1.2667e−02 | 1.2667e−02 | **1.2665e −02** | **1.2665e −02** |
| PVD | Mean | 3.6251e+04 + | 6.2133e+03 + | 6.4424e+03 ≈ | 6.4191e+03 ≈ | 7.2544e+03 + | 1.6285e+05 + | 2.8264e+04 + | 2.4127e+04 + | **6.0256e+03** ≈ | 6.1430e+03 |
| | Std | 1.8061e+04 | **9.3793e+01** | 4.9347e+02 | 4.3085e+02 | 5.8310e+02 | 1.1266e+05 | 4.8519e+04 | 3.9788e+04 | 2.2448e+02 | 2.7467e+02 |
| | Worst | 7.5381e+04 | **6.4514e+03** | 7.4845e+03 | 7.3185e+03 | 8.9427e+03 | 4.4564e+05 | 1.8957e+05 | 1.7924e+05 | 6.9004e+03 | 7.2253e+03 |
| | Best | 8.6785e+03 | **6.0645e+03** | 5.9161e+03 | 5.8854e+03 | 6.2704e+03 | 6.8021e+03 | 5.9645e+03 | 5.9011e+03 | **5.8853e+03** | 5.8856e+03 |
| CBD | Mean | 7.8125e+00 + | 6.9454e+00 + | 6.9189e+00 + | 6.9696e+00 + | 7.1243e+00 + | 7.5671e+00 + | 7.1238e+00 + | 6.9290e+00 + | 6.9017e+00 + | **6.8508e+00** |
| | Std | 4.8686e−01 | 2.6159e−02 | 4.6848e−02 | 1.5144e−01 | 1.0275e−01 | 1.1963e+00 | 2.4292e−01 | 2.3525e−01 | 5.1880e−02 | **2.6001e −02** |
| | Worst | 8.9215e+00 | 7.0205e+00 | 7.0675e+00 | 7.5277e+00 | 7.3869e+00 | 1.1578e+01 | 7.6922e+00 | 8.1309e+00 | 7.0522e+00 | **6.9835e+00** |
| | Best | 7.1279e+00 | 6.9013e+00 | 6.8589e+00 | 6.8436e+00 | 6.9460e+00 | 6.8477e+00 | 6.8622e+00 | **6.8430e+00** | 6.8465e+00 | **6.8430e+00** |
| WBD | Mean | 2.1521e+00 + | 1.6295e+00 + | 1.6161e+00 + | 1.7191e+00 + | 1.6554e+00 + | 2.0016e+00 + | 1.7934e+00 + | 2.0508e+00 + | 1.6136e+00 + | **1.5602e+00** |
| | Std | 1.8486e−01 | **2.6308e −02** | 9.7539e−02 | 2.3765e−01 | 5.8349e−02 | 4.2449e−01 | 2.5408e−01 | 5.8091e−01 | 1.6204e−01 | 8.5259e−02 |
| | Worst | 2.5315e+00 | **1.6936e+00** | 1.9286e+00 | 2.3648e+00 | 1.8024e+00 | 2.9821e+00 | 2.4056e+00 | 3.8193e+00 | 2.4100e+00 | 1.9613e+00 |
| | Best | 1.7753e+00 | 1.5857e+00 | 1.5250e+00 | 1.5168e+00 | 1.5638e+00 | 1.5459e+00 | 1.5317e+00 | 1.5436e+00 | 1.5186e+00 | **1.5168e+00** |
| +/≈/− summary: | | 4/0/0 | 4/0/0 | 3/1/0 | 3/1/0 | 4/0/0 | 4/0/0 | 3/1/0 | 4/0/0 | 2/2/0 | |

**Table 14** Average fitness value and classification accuracy on eight datasets

| Tasks | | PSO | DE | GBO | LCBO | ETWO | AOA | IAEO | CGO | VEGE | BCVEGE |
|---|---|---|---|---|---|---|---|---|---|---|---|
| CCR | Ave. fitness | 5.30e−02 | 4.27e−02 | 3.70e−02 | 5.51e−02 | **3.67e −02** | 5.29e−02 | 4.91e−02 | 5.13e−02 | 4.64e−02 | 4.23e−02 |
| | Ave. accuracy | 94.87% | 95.57% | 96.27% | 95.10% | **96.39%** | 94.87% | 95.10% | 94.75% | 95.22% | 96.27% |
| FH | Ave. fitness | 1.74e−01 | 1.65e−01 | 1.61e−01 | 1.62e−01 | 1.63e−01 | 1.73e−01 | 1.67e−01 | 1.70e−01 | 1.65e−01 | **9.23e −02** |
| | Ave. accuracy | 82.65% | 84.38% | 83.92% | 83.92% | 83.49% | 82.70% | 83.31% | 82.88% | 84.34% | **91.10%** |
| MP | Ave. fitness | 8.97e−02 | 8.46e−02 | 8.45e−02 | 9.26e−02 | 8.45e−02 | 8.88e−02 | 8.55e−02 | 9.78e−02 | 8.51e−02 | **8.44e −02** |
| | Ave. accuracy | 91.70% | 91.75% | 91.75% | 91.20% | 91.75% | 91.75% | 91.70% | 90.70% | 91.75% | **91.80%** |
| mushrooms | Ave. fitness | 1.93e−02 | 1.00e−02 | 8.91e−03 | 2.69e−02 | 7.75e−03 | 1.95e−02 | **7.60e −03** | 8.16e−03 | 1.08e−02 | 9.16e−03 |
| | Ave. accuracy | 98.58% | 99.20% | 99.24% | 95.79% | 99.41% | 98.23% | **99.80%** | 99.20% | 99.53% | 99.24% |
| WQ | Ave. fitness | 1.17e−01 | 1.05e−01 | 1.01e−01 | 1.18e−01 | 1.13e−01 | 1.03e−01 | 1.09e−01 | 1.03e−01 | 1.09e−01 | **9.84e −02** |
| | Ave. accuracy | 88.90% | 89.30% | 89.80% | 88.00% | 88.90% | 90.20% | 89.20% | 90.20% | 89.50% | **90.50%** |
| diabetes | Ave. fitness | 2.89e−02 | 2.51e−02 | **2.49e −02** | 5.00e−02 | 2.51e−02 | 3.49e−02 | 2.53e−02 | 2.88e−02 | 2.50e−02 | **2.49e −02** |
| | Ave. accuracy | **97.69%** | **97.69%** | **97.69%** | **97.69%** | **97.69%** | **97.69%** | **97.69%** | **97.69%** | **97.69%** | **97.69%** |
| HD | Ave. fitness | 5.14e−02 | 3.91e−02 | **3.80e −02** | 4.34e−02 | **3.80e −02** | 4.16e−02 | 3.87e−02 | 4.13e−02 | 3.83e−02 | **3.80e −02** |
| | Ave. accuracy | 95.74% | **96.39%** | **96.39%** | 95.74% | **96.39%** | 96.07% | **96.39%** | 96.07% | **96.39%** | **96.39%** |
| DB | Ave. fitness | 1.30e−01 | 1.32e−01 | 1.09e−01 | 1.35e−01 | 1.18e−01 | 1.21e−01 | 1.19e−01 | 1.35e−01 | **1.05e −01** | 1.16e−01 |
| | Ave. accuracy | 87.70% | 89.00% | 89.20% | **90.20%** | 88.30% | 88.40% | 88.30% | 87.40% | 89.70% | 88.50% |

