Expert Systems With Applications 280 (2025) 127465

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

KATSA: KNN Ameliorated Tree Seed Algorithm for complex optimization problems

Jianhua Jiang [a,b,*], Jiaqi Wu [a,b], Jinmeng Luo [a,b], Xianqiu Meng [c], Lize Qian [d], Keqin Li [e]

[a] Center for Artificial Intelligence, Jilin University of Finance and Economics, Changchun, 130117, PR China
[b] Jilin Province Key Laboratory of Fintech, Jilin University of Finance and Economics, Changchun, 130117, PR China
[c] College of Computer Science and Technology, Jilin University, Changchun, 130012, PR China
[d] Institute of Energy Research, Jiangxi Academy of Sciences, Nanchang, 330096, PR China
[e] Department of Computer Science, State University of New York, New Paltz, NY, 12561, USA

ARTICLE INFO

Keywords:
Tree Seed Algorithm
KNN
KATSA
Optimization problem
Swarm intelligence

ABSTRACT

Tree Seed Algorithm (TSA) is an outstanding algorithm for optimization problems, but it inevitably falls into the local optimum and has a low convergence speed in solving complex problems. This paper aims to address these above defects. Inspired by efficient learning from neighbors, a K-Nearest Neighbor (KNN) mechanism is adopted to enhance the tree or seed generation strategies for achieving the balance between exploitation and exploration. The proposed algorithm is named the KNN Ameliorated Tree Seed Algorithm (KATSA). First, based on the current best tree, the search space is divided into best and non-best neighbor areas by the KNN mechanism. Based on this division approach, the proposed seed generation strategy has a precise heuristic, and the convergence speed can be accelerated. Second, the proposed seed generation and tree migration strategies integrate the proposed dynamic regulation mechanism, which reduces the possibility of falling into a local optimum. Third, the proposed feedback mechanism can effectively balance exploration and exploitation. With these enhancements from the KNN mechanism, KATSA can converge to the global optima more effectively during its evolutionary process. The results obtained from IEEE CEC 2014 benchmark function evaluation verify the excellent performance of the KATSA when compared with some recent variants, including STSA, EST-TSA, fb_TSA, and MTSA. In addition, GWO, PSO, BOA, BA, GA, LSHADE, and RSA are also adopted for some benchmark comparative experiments. The applicability of the proposed KATSA is demonstrated by three real complex and constrained problems when compared to TSA, fb_TSA, LSHADE, RSA, GWO, ABC, and PSO. The experimental results show that the proposed KATSA can obtain stable and optimal results on these complex problems. The source code is available at www.jianhuajiang.com.

1. Introduction

The optimization problem is essential in the range of human productive activities, such as product design (Zhao et al., 2024; Zhou et al., 2021), resource allocation (Deng et al., 2022; Urooj et al., 2024), production scheduling (Chen et al., 2020; Goli et al., 2021; Shehadeh et al., 2024), path planning (Eswaran et al., 2024) and so on. Traditional optimization methods mainly rely on gradient-based information about the involved functions to find the optimal solution, such as Dynamic Programming (Zhang et al., 2002), Least Squares (Suykens & Vandewalle, 1999), and Newton's method (Qi & Jie, 1993). However, with the growing complexity of big data-driven optimizations, significantly since the number of feasible solutions has exponentially increased, it is hard to solve these problems based on these traditional methods.

Inspired by the wisdom of nature, population-based meta-heuristic algorithms provide new ideas for solving complex optimization problems (Deng & Liu, 2023; Salgotra et al., 2023). They mainly rely on nature-inspired computing, with little or no use of objective functions and constraints (Bejarbaneh et al., 2019; Kılıç et al., 2021). Hence, these algorithms have the advantage of being global, parallel, and versatile and do not require detailed problem information. Typical meta-heuristic algorithms include Genetic Algorithm (GA) (Holland & Reitman, 1977), Particle Swarm Optimization (PSO) (Kennedy & Eberhart, 1995), Ant Colony Optimization (ACO) (Dorigo et al., 2006),

* Corresponding author.
E-mail addresses: jjh@jlufe.edu.cn (J. Jiang), 5221191008@s.jlufe.edu.cn (J. Wu), 5221191007@s.jlufe.edu.cn (J. Luo), mengxq23@mails.jlu.edu.cn (X. Meng), qianlize@jxas.ac.cn (L. Qian), lik@newpaltz.edu (K. Li).
URL: https://www.jianhuajiang.com (J. Jiang).

https://doi.org/10.1016/j.eswa.2025.127465
Received 14 November 2023; Received in revised form 11 September 2024; Accepted 26 March 2025
Available online 8 April 2025

Artificial Bee Colony (ABC) (Karaboga & Ozturk, 2011), Cuckoo Search (CS) (Yang & Deb, 2014), Bat Algorithm (BA) (Yang & He, 2013), Besiege and Conquer Algorithm (BCA) (Jiang et al., 2024) and so on. These meta-heuristic algorithms are implemented by achieving a good balance between exploration and exploitation (Lin & Gen, 2009). However, no one algorithm can solve all optimization tasks according to the No Free Lunch (NFL) theorem (Droste et al., 2002).

Among them, Kiran proposed the Tree Seed Algorithm (TSA) inspired by tree and seed evolution (Kiran, 2015). TSA usually provides quick and easy access to the optimal or near-optimal solution (Beşkirli & Kiran, 2023; Jiang, Xu et al., 2020). In the TSA, each tree and seed represents a solution in the search space. The TSA is better than some meta-heuristic algorithms, such as PSO, GA, and some emerging heuristic algorithms (Jiang, Meng et al., 2020). In addition, it has a more straightforward structure, higher accuracy, and stronger robustness than some traditional intelligent optimization algorithms (Cinar et al., 2020; Jiang et al., 2019). However, it has disadvantages, such as premature convergence and easy stagnation in the local optimum (Jiang et al., 2019). This study aims to address these issues by proposing a TSA variant with modification and hybridization.

1.1. Motivations

In the high-dimensional and multi-modal problems, the search space and objective space have certain relationships (Li et al., 2023). That indicates the neighbor area of the best solution in the search space is of some heuristic meaning heuristic information that can be adopted to accelerate its convergence. The search in the neighbor area around the current best solution has more probability of enhancing the exploitation ability of the algorithm, while the K-Nearest Neighbor (KNN) is an excellent way to classify data from the spatial view (Fix, 1985). Based on the KNN, we can find the population located in the neighbor area of the current best solution. Therefore, by introducing this mechanism into the TSA, seeds will no longer be generated randomly but heuristic-guided. Therefore, the **motivations** of this paper are as follows:

- During the evolutional process of TSA, seeds are generated by its parent trees. However, this strategy will generate many useless seeds since its insufficient heuristic information. A search space division strategy is required to provide heuristic guidance for seeds. KNN can classify data from spatial dimensions which means that KNN can provide more heuristic information when dealing with high-dimensional and multi-modal problems.
- TSA adopts a constant search tendency (ST) parameter for all trees to balance the exploration and exploitation. This strategy is effective when solving simple problems. However, for complex problems, each tree has different gradients in objective space, and it is difficult to adapt to different situations through the same ST. Therefore, a dynamic ST strategy is required to balance the exploration and exploitation for each tree.
- TSA adopts a single way to generate seeds, which leads to an insufficient population diversity. TSA relies on the current tree and does not well exchange information with the current best tree. Therefore, the seeds are mainly distributed around its parent tree with insufficient diversity.
- TSA is easy to trap to the local optima because of its single seed generation strategy. Therefore, a variant that can avoid local optima is required to improve the TSA to solve the high-dimensional and multi-modal problems.

1.2. Contributions

In this study, we have proposed three mechanisms to improve the capability of TSA when solving high-dimensional and multi-modal problems based on the KNN principle. The **contributions** of this paper can be summarized as follows:

Table 1
Parameters used in TSA.

Parameters	Symbol
D	Problem dimension
N	Number of trees in a population
ns	Number of seeds in a tree
ST	Search tendency
$L_{j,min}$	Lower bound of the j_{th} dimension of search space
$H_{j,max}$	Higher bound of the j_{th} dimension of search space
$T_{i,j}$	j_{th} dimension of the i_{th} tree
$rand$	A random number in range of [0, 1]
$S_{i,j}$	j_{th} dimension of the i_{th} seed
B_j	the j_{th} dimension of best tree location
$T_{r,j}$	The neighbor of i_{th} tree in j_{th} dimension, r is the random index generate from the trees population
$\alpha_{i,j}$	Scaling factor of $S_{i,j}$ produced randomly in range of [−1, 1]
$Iters$	Number of iterations
$MaxIters$	Termination condition

- Search space division mechanism is proposed to divide the search space into two sub-spaces. The trees located in different areas are classified into two categories. Trees with different categories will generate more promising seeds.
- Dynamic search tendency mechanism is proposed to allocate different ST to each tree. This mechanism makes trees adjust their ST to balance exploration and exploitation adaptively.
- Binary-switched seed generation mechanism is proposed. This mechanism fully considers the best solution for the current generation. The population diversity is improved by proposing different seed generation methods. The tree evolutionary trajectory is recorded to adopt different seed generation methods dynamically for local optima avoidance.
- The proposed algorithm is evaluated on the IEEE CEC 2014 benchmark functions. The experiment results show that the proposed algorithm outperforms other comparative algorithms in high-dimensional and multi-modal problems.

To verify its performance, the proposed KATSA algorithm is experimented on the IEEE CEC 2014 benchmark functions, including 4 classes (unimodal, multimodal, composite, and hybrid) and 3 real complex and constrained engineering problems.

The remainder of this paper is organized as follows. Section 2 reviews the TSA and relevant works. Section 3 presents the inspiration and the proposed algorithm. Both qualitative and quantitative results of algorithms on various benchmark functions and real complex constrained engineering problems are presented and discussed in Section 4. Finally, Section 5 concludes the work and suggests several future research directions.

2. Relevant works

2.1. An overview of tree seed algorithm

Tree Seed Algorithm (TSA) is a novel meta-heuristic algorithm based on the evolution mechanism between the tree and its seeds (Kiran, 2015). TSA has been cited 336 according to Google Scholar. Table 1 lists all parameters associated with TSA.

- Eq. (1) generates a new initial tree position.

$$T_{i,j} = L_{j,min} + rand \times (H_{j,max} - L_{j,min}) \tag{1}$$

- Seed generation is realized by Eqs. (2) and (3). There are two ways to generate seeds: the first is for local search, while the second is for global search. The search tendency parameter ST is adopted to balance exploration and exploitation.

$$S_{i,j} = T_{i,j} + \alpha_{i,j} \times (B_j - T_{r,j}) \tag{2}$$

Eq. (2) plays a pivotal role in facilitating the exploitation phase of TSA around the current tree.

$$S_{i,j} = T_{i,j} + \alpha_{i,j} \times (T_{r,j} - T_{i,j}) \tag{3}$$

The Eq. (3) enables the TSA algorithm to perform a global search based on the current tree, encompassing a broader exploration of the solution space.

- Computation of the iterations (*Iters*) by Eq. (4).

$$Iters = Iters + 1 \tag{4}$$

where, $Iters$ on the right side represents the current number of iterations.

In the entire search space, excellent trees and seeds will continue to be iteratively updated, and new seeds will be generated continuously to complete both global and local searches. The seed generation mechanism (Eqs. (2) and (3)) is used to generate seeds depending on the search tendency (ST) parameter.

In the basic TSA, another important focus is the number of seeds produced by each tree. However, TSA has some deficiencies (Jiang et al., 2019; Jiang, Meng et al., 2020):

- The tree selection strategy is based on a randomly selected tree which affects its convergence efficiency.
- All seeds are generated in a single way, reduces diversity.
- The balance between exploration and exploitation depends on a constant ST without feedback, which is unreasonable.

2.2. An overview of TSA

Recently, TSA has gained attention in the algorithm design community with 336 citations. In the past few years, many excellent variants of TSA have been proposed. They mainly focus on tree migration, seed generation, feedback mechanisms, and real-world applications.

- For tree migration, the Migration Tree Seed Algorithm (MTSA) (Jiang et al., 2022) integrates hierarchical gravity learning and random-based migration inspired by Grey Wolf Optimizer (GWO) (Mirjalili et al., 2014) to address imbalance, local stagnation and premature convergence. Additionally, two tree migration mechanisms are proposed in the Triple Tree Seed Algorithm (TriTSA) (Jiang et al., 2021).
- For seed generation, the Tree Seed Algorithm with Sine–Cosine (TSASC) (Jiang, Han et al., 2020) incorporates the Sine–Cosine Algorithm (SCA) (Mirjalili, 2016). Previously, excessive reliance on specific group members weakened the ability to explore (Dehghani & Trojovskỳ, 2022). Meanwhile, the Sine Tree Seed Algorithm (STSA) (Jiang, Xu et al., 2020) hypothesizes that seeds can be generated from more to less, producing more in the early search stage and fewer later. To achieve this goal, STSA proposed an adaptive regulation mechanism k, which linearly varies with the number of iterations. To address TSA's lack of a mechanism to avoid local optima and failed seed classification, Kiran and Hakli (2021) integrate four different approaches (withering process, sequential seed generation, best-based solution update rule and dimensional selection) into basic TSA, naming it New Tree Seed Algorithm (NTSA).
- For the feedback mechanism, the feedback Tree Seed Algorithm (fb_TSA) (Jiang, Meng et al., 2020) is an outstanding variant that use a feedback mechanism to overcome the shortcoming of the Search Tendency (ST) mechanism. The value of ST and the numbers of seeds (ns) are dynamically adjusted during the optimization process. Jiang et al. (2019) pointed out that the basic TSA fails to balance exploration and exploitation and must fully consider the current optimal position. They redesigned the ST mechanism and proposed the Effective Search Tendency Tree Seed algorithm (EST-TSA).
- For the real-world application, SimLogicTSA (Cinar & Kiran, 2018) is applied to solving binary optimization by using logic gates (LogicTSA) and similarity measurement techniques (SimTSA). This variant of TSA is first used to resolve unconstrained optimization problems. CTSA (Babalik et al., 2018) introduced Deb's rules to select the promising trees and seeds based on the original TSA to resolve constrained optimization issues. The Levy search mechanism and a new seed updating equation have been added to TSA (I-TSA) (Ding et al., 2019) to solve the nonlinear hysteric parameter identification problem.

Given the above aspects, scholars have mainly focused on tree migration and seed generation mechanisms to enhance the evolutional mechanism of TSA. However, these proposed mechanisms need improvements in terms of slow convergence, lack of focus on the relationship between seeds, lower seed diversity, and other areas. This paper attempts to address these shortcomings.

2.3. An overview of KNN

The K-Nearest Neighbors (KNN) is a non-parametric classification method (Fix, 1985), which is simple but effective in many cases (Adithiyaa et al., 2020; Liang et al., 2020; Liu et al., 2022). Many scholars have proposed variants of KNN to improve the performance of classification tasks (Uddin et al., 2022).

- **Adaptive KNN (AdaNN)**: The adaptive KNN algorithm aims to find the optimal k value for each test example (Sun & Huang, 2010). The author proposed a sub-algorithm called 9NN to determine the optimal k of training examples. The main algorithm finds the nearest neighbor from the training example based on the Euclidean distance for each test example, and the test example inherits the k value of the nearest neighbor. The variant then adopts corresponding KNN algorithm to get the test examples' class labels.
- **Locally Adaptive KNN with Discrimination Class (DC-LAKNN)**: This variant of KNN utilizes the discrimination classes to determine the optimal k value (Pan et al., 2020). The quantity and distribution of the majority class neighbors and the second majority class neighbors in the k-neighborhood are considered by the discrimination classes concept. The algorithm defines discriminating classes through multiple stages, then selects one of those classes and forms a ranking table with various k values, distances from centroids, and their ratios.
- **Fuzzy KNN (F-KNN)**: The fuzzy KNN is proposed based on the concept of membership assignment (Keller et al., 1985). The variant tries to find the k-nearest neighbors of a testing dataset from the training dataset, similar to the original KNN. It then confers "membership" values to every class found in the k-nearest neighbor list. The fuzzy math method based on the weight of each class is employed to calculate the membership fitness. The class with the best membership fitness is chosen for the classification result.
- **K-Means Clustering-based KNN (KM-KNN)**: This variant incorporates two popular algorithms: k-means and 1NN. It uses the k-means algorithm to cluster the training dataset in advance (Cherif, 2018). The centroids of each cluster are computed, and a novel training dataset including all cluster centroids is generated. Subsequently, the 1NN algorithm is executed on this processed training dataset, employing the single nearest neighbor for classification.
- **Weight Adjusted KNN (W-KNN)**: This variant emphasizes the utilization of attribute weighting (Han et al., 2001). The algorithm first assigns weights to each training data point using a kernel function, assigning greater weight to closer points and lesser weight to farther points. The frequency of all nearest neighbors is then used to predict the output class of a given testing

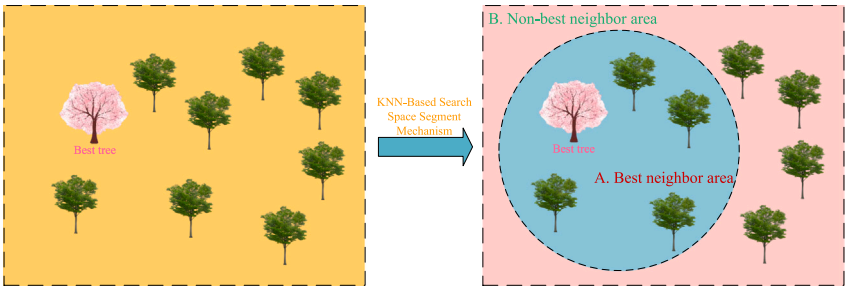


Fig. 1. Best and non-best neighbor areas are divided via search space division mechanism ($k = 5$).

data point. This variant considers the classification importance of different attributes when defining the kernel function for a multi-attribute dataset.

- **Hassanat Distance KNN (H-KNN)**: The Hassanat KNN algorithm emphasizes the distance measurement formula (Alkasassbeh et al., 2015). This variant introduces a novel approach to determining the distance between two data points. The Hassanat distance metric computes the nearest neighbors of a testing query and applies the majority voting rule, resembling the traditional KNN algorithm.
- **Generalized Mean Distance KNN (GMD-KNN)**: This variant is based on employing local vector constructions and repeated calculations of generalized mean distance (Gou et al., 2019). The algorithm stores sorted lists of k-nearest neighbors for each class, transformed them into local mean vectors, and performs repeated mean distance calculations to obtain the final distance value for each class relative to the testing query. The class with the shortest distance to the testing query is confirmed as the correct prediction.
- **Mutual KNN (M-KNN)**: The Mutual KNN algorithm focuses on mutual neighbors (Dhar et al., 2020). It transforms the training dataset by eliminating sets lacking mutual k-nearest neighbors with other sets, reducing anomalies and noise. The method identifies the k-nearest neighbors of the testing dataset's nearest neighbors by using the testing dataset to identify the k-nearest neighbors from the training dataset. The majority voting rule is then applied to classify the testing datasets.
- **Ensemble Approach KNN (EA-KNN)**: This approach solves the problem of having a fixed k parameter for classification (Hassanat et al., 2014). The algorithm finds the k-nearest neighbors of a testing data point by using a $Kmax$ value of $\sqrt{n}$, where n is the size of the training dataset. It sorts the list of nearest neighbors based on distance and conducts weight summation operations, involving iteratively adding an inverse logarithm function for k values ranging from 1 to $kmax$ in increments of 2. The class with the highest weight summation is identified as the predicted classification for the testing query.

The KNN algorithm classify data by measuring the distance between different eigenvalues. The working principle of the KNN algorithm can be summarized as follows.

For the input (training datasets): $D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}$, where $x_i \in X \subseteq R^n$ is an instance feature vector, $y_i \in Y = \{ c_1, c_2, \ldots, c_k \}$ is the class of the instance, i=1, 2, …,N.

For the output (the class y for the input x): firstly, according to a given distance metric, find x in the training set D such that x is among the k nearest neighbor points, covering the k points x as the field called $N_k(x)$. Secondly, determine the category (y) of the x: $y = argmax_{c_j} \sum_{x_i \in N_k(x)} I(y_i = c_i)$ according to the classification decision rules in $N_k(x)$ (e.g., majority voting). The result of KNN algorithm depends greatly on the choice of k (Abu et al., 2019; Li et al., 2024).

The KNN algorithm is described as follows:

Step 1: Calculate the distance between test data and each training data point.

Step 2: Sort by increasing distance.

Step 3: Select k points with the smallest distances.

Step 4: Determine the frequency of occurrence of the category in which the first k points belong.

Step 5: Return the most frequent category among the first k points as the predicted classification of test data.

Incorporating the KNN mechanism into the TSA algorithm necessitates developing novel balance mechanisms to regulate the equilibrium between global and local search phases. Consequently, Section 3 elaborates extensively on integrating the enhancements mentioned above into the TSA framework.

3. Method

In certain research literature concerning the TSA algorithm, numerous scholars have highlighted the potential occurrence of local optimal solutions or even the challenge of encountering local stagnation when applied to complex optimization problems (Gharehchopogh, 2022; Jiang et al., 2022, 2023). To address these concerns, integrating the KNN and a novel dynamic adjustment mechanism has been introduced to enhance the basic TSA algorithm.

