Computer Modeling in Engineering & Sciences

Tech Science Press

DOI: 10.32604/cmes.2023.031118

ARTICLE

An Improved JSO and Its Application in Spreader Optimization of Large Span Corridor Bridge

Shude Fu[1,2], Xinye Wu[1,2,*], Wenjie Wang[3], Yixin Hu[1,3,*], Zhengke Li[1] and Feng Jiang[3]

[1]Fujian Key Laboratory of Digital Simulations for Coastal Civil Engineering, School of Architecture and Civil Engineering, Xiamen University, Xiamen, 361005, China

[2]Xiamen Transportation Infrastructure Intelligent Management and Maintenance Engineering Technology Research Center, Xiamen, 361005, China

[3]Fuzhou Company of China Communications Construction Co., Ltd., Fuzhou, 350005, China

*Corresponding Authors: Xinye Wu. Email: wuxinye@xmu.edu.cn; Yixin Hu. Email: newhu915@163.com

Received: 15 May 2023 Accepted: 02 August 2023 Published: 15 December 2023

ABSTRACT

In this paper, given the shortcomings of jellyfish search algorithm with low search ability in the early stage and easy to fall into local optimal solution, this paper introduces adaptive weight function and elite strategy, improving the global search scope in the early stage and the ability to refine the local development in the later stage. In the numerical study, the benchmark problem of dimensional optimization with a 10-bar truss structure and simultaneous dimensional shape optimization with a 15-bar truss structure is adopted, and the corresponding penalty method is used for constraint treatment. The test results show that the improved jellyfish search algorithm can provide better truss sections as well as weights. Because when the steel main truss of the large-span covered bridge is lifted, the site is limited and the large lifting equipment cannot enter the site, and the original structure does not meet the problem of stress concentration and large deformation of the bolt group, so the spreader is used to lift, and the improved jellyfish search algorithm is introduced into the design optimization of the spreader. The results show that the improved jellyfish algorithm can efficiently and accurately find out the optimal shape and weight of the spreader, and through Midas Civil simulation, the spreader used can meet the requirements of weight and safety.

KEYWORDS

Truss optimization; improved JSO; size optimization; shape optimization

1 Introduction

In recent years, the emergence of various lifting equipment, construction site various installation and lifting technology is becoming more and more mature, but in the face of the problem of lifting the superstructure of the large-span covered bridge, due to the limitation of large-scale lifting equipment, coupled with the influence of the river bank, it is difficult to build a lifting structure, and can only consider the use of jacks for lifting on the original structure. At this point it is about whether the

weight of the structure to be lifted can meet the lifting limits of the jack and whether the force can meet the safety requirements.

Nowadays, the continuous in-depth research of intelligent algorithms by scholars from all over the world has introduced various algorithms into the weight, shape and topology optimization of structures, and solved many engineering problems.

Shi et al. [1] proposed a new hybrid algorithm, namely the improved plant growth simulation algorithm and the genetic mixing algorithm, and the optimization efficiency and effect of PGSA-GA are better than other algorithms and methods through the case studies of typical trusses and single-layer lattice shells; Kim et al. [2] proposed an efficient two-stage optimization program based on force method to correctly identify the location and extent of multiple damage in planar and space truss structures; Viet-Hung et al. [3] developed a robust method for dimensional optimization (RBDO) of truss structures by integrating nonlinear inelastic analysis, structural reliability analysis methods, and optimization methods based on differential evolution (DE) algorithms. Azad et al. [4] solved the problem of simultaneous optimization of dimensions and geometry of dynamically excited steel truss structures. Using the well-known Big Bang-Big Crunch algorithm, the minimum weight design of steel truss is carried out under periodic and a periodic excitation; Grzywinski et al. [5] proposed a novel and effective Jaya optimization algorithm for optimization the best quality of the supporting dome structure with natural frequency constraints; Habibi et al. [6] considered the geometric nonlinearity problem using the total Lagrangian formula, and obtain a nonlinear solution by introducing and minimizing the objective function constrained by the displacement type; Artar et al. [7] studied the optimal design of steel space truss towers under seismic load by using the Jaya optimization algorithm; Kaveh et al. [8] proposed a new Gaussian diagram-based Chaotic Firefly Algorithm (CGFA) for structural optimization problems. Wong et al. [9] investigated a new meta-heuristic algorithm called symbiotic organism search (SOS) for component size optimization of relatively large steel trusses; Ha et al. [10] developed an effective method to optimize nonlinear steel frames under several load combinations, considering the panel area for the first time in the optimization design. The double-layer board is designed to prevent shear deformation of the panel area; Gholizadeh et al. [11] proposed a new and efficient meta-heuristic, the Newtonian meta-heuristic (NMA). Seismic design optimization for steel bending moment frames based on discrete properties; Es-Haghi et al. [12] proposed an asymmetric genetic algorithm (AGA) to solve the optimization problem of steel frames, and optimized a 15-layer three-layer steel flat frame through AGA, AGA can reduce the analysis time, the number of analyses and the total weight of the structure; Truong et al. [13] proposed a gradient tree boosting (GTB) algorithm for the safety assessment of steel trusses, which was first generated using advanced analysis methods to consider the geometry of the structure and the non-linearity of the material. Then, four GTB models are proposed to predict the ultimate bearing capacity and displacement of the structure for safety evaluation of strength and suitability. Zhou et al. [14] proposed a hybrid strategy based on butterfly optimization algorithm (BOA) and differential evolution algorithm (DE), and experimental tests of 8-layer shear steel frame structure in the laboratory to evaluate its performance. Numerical and experimental results show that the proposed HBODEA is more powerful in detecting the stiffness reduction of limited sensors and contamination measurement results. Bigham et al. [15] proposed an improved electrical search algorithm to solve the topology optimization problem of nonlinear single-layer domes; Gholizadeh et al. [16] used the proposed algorithm to design a 6-layer and 12-layer steel bending moment frame and evaluate it. The optimal design of seismic performance and collapse capacity used the failure index and incremental dynamic analysis, and evaluated their seismic failure cost and adjusted collapse allowance ratio; Artar et al. [17] proposed teaching-based optimization (TLBO) and biogeography-based optimization (BBO) algorithms to study the optimal

discrete size design of steel truss steel bridges to minimize structural weight; Azad et al. [18] proposed a computationally efficient multi-stage guided random search algorithm for the optimization and standardization of real-size free steel double-layer grids. Carbas et al. [19] investigated the design of the optimal discrete dimensions of a steel plane truss containing the effects of seismic loads by teaching-based optimization (TLBO) and biogeography-based optimization (BBO) meta-heuristics; In order to obtain better vibration response data, based on the improved particle swarm optimization algorithm, Zhao et al. [20] proposed an optimal arrangement scheme of measurement points of long-span steel beams to improve the accuracy of the modal test results of long-span steel beams. Ojha et al. [21] adaptive search space decomposition method and a new formula based on gradient-free optimization for the anterior and post-buckling analysis of spatial truss structures. Eser et al. [22] proposed a volumetric controlled search (CCS) algorithm that is capable of handling dimensional optimization of particularly large steel frames under multi-strength and multi-displacement constraints. Sang-To et al. [23] presented a new shrimp tiger associative search algorithm (SGA) for efficient optimization of truss-based structural health monitoring (SHM). SGA can get rid of local optimization better and converge faster than the population-based algorithm. Azizi et al. [24] proposed a novel metaheuristic algorithm—Energy Valley Optimizer (EVO), and it can provide competitive and outstanding results in dealing with complex benchmarks and real-world problems. Kadkhoda et al. [25] proposed the coronavirus metamorphosis optimization algorithm (CMOA), and the CMOA is applied to three engineering problems including optimal design of a welded beam, a three-bar truss and a pressure vessel, showing its high potential in solving such practical problems and effectiveness in finding global optima.

This paper presents a modified variant of a recently introduced natural heuristic algorithm, the Jellyfish Search optimizer [26]. Two modifications are proposed in the basic JSO to form the modified variant, which is applied to the optimization of the main truss lifting spreader of the long span gallery bridge.

2 Jellyfish Search Algorithm

Jellyfish Search optimizer (JSO). The algorithm proposed by Chou et al. [26] in 2020, and the algorithm mainly simulates the characteristics of jellyfish drifting with ocean currents and the internal motion of jellyfish population, and introduces a time control mechanism. Compared with other bionic algorithms, this algorithm exhibits superior computational results.

Any meta-heuristic algorithm has two main stages: early exploration to enrich population diversity; later development, strengthen the approximation optimal solution. The JSO algorithm is also developed based on these two stages.

Before designing the algorithm, Chou et al. assumed the motion law of jellyfish as follows:

1. Jellyfish move in two ways, either with ocean currents or within populations, and the time control mechanism controls the transition between the two motions.
2. Jellyfish are more easily attracted to places with more food in the ocean.