optimization rapidly and wins the competitor algorithms, which is a pivotal factor contributing to the success of CVEGE.

Additionally, we shift our focus to the performance of all algorithms on $f_8$, which is a composition function characterized by a complex fitness landscape. As the dimension of $f_8$ increases, VEGE becomes prone to getting trapped in local optima easily. From Fig. 5, we estimate that the optimization speed of VEGE significantly slows down after reaching 5000, 10,000, and 20,000 FEs in 30-D, 50-D, and 100-D. In these instances, VEGE struggles to escape local optima and continues to converge slowly. Conversely, CVEGE demonstrates a remarkable ability to navigate away from local optima and approach more favorable solutions. This superior performance can be attributed to the unique mutation module and its effective cooperation with efficient exploration operations.

Finally, the statistical results of ablation experiments listed in Tables 11 and 12 practically investigate the contribution of each single strategy. Through the results, the proposed unique mutation module contributes mostly to the improvement of CVEGE and dominates the other two proposed strategies (i.e., the chaotic local search technique and the multiple seeding strategy). The multiple seeding strategy also accelerates the convergence of optimization in most instances, which is a success in improving the performance of CVEGE. However, some deterioration cases are observed in VEGE-1, such as in 50-D $f_7$. This situation demonstrates that the chaotic local search technique may not be good at dealing with this kind of problem. But owing to the presence of the No Free Lunch Theorem [86], we cannot deny the efficiency of this proposed technique in unknown optimization tasks. Therefore, we suggest combining the conventional VEGE and three proposed search operators together.

### 5.3 Performance analysis on engineering problems

The evaluation of optimization performance on four engineering problems serves as a valuable indicator of an algorithm's effectiveness in addressing real-world complex optimization challenges The statistical results, as summarized in Table 13, reveal that our proposed CVEGE performs at least comparably or significantly better than its competitor algorithms across all instances. Notably, in the majority of cases, CVEGE exhibits a significant performance advantage. Besides, when considering real-world applications, another critical metric is the stability of the algorithm. An algorithm may exhibit outstanding average performance, but if it is characterized by a standard deviation, it may not be suitable for real-world problems. Table 13 lists the standard deviation, the worst, and the best trial run among 30 independent experiments. From the statistical analysis and the stability perspective, our proposed CVEGE is quite competitive with state-of-the-art algorithms.

### 5.4 Performance analysis on feature selection tasks

Table 14 outlines the average fitness and classification accuracy in the wrapper-based feature selection tasks across eight datasets. Notably, our proposed CVEGE emerges as the top performer on five datasets, excelling in both average fitness and classification accuracy, which shows the competitiveness of our proposal. However, the degeneration of CVEGE compared with the original VEGE is observed in the case of