As shown in Eqs. (2) and (3), each agent updates its position driven by the ST threshold. Although the current local optima B_j is concerned by Eq. (2), seeds are generated randomly around the tree. These seeds may take the solution to a worse search space and slow down the convergence (Jiang et al., 2022). A lower ST has been verified to provide a more powerful solution (Jiang et al., 2019).

This section uses the KNN mechanism to optimize the balance parameter ST. The tree population around the current best tree is divided into two spaces: best neighbor area and the non-best neighbor area. Trees in different areas will generate seeds with different ST. If a tree is always beyond the best neighbor area, we will migrate it to the best neighbor area of the best tree by Eq. (8). Additionally, we improve the seed generation approach with Eq. 3.4 to fit our new mechanism. A binary-switched seed generation mechanism is introduced to control tree migration and avoid dismissing promising solution. The seed will converge to the best position more effectively in high-dimensional complex problems. Table 2 lists all the parameters used in KATSA.

3.1. Search space division mechanism

The seed position is essential in the TSA (Kiran, 2015). We divide the tree search space into two areas based on the relative distance to the current best tree: the best neighbor area and the non-best area as shown in Fig. 1.

3.1.1. Best neighbor area

Neighbor trees will have more opportunities to search for the best tree. This strategy of search will lead to an effective local search and enhanced exploitation capability. Therefore, we select two neighbor

Table 2
Parameters used in KATSA.

Parameters	Symbol
k	The number of neighbors in the best neighbor area
$rand$	A random number in range of [0, 1]
N	Number of trees in a population
$scp(i)$	The search control parameter of i_{th} tree
$bcp(i)$	The binary control parameter of i_{th} grownup seed
$S_{i,j}$	j_{th} dimension of the i_{th} seed
B_j	the j_{th} dimension of best tree location
$T_{i,j}$	j_{th} dimension of the i_{th} tree
$T_{n,j}$	Randomly selected from the non-best neighbor area in j_{th} dimension
$T_{r,j}$	The neighbor of i_{th} tree in j_{th} dimension, r is the random index generate from the trees population
$T_{b1,j}$	Randomly selected from the best neighbor area in j_{th} dimension
$T_{b2,j}$	Randomly selected from the best neighbor area, but different with $T_{b1,j}$ in j_{th} dimension

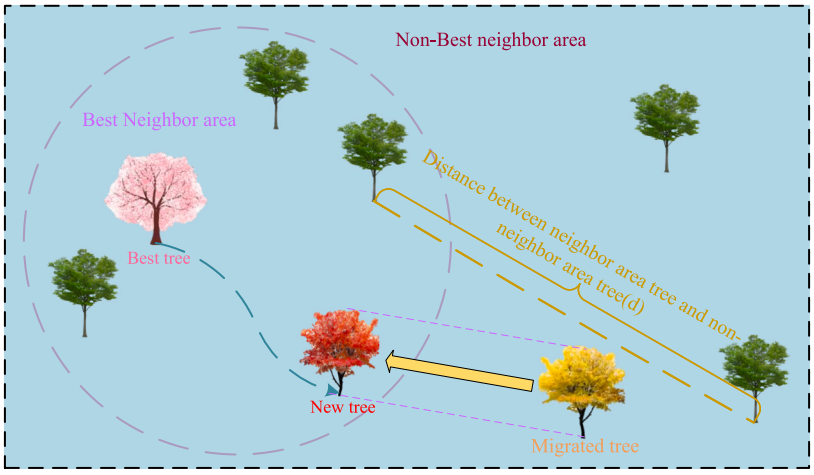


Fig. 2. Dynamic Serch Tendecy Regulation Mechanism — Tree Migration.

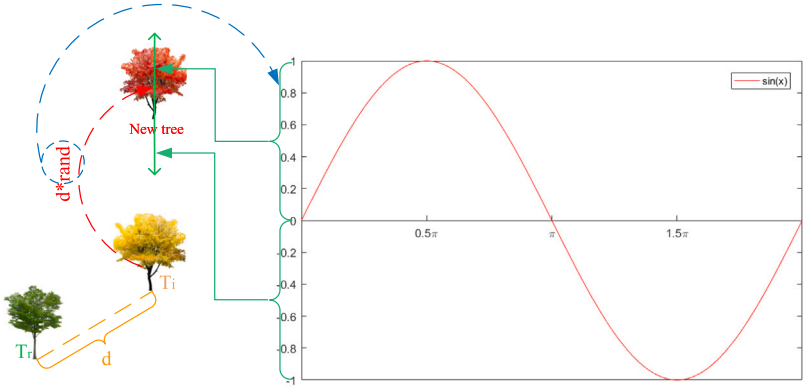


Fig. 3. Seed generation mechanism of the KATSA.

trees randomly to generate new seeds, and the indices of these trees can be generated using Eq. (5).

$$b_1, b_2 = fix(rand \times k) + 1 \tag{5}$$

3.1.2. Non-best neighbor area

The KNN mechanism divides the search space into two spaces. In the best neighbor area, seeds can be effectively generated around the best tree. In the non-best neighbor area, seeds should be generated in potentially valuable area to avoid local stagnation. The selection of non-neighbor tree's index as Eq. (6).

$$n = fix(rand \times (N - k - 1)) + 1 \tag{6}$$

3.2. Dynamic search tendency regulation mechanism

In different search sub-space, seeds should be generated using different balance strategies. A dynamic ST should be adopted for different areas. If a tree constantly searches far from the current best tree and cannot find a seed to replace it, the tree should be migrated to the best neighbor area. The search control parameter (scp) accomplishes this. The scp will be updated by Eq. (7).

$$scp(i) = scp(i) + 1 \tag{7}$$

If scp reaches a value of $threshold$, it means the tree generates N seeds, but it cannot be evolved in the non-best neighbor area. The tree

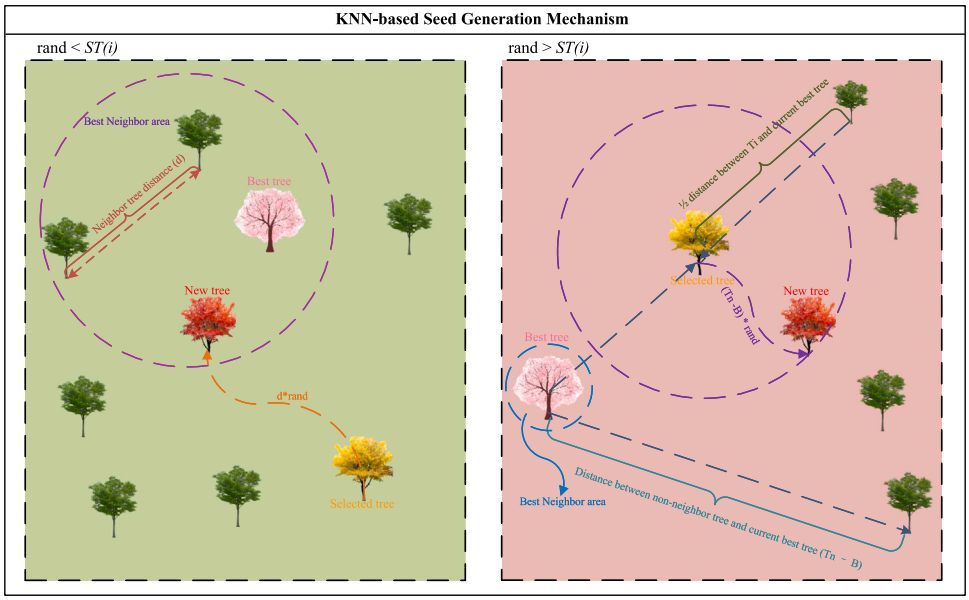


Fig. 4. The process of KNN-based seed generation mechanism.

will be migrated towards the best neighbor area through Eq. (8), and the *threshold* is calculated using Eq. (9). The tree migration process is shown in Fig. 2.

$$T_{i,j} = B_j + (T_{n,j} - T_{r,j}) \times cos(rand \times 2 \times \pi) \tag{8}$$

$$threshold = N/ns \tag{9}$$

After a tree has been migrated to the best neighbor area, its ST will be changed to 0.8 for enhancing local search, and its scp will be reinitialized. Tree migration can reduce local stagnation. After a tree is migrated to the best neighbor area, its seeds can cover the area near the best tree more completely.

3.3. Binary-switched seed generation mechanism

Trees will be migrated when the scp reaches a threshold value. Based on the previous mechanism, many unnecessary tree migrations may occur, resulting in many wasted evaluations. Therefore, if a seed replaces its tree, the new tree should not trigger the tree migration process. We give a flag named binary control parameter (bcp) and set its ST value to a high level, leading the new seed to search the vicinity. The binary control parameter is assigned to **False**.

When a seed replace its tree, the new tree will update its $bcp(i)$ value to **True** and reinitialize after producing the next generation of seeds. The new tree is expected to search its neighbors more efficiently, so the seed generation mechanism differs from the KNN-based seed generation mechanism in Eq. , and its effective search is shown in Fig. 3.

$$\begin{cases} S_{i,j} = T_{i,j} + (T_{i,j} - T_{r,j}) \times sin(rand \times 2 \times \pi) \times 0.05 & , \text{if } ST < rand, \\ S_{i,j} = \dfrac{T_{i,j} + B_j}{2} + (T_{n,j} - B_j) \times \cos(rand \times \pi \times 2) & , \text{if } ST \geq rand. \end{cases} \tag{10}$$

3.4. New algorithm: KNN ameliorated tree seed algorithm (KATSA)

The proposed KATSA algorithm uses the KNN method to divide the best and non-best neighbor areas based on the current best tree. An excessive value of k will affect the decision on the best neighbor area and its convergence speed.

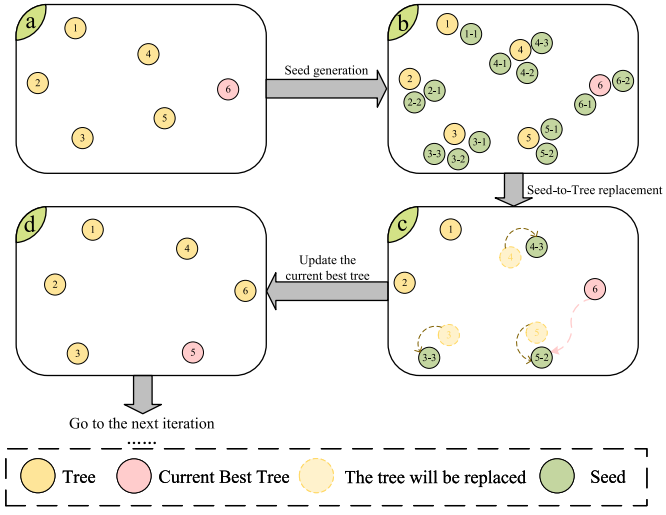


Fig. 5. Schematic diagram of the evolution process of TSA.

The seed is generated based on two trees in the best neighbor area when its random number is lower than ST. When the random number is larger than ST, the trees in the non-best neighbor area should also be considered. The equation is shown in Eq. 3.4. The mechanism is present vividly in Fig. 4.

$$\begin{cases} S_{i,j} = T_{i,j} + (T_{b_1,j} - T_{b_2,j}) \times \sin(rand \times \pi \times 2) & , \text{if } ST < rand, \\ S_{i,j} = \dfrac{T_{i,j} + B_j}{2} + (T_{n,j} - B_j) \times \cos(rand \times \pi \times 2) & , \text{if } ST \geq rand. \end{cases} \tag{11}$$

The workflow diagrams for TSA and KATSA are depicted in Figs. 5 and 6, respectively.

In Fig. 5-a, trees can be generated in the search space (e.g., 6 trees) and the function fitness of the trees is evaluated (Kiran, 2015).

In Fig. 5-b, the number of seeds for each tree is dynamic. For clearer observation, we chose some of its seeds to show.

In Figs. 5-(c, d) each seed is compared with its parent tree, and the best seed is retained for further evolution.

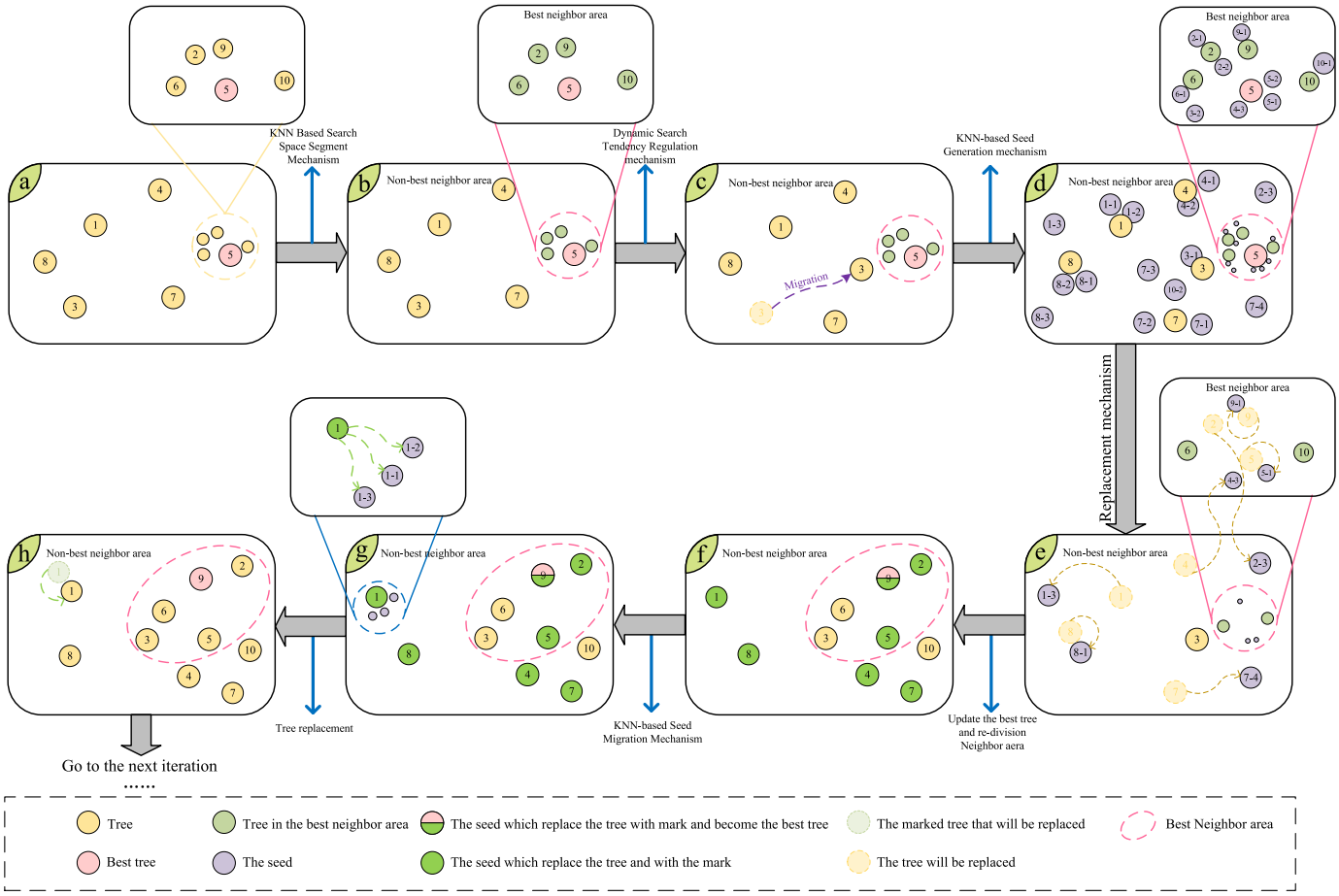


Fig. 6. Schematic diagram of the evolution process of KATSA.

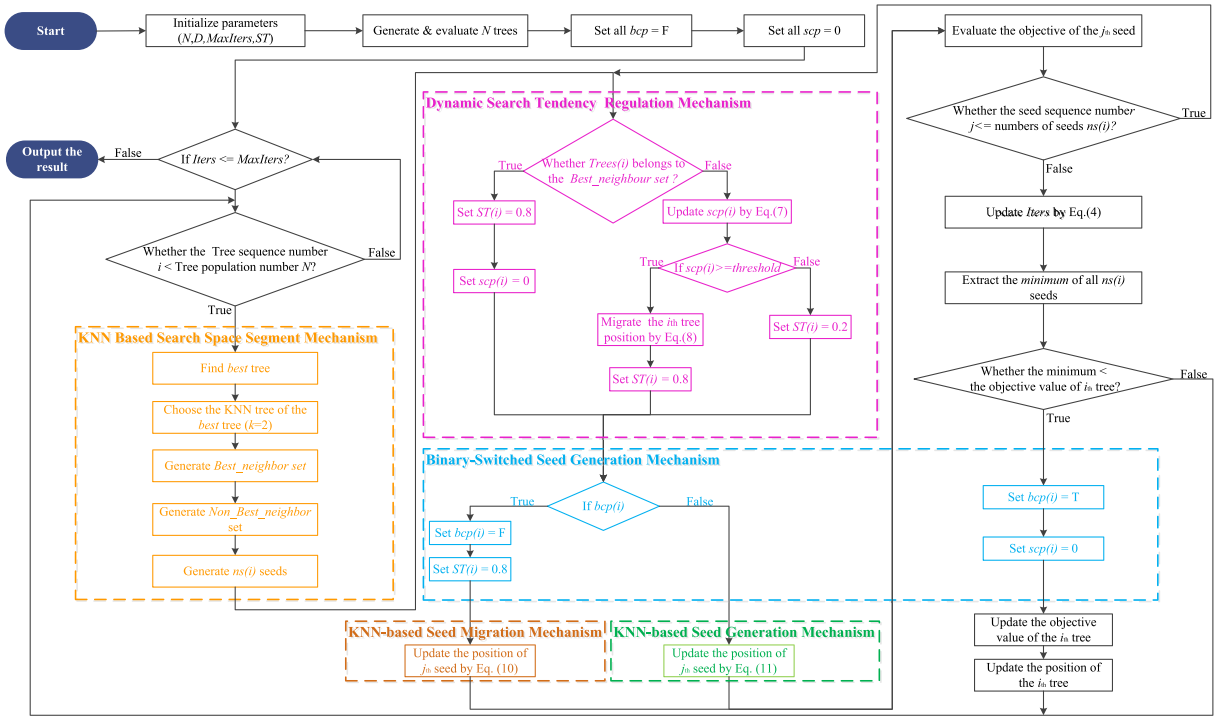


Fig. 7. The flow chart for the KATSA.

In Fig. 6-a, trees are planted in the search space, and the search space is segregated by the KNN mechanism into the best neighbor area and non-best neighbor area in Fig. 6-b.

If a tree cannot be replaced by its seeds for a long time, it will be migrated based on the dynamic search tendency regulation mechanism as shown in Fig. 6-c.

In Fig. 6-d, after tree migration, the seed will be generated via the KNN-based seed generation mechanism.

In Fig. 6-e, the generated seed will be as in TSA. In the KATSA, new trees will be marked. These markers are vividly represented in Fig. 6-f, which demonstrates that the new best neighbor area is set after tree replacement.

The marked tree will generate seeds by the KNN-based tree migration mechanism, and seeds will spawn in the immediate vicinity of the tree. The inset of Fig. 6-g shows the mechanism.

After all the mechanisms have been carried out, the iteration will go to the next loop, as shown in Fig. 6-h.

The primary steps are expressed in Algorithm 1, and the flowchart of KATSA is shown in Fig. 7.

3.5. Discussion on KATSA computational complexity

In this study, the complexity of the proposed KATSA is primarily influenced by several factors: the population size (N), the number of seeds generated per tree (ns), the problem's dimensionality (D) and the maximum number of iterations ($MaxIters$). These factors are the same as the basic TSA.

Regarding tree generation and initialization, N trees are created, each having D dimensions. Therefore, the time complexity for initialization is $O(N \times D)$.

For the entire population of trees' updates, the entire population needs to be updated for $MaxIters$ iterations. Thus, the time complexity incurred during this process is $O(MaxIters \times N)$.

Moreover, each tree requires generating seeds, with ns being the number of seeds generated per tree. Each seed corresponds to a solution for the problem, with each solution having D dimensions. Hence, for each tree, generating a generation of seeds has a time complexity of $O(ns \times D)$. As each tree generates a batch of seeds, the time complexity for seed generation across all trees is $O(N \times ns \times D)$.

Summing up the factors above, the overall complexity of the proposed algorithm can be expressed as $O(N \times D + MaxIters \times N \times ns \times D)$. This is the same as the basic TSA.

4. Experiments and discussion

In this section, we design three experiments to evaluate the performance of KATSA. Section 4.1 introduces the parameter settings in the following experiments, in Section 4.2, we try to obtain the optimal value of k by sensitivity analysis experiments, Section 4.3 introduces the qualitative analysis of KATSA, Section 4.4 introduces the quantitative analysis, Section 4.5 analyzes the experiments data from Section 4.4, Section 4.6 gives the statistical experiments to prove that KATSA performs significantly than other algorithms. To further prove the application of KATSA, we use KATSA to solve three real-world problems in Section 4.7.

4.1. Experiment setting

To further evaluate its performance, KATSA was compared with other state-of-the-art algorithms and several variants as shown in Table 3.

