Mathematical Biosciences and Engineering

http://www.aimspress.com/journal/mbe

MBE, 20(8): 13542–13561.
DOI: 10.3934/mbe.2023604
Received: 08 April 2023
Revised: 25 May 2023
Accepted: 01 June 2023
Published: 14 June 2023

Research article

QL-ADIFA: Hybrid optimization using Q-learning and an adaptive logarithmic spiral-levy firefly algorithm

Shuang Tan[1], **Shangrui Zhao**[1,*] **and Jinran Wu**[2]

[1] School of Science, Wuhan University of Technology, Wuhan 430070, China
[2] Institute for Learning Sciences & Teacher Education, Australian Catholic University, Brisbane 4000, Australia

* **Correspondence:** Email: zhaosr@whut.edu.cn.

Abstract: Optimization problems are ubiquitous in engineering and scientific research, with a large number of such problems requiring resolution. Meta-heuristics offer a promising approach to solving optimization problems. The firefly algorithm (FA) is a swarm intelligence meta-heuristic that emulates the flickering patterns and behaviour of fireflies. Although FA has been significantly enhanced to improve its performance, it still exhibits certain deficiencies. To overcome these limitations, this study presents the Q-learning based on the adaptive logarithmic spiral-Levy flight firefly algorithm (QL-ADIFA). The Q-learning technique empowers the improved firefly algorithm to leverage the firefly's environmental awareness and memory while in flight, allowing further refinement of the enhanced firefly. Numerical experiments demonstrate that QL-ADIFA outperforms existing methods on 15 benchmark optimization functions and twelve engineering problems: cantilever arm design, pressure vessel design, three-bar truss design problem, and 9 constrained optimization problems in CEC2020.

Keywords: Q-learning algorithm; firefly algorithm; Meta-heuristics; optimization problems

1. Introduction

Optimization problems are ubiquitous in engineering and scientific research, involving finding the optimal parameter values that meet specific objective functions while satisfying constraints [1–3]. Traditional gradient-based optimization algorithms have limitations and may not be effective in practical scenarios. In recent years, numerous meta-heuristic algorithms have been developed, showing better performance in terms of target function values while reducing computation costs. Among them, natural-inspired meta-heuristic algorithms have gained attention and have been applied to various optimization problems [4, 5].

In the field of meta-heuristic algorithms, there are four main categories: swarm intelligence optimization algorithms, evolution algorithms, physics-based algorithms, and human-based algorithms [6,7]. The firefly algorithm (FA) [8], an example of a swarm intelligence optimization algorithm, is inspired by the behaviour and flashing patterns of fireflies. It has been found to be more effective than other algorithms such as particle swarm optimization in solving optimization problems. Consequently, various versions of FA have been applied to a wide range of fields, including architectural material design [9], travel itinerary design [10], tumour classification [11], and image segmentation [12]. To further enhance the performance of FA, scholars have made improvements in various aspects.

The improvement of FA is mainly divided into two angles: algorithm modification and mixing with other algorithms. The modified FA is chiefly based on two important factors: Light variation and Attraction [13]. For example, [14] introduced Levy flights into FA to create the Levy-Flight Firefly Algorithm (LF-FA), which increased its global searching capability. A cooperative hybrid firefly algorithm is proposed by [1] with multiple firefly algorithm populations, where each FA maintains population diversity by hybridization and communication with each other to prevent the proposed algorithm from falling into local optimum. Many hybrid FAs are proposed when firefly algorithms incorporate machine learning, heuristics, hybridization, and other techniques. A novel adaptive hybrid evolutionary firefly algorithm (AHEFA) [15] mixes FA and the differential evolution (DE) algorithm, and selects mutation operators according to the results of iterations, balancing exploration and exploitation. Elitist techniques are adopted in the selection phase to carry on the viable solutions of the target individuals to the next generation. An example is the adaptive logarithmic Spiral-Levy FA (ADIFA) proposed by [16], which combines LF-FA with logarithmic spiral paths and an adaptive approach to balance exploration and exploitation capabilities. [17] provided a novel chaotic sine-cosine firefly (CSCF) algorithm with numerous variants, which integrates the chaotic form of the sine cosine algorithm (SCA) and the firefly algorithm (FA). CSCF chooses the best chaos variant from various chaotic forms, improving convergence speed and efficiency. To address FA's weaknesses in exploration and early convergence, [18] introduce an opposition-based method into FA and combine it with a symbiotic organisms search (SOS) algorithm, called IOFASOS. The impact of SOS algorithms on solutions is large in the early stages of IOFASOS implementation and becomes smaller and smaller as iterations progress.

While previous improvements to the firefly algorithm have been effective in improving the reliability of the firefly positions, they do not fully utilize the information generated during the last path changes. As a result, they do not make full use of the firefly's knowledge and memory of the environment when flying. Using this information as the basis for position changes could lead to faster discovery of the global optimal solution.

Q-learning involves the agent selecting actions based on the current state of the environment, receiving rewards or penalties based on the outcomes of its actions, accumulating knowledge, and making future predictions to maximize cumulative returns [19]. Many articles have mixed Q-learning with other algorithms and achieved good results. [20] utilizes a Q-learning model for adaptive parameter control in the differential evolution (DE) algorithm, where Q-learning uses information from its memory to select the best combination of parameters at the beginning of each iteration. Q-learning can also be used in the local reinforcement stage [21], which is dedicated to selecting the optimal state-action pair based on knowledge and completing the transition from one heuristic algorithm to another. Q-learning is applied to the marine predators algorithm to help leverage historical iteration information to balance exploration and development [19]. As a solution to the issue of not fully utilizing the information generated during previous path changes, this paper proposes the integration of Q-learning into AD-IFA, resulting in a new algorithm named Q-learning based on an adaptive logarithmic Spiral-Levy FA (QL-ADIFA). By incorporating Q-learning into AD-IFA, fireflies can choose the optimal strategy from two operations, thereby achieving a more balanced exploration and exploitation capability. The contributions of this paper are two-fold:

1) proposing the Q-learning based on the logarithmic Spiral-Levy firefly algorithm (QL-ADIFA) to solve the global optimization problem with faster convergence and superior solutions compared to the original improvement of firefly algorithms; and
2) testing QL-ADIFA on 15 benchmark functions and twelve engineering problems, demonstrating its improved convergence performance over the improved firefly algorithm.

The structure of this paper is organized as follows: Section 2 presents an overview of the improved firefly algorithm and the Q-learning algorithm. Section 3 provides a detailed description of the proposed Q-learning based on the adaptive logarithmic spiral-Levy flight firefly algorithm (QL-ADIFA). In Section 4, the performance of QL-ADIFA is evaluated using 15 benchmark functions. In Section 5, the effectiveness of the proposed algorithm is demonstrated through its application to twelve engineering problems. Finally, Section 6 provides a summary of the contributions and conclusions of this paper.

2. Related work

2.1. Q-learning

Q-learning is an off-policy temporal-difference method proposed by Watkins, which estimates the value of the Q-function for each state-action pair to determine the optimal action strategy [22]. The state represents the movement of the agent currently being taken, and the action represents a change from one state of the agent to another. Since the state and action spaces of the problem addressed in this paper are finite and discrete, the value function can be recorded using a matrix. The Reward table is used to store the reward or penalty for each state-action pair, while the Q-table records the corresponding Q-value for each pair. At each decision point, the optimal action strategy is selected by comparing the Q-values of each available action in the current state, and the Q-table is iteratively updated using the Bellman equation to minimize the difference between Q-values for adjacent states [23]. The equation is as follows:

$$Q(s^{Iter}, a^{Iter}) = Q(s^{Iter}, a^{Iter}) + \lambda \left\{ r_{Iter+1} + \theta \max_{a} Q(s^{Iter+1}, a) - Q(s^{Iter}, a^{Iter}) \right\}, \tag{1}$$

where s^{Iter} and a^{Iter} represent the state and action in this iteration respectively, λ is the learning rate, θ is the attenuation factor, and r_{Iter+1} is the immediate return.

Set the maximum iteration number $Max\_Iter = 100$, s_1 and s_2 represent different states, and a_1 and a_2 represent different actions. Then the pseudo-code for the Q-learning algorithm is provided in Algorithm 1.