2.1 Currents

The current contains a lot of nutrients, and jellyfish are attracted to it. The direction of the current is the convergence trend of the algorithm, which is defined as the average vector position of each jellyfish to the current best jellyfish position:

$$\overrightarrow{trend} = \frac{1}{n_{pop}} \sum \overrightarrow{trend_i} = \frac{1}{n_{pop}} \sum (X^* - e_c X_i) = X^* - e_c \frac{\sum X_i}{n_{pop}} = X^* - e_c \mu \tag{1}$$

$$Set\ df = e_c \mu \tag{2}$$

n_{pop} is the size of the jellyfish population; X^* is the optimal position of the jellyfish; e_c is the attraction control factor; μ is average position for all jellyfish; df is the difference between the optimal jellyfish position and the average position of all jellyfish. The original paper assumes that jellyfish are normally spatially distributed in all dimensions, and that all jellyfish may be distributed within a distance of $\pm\sigma\beta$ (σ is the standard deviation of the distribution, and $\beta = 3$ is the distribution coefficient) around the mean position, so there is the following definition:

$$\mathrm{d}f = \beta \times \sigma \times rand^{\mathrm{r}}\,(0, 1) \tag{3}$$

$$\text{Set } \sigma = rand^{\alpha}\,(0, 1) \times \mu \tag{4}$$

Ocean currents are calculated as follows:

$$\overrightarrow{trend} = X^* - \beta \times rand\,(0, 1) \times \mu \tag{5}$$

So the position update formula for each jellyfish as follows:

$$X_i\,(t + 1) = X_i\,(t) + rand\,(0, 1) \times (X^* - \beta \times rand\,(0, 1) \times \mu) \tag{6}$$

2.2 Jellyfish Swarms

The movement of jellyfish is mainly divided into passive movement (type A) and active movement (type B). Initially, when the jellyfish colony was first formed, most of the jellyfish only followed the movement of the population, so they showed A-type movement. Over time, jellyfish gradually exert their own initiative, so they increasingly exhibit B-type movements. Type A locomotion is the movement of a jellyfish around its own position:

$$X_i\,(t + 1) = X_i\,(t) + \gamma \times rand\,(0, 1) \times (U_b - L_b) \tag{7}$$

where U_b, L_b are the upper and lower limits of the search space, respectively, and $\gamma = 0.1$ is the motion coefficient. After the jellyfish swarm has formed a certain size, everyone begins to be motivated and approach companions who find more food, which is also known as the B-type movement:

$$\overrightarrow{Step} = X_i\,(t + 1) - X_i\,(t) \tag{8}$$

$$\overrightarrow{Step} = rand\,(0, 1) \times \overrightarrow{Direction} \tag{9}$$

$$\overrightarrow{Direction} = \begin{cases} X_j\,(t) - X_i\,(t) & if\ f\,(X_i) \geq f\left(X_j\right) \\ X_i\,(t) - X_j\,(t) & if\ f\,(X_i) < f\left(X_j\right) \end{cases} \tag{10}$$

$$X_i\,(t + 1) = X_i\,(t) + \overrightarrow{Step} \tag{11}$$

2.3 Time Control Mechanism

When a certain section of the ocean current contains abundant food, jellyfish gather in groups to feed each other. When the temperature or wind changes the current, the jellyfish swarm moves towards another current, forming another jellyfish swarm. Therefore, Chou et al. introduced a time

control mechanism to simulate this passive to active process. To regulate the movement of jellyfish following ocean currents and moving within the jellyfish swarm, time control mechanisms include the time control function $c(t)$ and the constant $C_0 = 0.5$:

$$c(t) = \left| \left(1 - \frac{t}{Max_{iter}}\right) \times (2 \times rand(0,1) - 1) \right| \tag{12}$$

When $c(t)$ is greater than C_0, the jellyfish will follow the currents, that is B-type motion; when $c(t)$ is less than C_0, the jellyfish will only move within the population, A-type motion.

2.4 Population Initialization

The JSO algorithm introduces a logistic chaos mapping as the initialization:

$$X_i(t+1) = \eta X_i(1 - X_i) \tag{13}$$

$X_0 \notin 0.0, 0.25, 0.75, 0.5, 1.0, \eta = 4.$

2.5 Boundary Restrictions

The JSO algorithm takes into account that the earth is circular, so a boundary buffering strategy is adopted:

$$\begin{cases} X'_{i,d} = \left(X_{i,d} - U_{b,d}\right) + L_b(d) \ \ if \ X_{i,d} > U_{b,d} \\ X'_{i,d} = \left(X_{i,d} - L_{b,d}\right) + U_b(d) \ \ if \ X_{i,d} < L_{b,d} \end{cases} \tag{14}$$

Through the analysis process of JSO algorithm, it can be seen that the algorithm has good robustness, and the buffer strategy of the boundary avoids the situation that individuals concentrate on searching the domain boundary, thereby improving the population diversity.

3 Modified the Jellyfish Search Algorithm (MJSO)

3.1 Elite Strategy

For the JSO algorithm in the early stage, it relies on determining the position of the optimal jellyfish, and the best position of the jellyfish generated at this time is still random, and the position update of the jellyfish is determined by the direction of the ocean current, and the ocean current is determined by the average of the random position generated by the jellyfish, which is easy to cause the global search ability of the algorithm in the early stage to be relatively low, so the elite strategy is introduced. After the population is initialized, the set of best positions of jellyfish is determined, and for the previous global search stage, the average value of the best position set of jellyfish ($\mu Elite$) is taken instead of the average of all jellyfish positions, and the updated formula is as follows:

$$X_i(t+1) = X_i(t) + rand(0,1) \times (X^* - \beta \times rand(0,1) \times \mu Elite) \tag{15}$$

$$\mu Elite = mean(n_{elite}) \tag{16}$$

$$n_{elite} = n_{pop} \times (1 - r) \tag{17}$$

Eq. (16) indicates that $\mu Elite$ equals the elite population mean, Eq. (17) is the equation for calculating the elite population, r denotes the golden mean ratio.

Elite is the set of the best individuals of a certain number of jellyfish in the population. $\mu Elite$ differs from Eq. (6) in that its mean value is obtained from the best individuals in the Elite set, not

from all individuals in the aggregate. Calculating the average of a certain number of elite individuals helps to direct individuals to the main areas of the search space where high-quality solutions exist. Therefore, when the local search process is initiated, the solution begins to search locally in areas where more promising, high-quality solutions exist.

3.2 Adaptive Weighting

In addition to the above elite strategy, aiming at the shortcomings of the overall process of JSO algorithm that the search range is not large in the early stage and the development is easy to fall into local optimization in the later stage, the adaptive weight function is introduced to improve the global search ability and calculation time, and the improved jellyfish algorithm is formed.

Adaptive weight inertia weight is a very important parameter in the particle swarm, when the inertia weight is large, the algorithm search ability is strong, can search a larger area, when the inertia weight is small, the algorithm later search ability is strong, can be finely searched around the optimal solution.

Introducing the particle swarm algorithm inertia weights and modifying them accordingly to obtain the adaptive weight function w, since the direction of ocean currents is determined by the optimal jellyfish position, adaptive weights are introduced to expand the jellyfish search capability in the early stage, and the local search ability of Type-A jellyfish is increased when the inertial weight is small in the later stage, so as to find the optimal solution. Weight function w is as follows:

$$\begin{cases} \varepsilon = \dfrac{(10 * it - 5 * Maxit)}{Maxit} \\ w = \dfrac{1}{1 + \sqrt{\varepsilon}} \end{cases} \tag{18}$$

where it is the number of iterations and Maxit is the maximum number of iterations. The function image is shown in Fig. 1.

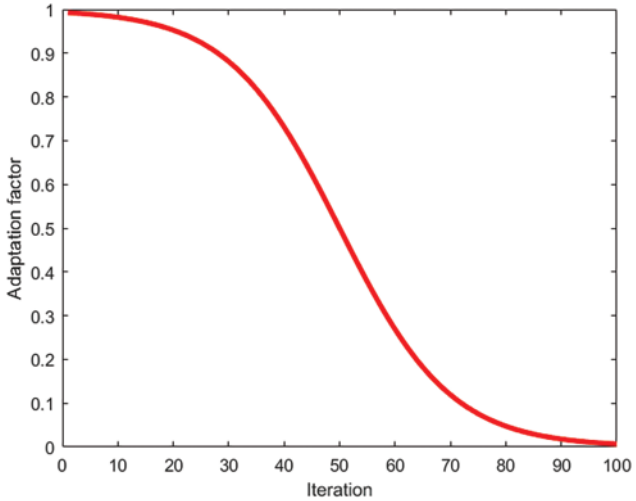

Figure 1: Adaptive function w graph

The improved jellyfish update formula is as follows:

The early stage follows the ocean current search stage:

$$X_i(t+1) = w * X_i(t) + rand(0,1) \times (X^* - \beta \times rand(0,1) \times \mu Elite) \tag{19}$$

A-type movement development stage:

$$X_i(t+1) = w * X_i(t) + \gamma \times rand(0,1) \times (U_b - L_b) \tag{20}$$

The MJSO pseudo-code is as follows:

MJSO processes

```
Begin
    Define objective function F(x)
    Set population size (Npop) and maximum iteration number (Maxit)
    Initialize jellyfish population Xi (i = 1,2...Npop)
    Calculate the fitness value (F(Xi)) of each Xi
    Define the size of Elite set (En)
    Find the best individual in the population (X*)
    Define the adaptive function (w)
    For it = 1 to maxit
        Sort the population by fitness value and add En best individuals to Elite
        Calculate (μElite) to Eq. (16)
        For i = 1 to Npop
            Calculate c(t) according to Eq. (12)
            If c(t) > Co then
                Define new location (Xnew) according to Eq. (19)
                Else
                If rand(0,1) > (1-c(t)) then
                  Define new location (Xnew) according to Eq. (20)
                  Else
                    Determine jellyfish direction according to Eq. (10)
                    Define new location (Xnew) according to Eq. (11)
                End If
            End If
            If F(Xnew) < F(Xi) then
              Xi = Xnew
                If F(Xnew) < F(X) then
                        X* = Xnew
                End If
            End If
        End For
    End For
Output best results
End
```

The MJSO algorithm pseudo-code can be seen that the proposed MJSO algorithm does not contain any additional loops. Only some modifications have been made to the search formula.

Therefore, the time complexity of MJSO (O = N × d × Maxit) is the same as the standard JSO algorithm.