To ensure the fairness of the experiment, this paper reviews the data sets and parameters used by some comparison algorithms in the original paper. This information is shown in Tables 4 and 5. According to the statistical results, we used the IEEE CEC 2014 benchmark function as the experiment data set for the experiment. More information about the CEC 2014 benchmark function can be found in Tables 13 and 14.

Algorithm 1 The pseudo code of the KATSA.

Step 1. Initialize of the algorithm
1.1 Set up the number of tree population (N);
1.2 Set the dimensionality of the problem (D);
1.3 Decide the termination condition ($MaxIters$);
1.4 Generate N random tree locations on the D-dimensional search space;
1.5 Evaluate the tree location using objective function specified for the problem;
1.6 Select the best solution;
1.7 Set the number of neighbor for best tree (k);
1.8 For all trees set the search control parameter (scp) as 0;
1.9 For all trees set the binary control parameter (bcp) as False.
1.10 Generate threshold by Eq. (9)
Step 2. Search with seeds
For all trees
2.1 Decide the number of seeds produced for this tree;
2.2 Calculate tree's Euclidean distance to best tree;
2.3 If tree belongs to the best neighbor area of best tree then;
Set this tree $ST = 0.8$;
Set this tree $scp = 0$;
Else
Update this tree scp by Eq. (7)
If this tree $scp \geq threshold$ then
Migrate this tree by Eq. (8);
Set this tree $ST = 0.8$;
Else
Set this tree $ST = 0.2$;
End if
End if
2.4 If this tree bcp is **True** then
Set this tree $bcp =$ **False**;
Set this tree $ST = 0.8$;
For all seeds
For all dimensions
Update this dimension using
End for
End for
Else
For all seeds
For all dimensions
Update this dimension using Section 3.4
End for
End for
End if
2.5 Select the best seed;
2.6 Seed substitutes for this tree, if the seed objective value is better than its tree.
End for
Step 3. Select of Best Solution
3.1 Select the best solution of population ;
3.2 New best solution substitutes for the previous best solution, set $scp = 0$ and set $bcp =$ **True**, if new best solution is better than the previous best solution;
Step 4. Experiment Termination condition
Go to Step 2, if termination condition is not met.
Step 5. Report
Report the best solution.

Table 3
Comparative algorithm used in the experiments.

Categories	Algorithms	Authors and years
TSA with its variants	TSA	Kiran (2015)
	EST_TSA	Jiang et al. (2019)
	STSA	Jiang, Xu et al. (2020)
	fb_TSA	Jiang, Meng et al. (2020)
	MTSA	Jiang et al. (2022)
Classic algorithms	GA	Holland and Reitman (1977)
	PSO	Kennedy and Eberhart (1995)
	BA	Yang and Hossein (2012)
	BOA	Arora and Singh (2019)
Recent popular algorithms	GWO	Mirjalili et al. (2014)
	RSA	Abualigah et al. (2022)
DE variant	LSHADE	Tanabe and Fukunaga (2014)

4.2. Sensitivity analysis

To obtain an optimal value of parameter k, we have conducted experiments by testing the KATSA algorithm on the IEEE CEC 2014 benchmark function sets with different dimensions of 30, 50 and 100. Furthermore, we designed experiments using seven different values of k to observe their impact on convergence performance.

The summary results are presented in Table 6. The number in Table 6 indicates how many first ranks are obtained in each dimension when $k = n$. Each experimental test involved 30 rounds with 500 iterations. More details can be found in Tables 15–17. The data indicate that for dimensions 30 and 100, using $k = 2$ yields better results. However, in the case of dimension 50, $k = 4$ produces superior results compared to $k = 2$. Based on the experimental results and for simplicity, we chose $k = 2$ for subsequent experiments.

4.3. Qualitative analysis

To make a qualitative analysis of the proposed algorithm, we conduct experiments to observe its performance. The experiments include:

- In Fig. 8(a), we record the tree's position in the 500 iteration experiments. The whole range is the search space. The iteration's change will vary the point color, and the color will change from green to blue. The last point is red, indicating global optimal solution. It shows a wide distribution of the trees in the search space, and the color shows the fast convergence to the global optima.
- We track the tree's position in the first dimension. From Fig. 8(b), we notice that in the former phase of the search history, Eq. 3.4 varies in a vast range, to the latter phase of the search history, Eq. applies, causing trees to change within a narrow range.
- Fig. 8(c) and (d) show the convergence of KATSA in 500 iterations. Fig. 8(c) records the average fitness history in the whole population, while Fig. 8(d) shows the global optima's convergence. The convergence curve shows that KATSA can converge to the global optima at high speed and efficiently.

By analyzing Fig. 8, we can conclude that KATSA can quickly converge to the global optima. The experimental data in Section 4.4 support this view.

4.4. Quantitative analysis

In this section, we design three experiments to prove the superiority of KATSA. The experiments' results support our views. In Section 4.4.1, KATSA is compared with other variants of TSA, and the results show that KATSA is better than the variants of TSA. In Section 4.4.2, we added other metaheuristic algorithms to implement the comparative experiments, and the results show that KATSA has a strong ability to solve complex problems. In Section 4.4.3, we draw the boxplot based on experiment results. These boxplots show the stability of KATSA.

4.4.1. Comparison of KATSA and variants of TSA

The comparison results for KATSA, EST-TSA, fb_TSA, TSA, STSA and MTSA in dimensions 30, 50 and 100 are presented in Tables 18–20. Each value in these tables represents the average of the best values obtained from 30 rounds of experiments, with 500 iterations per round. These results reflect the convergence performance of these algorithms, a crucial metric for evaluating their superiority.

Additionally, visual representations of the convergence curves for these algorithms are shown in Figs. 9–11, we documented the convergence process generated over 500 iterations for each round of the experiment. Specifically, for a given iteration within one of the 30 rounds, the algorithm produced 30 local optima. Taking the average of these 30 local optima at each iteration forms a point on the line in the graph. The slopes of these lines at specific points represent its convergence speed at those moments. These visualizations aid in analyzing both the convergence speed and solution quality of these algorithms.

These experimental results underscore that leveraging the KNN mechanism to partition the search space around the best tree enhances its convergence speed significantly. Furthermore, the improvements in the seed position update formula (Eq. 3.4) enable the KATSA to achieve better results compared to previous variants of TSA. Based on the experimental findings in this section, we assert that the proposed KATSA outperforms other TSA algorithms and their variants.

4.4.2. Comparison of KATSA and other algorithms

In order to further demonstrate the effectiveness of the proposed algorithm, additional experiments were conducted. We selected classical heuristic optimization algorithms (such as GA and PSO) and novel algorithms (like RSA and LSHADE) for comparative experiments with the innovative algorithm proposed in this paper. Additionally, the parameters for each algorithm were set as outlined in Table 5, adhering to the settings provided in their original papers.

Tables 21–23 document the average best values achieved by KATSA and other comparative algorithms. The tables also present the average rankings of each algorithm throughout the entire comparative experiment. The results in these tables indicate that KATSA performs well in solving multimodal problems. This suggests that the dynamic adjustment mechanism and the tree migration mechanism of KATSA contribute to its ability to avoid local optima.

Figs. 12–14 depict the convergence processes of KATSA compared to other comparative algorithms across different dimensions. In Figs. 12(a), 12(c), 13(d), 13(e), 14(b) and 14(f), it is observable that the convergence speed of KATSA is superior to that of other comparative algorithms, particularly in the initial iterations where KATSA's convergence curve slope is steeper than that of other algorithms. The KNN mechanism contributes to KATSA's convergence speed, while the binary seed generation mechanism prevents KATSA from getting trapped in local optima, particularly in multimodal problems. The ranking results of all the comparative algorithms are shown in Table 7.

Table 8 summarizes the experimental results in this section. The table displays the number of times KATSA wins, ties or loses in comparisons with different algorithms on each objective function. The results indicate that in low-dimensional functions, KATSA performs comparably to other comparative algorithms. However, in high-dimensional experiments, KATSA achieves significantly superior results than other algorithms. Therefore, we conclude that the proposed KATSA algorithm excels at solving high-dimensional problems compared to the other algorithms.

4.4.3. Stability analysis of the proposed KATSA

In this section, we present boxplots of experimental results that assist in analyzing the stability of the KATSA algorithm. In these boxplots (Figs. 15–17), the x-axis represents the proposed KATSA along with some comparative algorithms.

Table 4
The data sets adopted by some comparative algorithms.

Algorithms	Criteria
EST-TSA	19 benchmark functions & 1 real-world problem
fb_TSA	IEEE CEC 2014 benchmark functions & 4 real-world problems
TSA	24 benchmark functions
STSA	24 benchmark functions & 2 real-world problems
MTSA	IEEE CEC 2014 benchmark functions & 3 real-world problems
GWO	23 benchmark functions, 6 of which are from IEEE CEC 2005 benchmark functions
BA	8 real-world problems
BOA	IEEE CEC 2013 benchmark functions & 3 real-world problems
RSA	23 benchmark functions (13 from IEEE CEC 2017 and 10 from IEEE CEC 2019) & 7 real-world problems
LSHADE	IEEE CEC 2014 benchmark functions

Table 5
The initial parameters of comparative algorithms.

Algorithms	Parameters	Values
KATSA	neighbor tree number (k)	2
EST-TSA	ST	0.1
fb_TSA	null	null
TSA	ST	0.1
STSA	ST	0.1
MTSA	ST	0.1
GA	Type	Real coded
	Selection	Roulette wheel
	Crossover	Probability = 0.7
	Mutation	Probability = 0.2
PSO	Acceleration constants	0.2
GWO	$\vec{a}$	linearly decreased from 2 to 0
BA	Loudness(A)	0.5
	Pulse rate (a)	0.5
	Frequency minimum	0
	Frequency maximum	2
BOA	p	0.8
	power exponent	0.1
	sensory	modality = 0.01
RSA	Evolutionary Sence	$2 \times randn \times (1 - (iter/maxiter))$
	sensitive parameter controlling the exploration accuracy	0.005
	sensitive parameter controlling the exploitation accuracy	0.1
LSHADE	$r^{N^{init}}$	30
	r^{arc}	1.4
	p	0.11
	Historical memory size (H)	5

By comparing these boxplots, we observe that the KATSA algorithm demonstrates relatively consistent performance across various experiments, with high stability of the box and fewer outliers. This signifies that the KATSA algorithm consistently produces favorable outcomes in diverse experiment environments, validating its attribute of high stability.

Table 6
The summary results of the different k values in different dimensions.

Dimensions	k = 2	k = 3	k = 4	k = 5	k = 6	k = 7	k = 8
D = 30	9	3	3	6	3	6	6
D = 50	4	6	7	5	5	5	4
D = 100	15	6	2	3	2	1	1
Number of first ranks	28	15	12	14	10	12	11

4.5. Further analysis

To explain the performance of KATSA further, we need to analyze the experimental phenomena by combining the results of qualitative and quantitative experiments.

First, the KATSA algorithm is highly competitive compared to other TSA variants. This can be observed from the data in Tables 18–20. The KNN mechanism sorts out the best tree and other tree relationships and enables the proposed algorithm to converge quickly to the global optima.

Second, KATSA is better than other algorithms in solving high-dimensional complex problems, as evidenced by the rank first and average rank in Tables 21–23. However, solving low-dimensional problems is not significantly better than other algorithms. The reason is that when the algorithm faces low-dimensional problems, the KNN mechanism cannot divide best neighbor areas, reducing convergence speed. At the same time, this analysis also highlights that KATSA has room for improvement.

Third, the experimental results are shown in the boxplot Figs. 16(b), 17(c) and 17(f) demonstrate that the algorithm proposed in this article is more stable and of better quality than the comparative algorithms in this experiment. The experiment results of KATSA's

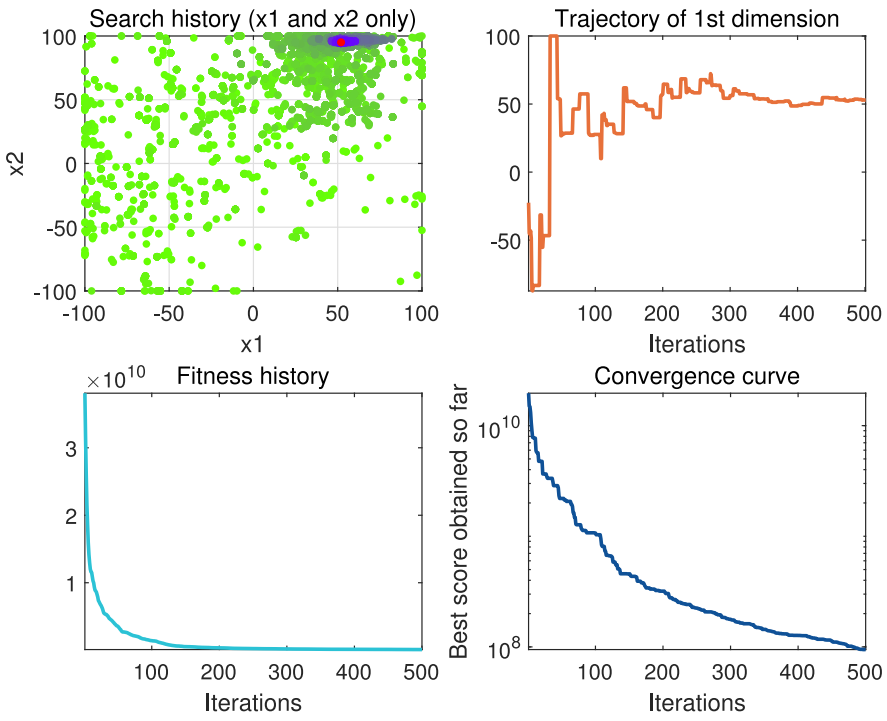
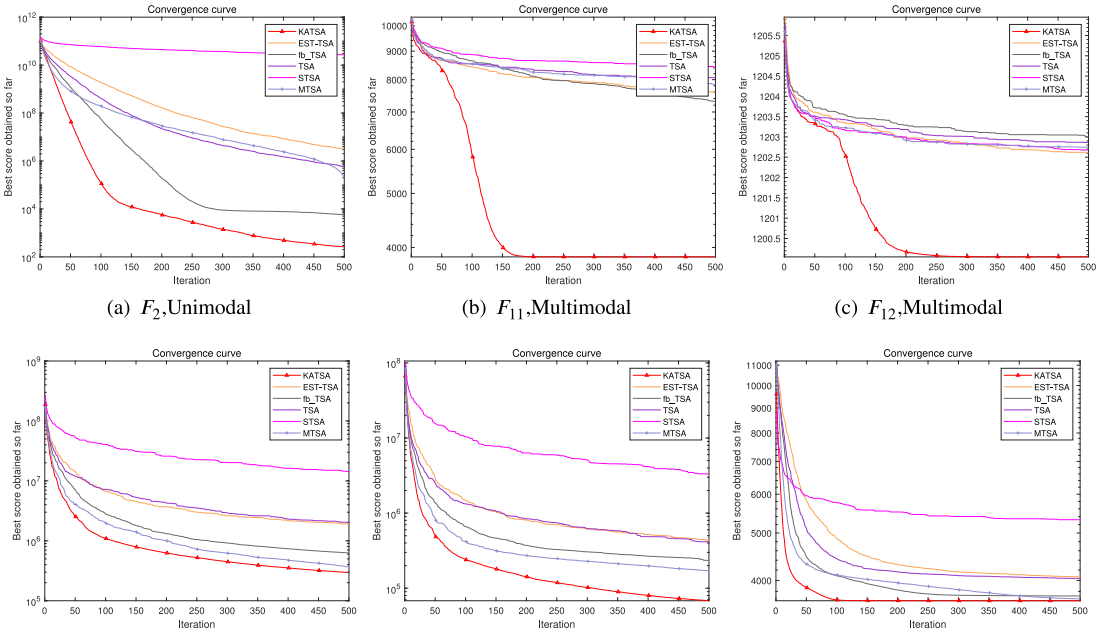


Fig. 8. The qualitative analysis of KATSA in Function 1 and dimension 100 (search history(a), trajectory history(b), average fitness history(c), convergence curve(d)).



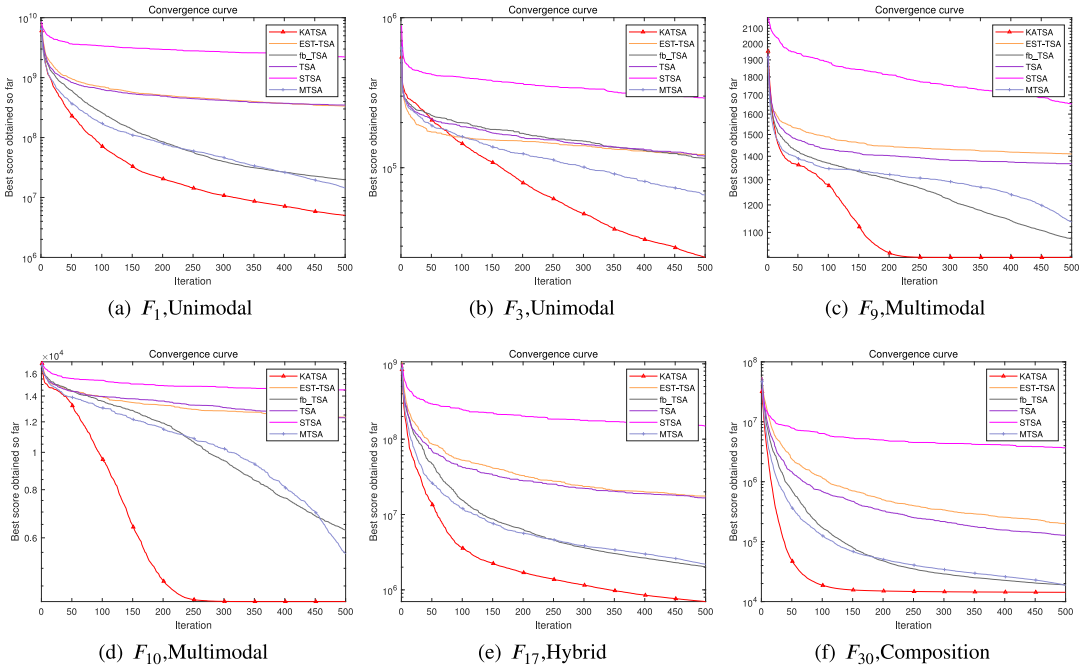
(a) F_2,Unimodal (b) F_{11},Multimodal (c) F_{12},Multimodal

(d) F_{17},Hybrid (e) F_{21},Hybrid (f) F_{28},Composition

Fig. 9. Convergence curve of the KATSA, TSA and its variants in 30 dimensions.

Table 7
The ranking results of all algorithms in the experiment.

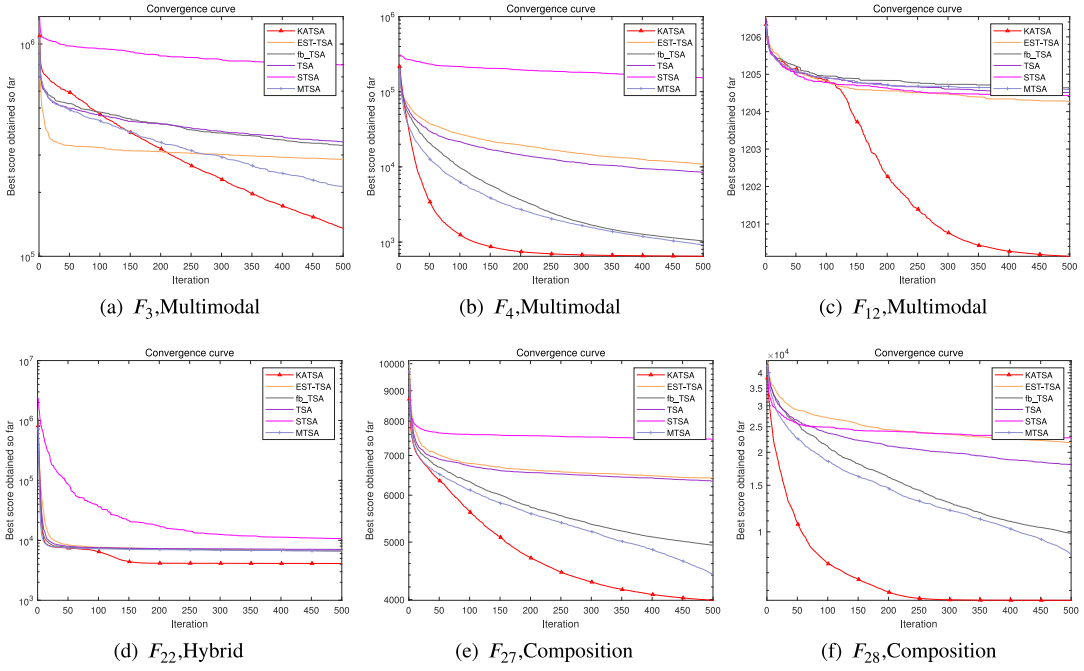
Dimensions	Rank type	KATSA	EST_TSA	fb_TSA	TSA	STSA	MTSA	GA	PSO	GWO	BA	BOA	RSA	LSHADE
D = 30	Rank first	**14**	1	5	0	0	1	0	0	0	0	5	3	5
	Average rank	**2.47**	5.9	3.73	5.97	9.07	3.53	9.57	6.7	7.37	12.93	9.8	10.2	3.77
D = 50	Rank first	**16**	1	0	0	1	1	0	0	0	0	5	3	7
	Average rank	**2.03**	6.37	4.1	6.53	9.83	3.63	9.03	6.47	6.9	12.97	9.63	9.8	3.7
D = 100	Rank first	**21**	2	1	0	1	0	0	0	0	0	5	4	2
	Average rank	**2.03**	6.27	4.93	7.47	10.47	3.9	8.67	6.37	5.9	13	9.3	8.63	4.07



(a) F_1,Unimodal (b) F_3,Unimodal (c) F_9,Multimodal

(d) F_{10},Multimodal (e) F_{17},Hybrid (f) F_{30},Composition

Fig. 10. Convergence curve of the KATSA, TSA and its variants in 50 dimensions.