**Table 15** Experimental results on ten knapsack tasks

| Tasks | | PSO | DE | GBO | LCBO | ETWO | AOA | IAEO | CGO | VEGE | BCVEGE |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $f_1$ | Mean | 259.50 | 284.23 | 289.33 | 244.17 | 264.83 | 251.83 | 287.43 | 277.00 | **290.37** | 288.87 |
| | Std | 26.80 | 17.10 | 11.05 | 38.47 | 53.38 | 31.20 | 13.96 | 19.78 | 6.15 | 12.39 |
| | Worst | 201.00 | 241.00 | 246.00 | 157.00 | −1.00 | 194.00 | 238.00 | 241.00 | 274.00 | 252.00 |
| | Best | 294.00 | 295.00 | 295.00 | 295.00 | 295.00 | 295.00 | 295.00 | 295.00 | 295.00 | 295.00 |
| $f_2$ | Mean | 872.53 | 984.67 | **1004.07** | 878.57 | 867.03 | 846.67 | 972.37 | 967.03 | 975.47 | 990.33 |
| | Std | 61.25 | 35.25 | 18.91 | 75.36 | 47.45 | 62.90 | 30.13 | 34.01 | 32.18 | 32.95 |
| | Worst | 720.00 | 860.00 | 949.00 | 738.00 | 795.00 | 728.00 | 914.00 | 883.00 | 885.00 | 889.00 |
| | Best | 1018.00 | 1024.00 | 1024.00 | 1024.00 | 1018.00 | 964.00 | 1024.00 | 1024.00 | 1024.00 | 1024.00 |
| $f_3$ | Mean | 33.97 | 34.33 | **35.00** | 33.27 | **35.00** | 34.23 | 34.50 | 34.57 | 34.93 | **35.00** |
| | Std | 2.12 | 1.80 | 0.00 | 2.86 | 0.00 | 1.45 | 1.38 | 1.36 | 0.36 | 0.00 |
| | Worst | 33.97 | 34.33 | 35.00 | 33.27 | 35.00 | 34.23 | 34.50 | 34.57 | 34.93 | 35.00 |
| | Best | 35.00 | 35.00 | 35.00 | 35.00 | 35.00 | 35.00 | 35.00 | 35.00 | 35.00 | 35.00 |
| $f_4$ | Mean | 22.43 | 22.90 | 22.93 | 22.50 | 22.93 | 22.57 | 22.93 | 22.97 | 22.93 | **23.00** |
| | Std | 1.56 | 0.30 | 0.25 | 1.28 | 0.25 | 1.15 | 0.25 | 0.18 | 0.25 | 0.00 |
| | Worst | 16.00 | 22.00 | 22.00 | 16.00 | 22.00 | 18.00 | 22.00 | 22.00 | 22.00 | 23.00 |
| | Best | 23.00 | 23.00 | 23.00 | 23.00 | 23.00 | 23.00 | 23.00 | 23.00 | 23.00 | 23.00 |
| $f_5$ | Mean | 374.75 | 438.57 | 452.31 | 351.69 | 377.13 | 369.57 | 433.24 | 399.00 | 435.29 | **459.21** |
| | Std | 45.67 | 31.05 | 29.94 | 35.97 | 35.65 | 36.28 | 31.09 | 41.21 | 31.01 | 24.18 |
| | Worst | 290.01 | 383.11 | 377.07 | 281.82 | 316.68 | 299.48 | 379.99 | 330.50 | 363.34 | 411.55 |
| | Best | 481.07 | 481.07 | 481.07 | 423.57 | 469.16 | 430.03 | 481.07 | 481.07 | 481.07 | 481.07 |
| $f_6$ | Mean | 49.40 | 51.20 | 51.70 | 48.30 | 50.53 | 49.43 | 51.57 | 51.07 | 51.70 | **51.80** |
| | Std | 3.18 | 1.33 | 0.53 | 2.75 | 1.50 | 2.75 | 0.96 | 1.26 | 0.64 | 0.54 |
| | Worst | 40.00 | 47.00 | 50.00 | 43.00 | 47.00 | 40.00 | 48.00 | 48.00 | 50.00 | 50.00 |
| | Best | 52.00 | 52.00 | 52.00 | 52.00 | 52.00 | 52.00 | 52.00 | 52.00 | 52.00 | 52.00 |
| $f_7$ | Mean | 95.13 | 101.90 | 104.87 | 93.43 | 105.03 | 101.00 | 105.23 | 104.87 | 104.60 | **105.90** |
| | Std | 8.77 | 6.16 | 3.18 | 12.53 | 3.40 | 7.06 | 2.53 | 1.93 | 2.74 | 2.04 |
| | Worst | 66.00 | 81.00 | 93.00 | 66.00 | 95.00 | 72.00 | 96.00 | 102.00 | 96.00 | 97.00 |
| | Best | 107.00 | 107.00 | 107.00 | 107.00 | 107.00 | 107.00 | 107.00 | 107.00 | 107.00 | 107.00 |
| $f_8$ | Mean | 9656.90 | **9741.27** | 9740.17 | 9661.77 | 9718.50 | 9694.83 | 9737.90 | 9711.27 | 9734.23 | 9738.40 |
| | Std | 164.68 | 9.85 | 11.30 | 116.66 | 16.13 | 44.24 | 7.94 | 33.02 | 10.11 | 9.72 |
| | Worst | 8818.00 | 9710.00 | 9714.00 | 9114.00 | 9669.00 | 9598.00 | 9726.00 | 9591.00 | 9712.00 | 9721.00 |
| | Best | 9747.00 | 9756.00 | 9756.00 | 9745.00 | 9746.00 | 9736.00 | 9756.00 | 9749.00 | 9753.00 | 9755.00 |
| $f_9$ | Mean | 122.90 | 127.40 | 129.20 | 120.70 | 127.30 | 122.00 | 128.10 | 126.90 | 128.40 | **129.60** |
| | Std | 8.71 | 6.09 | 2.99 | 9.08 | 5.60 | 8.17 | 5.05 | 5.82 | 4.08 | 2.15 |
| | Worst | 106.00 | 109.00 | 118.00 | 106.00 | 109.00 | 106.00 | 109.00 | 109.00 | 118.00 | 118.00 |
| | Best | 130.00 | 130.00 | 130.00 | 130.00 | 130.00 | 130.00 | 130.00 | 130.00 | 130.00 | 130.00 |
| $f_{10}$ | Mean | 862.87 | 984.53 | **1007.63** | 860.70 | 865.47 | 854.20 | 977.27 | 964.47 | 964.53 | 994.67 |
| | Std | 64.37 | 42.80 | 19.42 | 55.79 | 58.41 | 63.78 | 42.88 | 48.19 | 38.85 | 20.16 |
| | Worst | 765.00 | 891.00 | 943.00 | 730.00 | 693.00 | 692.00 | 870.00 | 831.00 | 886.00 | 944.00 |
| | Best | 1025.00 | 1025.00 | 1025.00 | 968.00 | 944.00 | 964.00 | 1025.00 | 1025.00 | 1019.00 | 1025.00 |

mushrooms and dry bean datasets, and we aim to shed light on this observation by considering the principles of the No Free Lunch Theorem. No Free Lunch Theorem states that every pair of optimization algorithms has identical average performance on all possible problems, and if an algorithm performs better on a certain category problem, it must deteriorate on the rest problems to maintain the same average performance level. Thus we infer our proposed CVEGE may not excel in tasks that closely resemble the structural characteristics of the feature selection tasks on mushrooms and dry bean datasets.