Algorithm 1: Q-learning algorithm pseudocode

```
1  Initialize Reward table ;
2  Set Q-table as a zero matrix of $m \times n$, $m = 2$, $n = 2$;
3  Set $Iter = 0$, $Max\_Iter = 100$;
4  Set state $s_i, i = 1, \cdots, m$ and action $a_i$, $i = 1, \cdots, n$;
5  State $s^0$ is randomly selected from $\{s_1, \cdots, s_m\}$;
6  while $Iter < Max\_Iter$ do
7      Select the best action $a^{Iter}$ from $s^{Iter}$ according to Q-table;
8      Execute action $a^{Iter}$ and get immediate feedback $r_{Iter+1}$;
9      Get the latest status $s^{Iter+1}$;
10     Acquiring the corresponding maximum value $s^{Iter+1}$ of Q-table;
11     Update Q-table according to Bellman equation;
12     Update status $s^{Iter+1}$;
13     $Iter = Iter + 1$;
14 return $Q$-$table$;
```

The Q-learning algorithm diagram is shown in Figure 1.

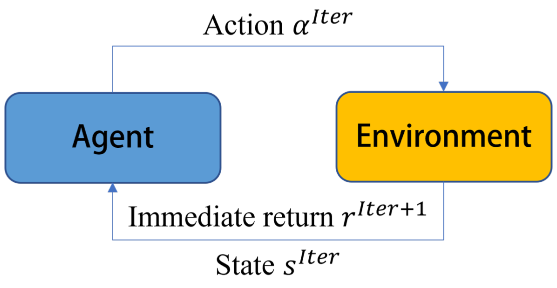

Figure 1. The flowchart of Q-learning.

2.2. *Improved firefly algorithm*

2.2.1. The firefly algorithm

The firefly algorithm is a meta-heuristic algorithm proposed in [24] based on the characteristics of firefly flashes, which is effective in dealing with nonlinear and multi-modal optimization problems. To simplify the firefly algorithm, fireflies follow the following three rules [8]:

1) All fireflies are of the same gender, and each firefly can only be attracted to the brighter ones.
2) The attraction of fireflies increases with brightness, while their brightness decreases with distance. Therefore, fireflies always move in the direction of the brighter ones, and the brightest firefly moves randomly.
3) The brightness of a firefly is equivalent to the value of the objective function.

The Euclidean distance between fireflies in d-dimension space can be expressed as:

$$r_{ij} = \sqrt{\sum_{p=1}^{d} (x_{i,p} - x_{j,p})^2}, \tag{2}$$

where $x_{i,p}$ is the pth component of the space coordinate of the ith firefly.

Since light is absorbed by the medium during propagation, the brightness of fireflies decreases with the increase of distance, so the attraction of fireflies can be expressed as:

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

where β_0 is the original attraction, γ is the light absorption coefficient, and r is the distance between fireflies.

The updated position of the ith firefly after it is attracted to a brighter jth firefly can be expressed as:

$$\mathbf{x}_{i,t+1} = \mathbf{x}_{i,t} + \beta_0 \cdot e^{-\gamma \cdot r_{ij}^2} \cdot (\mathbf{x}_{j,t} - \mathbf{x}_{i,t}) + \alpha \cdot (\mathbf{rand} - \mathbf{0.5}), \tag{4}$$

where $\mathbf{x}_{i,t}$ represents the position of the ith firefly in time t, **rand** is a d-dimensional uniform random vector between $[0, 1]^d$, and α is a parameter in $[0, 1]$.

2.2.2. The Levy-flight firefly algorithm

The firefly algorithm easily falls into the local minimum when dealing with global continuous optimization problems. To solve this problem, [25] proposed the Levy-flight firefly algorithm (LF-FA), inspired by the sudden and large turn of insects in straight flight.

Levy flying firefly algorithm replaces uniform distribution with Levy distribution, and updates the position of the u firefly after being attracted by brighter firefly j as follows:

$$\mathbf{x}_{i,t+1} = \mathbf{x}_{i,t} + \beta_0 \cdot e^{-\gamma \cdot r_{ij}{}^2} \cdot (\mathbf{x}_{j,t} - \mathbf{x}_{i,t}) + \alpha \cdot sign(\mathbf{rand} - \mathbf{0.5}) \otimes \mathbf{Levy}, \tag{5}$$

where $\otimes$ is the Hadamard product, and $sign(\cdot)$ is the sign function.

2.2.3. The logarithmic spiral path

Although the Levy flying firefly algorithm significantly enhances the exploration ability of the global space, it ignores the local development ability of the algorithm and the balance between exploration and development.

[26] proposed a logarithmic spiral path (LS) which could improve the local development ability of the algorithm by referring to the flight path taken by a peregrine falcon when looking for food. Therefore, [16] considered introducing the logarithmic spiral path as the direction of the improved firefly algorithm, and a new firefly position update mode can be designed as follows:

$$\mathbf{x}_{i,t+1} = \mathbf{x}_{i,t} + \beta_0 \cdot e^{-\gamma \cdot r_{ij}{}^2} \cdot (\mathbf{x}_{j,t} - \mathbf{x}_{i,t}) \otimes e^{b \cdot \mathbf{I}} \otimes \cos(2\pi \cdot \mathbf{I}), \tag{6}$$

where $\mathbf{I}$ is a d-dimensional uniform random vector in $[-1, 1]^d$ and is a constant used to define the shape of a logarithmic spiral.

2.2.4. An adaptive logarithmic spiral-Levy FA (AD-IFA)

The adaptive logarithmic spiral-Levy firefly algorithm (AD-IFA) proposed in [16] can solve the imbalance between exploration (Levy flight) and development (logarithmic spiral). An adaptive switch (ratio) method is proposed in AD-IFA, and the new position update formula is expressed as:

$$x_{i,t+1} = \begin{cases} x_{i,t} + \beta_0 e^{-\gamma \cdot r_{ij}{}^2}(x_{j,t} - x_{i,t}) + \alpha \cdot sign(\mathbf{rand} - 0.5) \otimes \mathbf{Levy}, & \text{if } u > R_t \\ x_{i,t} + \beta_0 e^{-\gamma \cdot r_{ij}{}^2}(x_{j,t} - x_{i,t}) \otimes e^{b \cdot \mathbf{I}} \otimes \cos(2\pi \cdot \mathbf{I}), & \text{if } u \leq R_t. \end{cases} \tag{7}$$

where u is a uniform random number between [0,1] and R_t is calculated in the last iteration. The value of R_{t+1} ranges in [0.5, 1], whose initial value is set as 0.5. Its specific expression is as follows,

$$R_{t+1} = \begin{cases} \dfrac{1}{1+\exp\left(-\frac{f_t}{f^*_{t-1}}\right)}, & \left\lfloor \lg \left| f^*_t \right| \right\rfloor \neq \left\lfloor \lg \left| f^*_{t-1} \right| \right\rfloor \\ \dfrac{1}{1+\exp\left(-\dfrac{f^*_t - \theta \cdot \left\lfloor \frac{f^*_f}{f} \right\rfloor}{f^*_{t-1} - \theta \cdot \left\lfloor \frac{f_{t-1}}{\theta} \right\rfloor}\right)}, & \text{else,} \end{cases} \tag{8}$$

where, $\theta = 10^{\lfloor \lg | f^*_t - f^*_{t-1} | \rfloor + 1}$. In Eq (8), f^*_t is the best fitness function value at the tth iteration, $\lg(\cdot) = \log_{10}(\cdot)$, and $\lfloor \cdot \rfloor$ is the floor function.

3. The proposed algorithm

This paper proposes the Q-learning based on the adaptive logarithmic spiral-Levy flight firefly algorithm (QL-ADIFA), which combines the strengths of the Q-learning algorithm and the adaptive logarithmic spiral-Levy flight firefly algorithm. QL-ADIFA leverages Q-learning to enhance the efficiency of the exploration and exploitation stages of the meta-heuristic algorithm. Additionally, it utilizes the meta-heuristic algorithm to better retain the information of the search space obtained during the iterative process. As a result, the QL-ADIFA algorithm achieves higher efficiency and effectiveness.

Table 1. Abbreviations used in this article.

Abbreviation	Full name
FA	Firefly Algorithm
LF-FA	The Levy-Flight Firefly Algorithm
AD-FIA	The adaptive logarithmic spiral-Levy flight firefly algorithm
QL-LSLFA	The Q-learning based on the logarithmic spiral-Levy flight firefly algorithm
QL-ADIFA	The Q-learning based on the adaptive logarithmic spiral-Levy flight firefly algorithm
NQL-ADIFA	The removed Q-learning based on the adaptive logarithmic spiral-Levy flight firefly algorithm

The proposed QL-ADIFA is divided into two parts. One is composed of the adaptive logarithmic spiral-Levy flight firefly algorithm (AD-IFA), that is, an adaptive switching (proportional) method is used to solve the problem of unbalanced exploration and exploitation. And the other part is composed of the Q-learning based on the logarithmic spiral-Levy flight firefly algorithm (QL-LSLFA), that is, Q-learning is used to solve the imbalance problem. During each iteration, one of the above two sections will be randomly selected to update the position of the firefly. And this random probability is set to 0.7 for choosing AD-IFA and 0.3 for choosing to use QL-LSLFA after testing in this article. To improve readability, abbreviations used in the article are recorded in Table 1.