(a) F_3,Multimodal (b) F_4,Multimodal (c) F_{12},Multimodal

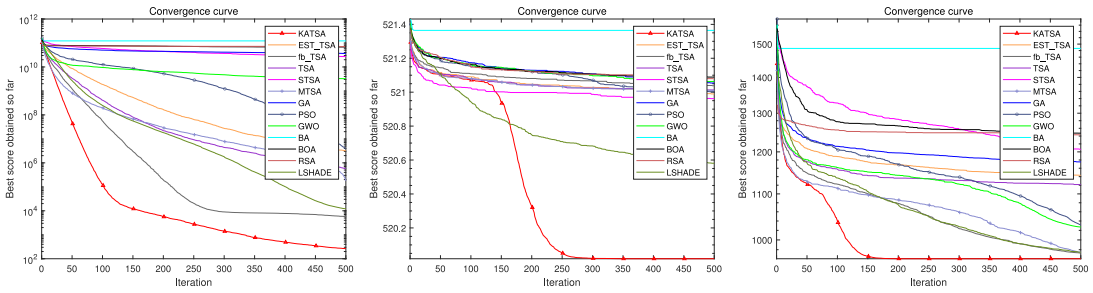
(d) F_{22},Hybrid (e) F_{27},Composition (f) F_{28},Composition

Fig. 11. Convergence curve of the KATSA, TSA and its variants in 100 dimensions.

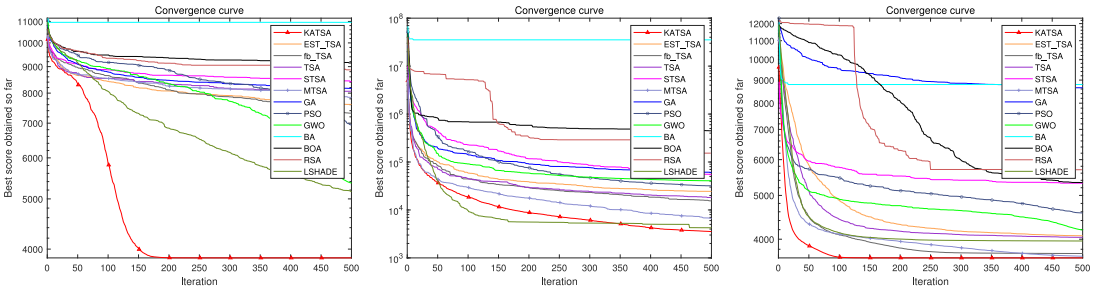
Table 8
The summary of the experimental results (KATSA compared with other algorithms).

Algorithms	EST-TSA	fb_TSA	TSA	STSA	MTSA	GA	PSO	GWO	BA	BOA	RSA	LSHADE
D = 30	26/0/4	18/0/12	28/0/2	29/0/1	20/0/10	29/0/1	30/0/0	29/0/1	30/0/0	25/0/5	27/0/3	25/0/5
D = 50	26/0/4	27/0/3	29/0/1	29/0/1	27/0/3	28/0/2	30/0/0	29/0/1	30/0/0	25/0/5	27/0/3	22/0/8
D = 100	25/0/5	29/0/1	29/0/1	29/0/1	27/0/3	27/0/3	30/0/0	27/0/3	30/0/0	24/0/6	25/0/5	27/0/3

boxes for some functions are smaller and lower than those of other comparative algorithms, highlighting its superior performance. Upon analysis, this superiority can be attributed to integrating the dynamic search tendency regulation mechanism and the binary-switched seed generation mechanism proposed in this article. These mechanisms enable the proposed algorithm to adaptively adjust the optimization

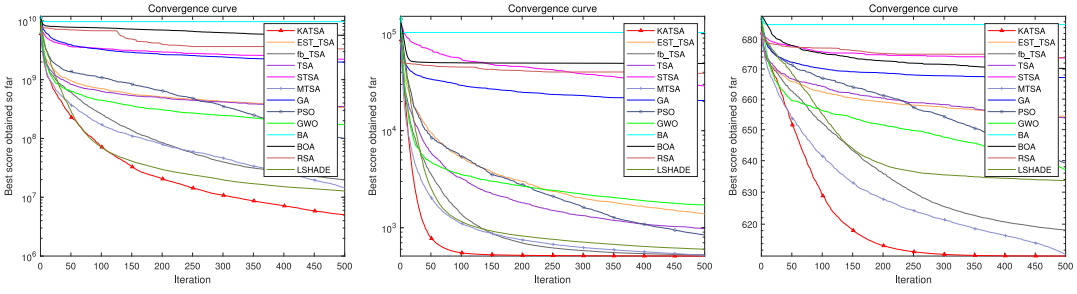


(a) F_2,Unimodal (b) F_5,Multimodal (c) F_9,Multimodal

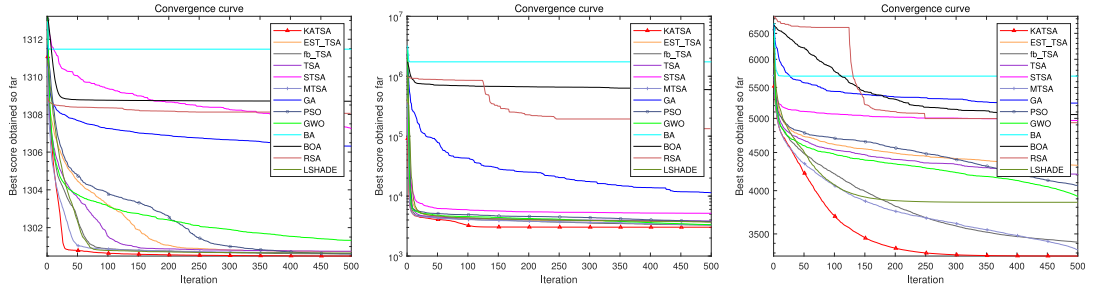


(d) F_{11},Multimodal (e) F_{20},Hybrid (f) F_{28},Composition

Fig. 12. The convergence curve for the KATSA and other algorithms in 30D.



(a) F_1,Unimodal (b) F_4,Multimodal (c) F_6,Multimodal



(d) F_{13},Multimodal (e) F_{22},Hybrid (f) F_{28},Composition

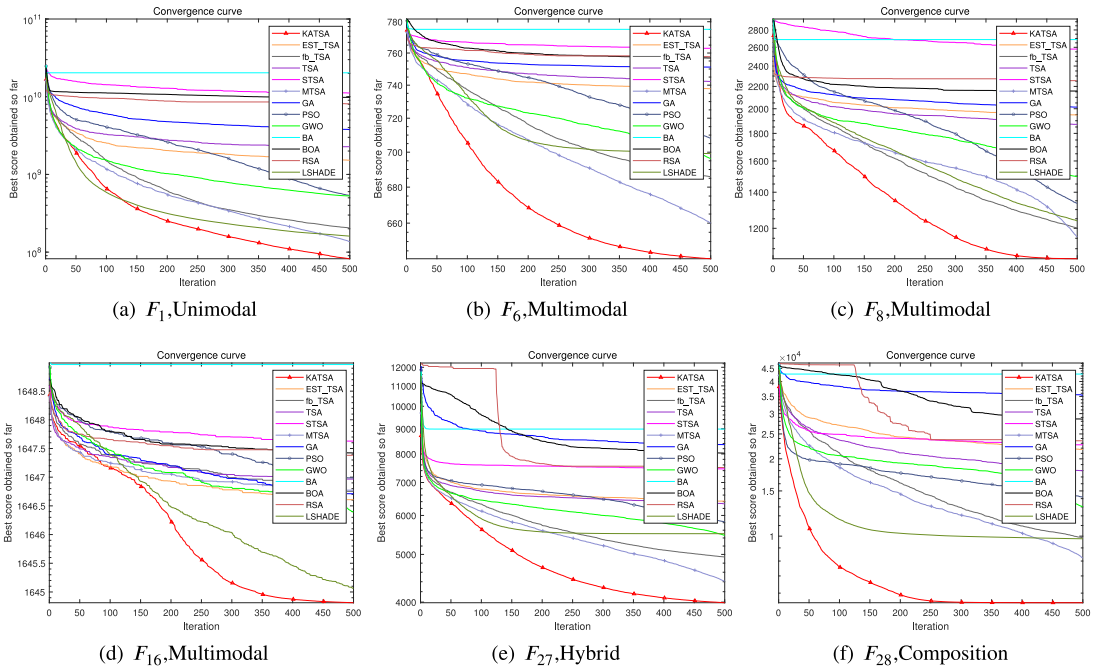
Fig. 13. The convergence curve for the KATSA and other algorithms in 50D.

process when facing different benchmark function experiments, leading to smaller standard deviations in multiple rounds of experiments and more reliable results.

4.6. Statistical experiments

Table 9 displays the results of Wilcoxon's Signed-Rank Test conducted on the experimental data of the KATSA algorithm compared to 12 other comparative algorithms. R^+ is the sum of ranks for the problems in which the KATSA outperformed the others, and R^- is the sum of ranks for the inverse (Derrac et al., 2011).

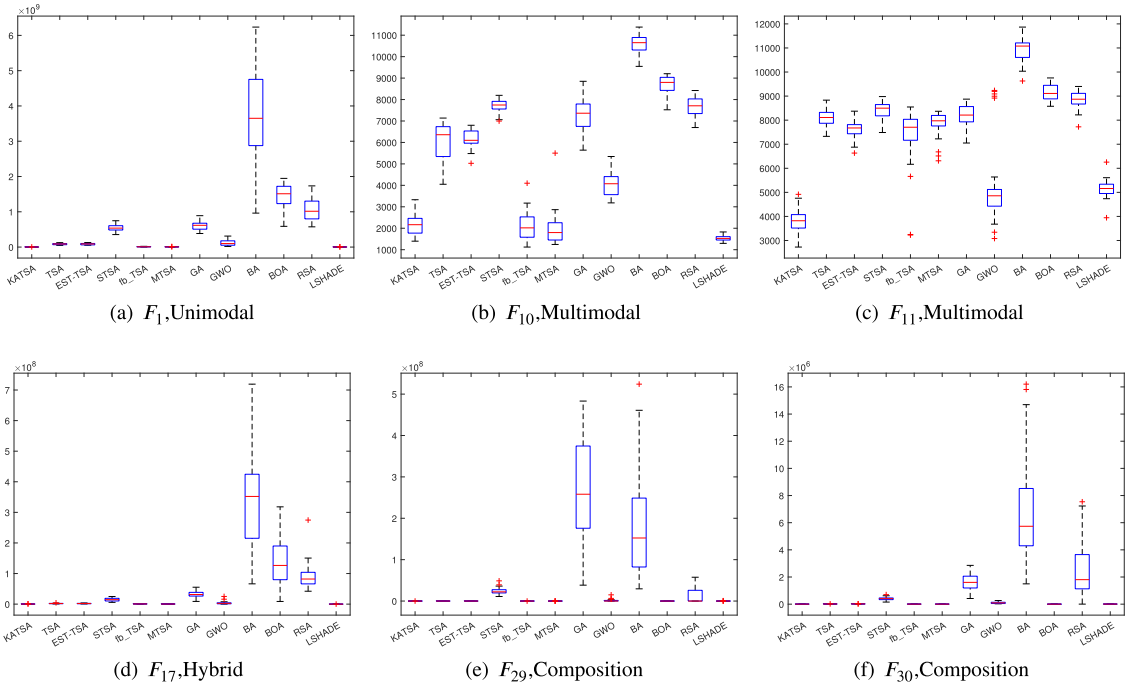
We posit that KATSA surpasses the performance of the other comparative algorithms. If the p-value resulting from the experiment between KATSA and a comparative algorithm is smaller than the significance level α, it signifies that KATSA's performance is superior to the compared algorithm. Table 9 shows that KATSA outperforms other algorithms in most dimensions. As the table states, KATSA shows a significant improvement over all the comparison algorithms with a level of significance $\alpha = 0.1$ in 50 dimensions and all algorithms with $\alpha = 0.01$ in 100 dimensions. This indicates that KATSA significantly outperforms the other algorithms in complex optimization problems.



(a) F_1,Unimodal (b) F_6,Multimodal (c) F_8,Multimodal

(d) F_{16},Multimodal (e) F_{27},Hybrid (f) F_{28},Composition

Fig. 14. The convergence curve for the KATSA and other algorithms in 100D.



(a) F_1,Unimodal (b) F_{10},Multimodal (c) F_{11},Multimodal

(d) F_{17},Hybrid (e) F_{29},Composition (f) F_{30},Composition

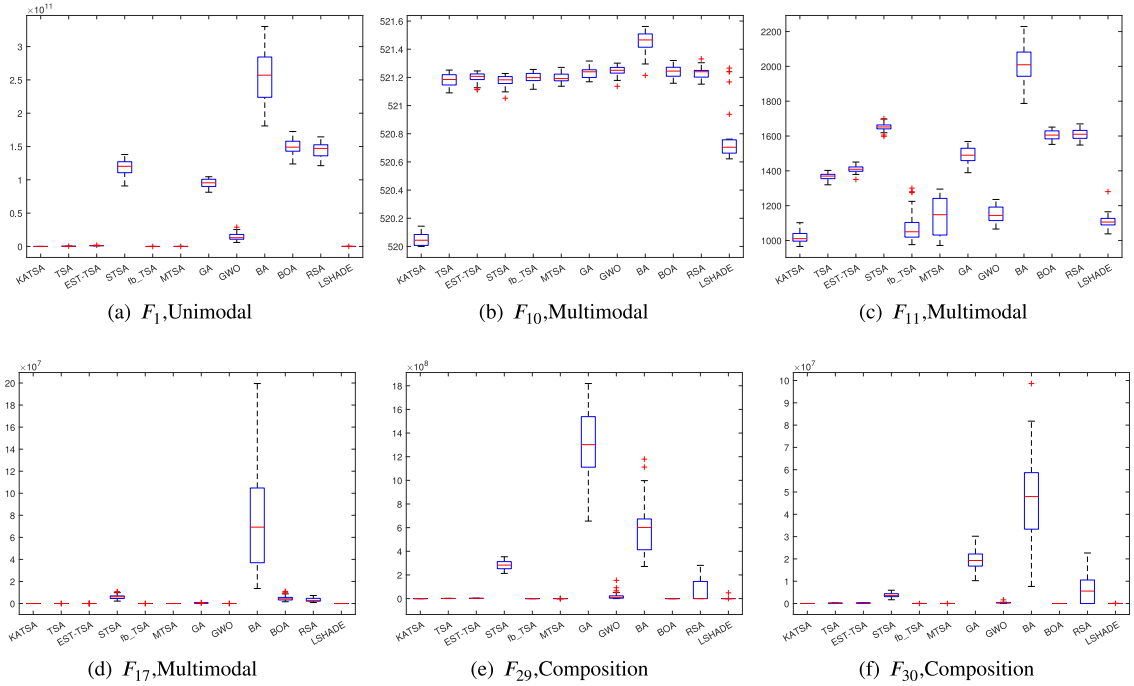
Fig. 15. Boxplot of all comparative algorithms in 30D.

4.7. Practical engineering problems of mathematical modeling

According to the further analysis results in Section 4.5, KATSA performs better in solving high-dimensional complex problems. Therefore, this section will adopt some problems with complex objective functions and many constraints to verify KATSA's ability to perform on complex constrained problems: Rolling Element Bearing (Rao et al., 2011), Topology Optimization (Sigmund, 2001) and Wind Farm Layout Problem (Wang et al., 2017). These problems are widely discussed in some literature, and these problems' definitions are from a test-suite (Kumar et al., 2020; Kunakote et al., 2022; Savsani et al., 2017; Smith & Randall, 2015). In the results of the Tables 10–12, *best* represents the best value obtained by the algorithm in 30 rounds of experiment, *mean* represents the average value results, and *std* represents the standard deviation of results, *worst* represents the worst result. The remaining x value represents the best solution obtained by the algorithm when obtaining the best value.

4.7.1. Example 1: Rolling element bearing

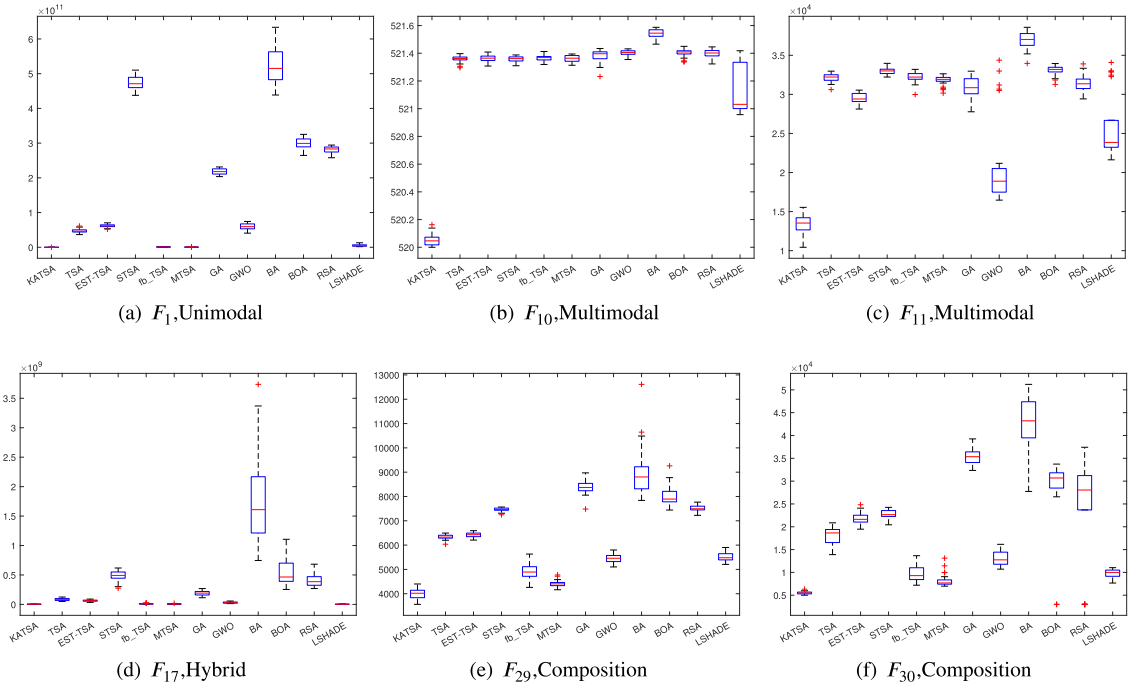
The objective of this task is to enhance the load-carrying capacity of a rolling element bearing through the optimization of five design variables and five design parameters. These design variables encompass



(a) F_1,Unimodal (b) F_{10},Multimodal (c) F_{11},Multimodal

(d) F_{17},Multimodal (e) F_{29},Composition (f) F_{30},Composition

Fig. 16. Boxplot of all comparative algorithms in 50D.



(a) F_1,Unimodal (b) F_{10},Multimodal (c) F_{11},Multimodal

(d) F_{17},Hybrid (e) F_{29},Composition (f) F_{30},Composition

Fig. 17. Boxplot of all comparative algorithms in 100D.

the pitch diameter (D_m), ball diameter (D_b), as well as the coefficients for outer and inner raceway curvatures (f_o and f_i), in addition to the total number of balls (Z). Correspondingly, the design parameters involve e, ϵ, ζ, $K_{D\max}$ and $K_{D\min}$, all of which are exclusively present within the constraints. These elements collectively constitute a set of ten variables — five design variables and five design parameters. The problem comprises nine intricate nonlinear constraints that are rooted in considerations of manufacturing and kinematic factors, as detailed by Rao et al. (2011).

Maximize :

$$f(\bar{x}) = \begin{cases} f_c Z^{2/3} D_b^{1.8} & , \text{ if } D_b \le 25.4 \text{ mm} \\ 3.647 f_c Z^{2/3} D_b^{1.4} & , \text{ otherwise} \end{cases} \tag{12}$$

Table 9
Results of the Wilcoxon's test for KATSA and other algorithms.