## 5.5 Performance analysis on 0/1 knapsack problems

Our experimental results, as summarized in Table 15, provide valuable insights into the performance of CVEGE

in 0/1 knapsack problems when compared to competitor algorithms. Remarkably, CVEGE emerges as the victor in six out of ten problem instances, demonstrating both its competitive edge and exceptional scalability in addressing transferable problems. Through the other metrics such as the standard deviation, the worst, and the best, the stability of our proposal on 0/1 knapsack problems becomes apparent. Moreover, in most cases, our proposed CVEGE can find the global optimum at least once except for $f_8$, which reveals the potential of CVEGE in tackling global combinatorial optimization problems.

### 5.6 Open topics for future research

The above numerical experiments and analysis show that our proposed CVEGE has achieved satisfactory performance on four various optimization performances. However, there exists ample room for further enhancements by introducing additional techniques into CVEGE. Here, we list some potential and open topics.

#### 5.6.1 Extending to various tasks

In this paper, our primary focus was on improving the overall performance of the original VEGE by augmenting its exploration capacity and its ability to escape local optima. As a result, we proposed the enhanced VEGE (CVEGE). To evaluate the performance and the scalability of CVEGE, we test it on four various optimization tasks, and the experimental results practically verify the competitive performance and distinguished scalability on transferable problems. In the future, we want to extend this distinguished scalability to other tasks such as more complex real-world simulation problems [4, 5], reinforcement learning [87, 88], classic traveling salesman problems (TSP) [89, 90], neural architecture search (NAS) [91, 92] and so other domains.

#### 5.6.2 Machine learning assisted VEGE

Machine learning techniques have been successfully introduced to the EC community, with one prominent approach being the use of surrogate models [93, 94]. The surrogate (ensemble) assisted methodology has found extensive application in the realm of expensive optimization problems, which leverages the relatively inexpensive surrogate model to estimate the promising solutions rather than directly evaluated by the computationally expensive objective function. In the future, the embedding of the surrogate model to CVEGE is a potential topic to effectively address expensive optimization problems.

## 6 Conclusion

In this study, we propose CVEGE as an advancement over the original VEGE algorithm. CVEGE incorporates several key improvements, including the integration of chaotic local search, the utilization of multiple seeding strategies, and the introduction of a specialized mutation module to address the limitations, such as optimization premature and unbalanced search capacity, of its predecessor. The proposed algorithm is rigorously evaluated across a range of scenarios, including CEC2020 benchmark functions of varying scales and real-world engineering problems. The experimental and statistical results underscore CVEGE's competitive performance against state-of-the-art optimization techniques. Additionally, the ablation experiments on 30-D and 50-D CEC2020 benchmark functions are conducted to investigate the contribution of the proposed strategies independently. Based on the experimental and statistical results, the mutation module dominates the other two proposed strategies while the chaotic local search and multiple seeding operator also accelerate the convergence of optimization. Therefore, we suggest incorporating these three strategies with the conventional VEGE simultaneously.

Furthermore, we extend CVEGE's capabilities by embedding the simplified sigmoid transfer function, leading to the development of BCVEGE, a binary variant. BCVEGE's efficiency is demonstrated through testing on eight instances of wrapper-based feature selection tasks and standard 0/1 knapsack benchmark functions. The remarkable performance and scalability of BCVEGE validate its potential across diverse optimization scenarios.

Concluding this paper, we outline potential topics for further developing CVEGE. In future research endeavors, we plan to harness the potential of these algorithms to address even more complex and challenging optimization tasks, expanding their application domains and continuing to advance the field of optimization.

**Acknowledgements** This work was supported by JST SPRING Grant Number JPMJSP2119.

**Author contributions** RZ: Conceptualization, Methodology, Investigation, Writing - original draft, Writing - review & editing, and Funding acquisition. CZ: Conceptualization and Writing - review & editing. JY: Investigation, Methodology, Formal Analysis, and Writing - review & editing, and Project administration.