3.3 Algorithm Performance Testing

3.3.1 Benchmark Testing

The MJSO algorithm and JSO algorithm were compared with the JSO algorithm for benchmark function testing, and 50 functions described in Table 1 of the literature [26], including separate, non-separable, single-modal and multi-modal functions, were tested with MATLAB R2021b, and the computer processor was: AMD Ryzen7 4800H with Radeon Graphics 2.90 GHz. The initial population size is 50, the number of iterations is 10,000, and 30 cycles are performed, taking the mean, standard deviation, and average time to each optimization. Table 2 shows the calculation results.

Table 1: 50 benchmark test functions

Function number	Function name	Dimension	Best	Rank
F1	Stepint	5	0	[−5.12, 5.12]
F2	Step	30	0	[−100, 100]
F3	Sphere	30	0	[−100, 100]
F4	SumSquares	30	0	[−10, 10]
F5	Quartic	30	0	[−1.28, 1.28]
F6	Beale	2	0	[−4.5, 4.5]
F7	Easom	2	−1	[−100, 100]
F8	Matyas	2	0	[−10, 10]
F9	Colville	4	0	[−10, 10]
F10	Trid6	6	−50	$[-D^2, D^2]$
F11	Trid10	10	−210	$[-D^2, D^2]$
F12	Zakharov	10	0	[−5, 10]
F13	Powell	24	0	[−4, 5]
F14	Schwefel 2.22	30	0	[−10, 10]
F15	Schwefel 1.2	30	0	[−100, 100]
F16	Rosenbrock	30	0	[−30, 30]
F17	Dixon-Price	30	0	[−10, 10]
F18	Foxholes	2	0.998	[−65.536, 65.536]
F19	Branin	2	0.398	[−5, 10] × [0,15]
F20	Bohachevsky1	2	0	[−100, 100]
F21	Booth	2	0	[−10, 10]
F22	Rastrigin	30	0	[−5.12, 5.12]
F23	Schwefel	30	−12,569.5	[−500, 500]
F24	Michalewicz2	2	−1.8013	$[0, \pi]$
F25	Michalewicz5	5	−4.6877	$[0, \pi]$
F26	Michalewicz10	10	−9.6602	$[0, \pi]$
F27	Schaffer	2	0	[−100, 100]
F28	Six Hump Camel Back	2	−1.03163	[−5, 5]
F29	Bohachevsky2	2	0	[−100, 100]

(Continued)

Table 1 (continued)

Function number	Function name	Dimension	Best	Rank
F30	Bohachevsky3	2	0	[−100, 100]
F31	Shubert	2	−186.73	[−10, 10]
F32	GoldStein-Price	2	3	[−2, 2]
F33	Kowalik	4	0.00031	[−5, 5]
F34	Shekel5	4	−10.15	[0, 10]
F35	Shekel7	4	−10.4	[0, 10]
F36	Shekel10	4	−10.53	[0, 10]
F37	Perm	4	0	[−D, D]
F38	Powersum	4	0	[0, 1]
F39	Hartman3	3	−3.86	[0, D]
F40	Hartman6	6	−3.32	[0, 1]
F41	Griewank	30	0	[−600, 600]
F42	Ackley	30	0	[−32, 32]
F43	Penalized	30	0	[−50, 50]
F44	Penalized2	30	0	[−50, 50]
F45	Langermann2	2	−1.08	[0, 10]
F46	Langermann5	5	−1.5	[0, 10]
F47	Langermann10	10	NA	[0, 10]
F48	Fletcher Powell2	2	0	$[-\pi, \pi]$
F49	Fletcher Powell5	5	0	$[-\pi, \pi]$
F50	Fletcher Powell10	10	0	$[-\pi, \pi]$

Table 2: Best, mean and std. values of benchmark test functions for JSO and MJSO

Function symbol	MJSO			JSO		
	Mean	Std.	Time	Mean	Std.	Time
F1	0	0	1.43	0	0	1.47
F2	0	0	1.31	0	0	1.36
F3	0	0	1.32	0	0	1.37
F4	0	0	1.3	0	0	1.33
F5	5.73E-05	2.06E-05	2.86	0.0000724	2.65E-05	2.92
F6	0	0	1.17	0	0	1.19
F7	−1	0	1.13	−1	0	1.16
F8	0	0	1.12	0	0	1.14
F9	0	0	1.14	0	0	1.18
F10	−50	5.35E-14	1.16	−50	2.59E-14	1.19
F11	−210	1.35E-11	1.22	−210	6.08E-12	1.25
F12	0	0	1.27	0	0	1.33
F13	2.32E-08	5.47E-08	1.93	3.08E-07	1.16E-06	1.99
F14	0	0	1.4	0	0	1.43

(Continued)

Table 2 (continued)

Function symbol	MJSO			JSO		
	Mean	Std.	Time	Mean	Std.	Time
F15	0	0	1.89	0	0	1.92
F16	5.1E-06	8.59E-06	1.43	0.847	3.58	1.43
F17	0	0	1.47	0.0000913	0.0005	1.46
F18	0.998	1.13E-16	3.85	0.998	1.13E-16	3.95
F19	0.398	0	1.55	0.398	0	1.46
F20	0	0	1.21	0	0	1.23
F21	0	0	1.29	0	0	1.32
F22	0	0	1.57	8.95	5.54	1.6
F23	−8420	575	2.6	−8050	490	2.74
F24	−1.8	9.03E-16	1.4	−1.8	9.03E-16	1.39
F25	−4.67	0.0336	1.64	−4.66	0.0488	1.63
F26	−9.45	0.191	1.96	−9.45	0.151	1.98
F27	0	0	1.27	0	0	1.28
F28	−1.03	6.78E-16	1.43	−1.03	6.78E-16	1.45
F29	0	0	1.26	0	0	1.27
F30	0	0	1.28	0	0	1.29
F31	−187	1.97E-14	1.61	−187	1.9E-14	1.64
F32	3	1.31E-15	1.29	3	2.1E-15	1.29
F33	0.000307	1.79E-19	1.35	0.000307	1.69E-19	1.35
F34	−10.2	7.23E-15	3.48	−10.2	7.17E-15	3.57
F35	−10.4	1.81E-15	3.46	−10.4	1.81E-15	3.55
F36	−10.5	1.68E-15	3.5	−10.5	1.65E-15	3.54
F37	0.0025	0.00214	2.46	0.00279	0.00217	2.5
F38	5.08E-05	9.34E-05	2.07	0.0000428	7.44E-05	2.03
F39	−3.86	2.71E-15	5.31	−3.86	2.71E-15	5.31
F40	−3.32	6.39E-16	6.72	−3.32	6.39E-16	6.81
F41	0	0	1.68	0	0	1.68
F42	0	0	1.72	0	0	1.71
F43	0.00346	0.0189	8.68	2.5E-30	1.06E-29	9.27
F44	0	0	5.42	0	0	5.59
F45	−1.08	4.52E-16	1.53	−1.08	4.52E-16	1.57
F46	−1.5	6.78E-16	1.65	−1.5	6.78E-16	1.68
F47	−0.736	0.22	2.03	−0.762	0.196	2.07
F48	0	0	1.58	0	0	1.53
F49	0	0	2.44	0	0	2.45
F50	0	0	2.55	0	0	2.55
Hit rate		80%			72%	
Total time (s)		107.39			109.4	

According to the optimal solution hit rate, the MJSO algorithm has a hit rate of 80% for the optimal solution calculated for 50 benchmark functions, and the JSO algorithm has a hit rate of 72%, which shows that the optimal solution optimization ability of the MJSO algorithm is better than that of the JSO algorithm. The time spent by the two algorithms is approximately the same, which also indicates that there is no increase in time complexity of the improved MJSO algorithm. The errors of both algorithms are approximately the same, and the original literature has verified that the robustness test is more stable than other metaheuristic algorithms.

It can be seen from Fig. 2 that the MJSO algorithm is optimized for unimodal separable (F3), unimodal inseparable function (F8), multimodal separable function (F23) and multimodal inseparable function (F42). The number of iterations required to converge the optimal solution is less than that of the JSO algorithm, and the overall image shows that the local search for the optimal solution of the MJSO algorithm is stronger, which can avoid falling into the local optimum.