Dimensions	Comparison	R^+	R^-	p-value	Comparison	R^+	R^-	p-value
D = 30	KATSA versus EST-TSA	413	52	1.74228E−04	KATSA versus PSO	465	0	1.73440E−06
	KATSA versus fb_TSA	307	158	1.98610E−01	KATSA versus GWO	446	19	3.88218E−06
	KATSA versus TSA	456	9	2.16302E−05	KATSA versus BA	465	0	1.73440E−06
	KATSA versus STSA	451	14	3.18168E−06	KATSA versus BOA	442	23	2.22483E−04
	KATSA versus MTSA	315	150	1.30592E−01	KATSA versus RSA	448	17	1.12654E−05
	KATSA versus GA	461	4	2.87860E−06	KATSA versus LSHADE	390	75	1.95692E−02
D = 50	KATSA versus EST-TSA	413	52	8.91873E−05	KATSA versus PSO	465	0	1.73440E−06
	KATSA versus fb_TSA	443	22	1.74228E−04	KATSA versus GWO	446	19	3.88218E−06
	KATSA versus TSA	464	1	9.31566E−06	KATSA versus BA	465	0	1.73440E−06
	KATSA versus STSA	462	3	2.87860E−06	KATSA versus BOA	445	20	2.83079E−04
	KATSA versus MTSA	424	41	8.18775E−05	KATSA versus RSA	451	14	1.49356E−05
	KATSA versus GA	459	6	3.51524E−06	KATSA versus LSHADE	373	92	5.70965E−02
D = 100	KATSA versus EST-TSA	423	42	1.74228E−04	KATSA versus PSO	465	0	1.73440E−06
	KATSA versus fb_TSA	463	2	1.79885E−05	KATSA versus GWO	430	35	1.12654E−05
	KATSA versus TSA	464	1	1.49356E−05	KATSA versus BA	465	0	1.73440E−06
	KATSA versus STSA	462	3	2.87860E−06	KATSA versus BOA	439	26	3.88111E−04
	KATSA versus MTSA	439	26	5.30699E−05	KATSA versus RSA	441	24	8.18775E−05
	KATSA versus GA	453	12	8.46608E−06	KATSA versus LSHADE	432	33	1.11380E−03

subject to :

$$g_1(\bar{x}) = Z - \frac{\phi_0}{2\sin^{-1}\left(D_b/D_m\right)} - 1 \le 0,$$
$$g_2(\bar{x}) = K_{D\min}(D-d) - 2D_b \le 0,$$
$$g_3(\bar{x}) = 2D_b - K_{D\max}(D-d) \le 0,$$
$$g_4(\bar{x}) = D_b - {}_w \le 0,$$
$$g_5(\bar{x}) = 0.5(D+d) - D_m \le 0,$$
$$g_6(\bar{x}) = D_m - (0.5+e)(D+d) \le 0,$$
$$g_7(\bar{x}) = \epsilon D_b - 0.5\left(D - D_m - D_b\right) \le 0,$$
$$g_8(\bar{x}) = 0.515 - f_i \le 0,$$
$$g_9(\bar{x}) = 0.515 - f_0 \le 0,$$

where,

$$f_c = 37.91\left\{1 + \left\{1.04\left(\frac{1-\gamma}{1+\gamma}\right)^{1.72}\left(\frac{f_i\left(2f_0-1\right)}{f_0\left(2f_i-1\right)}\right)^{0.41}\right\}^{10/3}\right\}^{-0.3},$$

$$\gamma = \frac{D_b\cos(\alpha)}{D_m}, f_i = \frac{r_i}{D_b}, f_0 = \frac{r_0}{D_b},$$

$$\phi_0 = 2\pi - 2 \times \cos^{-1}\left(\frac{\{(D-d)/2 - 3(T/4)\}^2 + \left\{D/2 - (T/4) - D_b\right\}^2 - \{d/2 + (T/4)\}^2}{2\{(D-d)/2 - 3(T/4)\}\left\{D/2 - (T/4) - D_b\right\}}\right),$$

$T = D - d - 2D_b, D = 160, d = 90, B_w = 30.$

with bounds :

$0.5(D+d) \le D_m \le 0.6(D+d),$

$0.15(D-d) \le D_b \le 0.45(D-d),$

$0.515 \le f_0 \le 0.6, \quad 0.515 \le f_i \le 0.6,$

$0.4 \le K_{D\min} \le 0.5, \quad 0.6 \le K_{D\max} \le 0.7,$

$4 \le Z \le 50, \quad 0.3 \le \epsilon \le 0.4,$

$0.02 \le e \le 0.1, \quad 0.6 \le \zeta \le 0.85.$

The Table 11 displays optimization outcomes for a rolling element bearing. The *best* and *std* rows show that KATSA, TSA, fb_TSA and LSHADE yield consistent optimal solutions, underscoring effective convergence. RSA and GWO exhibit more varied outcomes within this optimization study.

4.7.2. Example 2: Topology optimization

This problem primarily seeks to optimize the arrangement of materials to accommodate a specified range of loads. This optimization occurs within predefined design search parameters while adhering to constraints directly influencing system performance. The foundation of this problem rests upon the utilization of the power-law methodology. Mathematically, the problem can be articulated as presented in a prior work by Sigmund (Sigmund, 2001).

Minimize :

$$f(\bar{x}) = \mathbf{U}^T\mathbf{K}\mathbf{U} = \sum_{e=1}^{N}\left(x_e\right)^p u_e^T k_0 u_0 \tag{13}$$

subject to :

$$h_1(\bar{x}) = \frac{V(\bar{x})}{V_0} - f = 0,$$
$$h_2(\bar{x}) = \mathbf{K}\mathbf{U} - \mathbf{F} = 0,$$

with bounds :

$0 < \bar{x}_{\min} \le x \le 1.$

Table 10 presents results from topology optimization experiments. The *std* row shows that KATSA with the exact best solutions exhibit extremely low standard deviations, which means KATSA has a robust convergence ability. In summary, the table indicates that KATSA and RSA get the optimal results in topology optimization problem.

4.7.3. Example 3: Wind farm layout problem

The arrangement of wind turbines within a wind farm significantly influences the power generation over its operational lifespan. The overarching goal of wind farm layout design is to enhance the overall power output by strategically optimizing the positioning of individual turbines. This optimization task can be formalized as outlined in a study by Wang et al. (2017).

Minimize :

$$f = \sum_{i=1}^{N} E\left(P_i\right) \tag{14}$$

subject to :

$$\underline{x} + R \le x_i \le \bar{x} - R,$$
$$\underline{y} + R \le y_i \le \bar{y} - R,$$
$$\sqrt{\left(x_i - x_j\right)^2 + \left(y_i - y_j\right)^2} \ge 5R, j = 1, 2, \ldots, N \text{ and } j \ne i,$$

where

$$E(P_i) = \sum_{n=1}^{h}\xi_n\{P_r(e^{-(v_r/c_i'((\theta_{n-1}+\theta_n)/2))^{k_i((\theta_{n-1}+\theta_n)/2)}} - e^{-(v_{co}/(c_i'((\theta_{n-1}+\theta_n)/2))^{k_i((\theta_{n-1}+\theta_n)/2)})} + \sum_{j=1}^{s}(e^{-(v_{j-1}/(c_i'((\theta_{n-1}+\theta_n))/2))^{k_i((\theta_{n-1}+\theta_n)/2)}} - e^{-(v_j/(c_i'((\theta_{n-1}+\theta_n)/2)^{k_i((\theta_{n-1}+\theta_n)/2)})}\frac{e^{(v_{j-1}+v_j)/2}}{\alpha + \beta e^{(v_{j-1}+v_j)/2}}\}$$

and, ξ_n is the frequency of the interval $\left[\theta_{n-1}, \theta_n\right)$.

The experimental results are shown in Table 12. We can easily find that KATSA can obtain the best solution among the most compared algorithms from the result. That is saying KATSA can deal well with

Table 10
The result of topology optimization.

	KATSA	TSA	fb_TSA	LSHADE	RSA	GWO	ABC	PSO
best	2.6393E+00	2.6433E+00	2.6396E+00	2.6393E+00	2.6393E+00	2.7296E+00	2.6393E+00	2.6393E+00
mean	2.6393E+00	2.6445E+00	2.6399E+00	2.6538E+00	2.6393E+00	2.9379E+00	2.6393E+00	2.6393E+00
std	1.5384E−15	6.4121E−04	2.0046E−04	1.8124E−02	1.6718E−15	1.4334E−01	8.4485E−08	3.5296E−12
worst	2.6395E+00	2.6460E+00	2.6403E+00	2.7073E+00	2.6393E+00	3.2886E+00	2.6393E+00	2.6393E+00
×1	1.000	0.976	1.000	1.000	1.000	0.256	1.000	1.000
×2	1.000	0.996	0.999	1.000	1.000	0.312	1.000	1.000
×3	1.000	0.997	0.999	1.000	1.000	0.877	1.000	1.000
×4	1.000	1.000	1.000	1.000	1.000	0.479	1.000	1.000
×5	1.000	0.998	1.000	1.000	1.000	0.989	1.000	1.000
×6	1.000	0.999	1.000	1.000	1.000	0.999	1.000	1.000
×7	1.000	0.999	1.000	1.000	1.000	1.000	1.000	1.000
×8	1.000	0.999	1.000	1.000	1.000	0.999	1.000	1.000
×9	1.000	0.999	1.000	1.000	1.000	0.999	1.000	1.000
×10	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
×11	1.000	0.932	0.996	1.000	1.000	0.927	1.000	1.000
×12	1.000	0.993	0.999	1.000	1.000	0.028	1.000	1.000
×13	1.000	0.991	1.000	1.000	1.000	0.952	1.000	1.000
×14	1.000	0.993	0.999	1.000	1.000	0.983	1.000	1.000
×15	1.000	0.997	1.000	1.000	1.000	0.001	1.000	1.000
×16	1.000	0.999	1.000	1.000	1.000	0.997	1.000	1.000
×17	1.000	0.997	1.000	1.000	1.000	0.998	1.000	1.000
×18	1.000	0.999	1.000	1.000	1.000	1.000	1.000	1.000
×19	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
×20	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
×21	1.000	0.896	0.840	1.000	1.000	0.132	1.000	1.000
×22	1.000	0.743	0.995	1.000	1.000	0.961	1.000	1.000
×23	1.000	0.981	0.999	1.000	1.000	0.510	1.000	1.000
×24	1.000	0.977	1.000	1.000	1.000	0.211	1.000	1.000
×25	1.000	0.992	1.000	1.000	1.000	0.996	1.000	1.000
×26	1.000	0.997	1.000	1.000	1.000	0.806	1.000	1.000
×27	1.000	1.000	1.000	1.000	1.000	0.989	1.000	1.000
×28	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
×29	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000
×30	1.000	1.000	1.000	1.000	1.000	1.000	1.000	1.000

Table 11
The result of rolling element bearing.

	KATSA	TSA	fb_TSA	LSHADE	RSA	GWO	ABC	PSO
best	1.6958E+04	1.6958E+04	1.6958E+04	1.6958E+04	1.7768E+04	1.8006E+04	1.6958E+04	1.6958E+04
mean	1.6958E+04	1.6958E+04	1.6958E+04	1.6958E+04	1.9593E+04	2.0794E+04	1.6958E+04	1.6958E+04
std	7.4000E−12	1.1600E−11	7.4000E−12	7.4000E−12	1.4173E+03	1.5178E+03	4.1719E−02	5.2300E−06
worst	1.6958E+04	1.6958E+04	1.6958E+04	1.6958E+04	2.2627E+04	2.3920E+04	1.6958E+04	1.6958E+04
×1	131.200	131.200	131.200	131.200	125.000	127.474	131.200	131.200
×2	18.000	18.000	18.000	18.000	18.388	18.101	18.000	18.000
×3	5.192	5.353	4.850	5.264	4.510	5.240	5.407	4.545
×4	0.600	0.600	0.600	0.600	0.600	0.586	0.600	0.600
×5	0.600	0.600	0.600	0.600	0.600	0.596	0.600	0.600
×6	0.428	0.449	0.441	0.497	0.500	0.459	0.499	0.460
×7	0.681	0.675	0.677	0.659	0.700	0.622	0.687	0.666
×8	0.300	0.300	0.300	0.300	0.400	0.359	0.300	0.300
×9	0.086	0.087	0.061	0.033	0.100	0.087	0.066	0.062
×10	0.600	0.600	0.600	0.600	0.611	0.601	0.600	0.600

high-dimensional problems. The table further provides insights into the optimized wind farm layouts through the variable values displayed.

5. Conclusion and future work

This paper proposes a novel TSA variant called KATSA inspired by the KNN classifying principles. KATSA has improved the seed diversity to achieve a good balance between exploration and exploitation, strengthening its capability to optimize complex problems. The major **contributions** are outlined as follows:

Inspired by the KNN classifying principles, we proposed a search space division mechanism to divide the full search space into best and non-best neighbor areas for guiding the following search process. The IEEE CEC 2014 benchmark functions validate that the methodologies integrated into TSA significantly enhance performance. Notably, KATSA outperforms the comparative algorithms in 100 dimensions with a significance level of $\alpha = 0.01$. At the same time, the proposed KATSA provides more stable and excellent results on three complex real-world engineering problems compared with other algorithms. Thus, the primary goal of improving TSA for solving complex optimization problems has been successfully achieved.

Despite these successes, there remain some **limitations** in this paper. Firstly, KATSA shows less competitive experiment results on low-dimensional or single-modal problems. Second, the dynamic search tendency mechanism aims to achieve a better balance between exploration and exploitation. However, this mechanism uses an ST with

Table 12
The result of wind farm layout problem.

	KATSA	TSA	fb_TSA	LSHADE	RSA	GWO	ABC	PSO
best	−6.2460E+03	−5.3235E+03	−5.5185E+03	−5.5185E+03	−4.9099E+03	−5.9411E+03	−5.5641E+03	−5.9097E+03
mean	−6.1064E+03	−5.0676E+03	−5.1508E+03	−5.1508E+03	−4.7599E+03	−5.6572E+03	−5.3827E+03	−5.3385E+03
std	7.7296E+01	9.0202E+01	1.4086E+02	1.4086E+02	7.4967E+01	2.0717E+02	7.6716E+01	2.0454E+02
worst	−5.9070E+03	−4.9202E+03	−4.9556E+03	−4.9556E+03	−4.6131E+03	−5.2437E+03	−5.2554E+03	−4.9631E+03
×1	1895.916	585.875	1149.847	1149.847	167.760	1041.440	1960.000	724.803
×2	523.825	1271.793	1861.020	1861.020	895.593	304.502	434.503	1676.129
×3	975.875	134.327	1756.872	1756.872	1748.161	106.304	1911.896	1921.560
×4	40.035	908.955	415.095	415.095	1393.272	1507.836	1192.409	437.581
×5	192.448	1043.616	1612.290	1612.290	825.212	1872.907	348.930	337.832
×6	1959.981	1119.908	1684.734	1684.734	558.943	924.785	1960.000	1957.863
×7	1959.956	622.321	237.298	237.298	625.975	90.443	1791.350	1830.285
×8	50.250	1710.308	1600.633	1600.633	105.377	1958.814	1960.000	182.404
×9	1059.472	1788.275	1024.547	1024.547	1881.943	1583.425	634.962	986.434
×10	1959.994	1870.878	735.411	735.411	458.699	1578.262	824.766	666.333
×11	348.266	1768.606	1684.924	1684.924	1907.192	48.173	491.273	242.194
×12	1569.704	376.214	96.078	96.078	1828.379	569.838	1513.377	262.351
×13	792.698	998.545	371.165	371.165	1016.060	117.832	40.000	40.003
×14	761.137	544.819	344.509	344.509	1001.570	308.352	894.274	71.440
×15	1046.090	321.880	355.239	355.239	1199.557	932.031	1960.000	1711.544
×16	268.059	360.305	53.441	53.441	209.867	1955.269	40.000	1062.178
×17	1959.999	1274.236	710.145	710.145	497.671	1936.826	40.000	1746.829
×18	1959.986	906.892	1197.431	1197.431	1958.454	1957.136	40.000	1370.332
×19	1929.027	902.319	1686.420	1686.420	1226.287	1024.732	613.126	1756.546
×20	1502.610	116.772	1225.144	1225.144	1355.873	66.015	423.317	1643.034
×21	291.994	959.586	1614.032	1614.032	56.876	466.275	1260.684	1725.497
×22	1083.156	1555.426	983.320	983.320	1319.063	884.338	40.000	1960.000
×23	40.003	1788.275	256.535	256.535	1732.952	1797.135	1742.063	1781.735
×24	544.450	117.968	1880.624	1880.624	814.081	499.381	1466.655	673.525
×25	1958.940	1786.691	1714.245	1714.245	1198.716	53.898	199.888	595.292
×26	1085.224	672.415	625.987	625.987	1922.051	58.203	469.014	1366.645
×27	40.000	319.284	241.568	241.568	710.762	1926.542	40.000	803.065
×28	40.000	1842.032	1359.570	1359.570	1327.133	1201.946	1567.481	1088.728
×29	40.000	77.232	1045.318	1045.318	1467.651	398.544	1960.000	174.445
×30	40.000	1535.120	1588.430	1588.430	1629.077	1269.772	879.345	672.235

two different but constant values for dynamic regulation, which cannot adapt to different optimization problems. Third, the conclusions are based on single-objective optimization benchmark functions, while KATSA needs to be verified by using multi-objective ones. Fourth, KATSA's capability is evaluated by three real-world engineering optimization problems, but these experiments do not confirm that KATSA can solve all real-world problems across various fields.

For **future work**, we plan to enhance KATSA's capabilities in several aspects. First, a dynamic seed generation strategy should be proposed to address its low competitive performance in low-dimensional or single-modal problems. Second, a linear or non-linear dynamic ST will be necessary to adapt to different problems to achieve a better balance between exploration and exploitation. Third, when solving financial problems, the data that needs to be processed is usually huge in quantity and complex in attributes. This means that completing some tasks requires more time and space costs. For example, credit default risk prediction and supply chain optimization. Using evolutionary algorithms to select features from financial data has become a research hotspot. KATSA performs well in solving complex problems, so adopting KATSA incorporated with feature selection can better solve complex financial problems. Fourth, there are many saddle points in the deep learning model training process, which can easily lead to the stagnation of model training. The proposed KATSA algorithm can complete global optimization in complex problems, so it is expected to provide a better solution when solving deep learning model training.

CRediT authorship contribution statement

Jianhua Jiang: Conceptualization, Formal analysis, Code modification, Writing – review & editing, Validation, Supervision, Final review. **Jiaqi Wu:** Code modification, Writing – review & editing, Simulation experiment, Practical application testing. **Jinmeng Luo:** Writing – review & editing. **Xianqiu Meng:** Writing – review & editing. **Lize Qian:** Writing – review & editing. **Keqin Li:** Writing – review & editing, Supervision, Final review.

Declaration of competing interest

The authors declared that they have no conflicts of interest to this work. We declare that we do not have any commercial or associative interest that represents a conflict of interest in connection with the work submitted.

Acknowledgments

The authors thank the financial support from the Foundation of the Jilin Provincial Department of Science and Technology (No. YDZJ2022 01ZYTS565), the Foundation of Social Science of Jilin Province, China (No. 2022B84) and the Jilin Provincial Department of Education Science and Technology (No. JJKH20240198KJ).

Appendix A. IEEE CEC 2014 benchmark functions

See Tables 13 and 14.

Appendix B. The experiments results based on IEEE CEC 2014

See Tables 15–23.

Data availability

No data was used for the research described in the article.

Table 13
Definitions of the basic functions.

Function name	Function details
High Conditioned Elliptic Function	$f_1(X)=\sum_{i=1}^{D}\left(10^6\right)^{\frac{i-1}{D-1}}X_i^2$
Bent Cigar Function	$f_2(X)=X_1^2+10^6\sum_{i=1}^{D}X_i^2$
Discus Function	$f_3(X)=10^6X_i^2+\sum_{i=1}^{D}X_i^2$
Rosenbrock's Function	$f_4(X)=\sum_{i=1}^{D-1}\left(100\left(X_i^2-X_{i+1}\right)^2+\left(X_i-1\right)^2\right)$
Ackley's Function	$f_5(X)=-20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D}X_i^2}\right)-\exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos\left(2\pi X_i\right)\right)$
Weierstrass Function	$f_6(X)=\sum_{i=1}^{D}\left(\sum_{k=0}^{kmax}\left[a^k\cos\left(2\pi b^k\left(X_i+0.5\right)\right)\right]\right)-D\sum_{k=0}^{kmax}\left[a^k\cos\left(2\pi b^k\times0.5\right);a=0.5,b=3,kmax=20\right]$
Griewank's Function	$f_7(X)=\sum_{i=1}^{D}\frac{X_i^2}{4000}-\prod_{i=1}^{D}\cos\left(\frac{X_i}{\sqrt{i}}\right)+1$
Rastrigin's Function	$f_8(X)=\sum_{i=1}^{D}\left(X_i^2-10\cos\left(2\pi X_i\right)+10\right)$
Modified Schwefel's Function	$f_9=418.9829\times D-\sum_{i=1}^{D}g\left(z_i\right);z_i=x_i+4.209687462275036e+002;$
Katsuura Function	$f_{10}(X)=\frac{10}{D^2}\prod_{i=1}^{D}\left(1+i\sum_{j=1}^{32}\frac{\left\lvert 2^jX_i-\mathrm{round}(2^jX_i)\right\rvert}{2^j}\frac{10}{D^{12}}-\frac{10}{D^2}\right)$
HappyCat Function	$f_{11}(X)=\left\lvert\left(\sum_{i=1}^{D}X_i^2\right)^2-D\right\rvert^{1/4}+\left(0.5\sum_{i=1}^{D}X_i^2+\sum_{i=1}^{D}\right)/D+0.5$
HGBat Function	$f_{12}(X)=\left\lvert\left(\sum_{i=1}^{D}X_i^2\right)^2-\left(\sum_{i=1}^{D}X_i^2\right)\right\rvert^{1/2}+\left(0.5\sum_{i=1}^{D}X_i^2+\sum_{i=1}^{D}\right)/D+0.5$
Expanded Griewank's plus Rosenbrock's Function	$f_{13}=f_7\left(f_4\left(X_1,X_2\right)\right)+f_7\left(f_4\left(X_2,X_3\right)\right)+\ldots+f_7\left(f_4\left(X_{D-1},X_D\right)\right)+f_7\left(f_4\left(X_D,X_1\right)\right)$
Expanded Scaffer's F6 Function	$g(x,y)=0.5+\frac{\left(\sin^2\left(\sqrt{x^2+y^2}\right)-0.5\right)}{\left(1+0.001\left(x^2+y^2\right)\right)^2};f_{14}=g\left(X_1,X_2\right)+g\left(X_2,X_3\right)+\ldots+g\left(X_{D-1},X_D\right)+g\left(X_D,X_1\right)$

Table 14
Benchmark functions of IEEE CEC 2014.