**Data availability** This research code can be downloaded from https://github.com/RuiZhong961230/CVEGE.

## Declarations

**Conflict of interest** The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## References


1. Mirjalili S, Mirjalili SM, Lewis A (2014) Grey wolf optimizer. Adv Eng Softw 69:46–61. https://doi.org/10.1016/j.advengsoft.2013.12.007
2. Mirjalili S, Lewis A (2016) The whale optimization algorithm. Adv Eng Softw 95:51–67. https://doi.org/10.1016/j.advengsoft.2016.01.008
3. Yu J (2022) Vegetation evolution: an optimization algorithm inspired by the life cycle of plants. Int J Comput Intell Appl. https://doi.org/10.1142/S1469026822500109
4. Mehrpanahi A, Hamidavi A, Ghorbanifar A (2018) A novel dynamic modeling of an industrial gas turbine using condition monitoring data. Appl Therm Eng 143:507–520. https://doi.org/10.1016/j.applthermaleng.2018.07.081
5. Xiao Q, Li C, Tang Y, Pan J, Yu J, Chen X (2019) Multi-component energy modeling and optimization for sustainable dry gear hobbing. Energy 187:115911. https://doi.org/10.1016/j.energy.2019.115911
6. Pierezan J, Maidl G, Massashi Yamao E, dos Santos Coelho L, Cocco Mariani V (2019) Cultural coyote optimization algorithm applied to a heavy duty gas turbine operation. Energy Convers Manag 199:111932. https://doi.org/10.1016/j.enconman.2019.111932
7. Zhou Q, Yin Z, Zhang H, Wang T, Sun W, Tan C (2020) Performance analysis and optimized control strategy for a three-shaft, recuperated gas turbine with power turbine variable area nozzle. Appl Therm Eng 164:114353. https://doi.org/10.1016/j.applthermaleng.2019.114353
8. Taradeh M, Mafarja M, Heidari AA, Faris H, Aljarah I, Mirjalili S, Fujita H (2019) An evolutionary gravitational search-based feature selection. Inf Sci 497:219–239. https://doi.org/10.1016/j.ins.2019.05.038
9. Aljarah I, Habib M, Faris H, Al-Madi N, Heidari AA, Mafarja M, Elaziz MA, Mirjalili S (2020) A dynamic locality multi-objective salp swarm algorithm for feature selection. Comput Ind Eng 147:106628. https://doi.org/10.1016/j.cie.2020.106628
10. Song X-F, Zhang Y, Gong D-W, Gao X-Z (2022) A fast hybrid feature selection based on correlation-guided clustering and particle swarm optimization for high-dimensional data. IEEE Trans Cybern 52(9):9573–9586. https://doi.org/10.1109/TCYB.2021.3061152
11. Lu Z, Whalen I, Boddeti V, Dhebar Y, Deb K, Goodman E, Banzhaf W (2019) Nsga-net: neural architecture search using multi-objective genetic algorithm. In: Proceedings of the genetic and evolutionary computation conference. Association for Computing Machinery, New York, NY, USA, pp 419–427. https://doi.org/10.1145/3321707.3321729
12. Ahmad M, Abdullah M, Moon H, Yoo SJ, Han D (2020) Image classification based on automatic neural architecture search using binary crow search algorithm. IEEE Access 8:189891–189912. https://doi.org/10.1109/ACCESS.2020.3031599
13. Xue Y, Jiang P, Neri F, Liang J (2021) A multi-objective evolutionary approach based on graph-in-graph for neural architecture search of convolutional neural networks. Int J Neural Syst 31(09):2150035. https://doi.org/10.1142/S0129065721500350
14. Kennedy J, Eberhart R (1995) Particle swarm optimization. In: Proceedings of ICNN'95 - international conference on neural networks, 4, 1942–19484. https://doi.org/10.1109/ICNN.1995.488968
15. Li S, Chen H, Wang M, Heidari AA, Mirjalili S (2020) Slime mould algorithm: a new method for stochastic optimization. Futur Gener Comput Syst 111:300–323. https://doi.org/10.1016/j.future.2020.03.055
16. De Jong K (1988) Learning with genetic algorithms: an overview. Mach Learn 3:121–138. https://doi.org/10.1007/BF00113894
17. Storn R (1996) On the usage of differential evolution for function optimization. In: Proceedings of North American fuzzy information processing, pp 519–523. https://doi.org/10.1109/NAFIPS.1996.534789
18. Koza JR (1994) Genetic programming as a means for programming computers by natural selection. Stat Comput 4:87–112. https://doi.org/10.1007/BF00175355
19. Yao X, Liu Y, Lin G (1999) Evolutionary programming made faster. IEEE Trans Evol Comput 3(2):82–102. https://doi.org/10.1109/4235.771163
20. Zhang LM, Dahlmann C, Zhang Y (2009) Human-inspired algorithms for continuous function optimization. In: 2009 IEEE international conference on intelligent computing and intelligent systems, vol 1, pp 318–321. https://doi.org/10.1109/ICICISYS.2009.5357838
21. Rao RV, Savsani VJ, Vakharia DP (2011) Teaching-learning-based optimization: a novel method for constrained mechanical design optimization problems. Comput Aided Des 43(3):303–315. https://doi.org/10.1016/j.cad.2010.12.015
22. Dehghani M, Montazeri Z, Dehghani A, Ramirez-Mendoza R, Samet H, Guerrero J, Dhiman G (2020) Mlo: multi leader optimizer. Int J Intell Eng Syst 13:364–373
23. Dehghani M, Trojovska E, Zuščák T (2022) A new human-inspired metaheuristic algorithm for solving optimization problems based on mimicking sewing training. Sci Rep. https://doi.org/10.1038/s41598-022-22458-9
24. Mirjalili S (2016) Sca: a sine cosine algorithm for solving optimization problems. Knowl-Based Syst 96:120–133. https://doi.org/10.1016/j.knosys.2015.12.022
25. Talatahari S, Azizi M (2021) Chaos game optimization: a novel metaheuristic algorithm. Artif Intell Rev 54:917–1004. https://doi.org/10.1007/s10462-020-09867-w
26. Shaqfa M, Beyer K (2021) Pareto-like sequential sampling heuristic for global optimisation. Soft Comput 25:9077–9096. https://doi.org/10.1007/s00500-021-05853-8
27. Ahmadianfar I, Heidari AA, Gandomi AH, Chu X, Chen H (2021) Run beyond the metaphor: an efficient optimization algorithm based on runge kutta method. Expert Syst Appl 181:115079. https://doi.org/10.1016/j.eswa.2021.115079
28. Rashedi E, Nezamabadi-pour H, Saryazdi S (2009) Gsa: a gravitational search algorithm. Inf Sci 179(13):2232–2248. https://doi.org/10.1016/j.ins.2009.03.004. (**Special Section on High Order Fuzzy Sets**)
29. Eskandar H, Sadollah A, Bahreininejad A, Hamdi M (2012) Water cycle algorithm—a novel metaheuristic optimization method for solving constrained engineering optimization problems. Comput Struct 110–111:151–166. https://doi.org/10.1016/j.compstruc.2012.07.010
30. Dehghani M, Montazeri Z, Dehghani A, Seifi A (2017) Spring search algorithm: a new meta-heuristic optimization algorithm inspired by hooke's law. In: 2017 IEEE 4th international conference on knowledge-based engineering and innovation (KBEI), pp 0210–0214. https://doi.org/10.1109/KBEI.2017.8324975
31. Dehghani M, Samet H (2020) Momentum search algorithm: a new meta-heuristic optimization algorithm inspired by momentum conservation law. SN Appl Sci. https://doi.org/10.1007/s42452-020-03511-6
32. Lam AYS, Li VOK (2010) Chemical-reaction-inspired metaheuristic for optimization. IEEE Trans Evol Comput 14(3):381–399. https://doi.org/10.1109/TEVC.2009.2033580
33. Alatas B (2012) A novel chemistry based metaheuristic optimization method for mining of classification rules. Expert Syst Appl 39(12):11080–11088. https://doi.org/10.1016/j.eswa.2012.03.066