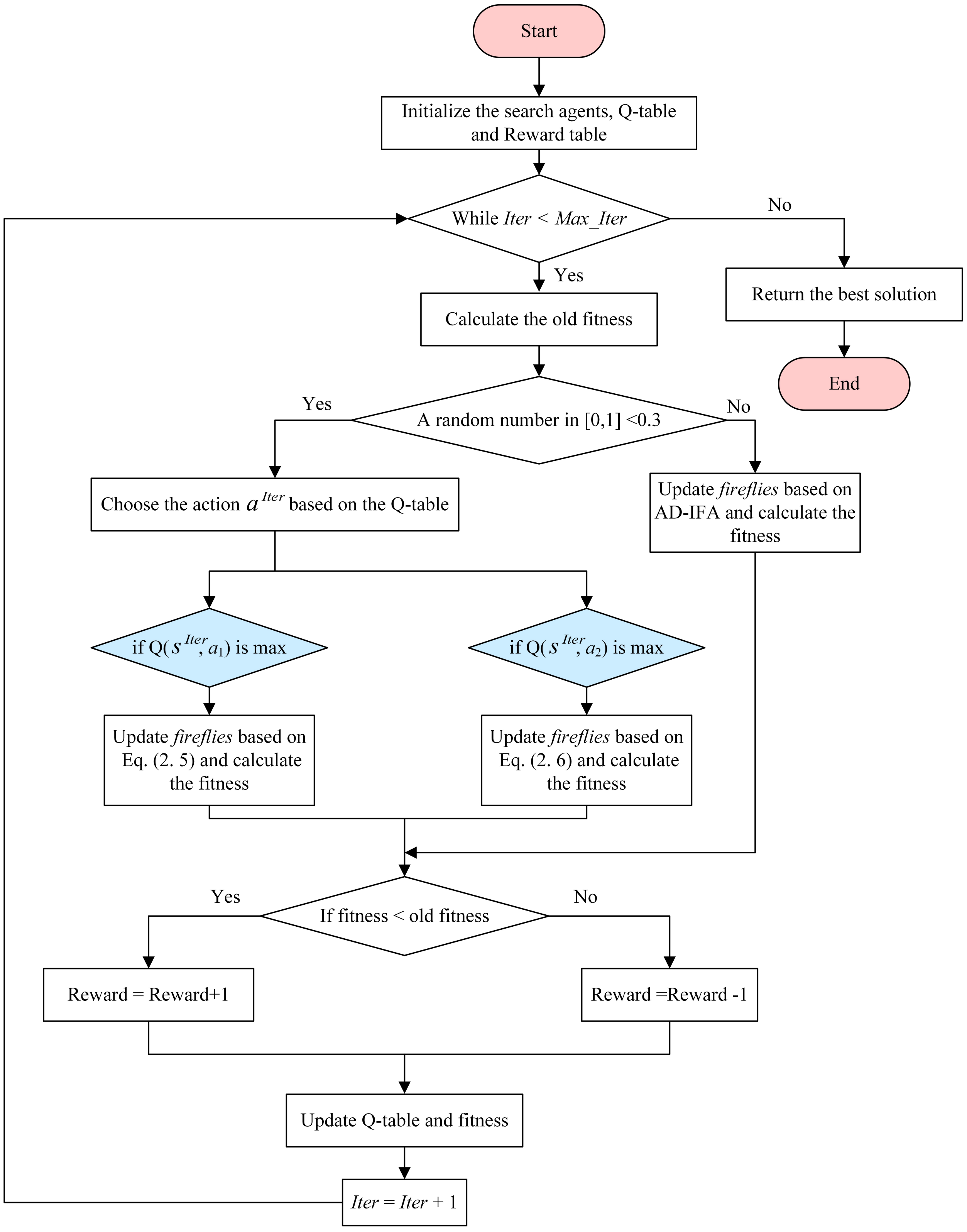
Start
Initialize the search agents, Q-table and Reward table
While $Iter < Max\_Iter$
No
Yes
Return the best solution
End
Calculate the old fitness
A random number in [0,1] <0.3
Yes
No
Choose the action a^{Iter} based on the Q-table
Update *fireflies* based on AD-IFA and calculate the fitness
if Q(s^{Iter}, a_1) is max
if Q(s^{Iter}, a_2) is max
Update *fireflies* based on Eq. (2. 5) and calculate the fitness
Update *fireflies* based on Eq. (2. 6) and calculate the fitness
If fitness < old fitness
Yes
No
Reward = Reward+1
Reward =Reward -1
Update Q-table and fitness
$Iter = Iter + 1$

Figure 2. The flowchart of QL-ADIFA.

The QL-LSLFA algorithm takes fireflies as agents, where there are two states of the agent: the exploration stage (the Levy-flight path) and the exploitation stage (the logarithmic spiral path) and two actions, i.e., switching from one stage to another stage. The flow chart of QL-LSLFA is shown in Figure 2. The Q-learning algorithm controls the action of fireflies by adapting to state transitions based on the Q-table. The fireflies learn good or bad behaviour based on the Reward table, which also rewards them for good behaviour (+1) and punishes them for bad behaviour (-1) to update the Reward table. The Q-table can better represent the firefly's performance in the process, allowing the firefly to obtain more appropriate actions. In QL-LSLFA, the position of the *i* firefly in the moment *t* changes based on:

$$\mathbf{x}_{i,t+1} = \begin{cases} \mathbf{x}_{i,t} + \beta_0 e^{-\gamma \cdot r_{ij}{}^2}(\mathbf{x}_{j,t} - \mathbf{x}_{i,t}) + \alpha \cdot sign(\mathbf{rand - 0.5}) \otimes \mathbf{Levy}, & \text{if } Q(s^{Iter}, a_1) \text{ is max} \\ \mathbf{x}_{i,t} + \beta_0 e^{-\gamma \cdot r_{ij}{}^2}(\mathbf{x}_{j,t} - \mathbf{x}_{i,t}) \otimes e^{b \cdot \mathbf{I}} \otimes \cos(2\pi \cdot \mathbf{I}), & \text{if } Q(s^{Iter}, a_2) \text{ is max.} \end{cases} \tag{9}$$

In Eq (9), a_1 represented the switch from the exploitation stage to the exploration stage, and a_2 represented the switch from the exploration stage to the exploitation stage.

In QL-LSLFA, fireflies are able to make adaptive judgments and choose the most appropriate actions according to the Q-learning algorithm. The improved steps of Q-learning can be summarized into five parts as follows:

1) The Q-table is initialized as a 2×2 zero matrix. The specific form of the Reward table is shown in Eq (10):

$$\begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix}. \tag{10}$$

2) According to each value of the Q-table in the current state, the action with the highest score is selected as the best action in the current stage.
3) Perform the selected action and calculate the new fitness value. The immediate reward is calculated as follows:

$$Reward = \begin{cases} Reward + 1, & \text{if the new function improves,} \\ Reward - 1, & \text{otherwise.} \end{cases} \tag{11}$$

4) Update the Q-table with Eq (1).
5) Update the location of the agent based on the new state.

Figure 2 shows the general flow chart of the proposed QL-ADIFA. From the initial phase of the QL-ADIFA, each search agent is independent of the other and continuously improves its behaviour based on Q-learning. The QL-ADIFA is executed iteratively until the termination condition is satisfied.

The feature of this algorithm is that it can effectively switch between different stages according to its own needs, so it can find the global solution effectively, and improve the efficiency of local searches.

The pseudo-code of the proposed QL-ADIFA algorithm is shown in Algorithm 2.