Unimodal functions:	
Rotated High Conditioned Elliptic Function	$F_1(x)=f_1\left(M\left(x-o_1\right)\right)+100$
Rotated Bent Cigar Function	$F_2(x)=f_2\left(M\left(x-o_2\right)\right)+200$
Rotated Discus Function	$F_3(x)=f_3\left(M\left(x-o_3\right)\right)+300$
Multimodal functions:	
Shifted and Rotated Rosenbrock's Function	$F_4(x)=f_4\left(M\left(\frac{2.048(x-o_4)}{100}\right)+1\right)+400$
Shifted and Rotated Ackley's Function	$F_5(x)=f_5\left(M\left(x-o_5\right)\right)+500$
Shifted and Rotated Weierstrass Function	$F_6(x)=f_6\left(M\left(\frac{0.5(x-o_6)}{100}\right)\right)+600$
Shifted and Rotated Griewank's Function	$F_7(x)=f_7\left(M\left(\frac{600(x-o_7)}{100}\right)\right)+700$
Shifted Rastrigin's Function	$F_8(x)=f_8\left(M\left(\frac{5.12(x-o_8)}{100}\right)\right)+800$
Shifted and Rotated Rastrigin's Function	$F_9(x)=f_8\left(M\left(\frac{5.12(x-o_9)}{100}\right)\right)+900$
Shifted Schwefel's Function	$F_{10}(x)=f_9\left(M\left(\frac{1000(x-o_{10})}{100}\right)\right)+1000$
Shifted and Rotated Schwefel's Function	$F_{11}(x)=f_9\left(M\left(\frac{1000(x-o_{11})}{100}\right)\right)+1100$
Shifted and Rotated Katsuura Function	$F_{12}(x)=f_{10}\left(M\left(\frac{5(x-o_{12})}{100}\right)\right)+1200$
Shifted and Rotated HappyCat Function	$F_{13}(x)=f_{11}\left(M\left(\frac{5(x-o_{13})}{100}\right)\right)+1300$
Shifted and Rotated HGBat Function	$F_{14}(x)=f_{12}\left(M\left(\frac{5(x-o_{14})}{100}\right)\right)+1400$
Shifted and Rotated Expanded Griewank's plus Rosenbrock's Function	$F_{15}(x)=f_{13}\left(M\left(\frac{5(x-o_{15})}{100}\right)+1\right)+1500$
Shifted and Rotated Expanded Scaffer's F6 Function	$F_{16}(x)=f_{14}\left(M\left(\left(x-o_{16}\right)\right)1\right)+1600$
Hybrid functions	
$F_{17}=f_9\left(M_1Z_1\right)+f_8\left(M_2Z_2\right)+f_3\left(M_3Z_3\right)+1700$	p = [0.3, 0.3, 0.4]
$F_{18}=f_2\left(M_1Z_1\right)+f_8\left(M_2Z_2\right)+f_3\left(M_3Z_3\right)+1800$	p = [0.3, 0.3, 0.4]
$F_{19}=f_7\left(M_1Z_1\right)+f_8\left(M_2Z_2\right)+f_3\left(M_3Z_3\right)+f_8\left(M_4Z_4\right)+1900$	p = [0.2, 0.2, 0.3, 0.3]
$F_{20}=f_{12}\left(M_1Z_1\right)+f_3\left(M_2Z_2\right)+f_{13}\left(M_3Z_3\right)+f_8\left(M_4Z_4\right)+2000$	p = [0.2, 0.2, 0.3, 0.3]
$F_{21}=f_{14}\left(M_1Z_1\right)+f_{12}\left(M_2Z_2\right)+f_4\left(M_3Z_3\right)+f_9\left(M_4Z_4\right)+f_1\left(M_5Z_5\right)+2100$	p = [0.1, 0.2, 0.2, 0.2, 0.3]
$F_{22}=f_{10}\left(M_1Z_1\right)+f_{11}\left(M_2Z_2\right)+f_{13}\left(M_3Z_3\right)+f_9\left(M_4Z_4\right)+f_5\left(M_5Z_5\right)+2200$	p = [0.1, 0.2, 0.2, 0.2, 0.3]
Notes:	
$Z_1=\left[y_{s_1},y_{s_1},\ldots\ldots,y_{s_{n1}}\right]$	
$Z_2=\left[y_{s_{n1+1}},y_{s_{n1+2}},\ldots\ldots,y_{s_{n1+n2}}\right]$	
$Z_N=\left[y_{s_{\sum_{i=1}^{N-1}n1+1}},y_{s_{\sum_{i=1}^{N-1}n1+2}},\ldots\ldots,y_{s_{5D}}\right]$	
$y=x-o_i,S=randperm(1:D),percentage\ of\ g_i(x)$	
$n_1=\left[p_1D\right],n_2=\left[p_2D\right],\ldots,n_{N-1}=\left[p_{N-1}D\right],n_N=D-\sum_{i=1}^{N-1}n+i$	
Composition functions	
$F_{23}=w_1*F_4'(x)+w_2*\left[1e^{-6}F_1'(x)+100\right]+w_3*\left[1e^{-26}F_2'(x)+200\right]$ $+w_4*\left[1e^{-6}F_3'(x)+300\right]+w_5*\left[1e^{-6}F_1'(x)+400\right]+2300$	$\sigma=[10,20,30,40,50]$
$F_{24}=w_1*F_{10}'(x)+w_2*\left[F_9'(x)+100\right]+w_3*\left[F_{14}'(x)+200\right]+2400$	$\sigma=[20,20,20]$
$F_{25}=w_1*0.25F_{11}'(x)+w_2*\left[F_9'(x)+100\right]+w_3*\left[1e^{-7}F_1'(x)+200\right]+2500$	$\sigma=[10,30,50]$
$F_{26}=w_1*0.25F_{11}'(x)+w_2*\left[F_{13}'(x)+100\right]+w_3*\left[1e^{-7}F_1'(x)+200\right]$ $+w_4*\left[2.5F_6'(x)+300\right]+w_5*\left[1e^{-6}F_{13}'(x)+400\right]+2700$	$\sigma=[10,10,10,10,10]$
$F_{27}=w_1*10F_{14}'(x)+w_2*\left[10F_9'(x)+100\right]+w_3*\left[2.5F_1'1(x)+200\right]$ $+w_4*\left[25F_{16}'(x)+300\right]+w_5*\left[1e^{-6}F_1'(x)+400\right]+2700$	$\sigma=[10,10,10,20,20]$

(*continued on next page*)

Table 14 (*continued*).

$F_{28} = w_1 * 2.5F'_{15}(x) + w_2 * \left[10F'_9(x) + 100\right] + w_3 * \left[2.5F'_1 1(x) + 200\right]$	
$+w_4 * \left[5e^{-4}F'_{16}(x) + 300\right] + w_5 * \left[1e^{-6}F'_1(x) + 400\right] + 2800$	$\sigma = [10, 20, 30, 40, 50]$
$F_{29} = w_1 * F'_{17}(x) + w_2 * \left[F'_{18}(x) + 100\right] + w_3 * \left[F'_{19}(x) + 200\right] + 2900$	$\sigma = [10, 30, 50]$
$F_{30} = w_1 * F'_{20}(x) + w_2 * \left[F'_{21}(x) + 100\right] + w_3 * \left[F'_{22}(x) + 200\right] + 3000$	$\sigma = [10, 30, 50]$

Notes:

$$w_i = \frac{1}{\sqrt{\sum_{j=1}^{D}\left(x_j - o_{ij}\right)}} \exp\left(-\frac{\sum_{j=1}^{D}\left(x_i - o_{ij}^2\right)}{2D\sigma_i^2}\right)$$

Table 15
The results for the different k values in 30 dimensions.

Function	k = 2	k = 3	k = 4	k = 5	k = 6	k = 7	k = 8
F_1	1.2399E+06	1.4234E+06	1.6675E+06	1.4659E+06	2.0003E+06	1.9826E+06	1.9325E+06
F_2	2.5335E+02	2.8716E+02	3.6411E+02	3.9891E+02	7.1506E+02	6.9511E+02	9.5834E+02
F_3	3.8485E+02	5.5192E+02	5.6787E+02	6.8117E+02	6.8758E+02	9.5878E+02	1.0623E+03
F_4	5.0944E+02	5.0205E+02	4.9966E+02	5.1709E+02	4.9873E+02	5.0556E+02	4.9732E+02
F_5	5.2002E+02	5.2003E+02	5.2002E+02	5.2002E+02	5.2004E+02	5.2004E+02	5.2004E+02
F_6	6.0360E+02	6.0381E+02	6.0404E+02	6.0416E+02	6.0385E+02	6.0453E+02	6.0412E+02
F_7	7.0000E+02	7.0000E+02	7.0000E+02	7.0000E+02	7.0000E+02	7.0000E+02	7.0000E+02
F_8	8.3785E+02	8.4418E+02	8.3958E+02	8.4151E+02	8.4169E+02	8.4156E+02	8.4402E+02
F_9	9.5845E+02	9.6159E+02	9.6153E+02	9.5594E+02	9.6209E+02	9.6188E+02	9.5927E+02
F_{10}	2.2036E+03	2.2790E+03	2.2800E+03	2.2348E+03	2.3727E+03	2.2214E+03	2.3734E+03
F_{11}	3.6262E+03	3.5632E+03	3.5258E+03	3.5374E+03	3.3566E+03	3.4248E+03	3.5623E+03
F_{12}	1.2000E+03	1.2000E+03	1.2000E+03	1.2000E+03	1.2000E+03	1.2000E+03	1.2000E+03
F_{13}	1.3004E+03	1.3003E+03	1.3004E+03	1.3004E+03	1.3004E+03	1.3004E+03	1.3004E+03
F_{14}	1.4003E+03	1.4003E+03	1.4003E+03	1.4003E+03	1.4003E+03	1.4003E+03	1.4003E+03
F_{15}	1.5037E+03	1.5039E+03	1.5042E+03	1.5046E+03	1.5042E+03	1.5044E+03	1.5040E+03
F_{16}	1.6115E+03	1.6115E+03	1.6116E+03	1.6115E+03	1.6115E+03	1.6115E+03	1.6114E+03
F_{17}	2.8267E+05	2.5555E+05	2.5031E+05	3.0179E+05	2.4748E+05	2.4444E+05	3.1877E+05
F_{18}	3.8308E+03	4.0167E+03	3.5662E+03	4.2625E+03	4.9803E+03	3.1772E+03	4.1728E+03
F_{19}	1.9079E+03	1.9061E+03	1.9076E+03	1.9061E+03	1.9079E+03	1.9081E+03	1.9063E+03
F_{20}	3.3374E+03	3.1287E+03	3.3568E+03	3.0813E+03	3.3442E+03	3.6977E+03	3.1405E+03
F_{21}	7.5033E+04	8.9001E+04	8.3445E+04	6.8184E+04	7.2323E+04	7.4842E+04	7.0220E+04
F_{22}	2.4947E+03	2.4965E+03	2.4592E+03	2.4745E+03	2.4624E+03	2.4700E+03	2.4293E+03
F_{23}	2.6152E+03	2.6152E+03	2.6152E+03	2.6152E+03	2.6152E+03	2.6152E+03	2.6152E+03
F_{24}	2.6260E+03	2.6262E+03	2.6255E+03	2.6258E+03	2.6272E+03	2.6263E+03	2.6255E+03
F_{25}	2.7050E+03	2.7054E+03	2.7049E+03	2.7050E+03	2.7050E+03	2.7045E+03	2.7048E+03
F_{26}	2.7003E+03	2.7037E+03	2.7037E+03	2.7070E+03	2.7004E+03	2.7004E+03	2.7004E+03
F_{27}	3.0941E+03	3.0994E+03	3.1014E+03	3.0917E+03	3.0847E+03	3.0972E+03	3.0855E+03
F_{28}	3.6821E+03	3.6592E+03	3.6362E+03	3.6357E+03	3.6482E+03	3.6278E+03	3.6448E+03
F_{29}	4.5530E+03	4.2879E+03	4.4294E+03	2.8446E+05	4.5669E+03	4.5211E+03	4.5848E+03
F_{30}	5.3154E+03	5.3057E+03	5.3639E+03	5.3792E+03	5.3476E+03	5.5197E+03	5.2759E+03
Ranking first	**9**	3	3	6	3	6	6

Table 16
The results for the different k values in 50 dimensions.

Function	k = 2	k = 3	k = 4	k = 5	k = 6	k = 7	k = 8
F_1	5.2154E+06	5.4943E+06	6.8816E+06	6.6041E+06	6.2877E+06	8.9924E+06	1.0121E+07
F_2	6.3796E+03	6.6519E+03	4.9554E+03	7.2051E+03	5.8905E+03	6.3065E+03	5.8428E+03
F_3	2.4389E+04	2.1355E+04	3.1414E+04	3.2837E+04	3.9966E+04	3.8281E+04	4.4150E+04
F_4	5.1685E+02	5.0443E+02	5.0218E+02	5.0176E+02	5.0428E+02	5.0256E+02	5.0506E+02
F_5	5.2004E+02	5.2003E+02	5.2004E+02	5.2004E+02	5.2007E+02	5.2011E+02	5.2026E+02
F_6	6.0947E+02	6.1020E+02	6.1041E+02	6.1039E+02	6.0928E+02	6.0922E+02	6.1046E+02
F_7	7.0000E+02	7.0000E+02	7.0000E+02	7.0000E+02	7.0000E+02	7.0000E+02	7.0000E+02
F_8	8.9425E+02	8.8875E+02	8.9624E+02	8.8771E+02	8.9087E+02	8.9592E+02	8.9318E+02
F_9	1.0253E+03	1.0137E+03	1.0200E+03	1.0239E+03	1.0218E+03	1.0265E+03	1.0325E+03
F_{10}	4.2077E+03	4.1388E+03	4.0238E+03	4.0988E+03	3.8465E+03	4.1316E+03	4.1948E+03
F_{11}	5.9877E+03	5.8561E+03	5.8395E+03	6.0922E+03	6.1968E+03	5.6920E+03	5.9447E+03
F_{12}	1.2000E+03	1.2000E+03	1.2000E+03	1.2000E+03	1.2001E+03	1.2001E+03	1.2002E+03
F_{13}	1.3005E+03	1.3005E+03	1.3005E+03	1.3005E+03	1.3005E+03	1.3005E+03	1.3005E+03
F_{14}	1.4003E+03	1.4003E+03	1.4004E+03	1.4003E+03	1.4004E+03	1.4004E+03	1.4004E+03
F_{15}	1.5088E+03	1.5094E+03	1.5086E+03	1.5091E+03	1.5106E+03	1.5109E+03	1.5130E+03
F_{16}	1.6209E+03	1.6207E+03	1.6212E+03	1.6209E+03	1.6210E+03	1.6209E+03	1.6210E+03
F_{17}	8.6319E+05	7.8413E+05	6.2351E+05	8.3087E+05	1.0387E+06	9.4011E+05	9.9017E+05
F_{18}	3.5405E+03	3.7379E+03	3.1118E+03	3.4408E+03	3.8611E+03	3.7239E+03	3.2990E+03
F_{19}	1.9371E+03	1.9304E+03	1.9381E+03	1.9356E+03	1.9400E+03	1.9357E+03	1.9387E+03
F_{20}	1.0132E+04	1.0626E+04	1.0508E+04	1.2514E+04	1.1132E+04	1.1133E+04	1.1471E+04
F_{21}	5.1048E+05	4.9104E+05	5.6615E+05	4.7950E+05	4.9915E+05	5.6077E+05	5.8586E+05
F_{22}	2.9602E+03	2.9807E+03	2.9702E+03	2.9567E+03	2.8751E+03	3.0101E+03	3.0044E+03
F_{23}	2.6440E+03	2.6440E+03	2.6440E+03	2.6440E+03	2.6440E+03	2.6440E+03	2.6440E+03
F_{24}	2.6720E+03	2.6717E+03	2.6711E+03	2.6709E+03	2.6714E+03	2.6714E+03	2.6708E+03
F_{25}	2.7119E+03	2.7126E+03	2.7128E+03	2.7115E+03	2.7107E+03	2.7116E+03	2.7114E+03
F_{26}	2.7305E+03	2.7273E+03	2.7410E+03	2.7239E+03	2.7317E+03	2.7142E+03	2.7139E+03
F_{27}	3.2761E+03	3.2694E+03	3.2549E+03	3.2670E+03	3.2859E+03	3.2622E+03	3.2538E+03
F_{28}	4.0723E+03	4.0451E+03	4.0479E+03	4.0556E+03	4.0213E+03	3.9961E+03	4.0027E+03
F_{29}	4.9907E+03	5.1392E+03	5.2049E+03	5.1517E+03	5.2663E+03	5.2874E+03	5.1948E+03
F_{30}	1.3761E+04	1.3878E+04	1.3781E+04	1.3335E+04	1.3651E+04	1.3064E+04	1.3394E+04
Ranking first	4	6	**7**	5	5	5	4

Table 17
The results for the different k values in 100 dimensions.

Function	k = 2	k = 3	k = 4	k = 5	k = 6	k = 7	k = 8
F_1	8.0345E+07	9.4068E+07	1.0720E+08	1.0709E+08	1.2528E+08	1.4147E+08	1.5969E+08
F_2	1.6475E+04	2.1158E+04	1.8957E+04	1.9533E+04	1.7662E+04	2.2956E+04	1.8688E+04
F_3	1.1775E+05	1.2189E+05	1.5437E+05	1.6458E+05	1.7861E+05	1.9970E+05	2.1592E+05
F_4	6.5147E+02	6.6097E+02	6.5007E+02	6.4579E+02	6.4930E+02	6.5332E+02	6.4855E+02
F_5	5.2006E+02	5.2009E+02	5.2011E+02	5.2027E+02	5.2071E+02	5.2093E+02	5.2116E+02
F_6	6.4086E+02	6.4197E+02	6.3985E+02	6.4189E+02	6.4525E+02	6.4972E+02	6.5381E+02
F_7	7.0000E+02	7.0001E+02	7.0001E+02	7.0001E+02	7.0002E+02	7.0002E+02	7.0003E+02
F_8	1.0397E+03	1.0608E+03	1.0709E+03	1.0717E+03	1.0996E+03	1.1381E+03	1.1808E+03
F_9	1.2355E+03	1.2235E+03	1.2487E+03	1.2421E+03	1.2333E+03	1.2517E+03	1.3037E+03
F_{10}	9.2691E+03	9.2768E+03	9.7716E+03	1.0092E+04	9.8290E+03	1.0280E+04	1.2301E+04
F_{11}	1.3428E+04	1.3240E+04	1.3309E+04	1.3265E+04	1.3323E+04	1.3478E+04	1.4274E+04
F_{12}	1.2002E+03	1.2002E+03	1.2003E+03	1.2004E+03	1.2006E+03	1.2009E+03	1.2012E+03
F_{13}	1.3006E+03	1.3006E+03	1.3006E+03	1.3006E+03	1.3006E+03	1.3007E+03	1.3006E+03
F_{14}	1.4003E+03	1.4004E+03	1.4004E+03	1.4003E+03	1.4003E+03	1.4003E+03	1.4004E+03
F_{15}	1.5399E+03	1.5480E+03	1.5613E+03	1.5698E+03	1.5833E+03	1.5881E+03	1.5908E+03
F_{16}	1.6447E+03	1.6446E+03	1.6449E+03	1.6448E+03	1.6450E+03	1.6453E+03	1.6453E+03
F_{17}	6.1426E+06	6.5246E+06	7.2279E+06	6.9276E+06	7.9676E+06	9.2253E+06	1.0825E+07
F_{18}	1.5282E+04	2.3598E+05	3.1582E+05	1.1928E+05	3.9624E+04	7.0901E+03	1.5774E+05
F_{19}	2.0037E+03	1.9969E+03	2.0028E+03	1.9995E+03	2.0076E+03	1.9993E+03	2.0079E+03
F_{20}	7.0292E+04	7.5122E+04	8.2086E+04	7.9698E+04	8.1568E+04	9.5210E+04	9.6048E+04
F_{21}	4.3063E+06	4.0552E+06	4.0116E+06	3.4985E+06	4.5304E+06	4.9802E+06	5.2648E+06
F_{22}	4.1101E+03	4.1603E+03	4.0305E+03	4.0847E+03	4.1134E+03	4.0827E+03	4.0953E+03
F_{23}	2.6482E+03	2.6483E+03	2.6483E+03	2.6483E+03	2.6483E+03	2.6483E+03	2.6483E+03
F_{24}	2.7845E+03	2.7833E+03	2.7847E+03	2.7841E+03	2.7853E+03	2.7840E+03	2.7838E+03
F_{25}	2.7486E+03	2.7474E+03	2.7483E+03	2.7471E+03	2.7485E+03	2.7527E+03	2.7532E+03
F_{26}	2.8019E+03	2.8019E+03	2.8027E+03	2.8067E+03	2.7974E+03	2.8022E+03	2.8095E+03
F_{27}	3.9753E+03	3.9841E+03	3.9919E+03	4.1269E+03	4.1588E+03	4.2410E+03	4.3080E+03
F_{28}	5.3101E+03	5.2895E+03	5.3185E+03	5.3759E+03	5.3048E+03	5.2639E+03	5.2406E+03
F_{29}	6.1194E+03	6.1961E+03	6.2651E+03	6.5243E+03	6.7378E+03	6.8921E+03	7.3048E+03
F_{30}	3.4693E+04	2.7692E+04	3.1284E+04	3.1255E+04	3.0705E+04	3.3703E+04	3.2656E+04
Ranking first	**15**	6	2	3	2	1	1

Table 18
The mean values for the KATSA, EAT-TSA, fb_TSA, TSA, STSA and MTSA in 30D.