34. Talatahari S, Azizi M, Gandomi AH (2021) Material generation algorithm: a novel metaheuristic algorithm for optimization of engineering problems. Processes. https://doi.org/10.3390/pr9050859
35. Zheng S, Janecek A, Tan Y (2013) Enhanced fireworks algorithm. In: 2013 IEEE congress on evolutionary computation, pp 2069–2077. https://doi.org/10.1109/CEC.2013.6557813
36. Yu J, Takagi H (2019) Accelerating vegetation evolution with mutation strategy and gbased growth strategy. In: 2019 IEEE symposium series on computational intelligence (SSCI), pp 3033–3039. https://doi.org/10.1109/SSCI44817.2019.9003027
37. Yu J, Takagi H (2020) Multi-species generation strategy-based vegetation evolution. In: 2020 IEEE congress on evolutionary computation (CEC), pp 1–6. https://doi.org/10.1109/CEC48606.2020.9185677
38. Zhong R, Peng F, Zhang E, Yu J, Munetomo M (2023) Vegetation evolution with dynamic maturity strategy and diverse mutation strategy for solving optimization problems. Biomimetics. https://doi.org/10.3390/biomimetics8060454
39. Mafarja M, Aljarah I, Heidari AA, Hammouri AI, Faris H, Al-Zoubi A, Mirjalili S (2018) Evolutionary population dynamics and grasshopper optimization approaches for feature selection problems. Knowl-Based Syst 145:25–45. https://doi.org/10.1016/j.knosys.2017.12.037
40. Gao S, Yu Y, Wang Y, Wang J, Cheng J, Zhou M (2021) Chaotic local search-based differential evolution algorithms for optimization. IEEE Trans Syst Man Cybern Syst 51(6):3954–3967. https://doi.org/10.1109/TSMC.2019.2956121
41. Kohli M, Arora S (2018) Chaotic grey wolf optimization algorithm for constrained optimization problems. J Comput Des Eng 5(4):458–472. https://doi.org/10.1016/j.jcde.2017.02.005
42. Thiétart RA, Forgues B (1995) Chaos theory and organization. Organ Sci 6(1):19–31. https://doi.org/10.1287/orsc.6.1.19
43. Kaur G, Arora S (2018) Chaotic whale optimization algorithm. J Comput Des Eng 5(3):275–284. https://doi.org/10.1016/j.jcde.2017.12.006
44. Yuzgec U, Eser M (2018) Chaotic based differential evolution algorithm for optimization of baker's yeast drying process. Egypt Inf J 19(3):151–163. https://doi.org/10.1016/j.eij.2018.02.001
45. Li Y, Han T, Han B, Zhao H, Wei Z (2019) Whale optimization algorithm with chaos strategy and weight factor. J Phys Conf Ser 1213:032004. https://doi.org/10.1088/1742-6596/1213/3/032004
46. Zhang J, Sanderson AC (2009) Jade: adaptive differential evolution with optional external archive. IEEE Trans Evol Comput 13(5):945–958. https://doi.org/10.1109/TEVC.2009.2014613
47. Yang D, Liu Z, Zhou J (2014) Chaos optimization algorithms based on chaotic maps with different probability distribution and search speed for global optimization. Commun Nonlinear Sci Numer Simul 19(4):1229–1246. https://doi.org/10.1016/j.cnsns.2013.08.017
48. Pei Y (2013) A chaotic ergodicity based evolutionary computation algorithm. In: 2013 ninth international conference on natural computation (ICNC), pp 454–459. https://doi.org/10.1109/ICNC.2013.6818019
49. Prado RS, Silva RCP, Guimarães FG, Neto OM (2010) Using differential evolution for combinatorial optimization: a general approach. In: 2010 IEEE international conference on systems, man and cybernetics, pp 11–18. https://doi.org/10.1109/ICSMC.2010.5642193
50. Liu Y, Yin M, Gu W (2014) An effective differential evolution algorithm for permutation flow shop scheduling problem. Appl Math Comput 248:143–159. https://doi.org/10.1016/j.amc.2014.09.010
51. Tu Q, Chen X, Liu X (2019) Hierarchy strengthened grey wolf optimizer for numerical optimization and feature selection. IEEE Access 7:78012–78028. https://doi.org/10.1109/ACCESS.2019.2921793
52. Mirjalili S, Lewis A (2013) S-shaped versus v-shaped transfer functions for binary particle swarm optimization. Swarm Evol Comput 9:1–14. https://doi.org/10.1016/j.swevo.2012.09.002
53. Emary E, Zawbaa HM, Hassanien AE (2016) Binary grey wolf optimization approaches for feature selection. Neurocomputing 172:371–381. https://doi.org/10.1016/j.neucom.2015.06.083
54. Ghosh KK, Singh PK, Hong J, Geem ZW, Sarkar R (2020) Binary social mimic optimization algorithm with x-shaped transfer function for feature selection. IEEE Access 8:97890–97906. https://doi.org/10.1109/ACCESS.2020.2996611
55. Mirjalili S, Zhang H, Mirjalili S, Chalup S, Noman N (2020) A novel u-shaped transfer function for binary particle swarm optimisation. In: Soft computing for problem solving 2019, pp 241–259. Springer, Singapore. https://doi.org/10.1007/978-981-15-3290-0_19
56. Guo S-s, Wang J-s, Guo M-w (2020) Z-shaped transfer functions for binary particle swarm optimization algorithm. Comput Intell Neurosci 2020:1–21. https://doi.org/10.1155/2020/6502807
57. Kristiyanti DA, Sitanggang IS, Annisa Nurdiati S (2023) Feature selection using new version of v-shaped transfer function for salp swarm algorithm in sentiment analysis. Computation. https://doi.org/10.3390/computation11030056
58. Faris H, Mafarja MM, Heidari AA, Aljarah I, Al-Zoubi AM, Mirjalili S, Fujita H (2018) An efficient binary salp swarm algorithm with crossover scheme for feature selection problems. Knowl-Based Syst 154:43–67. https://doi.org/10.1016/j.knosys.2018.05.009
59. Remeseiro B, Bolon-Canedo V (2019) A review of feature selection methods in medical applications. Comput Biol Med 112:103375. https://doi.org/10.1016/j.compbiomed.2019.103375
60. Bouaguel W (2022) Efficient multi-classifier wrapper feature-selection model: application for dimension reduction in credit scoring. Comput Sci. https://doi.org/10.7494/csci.2022.23.1.4120
61. Du D-Z, Ko K-I, Hu X (2012) Design and analysis of approximation algorithms, vol 62. Springer, Berlin. https://doi.org/10.1007/978-1-4614-1701-9
62. Ali IM, Essam D, Kasmarik K (2021) Novel binary differential evolution algorithm for knapsack problems. Inf Sci 542:177–194. https://doi.org/10.1016/j.ins.2020.07.013
63. Kellerer H, Pferschy U, Pisinger D (2013) Knapsack problems. Springer, Berlin
64. Yaseen ZM, Sulaiman SO, Deo RC, Chau K-W (2019) An enhanced extreme learning machine model for river flow forecasting: state-of-the-art, practical applications in water resource engineering area and future research direction. J Hydrol 569:387–408. https://doi.org/10.1016/j.jhydrol.2018.11.069
65. Abdollahzadeh B, Soleimanian Gharehchopogh F (2022) A multi-objective optimization algorithm for feature selection problems. Eng Comput. https://doi.org/10.1007/s00366-021-01369-9
66. Wang P, Xue B, Liang J, Zhang M (2023) Differential evolution-based feature selection: a niching-based multiobjective approach. IEEE Trans Evol Comput 27(2):296–310. https://doi.org/10.1109/TEVC.2022.3168052
67. Emary E, Zawbaa HM, Hassanien AE (2016) Binary ant lion approaches for feature selection. Neurocomputing 213:54–65. https://doi.org/10.1016/j.neucom.2016.03.101. (**Binary Representation Learning in Computer Vision**)
68. Chuang L-Y, Chang H-W, Tu C-J, Yang C-H (2008) Improved binary pso for feature selection using gene expression data. Comput Biol Chem 32(1):29–38. https://doi.org/10.1016/j.compbiolchem.2007.09.005
69. Yue CT, Price, PNSKV (2020) Problem definitions and evaluation criteria for the cec 2020 special session and competition on single objective bound constrained numerical optimization.