Algorithm 2: QL-ADIFA algorithm pseudo-code

```
1  Initialize Reward table ;
2  Set Q-table as a zero matrix of $m \times n$, $m = 2$ and $n = 2$;
3  Set $Iter = 0$, $Max\_Iter = 500$, and $N$ is the population size of fireflies;
4  Set state $s_i, i = 1, \cdots, m$ and action $a_i$, $i = 1, \cdots, n$;
5  State $s^0$ is randomly selected from $s_i$, $i = 1, \cdots, m$;
6  Construct fitness function $f(\mathbf{x})$, $\mathbf{x} = (\mathbf{x}_1, \cdots, \mathbf{x}_d)^T$;
7  Initialize firefly population $\mathbf{x}_i$ $(i = 1, \cdots, N)$;
8  Set various initial rates and define the light absorption coefficient $\gamma$;
9  while $Iter < Max\_Iter$ do
10     for $i = 1 : N$ do
11         for $j = 1 : N$ do
12             The light intensity $I_i$ of $\mathbf{x}_i$ is calculated by fitness function $I_i = f(\mathbf{x}_i)$;
13             if $I_j > I_i$ then
14                 Generate a random number $p$ in [0,1];
15                 if $p < 0.3$ then
16                     if The value of $Q(s^{Iter}, a_1)$ is the max in Q-table then
17                         According to Eq (5), enter the *exploration* stage and update the position
                             of $i$th firefly;
18                     else
19                         According to Eq (6), enter the *exploitation* stage and update the position
                             of $i$th firefly;
20                 else
21                     Use AD-IFA for the $i$th firefly's position updates;
22                 if The new function value has been improved then
23                     Reward = Reward + 1;
24                 else
25                     Reward = Reward - 1;
26     The attractive force varies with the distance $r$ according to $exp(-\gamma r^2)$;
27     Evaluate new solutions and update light intensity;
28     Rank fireflies and find the fitness value $f_l^*$ of fireflies in the best position;
29     Iter = Iter + 1;
30 return *Final result*;
```

4. Numerical simulations

Numerical simulations based on 15 benchmark functions are performed to verify the performance of the proposed method. Table 2 shows the details of these functions.

Table 2. The description of 15 reference functions.

Function	Description	Dimension (d)	Range	f_{min}
f_1	$f(x) = \sum_{i=0}^{d} \lvert x_i\rvert + \prod_{i=0}^{d} \lvert x_i\rvert$	30	[-10,10]	0
f_2	$f(x) = \sum_{i=1}^{d-1}[100(x_i^2 - x_{i+1})^2 + (1 - x_i)^2]$	30	[-30,30]	0
f_3	$f(x) = \sum_{i=1}^{d-1}\left[i\left(2x_i^2 - x_{i-1}\right)^2\right] + (x_1 - 1)^2$	50	[-10,10]	0
f_4	$f(x) = \sum_{i=0}^{d} ix_i^4 + \text{random}(0,1)$	50	[-1.28,1.28]	0
f_5	$f(x) = \sum_{i=1}^{d}\left(-x_i \sin\left(\sqrt{\lvert x_i\rvert}\right)\right)$	50	[-500,500]	$-418.9828\times d$
f_6	$f(x) = -20\exp\left(-0.2\sqrt{\frac{1}{d}\sum_{i=1}^{d} x_i^2}\right) - \exp\left(\frac{1}{d}\sum_{i=1}^{d}\cos(2\pi x_i)\right) + 20 + e$	50	[-32,32]	0
f_7	$f(x) = 1 + \frac{1}{4000}\sum_{i=1}^{d} x_i^2 - \prod_{i=1}^{d}\cos\left(\frac{x_i}{\sqrt{i}}\right)$	50	[-600,600]	0
f_8	$f(x) = \sum_{i=1}^{d} ix_i^2$	50	[-5.12,5.12]	0
f_9	$f(x) = \left(\frac{1}{500} + \sum_{j=1}^{25}\frac{1}{j+\sum_{i=1}^{2}\left(x_i - a_{ij}\right)}\right)^{-1}$	2	[-65,65]	0.9980
f_{10}	$f(x) = \sum_{i=1}^{11}\left[a_i - \frac{x_1\left(b_i^2+b_ix_2\right)}{b_1^2+b_1x_3+x_4}\right]^2$	4	[-5,5]	0.0003
f_{11}	$f(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$	2	[-5,5]	-1.0316
f_{12}	$\text{f(x)} = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos x_1 + 10$	2	[-5,5]	0.398
f_{13}	$f(x) = [1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)] \times[30 + (2x_1 - 3x_2)^2(18 - 32x_i + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)]$	2	[-2,2]	3
f_{14}	$f(x) = -\sum_{i=1}^{7}\left[(X - a_i)(X - a_i)^T + c_i\right]^{-1}$	4	[0,10]	-10.4029
f_{15}	$f(x) = -\sum_{i=1}^{10}\left[(X - a_i)(X - a_i)^T + c_i\right]^{-1}$	4	[0,10]	-10.5364

4.1. Parameter settings

This paper shows the results of the proposed QL-ADIFA and compares it with the original algorithm in different test functions. 100 iterations and 25 search agents are used to execute the proposed algorithm QL-ADIFA. To display the capabilities of the proposed method, the results of QL-LSLFA and the algorithms below 1) AD-IFA [16]; 2) FA [24]; 3) LFFA [16]; 4) the removed Q-learning based on the adaptive logarithmic spiral-Levy flight firefly algorithm (NQL-ADIFA); 5) A quasi-opposition learning and Q-learning based marine predators algorithm (QQLMPA) [19]; 6) an innovative optimizer named weighted mean of vectors (INFO) [27] carried out the comparison and analysis. The NQL-ADIFA here refers to eliminating the impact of Q-learning in QL-ADIFA, that is, in the QL-LSLFA part, the Q-learning algorithm is no longer used for exploration and exploitation strategy selection, but a random selection of strategies. The experiment was conducted in the Matlab R2017a environment, which has a 1.4 GHz quad-core Intel core i5 and 8 GB of RAM. The experimental settings for the parameters in all firefly algorithms (FA, LF-FA, AD-IFA, our QL-ADIFA) are as follows: the randomization parameter $\alpha = 0.2$, the fixed light absorption coefficient $\gamma = 1$ and the attractiveness at $r = 0$ is $\beta_0 = 1$.

To ensure the fairness of the experiment, this study obtained the average results with 25 runs. In particular, considering that the study rate θ is usually set to a high value at the beginning and then gradually decreases with time steps, it is set as follows:

$$\theta(Iter) = 1 - (0.9 \times \frac{Iter}{Max\_Iter}), \tag{12}$$

where $Iter$ is the current iteration and Max_Iter is the total number of iterations.

4.2. Result analysis

Tables 3 and 4 provide the performance details, where the mean (Avg) and standard deviation (Std) values are used to evaluate the results of QL-ADIFA and other algorithms. The best results are presented in bold, and all algorithms are ranked from best to worst based on their average performance. Additionally, the Wilcoxon rank sum test is conducted with a 95% confidence level to calculate the p-values and h-values, which demonstrates that QL-ADIFA is significantly different from other algorithms. 'NaN' represents 'not available'. Moreover, Figure 3 illustrates the convergence curves of six functions for all algorithms, which indicates that QL-ADIFA performs better than other algorithms in terms of searching for prey and achieving better results during fewer iterations. And QL-ADIFA has better exploration and development capabilities, making it easier to avoid local optima.

QL-ADIFA exhibits superior exploration capabilities that enable it to discover better solutions compared to the original algorithm. The results presented in Tables 3 and 4 show that QL-ADIFA achieves an Avg that is closer to the global optimal Avg when compared to other algorithms. However, compared with the two excellent algorithms of QQLMPA and INFO, QL-ADIFA still has a lot of room for improvement. Compared only to the FA series algorithms, QL-ADIFA performs the best in terms of Avg except for f_3. Furthermore, the Std value of QL-ADIFA is the lowest among 50% of the benchmark functions, indicating that its excellent performance is robust. The convergence curves of the six functions shown in Figure 3 illustrate that QL-ADIFA outperforms other algorithms in terms of convergence speed. Overall, the proposed algorithm achieves good performance in terms of accuracy and speed after the improvements are made.

Tables 3 and 4 present the p-values and h-values obtained from the non-parametric Wilcoxon rank sum statistical test. The test was conducted to determine whether QL-ADIFA performs significantly better than other algorithms. The results show that in most benchmark functions, the p-value is less than 0.05 and the h-value is 1 between QL-ADIFA and each of the other algorithms. This indicates that the advantage of QL-ADIFA over the other algorithms is credible and significant. Specifically, the h-value is 1 in 52% of functions, indicating a significant difference between QL-ADIFA and other algorithms. Additionally, the p-value is less than 5% in 52% of benchmark functions, indicating that QL-ADIFA is effective in solving most functions.

Table 3. Results of QL-ADIFA and other algorithms on 15 benchmark functions (f_1–f_9).