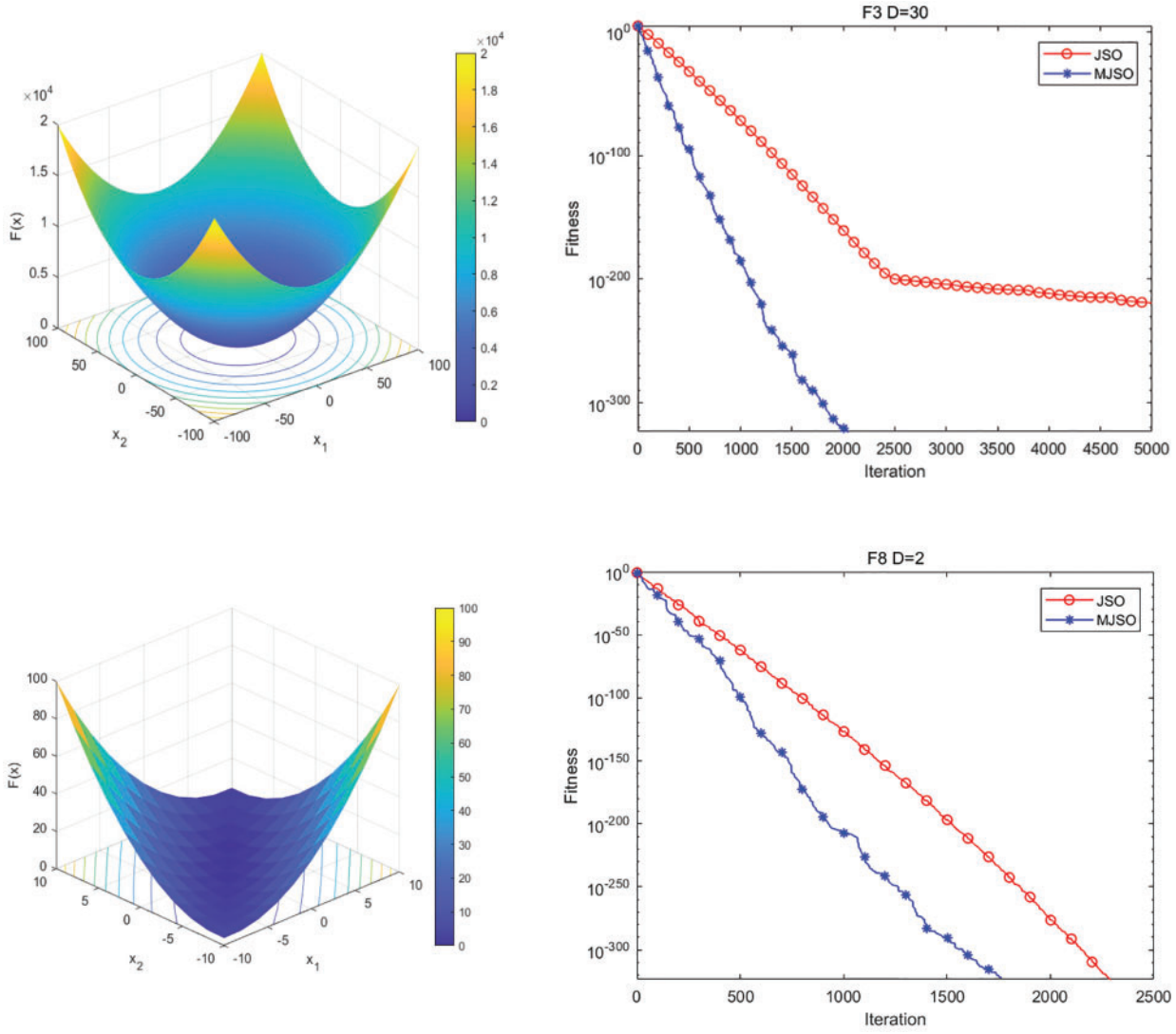

Figure 2: (Continued)

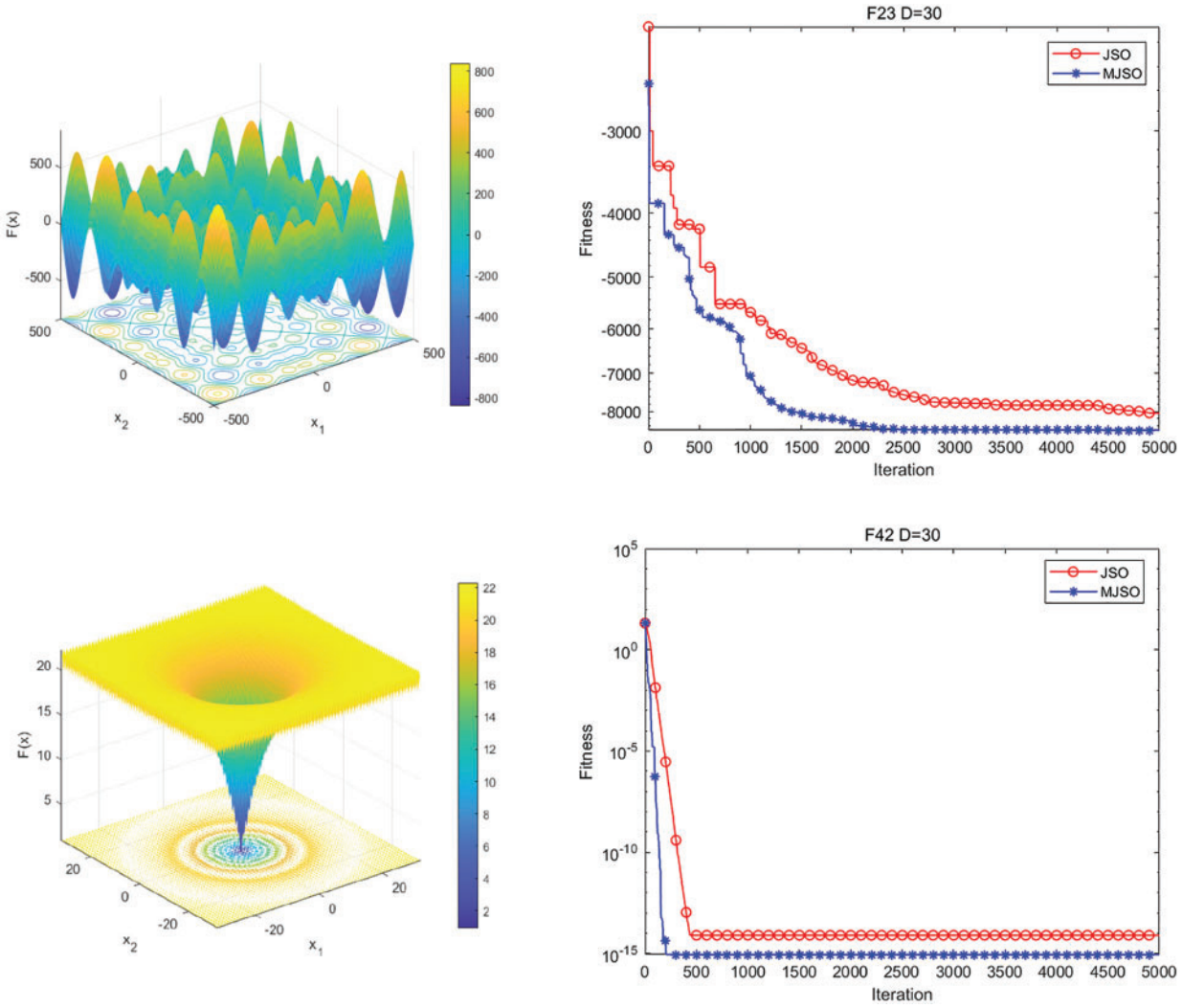

Figure 2: MJSO and JSO algorithm convergence comparison

3.3.2 10-Bar Truss Weight Optimization Test

Fig. 3 shows a 10-pole planar truss structure. The design variable is the cross-sectional area of the 10 truss elements. Table 3 lists the upper and lower bounds of material properties, load conditions, and design variables. The constraint is the allowable stress and allowable displacement of the planar truss. The maximum allowable displacement of each node in the $\pm$x and $\pm$y directions is equal to 2 in, while the maximum allowable stress for tension and compression is 25 ksi, with the goal of minimizing the weight of the structure under the specified constraints.

The average evolution of the objective function using 1000 iterations of the 10-bar truss with an initial population of 50 for MJSO, JSO, PSO, GWO and ACOR, respectively, as the number of iteration steps increases, is shown in Fig. 4, from which it can be seen that MJSO continuously provides better values than others, enabling it to provide a better adaptation of the results at each stage of execution.

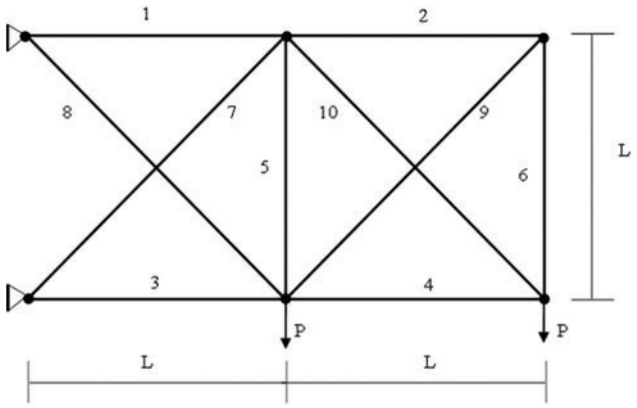

Figure 3: Diagram of a 10-bar truss

Table 3: 10-bar truss design parameters

Material properties	Value
Elastic modulus	107 psi
Material density	0.1 lb/in^3
Length L	360 in
Load P	100 kips
Design variable lower bounds	0.1 in^2
Design variable upper bound	35 in^2
The number of design variables	10

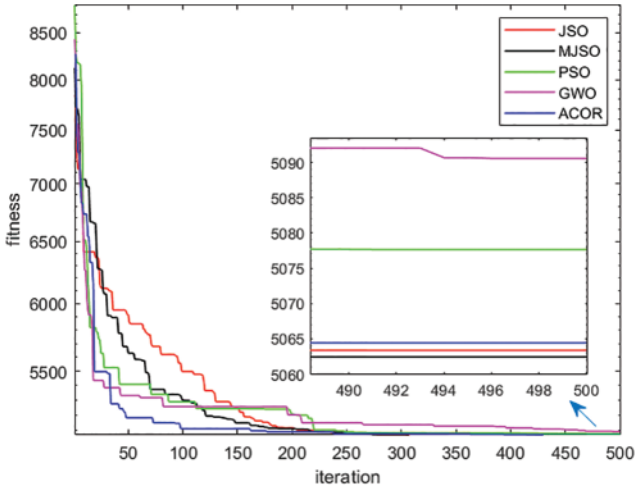

Figure 4: MJSO, JSO, PSO, GWO and ACOR the convergence process of the 10-bar truss

MJSO, JSO, PSO, GWO and ACOR algorithms were optimized for each unit section of the 10-bar truss, respectively, for 30 cycles, and the results of section optimization as well as the optimal weight optimum, average value and standard deviation after optimization are shown in Table 4.