In: Technical Report, Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore
70. Nguyen T (2020) A framework of Optimization Functions using Numpy (OpFuNu) for optimization problems. Zenodo. https://doi.org/10.5281/zenodo.3620960
71. Bayzidi H, Talatahari S, Saraee M, Lamarche C-P (2021) Social network search for solving engineering optimization problems. Comput Intell Neurosci 2021:1–32. https://doi.org/10.1155/2021/8548639
72. Heidari AA, Mirjalili S, Faris H, Aljarah I, Mafarja M, Chen H (2019) Harris hawks optimization: algorithm and applications. Futur Gener Comput Syst 97:849–872. https://doi.org/10.1016/j.future.2019.02.028
73. Yeniay O (2005) Penalty function methods for constrained optimization with genetic algorithms. Math Comput Appl 10:45–56. https://doi.org/10.3390/mca10010045
74. Kaggle. https://www.kaggle.com
75. Bhattacharjee KK, Sarmah SP (2014) Shuffled frog leaping algorithm and its application to 0/1 knapsack problem. Appl Soft Comput 19:252–263. https://doi.org/10.1016/j.asoc.2014.02.010
76. Storn R, Price K (1997) Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. J Global Optim 11:341–359. https://doi.org/10.1023/A:1008202821328
77. Ahmadianfar I, Bozorg-Haddad O, Chu X (2020) Gradient-based optimizer: a new metaheuristic optimization algorithm. Inf Sci 540:131–159. https://doi.org/10.1016/j.ins.2020.06.037
78. Khatri A, Gaba A, Rana K, Kumar V (2020) A novel life choice-based optimizer. Soft Comput. https://doi.org/10.1007/s00500-019-04443-z
79. Nguyen T, Hoang B, Nguyen G, Nguyen BM (2020) A new workload prediction model using extreme learning machine and enhanced tug of war optimization. Procedia Computer Science. The 11th International Conference on Ambient Systems, Networks and Technologies (ANT)/The 3rd International Conference on Emerging Data and Industry 4.0 (EDI40)/Affiliated Workshops, 170, 362–369. https://doi.org/10.1016/j.procs.2020.03.063
80. Hashim A, Hussain F, Houssein K, Mabrouk EM, Al-Atabany W (2021) Archimedes optimization algorithm: a new metaheuristic algorithm for solving optimization problems. Appl Intell 51:1–21. https://doi.org/10.1007/s10489-020-01893-z
81. Rizk-Allah RM, El-Fergany AA (2021) Artificial ecosystem optimizer for parameters identification of proton exchange membrane fuel cells model. Int J Hydrog Energy 46(75):37612–37627. https://doi.org/10.1016/j.ijhydene.2020.06.256. (**International Symposium on Sustainable Hydrogen 2019**)
82. Talatahari S, Azizi M (2021) Chaos game optimization: a novel metaheuristic algorithm. Artif Intell Rev. https://doi.org/10.1007/s10462-020-09867-w
83. Van Thieu N, Mirjalili S (2023) Mealpy: an open-source library for latest meta-heuristic algorithms in python. J Syst Arch 139:102871. https://doi.org/10.1016/j.sysarc.2023.102871
84. Yildiz BS, Kumar S, Panagant N, Mehta P, Sait SM, Yildiz AR, Pholdee N, Bureerat S, Mirjalili S (2023) A novel hybrid arithmetic optimization algorithm for solving constrained optimization problems. Knowl-Based Syst 271:110554. https://doi.org/10.1016/j.knosys.2023.110554
85. Holm S (1979) A simple sequentially rejective multiple test procedure. Scand J Stat 6(2):65–70
86. Wolpert DH, Macready WG (1997) No free lunch theorems for optimization. IEEE Trans Evol Comput 1(1):67–82. https://doi.org/10.1109/4235.585893
87. Han S, Sung Y (2019) Dimension-wise importance sampling weight clipping for sample-efficient reinforcement learning. arXiv:1905.02363
88. Wang Y, Zhang T, Chang Y, Wang X, Liang B, Yuan B (2022) A surrogate-assisted controller for expensive evolutionary reinforcement learning. Inf Sci 616:539–557. https://doi.org/10.1016/j.ins.2022.10.134
89. Panwar K, Deep K (2021) Discrete grey wolf optimizer for symmetric travelling salesman problem. Appl Soft Comput 105:107298. https://doi.org/10.1016/j.asoc.2021.107298
90. Demiral M (2021) Analysis of a hybrid whale optimization algorithm for traveling salesman problem. Mehmet Akif Ersoy Üniversitesi Fen Bilimleri Enstitüsü Dergisi 12, 469–476. https://doi.org/10.29048/makufebed.1003543
91. Fernandes Junior FE, Yen GG (2019) Particle swarm optimization of deep neural networks architectures for image classification. Swarm Evol Comput 49:62–74. https://doi.org/10.1016/j.swevo.2019.05.010
92. Termritthikun C, Jamtsho Y, Ieamsaard J, Muneesawang P, Lee I (2021) Eeea-net: an early exit evolutionary neural architecture search. Eng Appl Artif Intell 104:104397. https://doi.org/10.1016/j.engappai.2021.104397
93. Zhong R, Yu J, Zhang C, Munetomo M (2023) Surrogate ensemble-assisted hyper-heuristic algorithm for expensive optimization problems. Int J Comput Intell Syst. https://doi.org/10.1007/s44196-023-00346-y
94. Zhong R, Zhang E, Munetomo M (2023) Cooperative coevolutionary surrogate ensemble-assisted differential evolution with efficient dual differential grouping for large-scale expensive optimization problems. Complex Intell Syst. https://doi.org/10.1007/s40747-023-01262-6