Function		QL-ADIFA	NQL-ADIFA	AD-IFA	FA	LF-FA	QQLMPA	INFO	f_{min}
f_1	Avg	7.02E+01	8.89E+01	7.50E+01	8.42E+15	4.31E+14	2.15E-03	2.87E-07	0.00E+00
	Best	5.15E+01	5.83E+01	4.06E+01	2.16E+07	1.72E+04	1.41E-25	2.53E-10	
	Std	1.72E+01	2.24E+01	2.06E+01	2.63E+16	1.28E+15	1.33E-03	8.89E-07	
	Rank	3	5	4	7	6	2	1	
	$h-value$	-	1	1	1	1	0	0	
	$p-value$	-	1.70E-06	4.10E-04	1.67E-31	1.94E-31	8.96E-02	8.96E-02	
f_2	Avg	1.67E+07	1.92E+07	1.86E+07	5.10E+08	4.05E+08	4.88E+01	4.82E+01	0.00E+00
	Best	2.87E+06	1.12E+07	7.34E+06	4.42E+08	2.69E+08	4.87E+01	4.71E+01	
	Std	1.17E+07	7.84E+06	1.00E+07	4.68E+07	6.09E+07	5.51E-02	6.00E-01	
	Rank	3	5	4	7	6	2	1	
	$h-value$	-	1	1	1	1	0	0	
	$p-value$	-	3.39E-08	1.06E-04	1.05E-32	1.05E-32	8.96E-02	8.96E-02	
f_3	Avg	2.01E+05	2.51E+05	1.95E+05	5.78E+06	3.74E+06	7.70E-01	6.68E-01	0.00E+00
	Best	6.72E+04	6.79E+04	8.39E+04	4.27E+06	3.02E+06	6.79E-01	6.67E-01	
	Std	1.00E+05	1.52E+05	1.03E+05	6.57E+05	5.46E+05	8.34E-02	7.88E-04	
	Rank	4	5	3	7	6	2	1	
	$h-value$		1	0	1	1	0	0	
	$p-value$		1.48E-07	7.37E-01	1.05E-32	1.05E-32	8.96E-02	8.96E-02	
f_4	Avg	1.39E+01	1.62E+01	1.78E+01	2.54E+02	1.93E+02	6.40E-03	8.26E-03	0.00E+00
	Best	3.23E+00	3.37E+00	9.00E+00	1.80E+02	1.38E+02	1.92E-03	3.00E-03	
	Std	9.55E+00	1.14E+01	9.84E+00	3.23E+01	3.36E+01	4.08E-03	6.73E-03	
	Rank	3	5	4	7	6	2	1	
	$h-value$	-	1	1	1	1	0	0	
	$p-value$	-	3.61E-02	1.10E-03	1.04E-31	7.96E-32	8.96E-02	8.96E-02	
f_5	Avg	-6.11E+03	-5.51E+03	-5.95E+03	-2.67E+03	-3.10E+03	-7.66E+03	-1.25E+04	-2.09E+04
	Best	-8.47E+03	-6.94E+03	-7.59E+03	-3.37E+03	-3.86E+03	-9.40E+03	-1.35E+04	
	Std	1.42E+03	7.98E+02	9.34E+02	5.53E+02	6.14E+02	1.41E+03	8.21E+02	
	Rank	3	5	4	7	6	2	1	
	$h-value$	-	1	0	1	1	0	0	
	$p-value$	-	1.33E-03	3.98E-01	1.05E-32	1.96E-29	8.96E-02	8.96E-02	
f_6	Avg	1.26E+01	1.42E+01	1.45E+01	2.05E+01	2.02E+01	9.78E-04	4.67E-07	0.00E+00
	Best	6.75E+00	1.20E+01	1.27E+01	2.03E+01	2.00E+01	8.88E-16	8.49E-10	
	Std	3.39E-01	1.40E+00	1.08E+00	4.04E-01	1.01E-01	5.95E-04	9.18E-07	
	Rank	3	5	4	7	6	2	1	
	$h-value$	-	1	1	1	1	0	0	
	$p-value$	-	1.00E-05	6.62E-13	1.05E-32	1.05E-32	8.96E-02	8.96E-02	
f_7	Avg	1.47E+02	2.04E+02	1.56E+02	1.16E+03	1.14E+03	8.62E-05	1.58E-14	0.00E+00
	Best	1.06E+02	1.26E+02	7.48E+01	1.02E+03	9.98E+02	5.81E-06	0.00E+00	
	Std	3.68E+01	6.95E+01	5.33E+01	9.38E+01	9.49E+01	5.00E-05	4.48E-14	
	Rank	3	5	4	7	6	2	1	
	$h-value$	-	1	1	1	1	0	0	
	$p-value$	-	3.61E-02	1.10E-03	1.04E-31	7.96E-32	8.96E-02	8.96E-02	
f_8	Avg	1.12E+03	1.39E+03	1.35E+03	7.60E+03	5.03E+03	3.36E-06	8.26E-17	0.00E+00
	Best	5.56E+02	7.06E+02	9.69E+02	6.27E+03	3.89E+03	7.50E-07	6.03E-22	
	Std	4.17E+02	6.01E+02	4.59E+02	6.36E+02	5.21E+02	3.21E-06	1.61E-16	
	Rank	3	5	4	7	6	2	1	
	$h-value$	-	1	1	1	1	0	0	
	$p-value$	-	2.14E-11	3.62E-12	1.05E-32	1.05E-32	8.96E-02	8.96E-02	
f_9	Avg	-6.11E+03	-5.51E+03	-5.95E+03	-2.67E+03	-3.10E+03	-7.66E+03	-1.25E+04	9.98E-01
	Best	-8.47E+03	-6.94E+03	-7.59E+03	-3.37E+03	-3.86E+03	-9.40E+03	-1.35E+04	
	Std	1.42E+03	7.98E+02	9.34E+02	5.53E+02	6.14E+02	1.41E+03	8.21E+02	
	Rank	3	5	4	7	6	2	1	
	$h-value$	-	1	0	1	1	0	0	
	$p-value$	-	1.33E-03	3.98E-01	1.05E-32	1.96E-29	8.96E-02	8.96E-02	

Table 4. Results of QL-ADIFA and other algorithms on 15 benchmark functions (f_{10}–f_{15}).

Function		QL-ADIFA	NQL-ADIFA	AD-IFA	FA	LF-FA	QQLMPA	INFO	f_{min}
f_{10}	Avg	1.11E-03	5.17E-03	1.15E-03	7.35E-02	1.18E-02	4.20E-04	8.50E-03	4.20E-04
	Best	4.50E-04	7.77E-04	3.68E-04	8.29E-03	1.39E-03	3.08E-04	3.07E-04	
	Std	8.21E-03	8.11E-03	3.39E-04	7.28E-02	1.25E-02	1.27E-04	1.02E-02	
	Rank	2	4	3	7	6	1	5	
	$h-value$	-	1	1	1	1	0	0	
	$p-value$	-	9.30E-03	5.32E-12	1.07E-30	4.87E-16	8.96E-02	6.28E-01	
f_{11}	Avg	-1.03E+00	-1.03E+00	-1.03E+00	-6.86E-03	-1.03E+00	-1.03E+00	-1.03E+00	-1.03E+00
	Best	-1.03E+00	-1.03E+00	-1.03E+00	-1.03E+00	-1.03E+00	-1.03E+00	-1.03E+00	
	Std	2.52E-09	2.52E-09	1.13E-09	1.25E+00	3.84E-09	2.49E-08	2.22E-16	
	Rank	1	1	1	7	1	1	1	
	$h-value$	-	0	0	1	0	0	0	
	$p-value$	-	2.70E-01	1.86E-01	1.43E-29	5.39E-01	2.12E-01	8.96E-02	
f_{12}	Avg	3.98E-01	3.98E-01	3.98E-01	5.42E-01	3.98E-01	3.98E-01	3.98E-01	3.98E-01
	Best	3.98E-01	3.98E-01	3.98E-01	3.98E-01	3.98E-01	3.98E-01	3.98E-01	
	Std	8.24E-10	1.97E-09	7.87E-10	2.68E-01	3.48E-09	2.45E-06	0.00E+00	
	Rank	1	1	1	7	1	1	1	
	$h-value$	-	0	0	1	0	0	0	
	$p-value$	-	7.31E-01	5.31E-01	8.11E-26	3.90E-01	7.03E-01	8.96E-02	
f_{13}	Avg	3.00E+00	3.00E+00	3.00E+00	2.06E+01	3.00E+00	3.00E+00	3.00E+00	3.00E+00
	Best	3.00E+00	3.00E+00	3.00E+00	3.00E+00	3.00E+00	3.00E+00	3.00E+00	
	Std	2.61E-07	1.12E-07	1.19E-07	2.70E+01	6.12E-07	2.57E-07	3.56E-15	
	Rank	1	1	1	7	1	1	1	
	$h-value$	-	0	0	1	0	0	0	
	$p-value$	-	2.68E-01	2.12E-01	3.89E-27	5.01E-01	9.63E-02	8.96E-02	
f_{14}	Avg	-7.34E+00	-7.20E+00	-7.17E+00	-9.37E-01	-5.92E+00	-1.03E+01	-7.35E+00	-1.04E+01
	Best	-1.04E+01	-1.04E+01	-1.04E+01	-2.31E+00	-1.04E+01	-1.04E+01	-1.04E+01	
	Std	3.10E+00	3.45E+00	3.94E+00	6.52E-01	3.37E+00	2.81E-01	3.96E+00	
	Rank	3	4	5	7	6	1	2	
	$h-value$	-	0	1	1	1	0	0	
	$p-value$	-	1.34E-01	1.41E-02	1.31E-30	1.85E-06	8.96E-02	8.96E-02	
f_{15}	Avg	-8.49E+00	-6.30E+00	-5.92E+00	-1.72E+00	-4.76E+00	-1.05E+01	-8.41E+00	-1.05E+01
	Best	-1.05E+01	-1.05E+01	-1.05E+01	-3.30E+00	-1.05E+01	-1.05E+01	-1.05E+01	
	Std	3.07E+00	3.60E+00	3.37E+00	9.71E-01	3.18E+00	6.79E-02	3.49E+00	
	Rank	2	4	5	7	6	1	3	
	$h-value$	-	1	1	1	1	0	0	
	$p-value$	-	1.49E-12	2.55E-13	1.89E-26	9.33E-16	8.96E-02	1.88E-01	