Table 4: Comparison of 10-bar truss optimization results

	Unit number	MJSO	JSO	PSO	GWO	ACOR
Cross-sectional area (in^2)	1	30.2462	31.0510	30.1983	30.5486	30.5421
	2	0.1000	0.1011	0.1000	0.2715	0.1000
	3	23.4747	23.2819	24.4013	23.8667	23.4729
	4	15.060	15.0160	14.5457	14.6854	15.2586
	5	0.1000	0.1000	0.1000	0.1060	0.1000
	6	0.5470	0.5390	0.5570	0.1093	0.5696
	7	21.2643	20.9902	21.1165	21.2017	21.0661
	8	7.4561	7.4710	8.4860	8.5632	7.4754
	9	0.1004	0.1001	0.1000	0.2044	0.1000
	10	21.4588	21.2988	20.9266	20.8135	21.3055
Best weight (lb)		5060.671	5061.6664	5077.6475	5090.6388	5064.4658
Mean weight (lb)		5063.8834	5065.7621	5081.6322	5098.5763	5068.8955
Std.		6.426	7.231	6.811	7.312	6.945

It can be seen from the above table that the optimization results of 10-bar truss lightweight are superior and more robust than those of others.

3.3.3 15-Bar Truss Weight and Shape Optimization

Fig. 5 shows the structure of a 15-bar truss. The design variables for this problem are 15 unit cross-sectional areas (A1 to A15) and 8 nodal shape variables (X5 = X6, X3 = X4, Y1, Y2, Y3, Y4, Y5, Y6). Table 5 shows the material properties, loading conditions and discrete boundaries for cross-sectional area and continuous boundaries for shape variables. The constraint is the allowable stress of the plane truss with a maximum stress constraint of ±172.3689 MPa. No displacement limit is assumed. The objective is to optimize the shape and minimize the weight of the structure under the specified constraints.

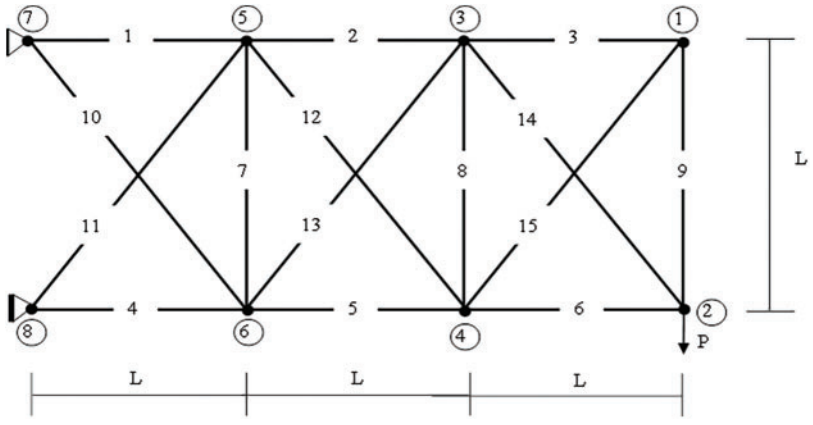

Figure 5: 15-bar truss

Table 5: 15-bar truss design parameters

Material properties	Value
Elastic modulus	68.95 GPa
Material density	2767.99 kg/m^3
Length L	3.048 m
Load P	44.45 kN
Cross-sectional value interval	D = (0.072 0.091 0.112 0.142 0.174 0.185 0.224 0.284 0.348 0.615 0.697 0.757 0.860 0.960 1.138 1.382 1.740 1.806 2.020 2.300 2.460 3.100 3.840 4.240 4.640 5.500 6.000 7.000 8.600 9.219 11.077 12.374) ($\times 10^{-3}$) (m^2)
Shape variable boundary constraints	$2.54 \leq X_5 \leq 3.556$ $5.588 \leq X_3 \leq 6.604$ $2.54 \leq Y_5 \leq 3.556$ $2.54 \leq Y_3 \leq 3.556$ $1.27 \leq Y_2 \leq 2.286$ $-0.508 \leq Y_6 \leq 0.508$ $-0.508 \leq Y_4 \leq 0.508$ $0.508 \leq Y_2 \leq 1.524$ (m)
The number of design variables	15 size variables and 8 shape variables

The convergence process of the objective function is shown in Fig. 6 as the number of iterations increases using 100 iterations of the 15-bar truss with an initial population of 50 by MJSO and JSO, respectively, from which it can be seen that the MJSO algorithm is clearly superior to the JSO algorithm in terms of search capability after the inclusion of the elite strategy in the early stage, and the entire convergence process MJSO algorithm shows superior exploration capability than the JSO algorithm.

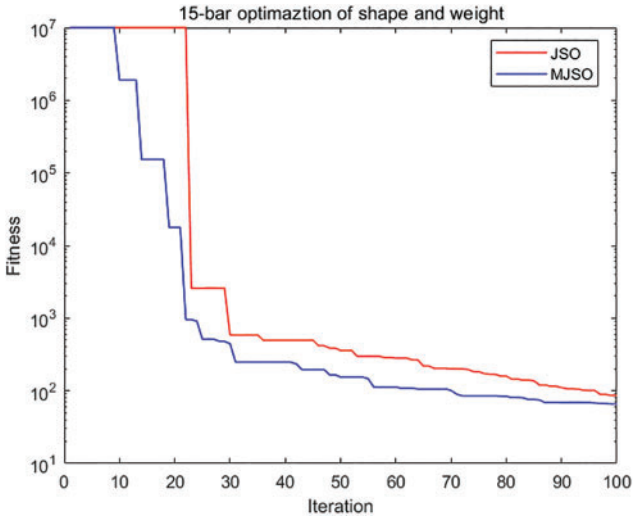

Figure 6: MJSO and JSO optimized the convergence process of 15-bar truss

Figs. 7 and 8 show the optimized structural shape of the truss after 100 iterations of the MJSO algorithm and the JSO algorithm. MJSO optimization is less than the original structure of two bars, JSO optimized only one bar.

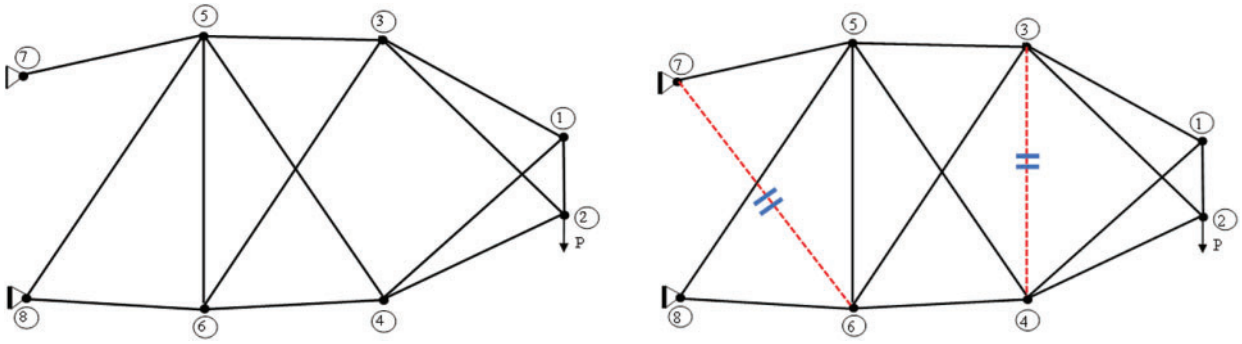

Figure 7: MJSO optimization results

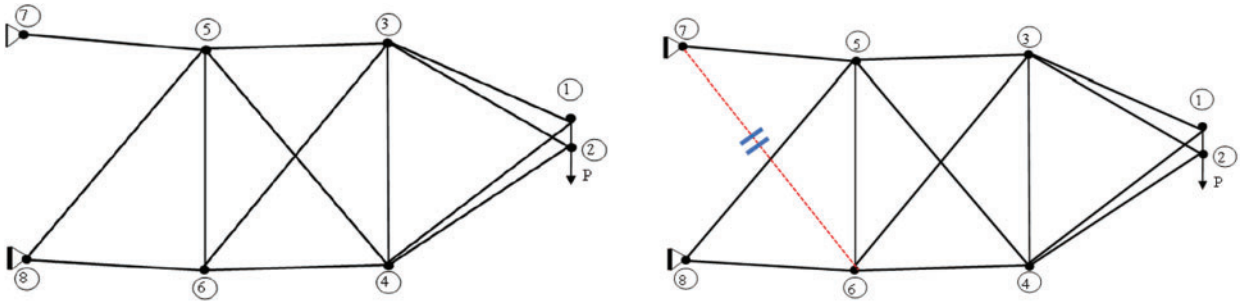

Figure 8: JSO optimization results

The MJSO and JSO algorithms were optimized for each unit section of the 15-bar truss, respectively, for 30 cycles, and the results of section optimization as well as the optimal weight optimum, average value and standard deviation after optimization are shown in Table 6.

Table 6: Comparison of optimization results of 15-bar truss

	Unit number	MJSO	JSO
Cross-sectional area ($\times 10^{-3}$) (m^2)	A_1	7.35	6.95
	A_2	5.96	5.7
	A_3	4.30	4.12
	A_4	2.32	1.30
	A_5	1.78	4.42
	A_6	0.933	1.81
	A_7	4.39	3.87
	A_8	0.00	4.40
	A_9	4.39	8.92
	A_{10}	0.00	0.00
	A_{11}	0.653	0.40
	A_{12}	2.69	4.72
	A_{13}	1.23	1.58

(Continued)

Table 6 (continued)

	Unit number	MJSO	JSO
	A_{14}	1.45	1.21
	A_{15}	4.04	3.29
Shape node displacement (m)	$X_5 = X_6$	3.0314	3.0902
	$X_3 = X_4$	5.7303	6.5812
	Y_1	1.7312	1.2844
	Y_2	0.8907	1.1111
	Y_3	3.3138	2.8395
	Y_4	0.0410	−0.1006
	Y_5	3.3346	2.8278
	Y_6	−0.0834	−0.1272
Best weight (kg)		29.857	38.325
Mean weight (kg)		30.634	41.793
Std.		2.57	4.41

From the above table, it can be seen that for the optimization of the 15-bar truss, the MJSO algorithm has one less string bar than the JSO algorithm when the constraints are satisfied, and the optimized weight average is 26% less than the JSO algorithm weight, which is more superior, and the overall standard deviation is smaller and more robust than the JSO algorithm.