Function	KATSA	EST-TSA	fb_TSA	TSA	STSA	MTSA
F_1	1.4607E+06	8.7063E+07	8.6922E+06	8.6306E+07	5.4322E+08	5.5935E+06
F_2	2.6432E+02	3.1964E+06	5.7720E+03	5.8006E+05	2.7091E+10	1.7527E+05
F_3	5.1213E+02	4.0665E+04	1.1325E+04	3.8908E+04	8.7548E+04	4.0076E+03
F_4	5.0198E+02	5.9385E+02	4.9538E+02	5.5518E+02	2.8800E+03	5.0492E+02
F_5	5.2002E+02	5.2099E+02	5.2104E+02	5.2101E+02	5.2096E+02	5.2101E+02
F_6	6.0386E+02	6.2561E+02	6.0297E+02	6.2231E+02	6.3864E+02	6.0310E+02
F_7	7.0000E+02	7.0041E+02	7.0000E+02	7.0005E+02	9.5471E+02	7.0031E+02
F_8	8.4079E+02	9.8332E+02	8.3501E+02	9.6593E+02	1.0826E+03	8.3268E+02
F_9	9.6182E+02	1.1431E+03	9.7305E+02	1.1215E+03	1.2061E+03	9.7489E+02
F_{10}	2.1722E+03	6.1720E+03	2.1228E+03	6.0724E+03	7.7056E+03	1.9607E+03
F_{11}	3.8468E+03	7.6077E+03	7.3170E+03	8.0662E+03	8.4176E+03	7.8397E+03
F_{12}	1.2000E+03	1.2026E+03	1.2030E+03	1.2029E+03	1.2027E+03	1.2027E+03
F_{13}	1.3003E+03	1.3005E+03	1.3004E+03	1.3005E+03	1.3043E+03	1.3004E+03
F_{14}	1.4003E+03	1.4003E+03	1.4003E+03	1.4003E+03	1.4841E+03	1.4003E+03
F_{15}	1.5045E+03	1.5226E+03	1.5145E+03	1.5197E+03	1.4702E+04	1.5160E+03
F_{16}	1.6114E+03	1.6126E+03	1.6125E+03	1.6126E+03	1.6133E+03	1.6121E+03
F_{17}	2.9416E+05	1.9170E+06	6.2795E+05	2.0408E+06	1.4359E+07	3.6363E+05
F_{18}	3.7917E+03	2.2446E+03	2.1713E+03	2.4866E+03	1.2643E+08	2.8880E+03
F_{19}	1.9085E+03	1.9086E+03	1.9063E+03	1.9079E+03	2.0036E+03	1.9081E+03
F_{20}	3.5500E+03	2.4029E+04	1.5481E+04	1.8209E+04	5.4708E+04	6.6143E+03
F_{21}	6.8118E+04	4.3493E+05	2.3433E+05	4.1483E+05	3.3016E+06	1.7046E+05
F_{22}	2.5094E+03	2.7087E+03	2.4130E+03	2.7233E+03	3.2666E+03	2.3558E+03
F_{23}	2.6152E+03	2.5964E+03	2.6152E+03	2.6153E+03	2.7188E+03	2.6152E+03
F_{24}	2.6259E+03	2.6000E+03	2.6244E+03	2.6276E+03	2.6006E+03	2.6217E+03
F_{25}	2.7048E+03	2.7000E+03	2.7103E+03	2.7227E+03	2.7529E+03	2.7060E+03
F_{26}	2.7004E+03	2.7568E+03	2.7104E+03	2.7006E+03	2.7040E+03	2.7104E+03
F_{27}	3.1036E+03	3.2763E+03	3.0410E+03	3.2588E+03	3.7034E+03	3.0740E+03
F_{28}	3.6367E+03	4.0703E+03	3.7191E+03	4.0380E+03	5.3226E+03	3.6677E+03
F_{29}	4.4718E+03	3.9309E+04	3.9622E+03	3.8750E+04	2.3860E+07	4.0614E+03
F_{30}	5.3411E+03	1.8585E+04	5.1298E+03	1.3683E+04	4.0830E+05	4.7511E+03
Rank_first	16	3	7	0	0	4

Table 19
The mean values for the KATSA, EAT-TSA, fb_TSA, TSA, STSA and MTSA in 50D.

Function	KATSA	EST_TSA	fb_TSA	TSA	STSA	MTSA
F_1	5.0363E+06	3.3614E+08	1.9662E+07	3.4820E+08	2.2216E+09	1.4392E+07
F_2	6.9643E+03	1.2539E+09	3.1315E+04	2.0658E+08	1.1870E+11	7.5336E+06
F_3	2.5230E+04	1.2229E+05	1.1501E+05	1.1979E+05	2.9148E+05	6.5874E+04
F_4	5.1048E+02	1.3890E+03	5.2589E+02	9.6872E+02	2.8739E+04	5.2063E+02
F_5	5.2005E+02	5.2120E+02	5.2120E+02	5.2118E+02	5.2118E+02	5.2119E+02
F_6	6.1020E+02	6.5440E+02	6.1818E+02	6.5379E+02	6.7370E+02	6.1067E+02
F_7	7.0000E+02	7.1003E+02	7.0009E+02	7.0222E+02	1.8093E+03	7.0107E+02
F_8	8.9434E+02	1.2505E+03	8.8771E+02	1.1910E+03	1.4593E+03	8.8478E+02
F_9	1.0162E+03	1.4101E+03	1.0772E+03	1.3672E+03	1.6533E+03	1.1366E+03
F_{10}	4.0992E+03	1.2333E+04	6.2812E+03	1.2273E+04	1.4523E+04	5.4532E+03
F_{11}	6.1584E+03	1.3956E+04	1.4513E+04	1.4740E+04	1.5237E+04	1.4465E+04
F_{12}	1.2000E+03	1.2036E+03	1.2039E+03	1.2038E+03	1.2036E+03	1.2037E+03
F_{13}	1.3005E+03	1.3007E+03	1.3006E+03	1.3008E+03	1.3072E+03	1.3006E+03
F_{14}	1.4003E+03	1.4005E+03	1.4004E+03	1.4005E+03	1.6847E+03	1.4004E+03
F_{15}	1.5092E+03	3.5658E+03	1.5313E+03	1.8002E+03	6.0721E+06	1.5357E+03
F_{16}	1.6211E+03	1.6224E+03	1.6224E+03	1.6224E+03	1.6231E+03	1.6222E+03
F_{17}	6.9887E+05	1.7587E+07	2.0233E+06	1.6607E+07	1.4951E+08	2.1825E+06
F_{18}	3.2455E+03	2.9417E+03	2.5798E+03	2.7540E+03	2.5859E+09	2.4522E+03
F_{19}	1.9331E+03	1.9756E+03	1.9400E+03	1.9641E+03	2.3748E+03	1.9437E+03
F_{20}	1.0804E+04	4.6997E+04	3.7856E+04	4.0498E+04	2.7764E+05	1.5918E+04
F_{21}	4.5278E+05	4.8697E+06	2.0389E+06	6.3323E+06	4.3273E+07	1.1088E+06
F_{22}	3.0274E+03	3.7307E+03	3.2739E+03	3.8559E+03	5.1720E+03	3.3147E+03
F_{23}	2.6440E+03	2.5206E+03	2.6440E+03	2.6459E+03	3.4021E+03	2.6441E+03
F_{24}	2.6716E+03	2.6000E+03	2.6731E+03	2.6954E+03	2.8894E+03	2.6624E+03
F_{25}	2.7125E+03	2.7000E+03	2.7223E+03	2.7700E+03	2.9073E+03	2.7155E+03
F_{26}	2.7280E+03	2.8000E+03	2.7677E+03	2.7396E+03	2.7082E+03	2.7607E+03
F_{27}	3.2689E+03	4.3258E+03	3.4128E+03	4.2045E+03	4.9667E+03	3.3321E+03
F_{28}	4.0286E+03	6.4561E+03	4.4477E+03	5.6757E+03	9.4078E+03	4.3436E+03
F_{29}	5.1272E+03	1.8796E+06	4.6571E+03	1.0955E+06	2.8321E+08	8.3085E+03
F_{30}	1.4278E+04	2.0078E+05	1.8954E+04	1.2699E+05	3.6664E+06	1.8704E+04
Rank_first	23	3	1	0	1	2

Table 20
The mean values for the KATSA, EAT-TSA, fb_TSA, TSA, STSA and MTSA in 100D.

Function	KATSA	EST_TSA	fb_TSA	TSA	STSA	MTSA
F_1	8.2210E+07	1.5240E+09	2.0388E+08	2.2730E+09	1.1172E+10	1.3602E+08
F_2	1.7281E+04	6.1679E+10	9.9583E+08	4.7028E+10	4.7471E+11	4.5200E+08
F_3	1.3522E+05	2.8642E+05	3.3327E+05	3.4571E+05	7.9781E+05	2.1276E+05
F_4	6.4598E+02	1.0911E+04	1.0374E+03	8.4333E+03	1.5327E+05	9.1537E+02
F_5	5.2006E+02	5.2136E+02	5.2137E+02	5.2136E+02	5.2136E+02	5.2136E+02
F_6	6.4079E+02	7.3778E+02	6.8590E+02	7.4232E+02	7.6299E+02	6.5997E+02
F_7	7.0000E+02	1.2913E+03	7.1057E+02	1.1144E+03	4.9532E+03	7.0571E+02
F_8	1.0538E+03	1.9457E+03	1.2019E+03	1.8701E+03	2.5803E+03	1.1562E+03
F_9	1.2378E+03	2.2219E+03	1.6268E+03	2.0922E+03	3.0204E+03	1.5772E+03
F_{10}	9.3960E+03	2.8918E+04	1.8425E+04	2.9141E+04	3.2907E+04	1.6463E+04
F_{11}	1.3319E+04	2.9489E+04	3.2196E+04	3.2175E+04	3.2977E+04	3.1822E+04
F_{12}	1.2001E+03	1.2043E+03	1.2046E+03	1.2045E+03	1.2044E+03	1.2046E+03
F_{13}	1.3006E+03	1.3040E+03	1.3007E+03	1.3031E+03	1.3118E+03	1.3008E+03
F_{14}	1.4004E+03	1.5747E+03	1.4005E+03	1.5257E+03	2.6329E+03	1.4005E+03
F_{15}	1.5374E+03	4.3474E+05	1.7911E+03	5.2213E+05	2.0713E+08	1.6086E+03
F_{16}	1.6448E+03	1.6466E+03	1.6468E+03	1.6470E+03	1.6476E+03	1.6468E+03
F_{17}	5.9181E+06	1.6934E+08	2.5990E+07	2.1580E+08	1.1318E+09	1.5923E+07
F_{18}	1.6664E+05	6.0633E+04	2.9796E+03	3.1619E+03	2.2681E+10	4.7595E+04
F_{19}	2.0078E+03	2.2465E+03	2.0209E+03	2.1528E+03	6.3789E+03	2.0238E+03
F_{20}	5.8714E+04	2.2366E+05	2.0388E+05	2.6327E+05	4.2154E+06	1.2110E+05
F_{21}	3.2889E+06	6.4248E+07	1.0193E+07	8.5766E+07	4.8969E+08	6.5820E+06
F_{22}	4.1157E+03	6.7345E+03	6.6483E+03	7.0708E+03	1.0716E+04	6.6634E+03
F_{23}	2.6482E+03	2.5000E+03	2.6549E+03	2.7630E+03	5.8884E+03	2.6568E+03
F_{24}	2.7847E+03	2.6000E+03	2.8125E+03	2.9880E+03	3.9436E+03	2.7788E+03
F_{25}	2.7504E+03	2.7000E+03	2.8102E+03	3.0256E+03	3.8531E+03	2.7752E+03
F_{26}	2.8051E+03	2.8000E+03	2.8125E+03	2.9680E+03	2.7190E+03	2.8043E+03
F_{27}	3.9883E+03	6.4153E+03	4.9412E+03	6.3373E+03	7.4619E+03	4.4101E+03
F_{28}	5.5049E+03	2.1767E+04	9.8599E+03	1.7988E+04	2.2682E+04	8.2316E+03
F_{29}	6.1287E+03	5.6233E+07	3.8612E+04	2.1790E+07	1.4457E+09	1.6351E+05
F_{30}	3.1288E+04	3.9067E+06	1.9775E+05	2.4575E+06	5.9225E+07	1.4632E+05
Rank_first	25	3	1	0	1	0

Table 21
The mean values for the KATSA and other algorithms with 30D.

Function		KATSA	EST_TSA	fb_TSA	TSA	STSA	MTSA	GA	PSO	GWO	BA	BOA	RSA	LSHADE
Unimodal functions	F_1	1.4607E+06	8.7063E+07	8.6922E+06	8.6306E+07	5.4322E+08	5.5935E+06	6.0717E+08	3.5844E+07	1.1847E+08	3.7428E+09	1.4643E+09	1.0388E+09	2.5276E+06
	F_2	2.6432E+02	3.1964E+06	5.7720E+03	5.8006E+05	2.7091E+10	1.7527E+05	3.6603E+10	3.9648E+06	3.2506E+09	1.2222E+11	6.8039E+10	7.0807E+10	1.2070E+04
	F_3	5.1213E+02	4.0665E+04	1.1325E+04	3.8908E+04	8.7548E+04	4.0076E+03	6.8924E+04	2.7280E+04	5.7768E+04	3.8418E+06	8.1344E+04	8.0452E+04	5.6902E+02
Simple multimodal functions	F_4	5.0198E+02	5.9385E+02	4.9538E+02	5.5518E+02	2.8800E+03	5.0492E+02	5.6725E+03	6.4525E+02	6.9892E+02	3.6710E+04	1.5760E+04	9.5744E+03	5.0936E+02
	F_5	5.2002E+02	5.2099E+02	5.2104E+02	5.2101E+02	5.2096E+02	5.2101E+02	5.2105E+02	5.2100E+02	5.2106E+02	5.2136E+02	5.2109E+02	5.2108E+02	5.2058E+02
	F_6	6.0386E+02	6.2561E+02	6.0297E+02	6.2231E+02	6.3864E+02	6.0310E+02	6.3662E+02	6.1733E+02	6.1630E+02	6.4883E+02	6.3863E+02	6.3854E+02	6.1470E+02
	F_7	7.0000E+02	7.0041E+02	7.0000E+02	7.0005E+02	9.5471E+02	7.0031E+02	1.0828E+03	7.0100E+02	7.3043E+02	1.8557E+03	1.4930E+03	1.2921E+03	7.0007E+02
	F_8	8.4079E+02	9.8332E+02	8.3501E+02	9.6593E+02	1.0826E+03	8.3268E+02	1.0519E+03	8.5344E+02	9.0471E+02	1.2953E+03	1.1218E+03	1.1614E+03	8.1116E+02
	F_9	9.6182E+02	1.1431E+03	9.7305E+02	1.1215E+03	1.2061E+03	9.7489E+02	1.1754E+03	1.0313E+03	1.0271E+03	1.4866E+03	1.2474E+03	1.2431E+03	9.7476E+02
	F_{10}	2.1722E+03	6.1720E+03	2.1228E+03	6.0724E+03	7.7056E+03	1.9607E+03	7.2462E+03	3.1039E+03	4.0351E+03	1.0578E+04	8.6717E+03	7.6871E+03	1.5360E+03
	F_{11}	3.8468E+03	7.6077E+03	7.3170E+03	8.0662E+03	8.4176E+03	7.8397E+03	8.1694E+03	6.9399E+03	5.3667E+03	1.0957E+04	9.1470E+03	8.8673E+03	5.1649E+03
	F_{12}	1.2000E+03	1.2026E+03	1.2030E+03	1.2029E+03	1.2027E+03	1.2027E+03	1.2028E+03	1.2026E+03	1.2029E+03	1.2067E+03	1.2035E+03	1.2033E+03	1.2010E+03
	F_{13}	1.3003E+03	1.3005E+03	1.3004E+03	1.3005E+03	1.3043E+03	1.3004E+03	1.3054E+03	1.3005E+03	1.3007E+03	1.3105E+03	1.3087E+03	1.3077E+03	1.3005E+03
	F_{14}	1.4003E+03	1.4003E+03	1.4003E+03	1.4003E+03	1.4841E+03	1.4003E+03	1.5458E+03	1.4003E+03	1.4054E+03	1.7934E+03	1.7024E+03	1.5659E+03	1.4003E+03
	F_{15}	1.5045E+03	1.5226E+03	1.5145E+03	1.5197E+03	1.4702E+04	1.5160E+03	1.8072E+04	1.5208E+03	1.7862E+03	1.4262E+07	3.3481E+05	1.8967E+05	1.5127E+03
	F_{16}	1.6114E+03	1.6126E+03	1.6125E+03	1.6126E+03	1.6133E+03	1.6121E+03	1.6129E+03	1.6127E+03	1.6123E+03	1.6144E+03	1.6133E+03	1.6134E+03	1.6115E+03
Hybrid functions	F_{17}	2.9416E+05	1.9170E+06	6.2795E+05	2.0408E+06	1.4359E+07	3.6363E+05	3.1593E+07	2.0395E+06	3.9718E+06	3.4698E+08	1.4074E+08	9.0884E+07	3.9775E+04
	F_{18}	3.7917E+03	2.2446E+03	2.1713E+03	2.4866E+03	1.2643E+08	2.8880E+03	7.0673E+08	4.1238E+05	1.7930E+07	1.0421E+10	3.3257E+09	4.9397E+09	2.1377E+03
	F_{19}	1.9085E+03	1.9086E+03	1.9063E+03	1.9079E+03	2.0036E+03	1.9081E+03	2.1630E+03	1.9291E+03	1.9545E+03	3.2994E+03	2.4514E+03	2.2551E+03	1.9144E+03
	F_{20}	3.5500E+03	2.4029E+04	1.5481E+04	1.8209E+04	5.4708E+04	6.6143E+03	6.0670E+04	3.1789E+04	4.0613E+04	3.5309E+07	4.4758E+05	1.5303E+05	4.2499E+03
	F_{21}	6.8118E+04	4.3493E+05	2.3433E+05	4.1483E+05	3.3016E+06	1.7046E+05	8.7241E+06	6.9661E+05	1.5722E+06	1.9906E+08	3.4659E+07	4.4416E+07	1.0102E+04
	F_{22}	2.5094E+03	2.7087E+03	2.4130E+03	2.7233E+03	3.2666E+03	2.3558E+03	3.6332E+03	2.6350E+03	2.6643E+03	3.7620E+04	5.5592E+04	1.0670E+04	2.6585E+03
	F_{23}	2.6152E+03	2.5964E+03	2.6152E+03	2.6153E+03	2.7188E+03	2.6152E+03	2.8144E+03	2.6180E+03	2.6509E+03	4.3393E+03	2.5000E+03	2.5000E+03	2.6155E+03
Composition functions	F_{24}	2.6259E+03	2.6000E+03	2.6244E+03	2.6276E+03	2.6006E+03	2.6217E+03	2.6228E+03	2.6404E+03	2.6001E+03	2.8782E+03	2.6000E+03	2.6000E+03	2.6386E+03
	F_{25}	2.7048E+03	2.7000E+03	2.7103E+03	2.7227E+03	2.7529E+03	2.7060E+03	2.7060E+03	2.7151E+03	2.7119E+03	2.9176E+03	2.7000E+03	2.7000E+03	2.7081E+03
	F_{26}	2.7004E+03	2.7568E+03	2.7104E+03	2.7006E+03	2.7040E+03	2.7104E+03	2.7118E+03	2.7413E+03	2.7371E+03	2.9267E+03	2.7831E+03	2.7969E+03	2.7171E+03
	F_{27}	3.1036E+03	3.2763E+03	3.0410E+03	3.2588E+03	3.7034E+03	3.0740E+03	3.8177E+03	3.3599E+03	3.4197E+03	4.5162E+03	3.5248E+03	4.0861E+03	3.2647E+03
	F_{28}	3.6367E+03	4.0703E+03	3.7191E+03	4.0380E+03	5.3226E+03	3.6677E+03	8.6542E+03	4.5722E+03	4.1941E+03	8.7983E+03	5.3382E+03	5.6994E+03	3.9640E+03
	F_{29}	4.4718E+03	3.9309E+04	3.9622E+03	3.8750E+04	2.3860E+07	4.0614E+03	2.6376E+08	2.8026E+06	1.4053E+06	1.8043E+08	3.1000E+03	1.5199E+07	3.1440E+04
	F_{30}	5.3411E+03	1.8585E+04	5.1298E+03	1.3683E+04	4.0830E+05	4.7511E+03	1.6505E+06	2.2085E+04	1.0430E+05	7.0220E+06	3.2000E+03	2.5314E+06	6.8634E+03
Ranking first		**14**	1	5	0	0	1	0	0	0	0	5	3	5
Average ranking		**2.47**	5.90	3.73	5.97	9.07	3.53	9.57	6.70	7.37	12.93	9.80	10.20	3.77

Table 22
The mean values for the KATSA and other algorithms with 50D.