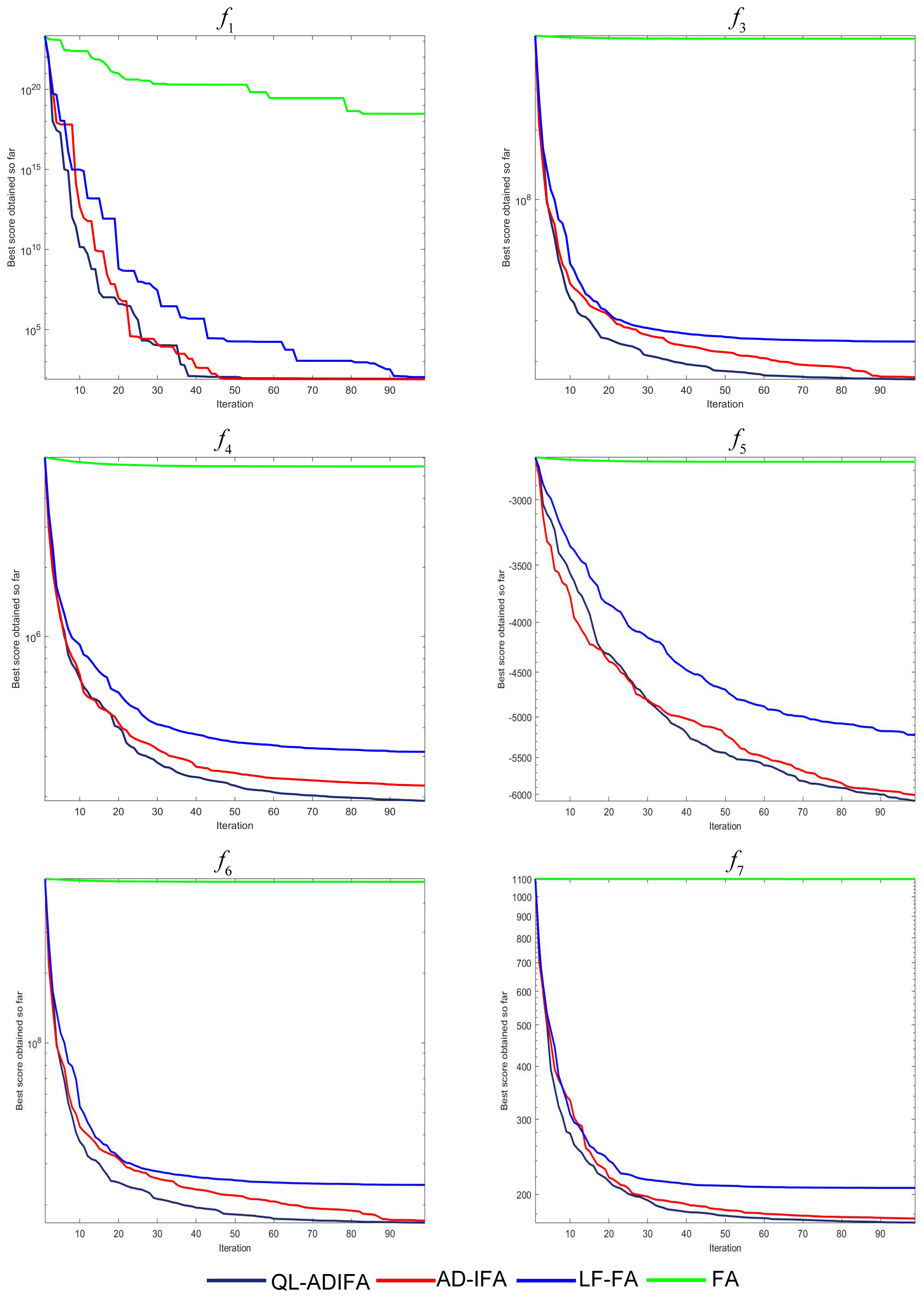

Figure 3. Select the convergence curve of the comparison algorithm on the reference function.

5. Engineering design problems

This section presents the practical application of QL-ADIFA in solving engineering design problems, specifically the cantilever beam design and pressure vessel design. Similar to the benchmark function test, QL-ADIFA is evaluated using 100 iterations repeated 25 times for each problem.

5.1. The cantilever beam design

The first practical engineering problem involves the weight optimization of a square-section cantilever beam, specifically the design of a cantilever beam. The beam is made up of 5 hollow square blocks of constant thickness, where the height of each block is a decision variable and the thickness is fixed. One end of the beam is rigidly supported, and a vertical force acts on the free node of the cantilever. The objective is to minimize the weight of the cantilever beam and can be expressed as follows:

$$\text{Minimize } f(x) = 0.0624(x_1 + x_2 + x_3 + x_4 + x_5).$$

$$\text{Subject to:}$$

$$\begin{cases} g(x) = \dfrac{61}{x_1^3} + \dfrac{37}{x_2^3} + \dfrac{19}{x_3^3} + \dfrac{7}{x_4^3} + \dfrac{1}{x_5^3} - 1 \le 0, \\ 0.01 \le x_i \le 100, \quad i = 1, 2, 3, 4, 5. \end{cases}$$

Table 5 presents a comparison of the best results achieved by different algorithms. The best result obtained by QL-ADIFA outperforms the best result of the other algorithms. The optimal value of QL-ADIFA is 1.7505 when the decision variables $x_1 = 4.7861$, $x_2 = 7.9613$, $x_3 = 7.7762$, $x_4 = 4.6585$, and $x_5 = 2.8702$.

Table 5. Comparison of optimization design of cantilever beam by different algorithms.

	QL-ADIFA	NQL-ADIFA	AD-IFA	FA	LF-FA	QQLMPA	INFO
x_1	8.72	4.79	5.22	8.73	8.51	22.05	19.97
x_2	9.56	5.85	10.20	16.70	8.80	4.55	55.09
x_3	4.67	30.14	26.36	12.58	2.84	29.85	2.83
x_4	2.25	3.81	3.11	41.58	31.59	9.91	19.81
x_5	6.59	5.68	1.53	39.19	38.42	20.65	14.47
Optimal value	4.43	5.89	5.37	9.26	7.94	2.22	7.39

5.2. The pressure vessel design

The second example pertains to the design of a pressure vessel. The objective is to minimize the total cost, which includes welding, material, and pressure vessel moulding costs. The problem involves four design variables, namely, 1) x_1, which is the thickness of the shell, 2) x_2, which is the thickness of the head, 3) x_3, which is the inner radius of the head, and 4) x_4, which is the length of the cylindrical section of the container. The problem can be mathematically expressed as:

$$\text{Minimize } f(x) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3.$$

Subject to:

$$\begin{cases} g_1(x) = -x_1 + 0.0193x_3 \le 0, \\ g_2(x) = -x_2 + 0.00954x_3 \le 0, \\ g_3(x) = -\pi x_3^2 x_4 - \dfrac{4}{3}\pi x_3^3 + 1296000 \le 0, \\ g_4(x) = x_4 - 240 \le 0, \\ 1 \le x_1 \le 99, \quad 1 \le x_2 \le 99, \quad 10 \le x_3 \le 200, \quad 10 \le x_4 \le 200. \end{cases}$$

The table provided in Table 6 displays the best results achieved by various algorithms for the pressure vessel design problem. QL-ADIFA outperforms all other algorithms with the best result. Specifically, when the design variables are $x_1 = 1$, $x_2 = 1$, $x_3 = 51.2031$, and $x_4 = 89.0799$, the optimal value obtained by QL-ADIFA is 8798.52.