4 Optimization of Steel Truss Lifting Spreader for Long-Span Covered Bridges

The project is a 54 m span corridor bridge with steel truss as the main structure. In the steel truss lifting stage, due to the site limitation, large lifting equipment cannot enter the site, so the project can only set up lifting brackets on the structure of Guanjiang Pavilion on both sides and use 4 lifting power of 70 tons jack to lift the steel truss.

The steel truss beam model is established according to the material and cross section provided by the design, the hoisting bracket adopts the beam unit, the steel beam adopts the truss unit, the wire rope adopts the cable unit for simulation, and the vertical rigid frame of the hoisting bracket and Guanjiang Pavilion adopts the rigid connection. Midas civil was used for the lifting analysis without considering the effect of live load.

4.1 No Spreader Lift

The jack is directly connected to the upper lugs of the steel joist through the steel strand to lift the steel joist with the structure arrangement as shown in the Fig. 9.

The structure is simulated by Midas Civil to analyze the construction process of the steel truss as a whole. The calculation shows that the maximum stress of the steel main truss of the spreading gallery is 203 MPa < 300 MPa, which meets the structural stress limit; the maximum deflection is 37.6 mm < L/400 = 135 mm, which meets the deflection limit. The relative displacement of adjacent section beam is 33 mm > 0.75 mm, which does not satisfy the construction condition of group bolts. Therefore, the lifting method needs to be reconsidered.

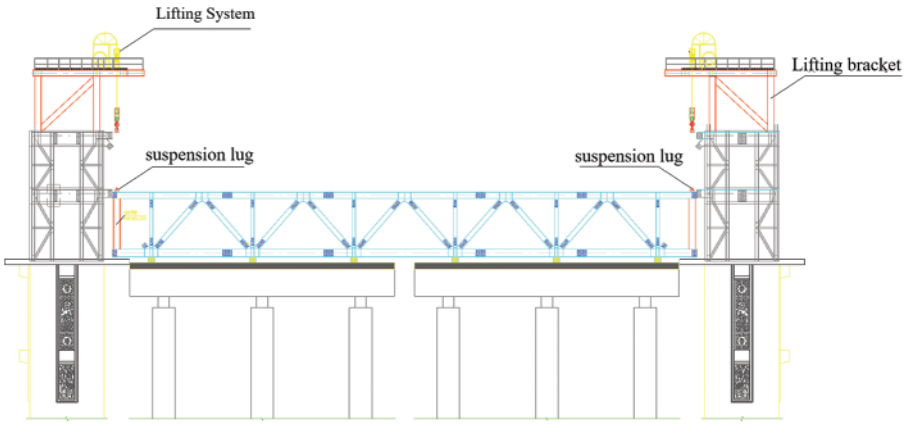

Figure 9: Lifting without spreader

4.2 Lifting with Rigid Flat Beam Spreader

For the common method of overall lifting construction, a rigid flat beam spreader is designed to meet the construction conditions of group bolts by connecting the steel strand with the rigid flat beam spreader to reduce the deformation between the beams of the steel joist. The structure is shown Fig. 10.

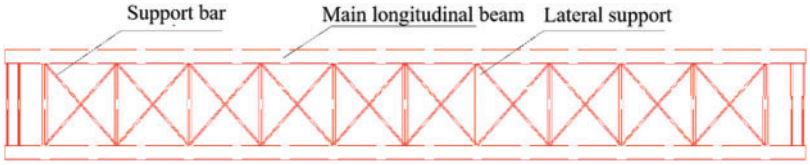

Figure 10: Spreader with rigid flat beam

The main longitudinal beam structure adopts square steel beam with cross-section height 1.4 m, width 0.9 m, web thickness 25 mm, top and bottom plate thickness 30 mm, cross brace structure adopts HM588 × 300 section steel, support bar structure adopts I32a.

Using Midas civil calculations, it can be obtained that the maximum stress of the steel truss is 45 Mpa, the maximum deflection is 13.5 mm, and the relative displacement of the adjacent internode beam is 0.7 mm, all of which meet the limit requirements, but the weight of the spreader itself is 120 tons, plus the weight of the steel truss 180 tons, exceeding the lifting power limit of the jack, required weight optimization of the spreader.

4.3 Optimization of Rigid Flat Stretcher Spreader Based on MJSO Algorithm

4.3.1 Model Construction

In optimizing the steel joist structure, section A with nodal vertical displacement X is set as the design variable. The maximum load is kept constant and the known permissible stresses, elastic modulus, stress parameters and permissible displacements of the nodes are the constraints in order to achieve the minimum mass of the structure.

① Design variables

$$A = [A_1, A_2, \cdots, A_n, X_I, X_2, \cdots, X_n]^T \tag{21}$$

A is the design variable of the structure; n is the number of section variable groups of continuous bars.

② Objective function

$$\min f(x) = \sum_{i=1}^{n} \rho_i A_i L_i \tag{22}$$

$f(x)$ is the weight of the structure; ρ is the material density of group I members; A is the cross-sectional area of group i members; L is the length of group i members.

③ Constraints

$$s.t. \begin{cases} g_i^{\sigma}(A) = [\sigma_i] - \sigma_i \geq 0, i = 1, 2, \cdots, K \\ g_{jl}^{u}(A) = \left[u_{jl}\right] - u_{jl} \geq 0, j = 1, 2, \cdots, m; \\ \qquad\qquad l = 1, 2, \cdots, \omega \\ A_{min} \leq A \leq A_{max} \\ x_b^a \leq x_b \leq x_b^c, b = 1, 2, \cdots, z \end{cases} \tag{23}$$

K is the total number of bars; m is the total number of nodes; $g_i^{\sigma}(A)$ is the stress constraint; $g_{jl}^{u}(A)$ is the displacement constraint; A_{min} is the minimum cross-sectional area of the member; A_{max} is the maximum cross-sectional area; ω is the number of dimensions constrained by the node displacement; σ_i is the most unfavorable stress value of the ith group of members; u_{jl} the maximum displacement of node j; and the upper and lower limits of the x_b coordinate b.

④ Treatment of constrained condition

As with other direct search methods, MJSO cannot be used directly for constrained optimization problems. The penalty function method is one of the most widely used constraint processing techniques [27]. In this paper, a self-adaptive penalty function strategy is proposed to solve constrained optimization problems. Based on this method, Tao et al. [28] proposed a self-adaptive penalty function strategy for solving constrained optimization problems and the fitness function is written as Eqs. (24)–(26):

$$W(x) = f(x) + P \sum_{i=1}^{P} h(t)\, g_i(x) f(x) \tag{24}$$

$$g_i(x) = \frac{\sigma_i}{\sigma_{iall}} - 1 \leq 0 \tag{25}$$

$$h(t) = 1 + t/T \tag{26}$$

where $W(x)$ is an unconstrained objective function (the objective function after penalty); σ_i and σ_{iall} are the actual value and allowable value of ith constraint; $h(t)$ is the penalty parameter.

The MJSO algorithm is introduced according to the steel joist structure optimization model, and its optimization process is shown in Fig. 11.

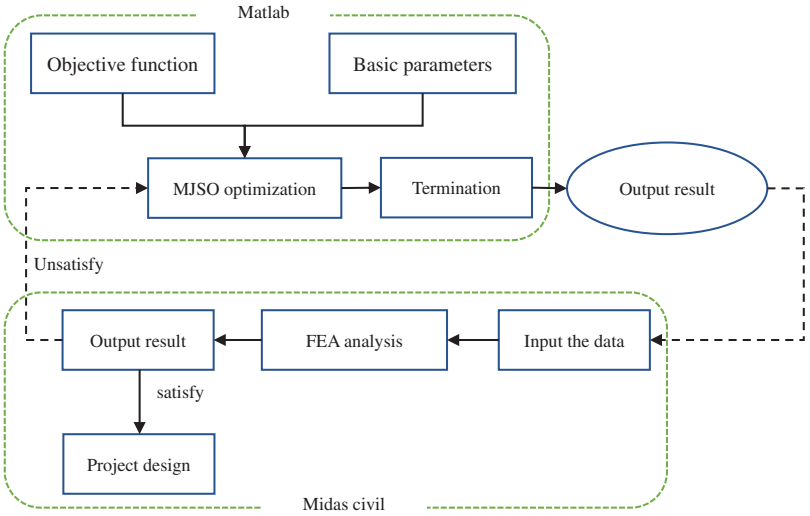

Figure 11: Flow chart of optimized steel truss by MJSO

4.3.2 Optimized Design of MJSO Algorithm Spreader

Based on the flat beam model, a truss model is established as in Fig. 12, with steel strength of Q345 steel, density of 7850 kg/m^3 and modulus of elasticity of 2.06E+08 kN/m^2. The cross-section size of the string bar is HM588 × 300, and the cross-section size of the middle support bar is I32a. 18 nodes and 26 units are set up, and the constraints are node 1 and node 10 are rigidly connected, while other nodes are not constrained. In this node 1, node 10 vertical applied jack maximum lifting weight of 70 tons of force P. The upper limit of stress constraint is 125768.5 kN/m^2, the maximum vertical displacement is 0.135 m, the cross-sectional area optimization boundary between 0–0.121 m^2, the node optimization boundary between 2–4 m. The design variables are section A1–A26, Y11 = Y18, Y12 = Y17, Y13 = Y16, Y14 = Y15, total 30 variables. The calculation was performed by MATLAB with 200 iterations and an initial population of 50, and the optimization results of MJSO and JSO are shown in the Table 7.