# Terms and Conditions

Springer Nature journal content, brought to you courtesy of Springer Nature Customer Service Center GmbH (“Springer Nature”).
Springer Nature supports a reasonable amount of sharing of research papers by authors, subscribers and authorised users (“Users”), for small-scale personal, non-commercial use provided that all copyright, trade and service marks and other proprietary notices are maintained. By accessing, sharing, receiving or otherwise using the Springer Nature journal content you agree to these terms of use (“Terms”). For these purposes, Springer Nature considers academic use (by researchers and students) to be non-commercial.
These Terms are supplementary and will apply in addition to any applicable website terms and conditions, a relevant site licence or a personal subscription. These Terms will prevail over any conflict or ambiguity with regards to the relevant terms, a site licence or a personal subscription (to the extent of the conflict or ambiguity only). For Creative Commons-licensed articles, the terms of the Creative Commons license used will apply.
We collect and use personal data to provide access to the Springer Nature journal content. We may also use these personal data internally within ResearchGate and Springer Nature and as agreed share it, in an anonymised way, for purposes of tracking, analysis and reporting. We will not otherwise disclose your personal data outside the ResearchGate or the Springer Nature group of companies unless we have your permission as detailed in the Privacy Policy.
While Users may use the Springer Nature journal content for small scale, personal non-commercial use, it is important to note that Users may not:

1. use such content for the purpose of providing other users with access on a regular or large scale basis or as a means to circumvent access control;
2. use such content where to do so would be considered a criminal or statutory offence in any jurisdiction, or gives rise to civil liability, or is otherwise unlawful;
3. falsely or misleadingly imply or suggest endorsement, approval , sponsorship, or association unless explicitly agreed to by Springer Nature in writing;
4. use bots or other automated methods to access the content or redirect messages
5. override any security feature or exclusionary protocol; or
6. share the content in order to create substitute for Springer Nature products or services or a systematic database of Springer Nature journal content.

In line with the restriction against commercial use, Springer Nature does not permit the creation of a product or service that creates revenue, royalties, rent or income from our content or its inclusion as part of a paid for service or for other commercial gain. Springer Nature journal content cannot be used for inter-library loans and librarians may not upload Springer Nature journal content on a large scale into their, or any other, institutional repository.
These terms of use are reviewed regularly and may be amended at any time. Springer Nature is not obligated to publish any information or content on this website and may remove it or features or functionality at our sole discretion, at any time with or without notice. Springer Nature may revoke this licence to you at any time and remove access to any copies of the Springer Nature journal content which have been saved.
To the fullest extent permitted by law, Springer Nature makes no warranties, representations or guarantees to Users, either express or implied with respect to the Springer nature journal content and all parties disclaim and waive any implied warranties or warranties imposed by law, including merchantability or fitness for any particular purpose.
Please note that these rights do not automatically extend to content, data or other material published by Springer Nature that may be licensed from third parties.
If you would like to use or distribute our Springer Nature journal content to a wider audience or on a regular basis or in any other manner not expressly permitted by these Terms, please contact Springer Nature at

onlineservice@springernature.com