Table 6. Comparison of optimization design of pressure vessel by different algorithms.

	QL-ADIFA	NQL-ADIFA	AD-IFA	FA	LF-FA	QQLMPA	INFO
x_1	1	1	1	1	1	1	1
x_2	1	1	1	1	1	1	1
x_3	51.20	51.74	51.11	50.90	49.8389	47.39	49.15
x_4	89.0799	85.22	89.7867	92.4038	99.8011	121.61	105.22
Optimal value	8798.52	8799.90	8799.79	8835.10	8817.23	8800.39	9084.11

5.3. The three-bar truss design problem

The three-bar truss design problem [28] is a classic engineering problem that involves determining the optimal design of a simple truss structure made of three bars. The truss must resist a set of external forces while minimizing the total weight of the structure. The problem involves two design variables, namely, 1) x_1, which is the area of bar 1 and the area of bar 3, and 2) x_2, which is the area of bar 2. The problem can be mathematically expressed as:

$$\text{Minimize } f(x) = \left(2\sqrt{2}x_1 + x_2\right) \times L$$

Subject to:

$$\begin{cases} g_1(x) = \dfrac{\sqrt{2}x_1 + x_2}{\sqrt{2}x_1^2 + 2x_1x_2}P - \sigma \quad \le 0 \\ g_2(x) = \dfrac{x_2}{\sqrt{2}x_1^2 + 2x_1x_2}P - \sigma \quad \le 0 \\ g_3(x) = \dfrac{1}{\sqrt{2}x_2 + x_1}P - \sigma \quad \le 0 \\ 0.00 \le x_1 \le 1.00, \quad 0.00 \le x_2 \le 1.00. \end{cases}$$

Here, the decision variables are the area of bars 1 and 3 (x_1), and the area of bar 2 (x_2). $L = 100$ cm, $P = 2$ N/cm^2, $\sigma = 2$ N/cm^2.

The table provided in Table 7 displays the best results achieved by various algorithms for the three-bar truss design problem. QL-ADIFA outperforms all other algorithms with the best result. Specifically, when the design variables are $x_1 = 0.79$, $x_2 = 0.41$, the optimal value obtained by QL-ADIFA is 263.85.

Table 7. Comparison of optimization design of three-bar truss design by different algorithms.

	QL-ADIFA	NQL-ADIFA	AD-IFA	FA	LF-FA	QQLMPA	INFO
x_1	0.79	0.79	0.79	0.79	0.79	0.79	0.79
x_2	0.41	0.41	0.41	0.41	0.41	0.41	0.40
Optimal value	263.85	265.75	263.85	264.08	263.97	263.86	263.85

5.4. The CEC2020 functions

Table 8. Results of QL-ADIFA and other algorithms on 9 constrained optimization problems.

Function		QL-ADIFA	NQL-ADIFA	AD-IFA	FA	LF-FA	QQLMPA	INFO	f_{min}
	Avg	-3.53E-02	-2.07E-02	-7.10E-02	1.83E+02	2.53E+00	-1.38E-01	-3.05E-01	-3.88E-01
$F1$	Best	-3.28E-01	-1.73E-01	-3.70E-01	-3.15E-01	-3.67E-01	-3.88E-01	-3.69E-01	
	Std	1.03E-01	5.44E-02	1.49E-01	2.37E+02	4.03E+00	1.57E-01	1.00E-01	
	Avg	7.82E+05	2.23E+06	8.65E+05	1.46E+10	1.49E+09	-1.38E-01	-3.05E-01	-4.00E+02
$F2$	Best	-1.87E-02	-2.78E+02	-1.57E-02	2.47E+08	6.17E+07	-3.88E-01	-3.69E-01	
	Std	2.43E+06	3.67E+06	1.85E+06	1.73E+10	2.45E+09	1.57E-01	1.00E-01	
	Avg	9.98E-01	9.98E-01	9.98E-01	1.21E+00	1.13E+00	9.98E-01	9.98E-01	1.86E+00
$F3$	Best	9.98E-01	9.98E-01	9.98E-01	1.06E+00	1.06E+00	9.98E-01	9.98E-01	
	Std	2.34E-16	2.34E-16	2.34E-16	1.29E-01	5.57E-02	2.34E-16	2.34E-16	
	Avg	2.00E+00	2.00E+00	2.00E+00	2.00E+00	2.00E+00	2.00E+00	2.00E+00	2.00E+00
$F4$	Best	2.00E+00	2.00E+00	2.00E+00	2.00E+00	2.00E+00	2.00E+00	2.00E+00	
	Std	4.94E-06	8.45E-07	4.08E-06	9.05E-07	1.81E-06	6.89E-06	3.75E-07	
	Avg	2.69E+04	2.69E+04	2.69E+04	1.15E+06	6.94E+04	2.69E+04	2.69E+04	2.69E+04
$F5$	Best	2.69E+04	2.69E+04	2.69E+04	3.01E+04	2.78E+04	2.69E+04	2.69E+04	
	Std	1.63E+01	8.75E+01	4.61E-07	1.70E+06	1.17E+05	1.22E-05	7.67E-12	
	Avg	3.03E+03	3.03E+03	3.04E+03	4.26E+03	3.74E+03	2.69E+04	2.69E+04	2.99E+03
$F6$	Best	3.00E+03	3.01E+03	3.01E+03	3.35E+03	3.08E+03	2.69E+04	2.69E+04	
	Std	1.03E+01	1.32E+01	4.60E+01	5.78E+02	7.55E+02	1.22E-05	7.67E-12	
	Avg	7.39E+03	7.20E+03	8.74E+03	3.93E+04	3.36E+04	6.14E+03	6.73E+03	5.89E+03
$F7$	Best	6.22E+03	6.17E+03	6.21E+03	1.96E+04	1.71E+04	6.06E+03	6.08E+03	
	Std	1.00E+03	1.17E+03	5.08E+03	1.28E+04	1.10E+04	7.43E+01	4.68E+02	
	Avg	1.88E+00	1.87E+00	1.93E+00	3.29E+00	2.32E+00	1.70E+00	1.75E+00	1.67E+00
$F8$	Best	1.73E+00	1.70E+00	1.74E+00	1.80E+00	1.77E+00	1.67E+00	1.67E+00	
	Std	1.60E-01	1.62E-01	1.71E-01	9.25E-01	4.04E-01	1.65E-02	1.61E-01	
	Avg	-7.26E+02	-7.24E+02	-7.25E+02	-6.74E+02	-7.16E+02	-7.34E+02	-7.39E+02	8.91E-02
$F9$	Best	-7.28E+02	-7.28E+02	-7.32E+02	-7.10E+02	-7.23E+02	-7.41E+02	-7.42E+02	
	Std	2.01E+00	2.53E+00	3.11E+00	6.55E+01	4.57E+00	5.95E+00	3.68E+00	

There are 57 constrained optimization problems in CEC2020 with varying dimensions, ranging from 2 variables to 158 variables, and a range of equality and inequality constraints, ranging from 2 constraints to 148 constraints. Comprehensive information regarding these problems can be found in [29]. To evaluate the performance of the QL-ADIFA algorithm proposed in this study, 9 constrained optimization problems were selected for a series of experiments.

The detailed results of some CEC2020 functions obtained from the proposed QL-ADIFA and other contrast algorithms are presented in Table 8, including the average (Avg), best (Best), the standard deviation (Std) of each problem. As shown in Table 8, QL-ADIFA significantly improves average values, stability, and convergence speed compared with AD-IFA.

6. Conclusions

The present study proposes a new optimization algorithm, called QL-ADIFA, which is a hybrid of Q-learning based on the Logarithmic Spiral-Levy Firefly Algorithm (QL-LSLFA) and the adaptive Logarithmic Spiral-Levy Firefly Algorithm (AD-IFA). The QL-LSLFA algorithm improves the efficiency of the original FA by introducing Q-learning, which enables the fireflies to better adapt to state transitions and use the information obtained from previous iterations. In addition, with the union with AD-IFA, the QL-LSLFA can be avoided falling into local optimums as much as possible. In order to evaluate the performance of QL-ADIFA, the algorithm was tested on 15 benchmark functions and twelve engineering problems. The experimental results demonstrated that QL-ADIFA outperforms other algorithms in terms of solution quality, stability, and convergence speed for most of the tested functions and problems. The proposed hybrid algorithm thus represents an effective and promising approach to solving global optimization problems.