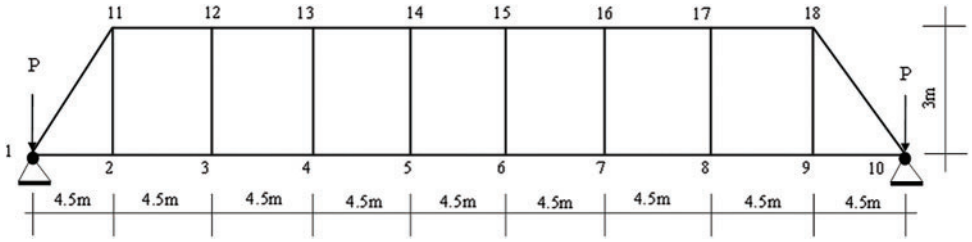

Figure 12: Spreader truss model diagram

Table 7: Spreader optimization results

	Unit number	MJSO	JSO
Cross-sectional area (m^2)	A_1	0.01825	0.01935
	A_2	0.01833	0.02013
	A_3	0.01842	0.02045
	A_4	0.01906	0.02132
	A_5	0.01913	0.02195
	A_6	0.01901	0.02183
	A_7	0.01895	0.02064
	A_8	0.01887	0.02051
	A_9	0.01841	0.02033
	A_{10}	0.01306	0.01623
	A_{11}	0.01245	0.01589
	A_{12}	0.01289	0.01577
	A_{13}	0.01244	0.01602
	A_{14}	0.01345	0.01679
	A_{15}	0.01268	0.01583
	A_{16}	0.01224	0.01494
	A_{17}	0.01339	0.01569
	A_{18}	0.01244	0.01502
	A_{19}	0.01321	0.01591
	A_{20}	0.01221	0.01488
	A_{21}	0.01329	0.01421
	A_{22}	0.01316	0.01473
	A_{23}	0.01269	0.01562
	A_{24}	0.01278	0.01455
	A_{25}	0.01282	0.01454
	A_{26}	0.01255	0.01432
Shape node displacement (m)	$Y_{11} = Y_{18}$	2.45	2.88
	$Y_{12} = Y_{17}$	3.26	3.43
	$Y_{13} = Y_{16}$	3.58	3.76
	$Y_{14} = Y_{15}$	3.73	3.90
Weight (kg)		14001.48	22315.56

Figs. 13 and 14 show the optimized shape of the spreader for 100 *vs.* 200 iterations of MJSO, respectively. As can be seen from Fig. 15, MJSO has better optimization efficiency than JSO, and as can be seen from Table 6, MSJO is 37% lighter than JSO in optimizing spreader, and MSJO has better overall performance than JSO.

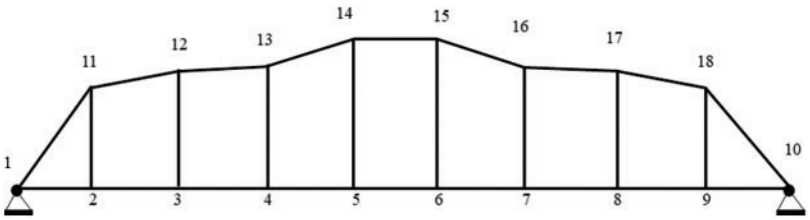

Figure 13: MJSO iteration 100th model

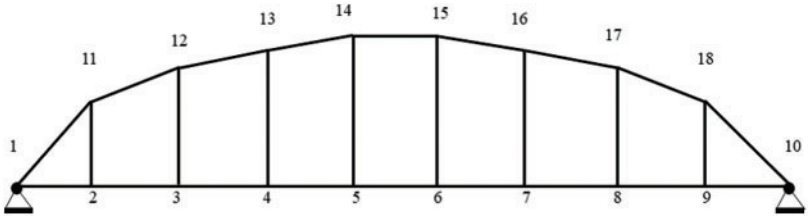

Figure 14: MJSO iteration 200th model

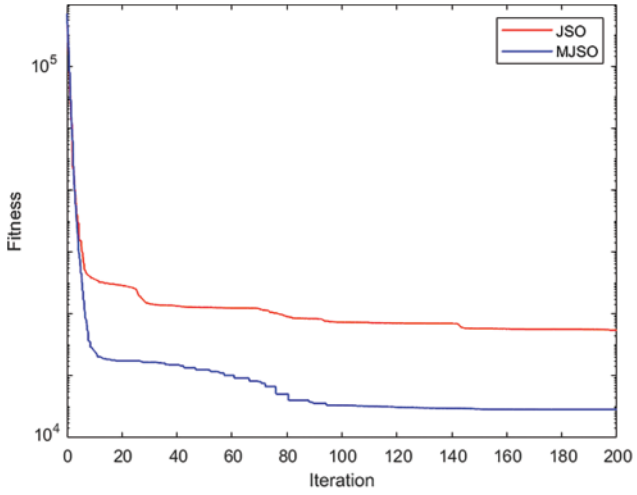

Figure 15: MJSO and JSO optimizes spreader convergence process

4.3.3 Finite Element Analysis after Optimization

Finite element analysis is performed on the optimized structure, and the model is established and calculated according to the actual situation and actual construction working conditions. The size is selected according to the optimized cross-section, and the overall shape such as arch structure is considered after optimization, i.e., the curve is selected for monolization. The two arch spreaders are connected by support bars and props are set in each span, and the overall spreader is shown in Fig. 16. In this study, Midas civil is used for lifting operation of the optimized spreader, and the force as well as deformation of the optimized spreader are analyzed.

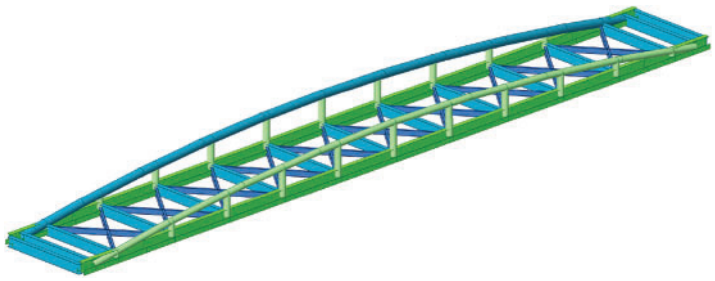

Figure 16: Schematic diagram of arch truss spreader

According to the optimization results, according to the Chinese steel design code standard for section selection, the main longitudinal beam adopts 2HM588 × 300 mm section steel, the arch support and upright bar all adopt Φ426 ∗ 10 mm steel pipe, the support bar selects HM588 × 300 mm, the spreader elevation and plane are shown in Figs. 17 and 18.

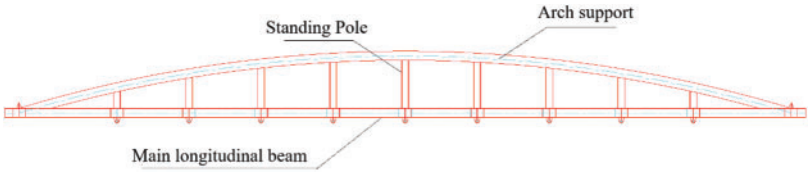

Figure 17: Elevation diagram of arch truss spreader

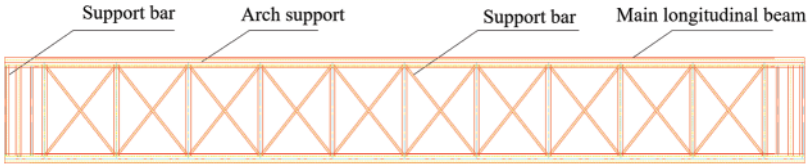

Figure 18: Schematic diagram of arch truss spreader plan

Installed under no wind or light wind condition, the load combination is as follows:

Self-weight of lifting bracket + Weight of spreader + Weight of main truss;

After software calculation, the maximum stress of all bars is 180 MPa, which is less than the strength design value [f] = 215 MPa, so it meets the requirements.

The maximum deformation under this working condition is 77.5 mm, and the deformation is as follows:

L/200 = 51910/400 = 129 mm;

which meets the requirement according to the cantilever member allowable deflection value specified in the Code for design of steel structures. The deformation difference of upper and lower chords is as follows:

74.899 − 74.775 = 0.114 mm < 0.75 mm;

which meets the requirements of group bolt construction. The spreader weighs about 50 tons, and the main truss weighs 110 tons, which is 46% lighter than the flat beam spreader and the requirement of jacking lifting limit.

5 Conclusion

To solve the truss shape and size optimization problem more effectively and efficiently, this paper proposes an improved jellyfish search algorithm (MJSO) by integrating the advantages of adaptive weight function and elite strategy through the jellyfish search algorithm. This algorithm improves the global search ability in the early stage of the algorithm by the elite strategy and the adaptive weight function and enhances the local search ability in the late stage of the algorithm by the adaptive weight function to avoid falling into the local optimum. Through testing and application, the advantages of the MJSO algorithm are as follows:

(1) In the benchmark function test, the MJSO algorithm performs better than the JSO algorithm overall for unimodal, multi-modal, separable, and non-separable functions.

(2) The MJSO algorithm is more robust than the JSO, PSO, GWO, and ACOR algorithms in terms of the effect of truss lightweight, and the optimization results are better.

(3) In the test of simultaneous optimization of truss shape and weight, the MJSO algorithm reduced two chords while JSO reduced only one chord under the constraints, and the weight of MSJO was 26% lighter than that of JSO, and the standard deviation of MJSO through 30 cycles was 1.84 smaller than that of JSO, and the robustness of MJSO is also superior. In general, MSJO is superior to JSO in optimizing the shape and weight of trusses.

(4) For the restricted construction environment of the site, the structure needs to be lifted using a spreader. Through the optimization of the lifting spreader by MJSO and JSO, the result of lightweight spreader by MJSO is better than JSO, and the problem of large lifting weight of steel joist as a whole is solved.

Since the MJSO algorithm improvement is only optimized for the passive motion in the early ocean currents and the later local search, it lacks the optimization for the active motion part, which needs to be enhanced here.

Acknowledgement: Open Access funding enabled and organized by Xiamen Transportation Infrastructure Intelligent Management and Maintenance Engineering Technology Research Center. The authors gratefully acknowledge the support.

Funding Statement: This research was funded by the National Natural Science Foundation of China (Grant No. 51305372), the Open Fund Project of the Transportation Infrastructure Intelligent Management and Maintenance Engineering Technology Center of Xiamen City (Grant No. TCIMI201803) and the Project of the 2011 Collaborative Innovation Center of Fujian Province (Grant No. 2016BJC019).

Author Contributions: The authors confirm contribution to the paper as follows: study conception and design: X.W., S.F. and Y.H.; data collection: S.F., W.W., Z.L. and F.J.; analysis and interpretation of results: X.W., S.F., W.W. and Y.H.; draft manuscript preparation: S.F., X.W. and Y.H. All authors reviewed the results and approved the final version of the manuscript.

Availability of Data and Materials: Not applicable.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

1. Shi, K., Ruan, Z., Jiang, Z., Lin, Q., Wang, L. (2018). Improved plant growth simulation and genetic hybrid algorithm (PGSA-GA) and its structural optimization. *Engineering Computations, 35(1),* 268–286. https://doi.org/10.1108/EC-03-2017-0113
2. Kim, N. I., Kim, Y. W., Lee, J., Kang, J. W. (2018). Two-stage optimization based on force method for damage identification of planar and space trusses. *International Journal of Steel Structures, 18(1),* 1–12. https://doi.org/10.1007/s13296-018-0301-x
3. Viet-Hung, T., Seung-Eock, K. (2018). Reliability-based design optimization of nonlinear inelastic trusses using improved differential evolution algorithm. *Advances in Engineering Software, 121(4),* 59–74. https://doi.org/10.1016/j.advengsoft.2018.03.006
4. Azad S. K., Bybordiani M., Azad S. K., Jawad F. K. J. (2018). Simultaneous size and geometry optimization of steel trusses under dynamic excitations. *Structural and Multidisciplinary Optimization, 58(6),* 2545–2563. https://doi.org/10.1007/s00158-018-2039-7
5. Grzywinski, M., Dede, T., Ozdemir, Y. I. (2019). Optimization of the braced dome structures by using Jaya algorithm with frequency constraints. *Steel and Composite Structures, 30,* 47–55. https://doi.org/10.12989/scs.2019.30.1.047
6. Habibi, A., Bidmeshki, S. (2019). An optimized approach for tracing pre- and post-buckling equilibrium paths of space trusses. *International Journal of Structural Stability and Dynamics, 19(4),* 1950040. https://doi.org/10.1142/S0219455419500408
7. Artar, M., Daloglu, A. T. (2019). Optimum design of steel space truss towers under seismic effect using Jaya algorithm. *Structural Engineering and Mechanics, 71,* 1–12. https://doi.org/10.12989/sem.2019.71.1.001
8. Kaveh, A., Moghanni, R. M., Javadi, S. M. (2019). Optimum design of large steel skeletal structures using chaotic firefly optimization algorithm based on the Gaussian map. *Structural and Multidisciplinary Optimization, 60(3),* 879–894. https://doi.org/10.1007/s00158-019-02263-1
9. Wong, F. T., Prayogo, D., Putra, R. E., Joseph, J. (2020). Member sizing optimization of large scale steel space trusses using a symbiotic organisms search algorithm. *Journal of Physics: Conference Series,* 1625. https://doi.org/10.1088/1742-6596/1625/1/012019
10. Ha, M. H., Vu, Q. V., Truong, V. H. (2020). Optimization of nonlinear inelastic steel frames considering panel zones. *Advances in Engineering Software, 142,* 102771. https://doi.org/10.1016/j.advengsoft.2020.102771
11. Gholizadeh, S., Danesh, M., Gheyratmand, C. (2020). A new Newton metaheuristic algorithm for discrete performance-based design optimization of steel moment frames. *Computers & Structures, 234*. https://doi.org/10.1016/j.compstruc.2020.106250
12. Es-Haghi, M. S., Shishegaran, A., Rabczuk, T. (2020). Evaluation of a novel asymmetric genetic algorithm to optimize the structural design of 3D regular and irregular steel frames. *Frontiers of Structural and Civil Engineering, 14,* 1110–1130. https://doi.org/10.1007/s11709-020-0643-2
13. Truong, V. H., Vu, Q. V., Thai, H. T., Ha, M. H. (2020). A robust method for safety evaluation of steel trusses using gradient tree boosting algorithm. *Advances in Engineering Software, 147(5),* 102825. https://doi.org/10.1016/j.advengsoft.2020.102825
14. Zhou, H., Zhang, G., Wang, X., Ni, P., Zhang, J. (2020). A hybrid identification method on butterfly optimization and differential evolution algorithm. *Smart Structures and Systems, 26,* 345–360. https://doi.org/10.12989/sss.2020.26.3.345
15. Bigham, A., Gholizadeh, S. (2020). Topology optimization of nonlinear single-layer domes by an improved electro-search algorithm and its performance analysis using statistical tests. *Structural and Multidisciplinary Optimization, 62(4),* 1821–1848. https://doi.org/10.1007/s00158-020-02578-4

16. Gholizadeh, S., Fattahi, F. (2021). Multi-objective design optimization of steel moment frames considering seismic collapse safety. *Engineering with Computers, 37(2),* 1315–1328. https://doi.org/10.1007/s00366-019-00886-y
17. Artar, M., Carbas, S. (2021). Discrete sizing design of steel truss bridges through teaching-learning-based and biogeography-based optimization algorithms involving dynamic constraints. *Structures, 34(11),* 3533–3547. https://doi.org/10.1016/j.istruc.2021.09.101
18. Azad, S. K., Aminbakhsh, S., Shaban, S. S. S. (2021). Multi-stage guided stochastic search for optimization and standardization of free-form steel double-layer grids. *Structures, 34(1),* 678–699. https://doi.org/10.1016/j.istruc.2021.07.068
19. Carbas, S., Artar, M. (2022). Optimum discrete design of steel planar trusses comprising earthquake load impact. In: Kim, J., Deep, K., Geem, Z., Sadollah, A., Yadav, A. (Eds.), *Proceedings of 7th International Conference on Harmony Search, Soft Computing and Applications,* pp. 369–379. https://doi.org/10.1007/978-981-19-2948-9_36
20. Zhao, J., Wang, K., Wu, D., Huang, Q., Yu, M. (2022). Optimization strategy for modal test measurement points of large-span steel beams based on improved particle swarm optimization algorithm with random weights. *Applied Sciences, 12(23),* 12082. https://doi.org/10.3390/app122312082
21. Ojha, V., Panto, B., Nicosia, G. (2023). Adaptive search space decomposition method for pre- and post-buckling analyses of space truss structures. *Engineering Applications of Artificial Intelligence, 117*. https://doi.org/10.1016/j.engappai.2022.105593
22. Eser, H., Hasancebi, O. (2023). Capacity controlled search: A new and efficient design-driven method for discrete size optimization of steel frames. *Computers & Structures, 275*. https://doi.org/10.1016/j.compstruc.2022.106937
23. Sang-To, T., Le-Minh, H., Abdel Wahab, M., Thanh, C. L. (2023). A new metaheuristic algorithm: Shrimp and Goby association search algorithm and its application for damage identification in large-scale and complex structures. *Advances in Engineering Software, 176(1),* 103363. https://doi.org/10.1016/j.advengsoft.2022.103363
24. Azizi, M., Aickelin, U., Khorshidi, H. A., Baghalzadeh Shishehgarkhaneh, M. (2023). Energy valley optimizer: A novel metaheuristic algorithm for global and engineering optimization. *Scientific Reports, 13(1),* 1–23. https://doi.org/10.1038/s41598-022-27344-y
25. Kadkhoda Mohammadi, S., Nazarpour, D., Beiraghi, M. (2023). A novel metaheuristic algorithm inspired by COVID-19 for real-parameter optimization. *Neural Computing and Applications, 35(14),* 10147–10196. https://doi.org/10.1007/s00521-023-08229-1
26. Chou, J. S., Truong, D. N. (2021). A novel metaheuristic optimizer inspired by behavior of jellyfish in ocean. *Applied Mathematics and Computation, 389,* 125535. https://doi.org/10.1016/j.amc.2020.125535
27. Coello, C. A. (2002). Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: A survey of the state of the art. *Computer Methods in Applied Mechanics and Engineering, 191(11–12),* 1245–1287. https://doi.org/10.1016/S0045-7825(01)00323-1
28. Tao, R., Meng, Z., Zhou, H. (2021). A self-adaptive strategy based firefly algorithm for constrained engineering design problems. *Applied Soft Computing, 107(Suppl 1),* 107417. https://doi.org/10.1016/j.asoc.2021.107417