In future work, combining Q-learning with other variants of the Firefly algorithm can be considered to improve the convergence speed of the algorithm. In addition, we can also try adding pre-experiments for parameter settings to optimize the performance of the algorithm.

Use of AI tools declaration

The authors declare they have not used Artificial Intelligence (AI) tools in the creation of this article.

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. A. M. Altabeeb, A. M. Mohsen, L. Abualigah, A. Ghallab, Solving capacitated vehicle routing problem using cooperative firefly algorithm, *Appl. Soft Comput.*, **108** (2021), 107403. https://doi.org/10.1016/j.asoc.2021.107403
2. M. H. Nadimi-Shahraki, H. Zamani, S. Mirjalili, Eqnhanced whale optimization algorithm for medical feature selection: A COVID-19 case study, *Comput. Biol. Med.*, **148** (2022), 105858. https://doi.org/10.1016/j.compbiomed.2022.105858

3. H. Zhang, Y. Shi, X. Yang, R. Zhou, A firefly algorithm modified support vector machine for the credit risk assessment of supply chain finance, *Res. Int. Bus. Finance*, **58** (2021), 101482. https://doi.org/10.1016/j.ribaf.2021.101482

4. J. S. Pan, L. G. Zhang, R. B. Wang, V. Snášel, S. C. Chu, Gannet optimization algorithm: A new metaheuristic algorithm for solving engineering optimization problems, *Math. Comput. Simul.*, **202** (2022), 343–373. https://doi.org/10.1016/j.matcom.2022.06.007

5. G. Dhiman, K. K. Singh, M. Soni, A. Nagar, M. Dehghani, A. Slowik, et al., Mosoa: A new multi-objective seagull optimization algorithm, *Expert Syst. Appl.*, **167** (2021), 114150. https://doi.org/10.1016/j.eswa.2020.114150

6. Z. Cui, X. Hou, H. Zhou, W. Lian, J. Wu, Modified slime mould algorithm via levy flight, in *2020 13th International Congress on Image and Signal Processing, Biomedical Engineering and Informatics (CISP-BMEI)*, IEEE, Public Library of Science San Francisco, USA, (2020), 1109–1113. https://doi.org/10.1109/cisp-bmei51763.2020.9263669

7. Y. Yang, Y. Gao, S. Tan, S. Zhao, J. Wu, S. Gao, et al., An opposition learning and spiral modelling based arithmetic optimization algorithm for global continuous optimization problems, *Eng. Appl. Artif. Intell.*, **113** (2022), 104981. https://doi.org/10.1016/j.engappai.2022.104981

8. X. S. Yang, A. Slowik, Firefly algorithm, in *Swarm Intelligence Algorithms*, CRC Press, (2020), 163–174. https://doi.org/10.1201/9780429289071-12

9. N. Bacanin, T. Bezdan, K. Venkatachalam, F. Al-Turjman, Optimized convolutional neural network by firefly algorithm for magnetic resonance image classification of glioma brain tumor grade, *J. Real Time Image Process.*, **18** (2021), 1085–1098. https://doi.org/10.1007/s11554-021-01106-x

10. J. Zhang, Y. Huang, G. Ma, Y. Yuan, B. Nener, Automating the mixture design of lightweight foamed concrete using multi-objective firefly algorithm and support vector regression, *Cem. Concr. Compos.*, **121** (2021), 104103. https://doi.org/10.1016/j.cemconcomp.2021.104103

11. M. Rigakis, D. Trachanatzi, M. Marinaki, Y. Marinakis, Tourist group itinerary design: When the firefly algorithm meets the n-person battle of sexes, *Knowledge-Based Syst.*, **228** (2021), 107257. https://doi.org/10.1016/j.knosys.2021.107257

12. A. Sharma, R. Chaturvedi, A. Bhargava, A novel opposition based improved firefly algorithm for multilevel image segmentation, *Multimedia Tools. Appl.*, **81** (2022), 15521–15544. https://doi.org/10.1007/s11042-022-12303-6

13. V. Kumar, D. Kumar, A systematic review on firefly algorithm: past, present, and future, *Arch. Comput. Methods Eng.*, **28** (2021), 3269–3291. https://doi.org/10.36227/techrxiv.12122748

14. X. S. Yang, Firefly algorithm, Lévy flights and global optimization, in *Research and Development in Intelligent Systems XXVI*, (2010), 209–218. https://doi.org/10.1007/978-1-84882-983-1_15

15. Q. X. Lieu, D. T. Do, J. Lee, An adaptive hybrid evolutionary firefly algorithm for shape and size optimization of truss structures with frequency constraints, *Comput. Struct.*, **195** (2018), 99–112. https://doi.org/10.36227/techrxiv.12122748

16. J. Wu, Y. G. Wang, K. Burrage, Y. C. Tian, B. Lawson, Z. Ding, An improved firefly algorithm for global continuous optimization problems, *Expert Syst. Appl.*, **149** (2020), 113340. https://doi.org/10.1016/j.eswa.2020.113340

17. B. A. Hassan, Cscf: a chaotic sine cosine firefly algorithm for practical application problems, *Neural. Comput. Appl.*, **33** (2021), 7011–7030. https://doi.org/10.1007/s00521-020-05474-6
18. M. J. Goldanloo, F. S. Gharehchopogh, A hybrid obl-based firefly algorithm with symbiotic organisms search algorithm for solving continuous optimization problems, *J. Supercomput.*, **78** (2022), 3998–4031. https://doi.org/10.1007/s11227-021-04015-9
19. S. Zhao, Y. Wu, S. Tan, J. Wu, Z. Cui, Y. G. Wang, Qqlmpa: A quasi-opposition learning and q-learning based marine predators algorithm, *Expert Syst. Appl.*, **213** (2023), 119246. https://doi.org/10.1016/j.eswa.2022.119246
20. T. N. Huynh, D. T. Do, J. Lee, Q-learning-based parameter control in differential evolution for structural optimization, *Appl. Soft Comput.*, **107** (2021), 107464. https://doi.org/10.1016/j.asoc.2021.107464
21. R. Qi, J. Q. Li, J. Wang, H. Jin, Y. Y. Han, Qmoea: A q-learning-based multiobjective evolutionary algorithm for solving time-dependent green vehicle routing problems with time windows, *Inf. Sci.*, **608** (2022), 178–201. https://doi.org/10.1016/j.ins.2022.06.056
22. B. Jang, M. Kim, G. Harerimana, J. W. Kim, Q-learning algorithms: A comprehensive classification and applications, *IEEE Access*, **7** (2019), 133653–133667. https://doi.org/10.1109/access.2019.2941229
23. R. H. Crites, A. G. Barto, Elevator group control using multiple reinforcement learning agents, *Mach. Learn.*, **33** (1998), 235–262. https://doi.org/10.36227/techrxiv.21197626
24. X. S. Yang, Firefly algorithm, stochastic test functions and design optimisation, *Int. J. Bio-Inspired Comput.*, **2** (2010), 78–84. https://doi.org/10.1504/ijbic.2010.032124
25. A. M. Reynolds, M. A. Frye, Free-flight odor tracking in drosophila is consistent with an optimal intermittent scale-free search, *PloS One*, **2** (2007), e354. https://doi.org/10.1371/journal.pone.0000354
26. V. A. Tucker, A. E. Tucker, K. Akers, J. H. Enderson, Curved flight paths and sideways vision in peregrine falcons (falco peregrinus), *J. Exp. Biol.*, **203** (2000), 3755–3763. https://doi.org/10.1242/jeb.203.24.3755
27. I. Ahmadianfar, A. A. Heidari, S. Noshadian, H. Chen, A. H. Gandomi, Info: An efficient optimization algorithm based on weighted mean of vectors, *Expert Syst. Appl.*, **195** (2022), 116516. https://doi.org/10.1016/j.eswa.2022.116516
28. X. Tao, W. Guo, X. Li, Q. He, R. Liu, J. Zou, Fitness peak clustering based dynamic multi-swarm particle swarm optimization with enhanced learning strategy, *Expert Syst. Appl.*, **191** (2022), 116301. https://doi.org/10.1016/j.eswa.2021.116301
29. A. Kumar, G. Wu, M. Z. Ali, R. Mallipeddi, P. N. Suganthan, S. Das, A test-suite of non-convex constrained optimization problems from the real-world and some baseline results, *Swarm Evol. Comput.*, **56** (2020), 100693. https://doi.org/10.1016/j.swevo.2020.100693