Function		KATSA	EST_TSA	fb_TSA	TSA	STSA	MTSA	GA	PSO	GWO	BA	BOA	RSA	LSHADE
Unimodal functions	F_1	5.0363E+06	3.3614E+08	1.9662E+07	3.4820E+08	2.2216E+09	1.4392E+07	1.9795E+09	9.9335E+07	1.7144E+08	9.6998E+09	5.6590E+09	3.3341E+09	1.2775E+07
	F_2	6.9643E+03	1.2539E+09	3.1315E+04	2.0658E+08	1.1870E+11	7.5336E+06	9.5198E+10	3.4985E+08	1.4427E+10	2.5526E+11	1.4995E+11	1.4462E+11	3.9796E+07
	F_3	2.5230E+04	1.2229E+05	1.1501E+05	1.1979E+05	2.9148E+05	6.5874E+04	1.3038E+05	1.6085E+05	1.1120E+05	3.5046E+06	1.7197E+05	1.5100E+05	1.7990E+04
Simple multimodal functions	F_4	5.1048E+02	1.3890E+03	5.2589E+02	9.6872E+02	2.8739E+04	5.2063E+02	2.0456E+04	8.4204E+02	1.7183E+03	1.0289E+05	4.9272E+04	3.9017E+04	6.0016E+02
	F_5	5.2005E+02	5.2120E+02	5.2120E+02	5.2118E+02	5.2118E+02	5.2119E+02	5.2123E+02	5.2117E+02	5.2124E+02	5.2145E+02	5.2124E+02	5.2123E+02	5.2077E+02
	F_6	6.1020E+02	6.5440E+02	6.1818E+02	6.5379E+02	6.7370E+02	6.1067E+02	6.6714E+02	6.3927E+02	6.3698E+02	6.8506E+02	6.7012E+02	6.7386E+02	6.3364E+02
	F_7	7.0000E+02	7.1003E+02	7.0009E+02	7.0222E+02	1.8093E+03	7.0107E+02	1.6021E+03	7.0437E+02	8.0509E+02	3.0048E+03	2.2098E+03	2.0372E+03	7.0151E+02
	F_8	8.9434E+02	1.2505E+03	8.8771E+02	1.1910E+03	1.4593E+03	8.8478E+02	1.3298E+03	9.4523E+02	1.0440E+03	1.7106E+03	1.4251E+03	1.4805E+03	8.7260E+02
	F_9	1.0162E+03	1.4101E+03	1.0772E+03	1.3672E+03	1.6533E+03	1.1366E+03	1.4906E+03	1.2125E+03	1.1508E+03	2.0084E+03	1.6049E+03	1.6098E+03	1.1121E+03
	F_{10}	4.0992E+03	1.2333E+04	6.2812E+03	1.2273E+04	1.4523E+04	5.4532E+03	1.3800E+04	6.1291E+03	7.8038E+03	1.7816E+04	1.5452E+04	1.4506E+04	3.6771E+03
	F_{11}	6.1584E+03	1.3956E+04	1.4513E+04	1.4740E+04	1.5237E+04	1.4465E+04	1.4437E+04	1.4238E+04	9.3696E+03	1.7987E+04	1.5891E+04	1.5361E+04	1.0020E+04
	F_{12}	1.2000E+03	1.2036E+03	1.2039E+03	1.2038E+03	1.2036E+03	1.2037E+03	1.2036E+03	1.2035E+03	1.2039E+03	1.2074E+03	1.2046E+03	1.2043E+03	1.2016E+03
	F_{13}	1.3005E+03	1.3007E+03	1.3006E+03	1.3008E+03	1.3072E+03	1.3006E+03	1.3063E+03	1.3006E+03	1.3013E+03	1.3115E+03	1.3087E+03	1.3081E+03	1.3007E+03
	F_{14}	1.4003E+03	1.4005E+03	1.4004E+03	1.4005E+03	1.6847E+03	1.4004E+03	1.6258E+03	1.4006E+03	1.4289E+03	2.0234E+03	1.7928E+03	1.7050E+03	1.4004E+03
	F_{15}	1.5092E+03	3.5658E+03	1.5313E+03	1.8002E+03	6.0721E+06	1.5357E+03	4.9764E+05	1.5639E+03	5.2023E+03	7.4978E+07	4.8995E+06	3.3751E+06	1.5550E+03
	F_{16}	1.6211E+03	1.6224E+03	1.6224E+03	1.6224E+03	1.6231E+03	1.6222E+03	1.6226E+03	1.6224E+03	1.6218E+03	1.6241E+03	1.6231E+03	1.6230E+03	1.6210E+03
Hybrid functions	F_{17}	6.9887E+05	1.7587E+07	2.0233E+06	1.6607E+07	1.4951E+08	2.1825E+06	2.3789E+08	1.0572E+07	1.1227E+07	1.3395E+09	6.2598E+08	4.9092E+08	3.7852E+05
	F_{18}	3.2455E+03	2.9417E+03	2.5798E+03	2.7540E+03	2.5859E+09	2.4522E+03	6.7924E+09	2.2367E+05	1.9755E+08	3.1519E+10	1.8986E+10	1.2539E+10	3.1691E+03
	F_{19}	1.9331E+03	1.9756E+03	1.9400E+03	1.9641E+03	2.3748E+03	1.9437E+03	2.6671E+03	1.9801E+03	2.0289E+03	8.3801E+03	5.2969E+03	4.3197E+03	1.9629E+03
	F_{20}	1.0804E+04	4.6997E+04	3.7856E+04	4.0498E+04	2.7764E+05	1.5918E+04	7.2512E+04	8.8892E+04	5.1698E+04	4.3617E+07	4.6730E+05	3.4133E+05	1.0586E+04
	F_{21}	4.5278E+05	4.8697E+06	2.0389E+06	6.3323E+06	4.3273E+07	1.1088E+06	1.4779E+07	6.1629E+06	5.9534E+06	4.7278E+08	1.2196E+08	9.0059E+07	1.9346E+05
	F_{22}	3.0274E+03	3.7307E+03	3.2739E+03	3.8559E+03	5.1720E+03	3.3147E+03	1.1405E+04	3.7540E+03	3.3408E+03	1.7380E+06	5.9611E+05	1.3343E+05	3.6825E+03
	F_{23}	2.6440E+03	2.5206E+03	2.6440E+03	2.6459E+03	3.4021E+03	2.6441E+03	2.9378E+03	2.6564E+03	2.7781E+03	6.2080E+03	2.5000E+03	2.5000E+03	2.6474E+03
Composition functions	F_{24}	2.6716E+03	2.6000E+03	2.6731E+03	2.6954E+03	2.8894E+03	2.6624E+03	2.6630E+03	2.6966E+03	2.6003E+03	3.1985E+03	2.6000E+03	2.6000E+03	2.6959E+03
	F_{25}	2.7125E+03	2.7000E+03	2.7223E+03	2.7700E+03	2.9073E+03	2.7155E+03	2.7118E+03	2.7364E+03	2.7247E+03	3.2255E+03	2.7000E+03	2.7000E+03	2.7308E+03
	F_{26}	2.7280E+03	2.8000E+03	2.7677E+03	2.7396E+03	2.7082E+03	2.7607E+03	2.7969E+03	2.7894E+03	2.8023E+03	3.2533E+03	2.7963E+03	2.7969E+03	2.7672E+03
	F_{27}	3.2689E+03	4.3258E+03	3.4128E+03	4.2045E+03	4.9667E+03	3.3321E+03	5.2414E+03	4.0639E+03	3.9295E+03	5.6965E+03	5.0586E+03	4.9335E+03	3.8589E+03
	F_{28}	4.0286E+03	6.4561E+03	4.4477E+03	5.6757E+03	9.4078E+03	4.3436E+03	1.5820E+04	6.9114E+03	6.2769E+03	1.7731E+04	8.8334E+03	1.0968E+04	5.2327E+03
	F_{29}	5.1272E+03	1.8796E+06	4.6571E+03	1.0955E+06	2.8321E+08	8.3085E+03	1.2888E+09	3.4687E+07	2.3266E+07	5.8951E+08	3.1000E+03	6.5755E+07	1.6652E+06
	F_{30}	1.4278E+04	2.0078E+05	1.8954E+04	1.2699E+05	3.6664E+06	1.8704E+04	1.9129E+07	1.2426E+05	3.8648E+05	4.7834E+07	3.2000E+03	6.1704E+06	2.2495E+04
Ranking first		**16**	1	0	0	1	1	0	0	0	0	5	3	7
Average ranking		**2.03**	6.37	4.10	6.53	9.83	3.63	9.03	6.47	6.90	12.97	9.63	9.80	3.70

Table 23
The mean values for the KATSA and other algorithms with 100D.

Function		KATSA	EST_TSA	fb_TSA	TSA	STSA	MTSA	GA	PSO	GWO	BA	BOA	RSA	LSHADE
Unimodal functions	F_1	8.2210E+07	1.5240E+09	2.0388E+08	2.2730E+09	1.1172E+10	1.3602E+08	3.7672E+09	5.3502E+08	5.2882E+08	2.0318E+10	9.8104E+09	8.1064E+09	1.6058E+08
	F_2	1.7281E+04	6.1679E+10	9.9583E+08	4.7028E+10	4.7471E+11	4.5200E+08	2.1761E+11	9.7341E+09	5.9235E+10	5.2440E+11	2.9888E+11	2.8168E+11	5.1143E+09
	F_3	1.3522E+05	2.8642E+05	3.3327E+05	3.4571E+05	7.9781E+05	2.1276E+05	2.9129E+05	4.2901E+05	2.7003E+05	3.5831E+06	3.2300E+05	3.0644E+05	9.4005E+04
Simple multimodal functions	F_4	6.4598E+02	1.0911E+04	1.0374E+03	8.4333E+03	1.5327E+05	9.1537E+02	5.3283E+04	2.3913E+03	6.4254E+03	2.2998E+05	1.0689E+05	7.9457E+04	1.4725E+03
	F_5	5.2006E+02	5.2136E+02	5.2137E+02	5.2136E+02	5.2136E+02	5.2136E+02	5.2138E+02	5.2136E+02	5.2140E+02	5.2154E+02	5.2140E+02	5.2140E+02	5.2113E+02
	F_6	6.4079E+02	7.3778E+02	6.8590E+02	7.4232E+02	7.6299E+02	6.5997E+02	7.5119E+02	7.0818E+02	6.9629E+02	7.7508E+02	7.5690E+02	7.5770E+02	6.9918E+02
	F_7	7.0000E+02	1.2913E+03	7.1057E+02	1.1144E+03	4.9532E+03	7.0571E+02	2.9194E+03	7.9020E+02	1.1671E+03	5.5092E+03	3.8609E+03	3.5575E+03	7.5332E+02
	F_8	1.0538E+03	1.9457E+03	1.2019E+03	1.8701E+03	2.5803E+03	1.1562E+03	2.0136E+03	1.3350E+03	1.4975E+03	2.6872E+03	2.1571E+03	2.2549E+03	1.2390E+03
	F_9	1.2378E+03	2.2219E+03	1.6268E+03	2.0922E+03	3.0204E+03	1.5772E+03	2.2631E+03	1.8576E+03	1.6358E+03	3.1608E+03	2.4130E+03	2.3727E+03	1.6434E+03
	F_{10}	9.3960E+03	2.8918E+04	1.8425E+04	2.9141E+04	3.2907E+04	1.6463E+04	3.1019E+04	1.8098E+04	1.8174E+04	3.6663E+04	3.3145E+04	3.0994E+04	1.3614E+04
	F_{11}	1.3319E+04	2.9489E+04	3.2196E+04	3.2175E+04	3.2977E+04	3.1822E+04	3.0744E+04	3.1860E+04	2.0678E+04	3.6908E+04	3.3046E+04	3.1336E+04	2.5680E+04
	F_{12}	1.2001E+03	1.2043E+03	1.2046E+03	1.2045E+03	1.2044E+03	1.2046E+03	1.2044E+03	1.2043E+03	1.2047E+03	1.2070E+03	1.2050E+03	1.2050E+03	1.2025E+03
	F_{13}	1.3006E+03	1.3040E+03	1.3007E+03	1.3031E+03	1.3118E+03	1.3008E+03	1.3080E+03	1.3007E+03	1.3040E+03	1.3129E+03	1.3096E+03	1.3091E+03	1.3007E+03
	F_{14}	1.4004E+03	1.5747E+03	1.4005E+03	1.5257E+03	2.6329E+03	1.4005E+03	2.0504E+03	1.4239E+03	1.5491E+03	2.8509E+03	2.3274E+03	2.2355E+03	1.4034E+03
	F_{15}	1.5374E+03	4.3474E+05	1.7911E+03	5.2213E+05	2.0713E+08	1.6086E+03	4.5518E+06	1.6226E+04	9.0649E+04	4.0532E+08	2.5586E+07	1.3696E+07	2.5839E+03
	F_{16}	1.6448E+03	1.6466E+03	1.6468E+03	1.6470E+03	1.6476E+03	1.6468E+03	1.6467E+03	1.6470E+03	1.6464E+03	1.6490E+03	1.6474E+03	1.6474E+03	1.6451E+03
Hybrid functions	F_{17}	5.9181E+06	1.6934E+08	2.5990E+07	2.1580E+08	1.1318E+09	1.5923E+07	7.4381E+08	8.6819E+07	6.2149E+07	4.0084E+09	1.7318E+09	1.2478E+09	6.4466E+06
	F_{18}	1.6664E+05	6.0633E+04	2.9796E+03	3.1619E+03	2.2681E+10	4.7595E+04	2.1905E+10	9.5145E+06	1.3817E+09	7.5090E+10	3.6866E+10	3.3202E+10	4.0277E+05
	F_{19}	2.0078E+03	2.2465E+03	2.0209E+03	2.1528E+03	6.3789E+03	2.0238E+03	5.9204E+03	2.1467E+03	2.4195E+03	2.2114E+04	1.2860E+04	1.0180E+04	2.0826E+03
	F_{20}	5.8714E+04	2.2366E+05	2.0388E+05	2.6327E+05	4.2154E+06	1.2110E+05	4.8363E+05	2.7018E+05	1.5082E+05	9.6722E+07	1.6018E+06	8.2949E+05	6.7158E+04
	F_{21}	3.2889E+06	6.4248E+07	1.0193E+07	8.5766E+07	4.8969E+08	6.5820E+06	1.8993E+08	3.3551E+07	3.1073E+07	1.7613E+09	5.4864E+08	4.1426E+08	2.8796E+06
	F_{22}	4.1157E+03	6.7345E+03	6.6483E+03	7.0708E+03	1.0716E+04	6.6634E+03	1.9103E+04	6.4796E+03	4.7552E+03	2.2933E+06	3.8885E+05	1.2615E+05	5.9861E+03
	F_{23}	2.6482E+03	2.5000E+03	2.6549E+03	2.7630E+03	5.8884E+03	2.6568E+03	3.1142E+03	2.7157E+03	3.0605E+03	9.5086E+03	2.5000E+03	2.5000E+03	2.7172E+03
Composition functions	F_{24}	2.7847E+03	2.6000E+03	2.8125E+03	2.9880E+03	3.9436E+03	2.7788E+03	2.7527E+03	2.9186E+03	2.6012E+03	4.1511E+03	2.6000E+03	2.6000E+03	2.8977E+03
	F_{25}	2.7504E+03	2.7000E+03	2.8102E+03	3.0256E+03	3.8531E+03	2.7752E+03	2.7301E+03	2.8581E+03	2.7067E+03	3.9628E+03	2.7000E+03	2.7000E+03	2.7972E+03
	F_{26}	2.8051E+03	2.8000E+03	2.8125E+03	2.9680E+03	2.7190E+03	2.8043E+03	2.8022E+03	2.8338E+03	2.8007E+03	3.9629E+03	2.8000E+03	2.8000E+03	2.8032E+03
	F_{27}	3.9883E+03	6.4153E+03	4.9412E+03	6.3373E+03	7.4619E+03	4.4101E+03	8.3576E+03	5.8202E+03	5.4642E+03	8.9874E+03	8.0255E+03	7.5058E+03	5.5091E+03
	F_{28}	5.5049E+03	2.1767E+04	9.8599E+03	1.7988E+04	2.2682E+04	8.2316E+03	3.5383E+04	1.4224E+04	1.3043E+04	4.2766E+04	2.8543E+04	2.3517E+04	9.7682E+03
	F_{29}	6.1287E+03	5.6233E+07	3.8612E+04	2.1790E+07	1.4457E+09	1.6351E+05	1.8889E+09	2.4098E+07	1.6552E+08	2.2187E+09	3.1000E+03	3.1000E+03	5.4894E+07
	F_{30}	3.1288E+04	3.9067E+06	1.9775E+05	2.4575E+06	5.9225E+07	1.4632E+05	1.4058E+08	1.0129E+06	6.5349E+06	5.7193E+08	3.2000E+03	1.3849E+07	1.7309E+05
Ranking first		**21**	2	1	0	1	0	0	0	0	0	5	4	2
Average ranking		**2.03**	6.27	4.93	7.47	10.47	3.90	8.67	6.37	5.90	13.00	9.30	8.63	4.07

References

Abu, H. A., Hassanat, A. B., Lasassmeh, O., Tarawneh, A. S., Alhasanat, M. B., Eyal Salman, H. S., & Prasath, V. S. (2019). Effects of distance measure choice on k-nearest neighbor classifier performance: a review. *Big Data*, *7*(4), 221–248.

Abualigah, L., Abd Elaziz, M., Sumari, P., Geem, Z. W., & Gandomi, A. H. (2022). Reptile search algorithm (RSA): a nature-inspired meta-heuristic optimizer. *Expert Systems with Applications*, *191*, Article 116158.

Adithiyaa, T., Chandramohan, D., & Sathish, T. (2020). Optimal prediction of process parameters by GWO-KNN in stirring-squeeze casting of AA2219 reinforced metal matrix composites. *Materials Today: Proceedings*, *21*, 1000–1007.

Alkasassbeh, M., Altarawneh, G. A., & Hassanat, A. (2015). On enhancing the performance of nearest neighbour classifiers using hassanat distance metric. ArXiv preprint arXiv:1501.00687.

Arora, S., & Singh, S. (2019). Butterfly optimization algorithm: a novel approach for global optimization. *Soft Computing*, *23*, 715–734.

Babalik, A., Cinar, A. C., & Kiran, M. S. (2018). A modification of tree-seed algorithm using Deb's rules for constrained optimization. *Applied Soft Computing*, *63*, 289–305.

Bejarbaneh, E. Y., Bagheri, A., Bejarbaneh, B. Y., Buyamin, S., & Chegini, S. N. (2019). A new adjusting technique for PID type fuzzy logic controller using PSOSCALF optimization algorithm. *Applied Soft Computing*, *85*, Article 105822.

Beşkirli, M., & Kiran, M. S. (2023). Optimization of butterworth and bessel filter parameters with improved tree-seed algorithm. *Biomimetics*, *8*(7), 540.

Chen, X., An, Y., Zhang, Z., & Li, Y. (2020). An approximate nondominated sorting genetic algorithm to integrate optimization of production scheduling and accurate maintenance based on reliability intervals. *Journal of Manufacturing Systems*, *54*, 227–241.

Cherif, W. (2018). Optimization of K-NN algorithm by clustering and reliability coefficients: application to breast-cancer diagnosis. *Procedia Computer Science*, *127*, 293–299.

Cinar, A. C., & Kiran, M. S. (2018). Similarity and logic gate-based tree-seed algorithms for binary optimization. *Computers & Industrial Engineering*, *115*, 631–646.

Cinar, A. C., Korkmaz, S., & Kiran, M. S. (2020). A discrete tree-seed algorithm for solving symmetric traveling salesman problem. *Engineering Science and Technology, an International Journal*, *23*(4), 879–890.

Dehghani, M., & Trojovskỳ, P. (2022). Hybrid leader based optimization: a new stochastic optimization algorithm for solving optimization applications. *Scientific Reports*, *12*(1), 1–16.

Deng, L., & Liu, S. (2023). Snow ablation optimizer: A novel metaheuristic technique for numerical optimization and engineering design. *Expert Systems with Applications*, *225*, Article 120069.

Deng, W., Ni, H., Liu, Y., Chen, H., & Zhao, H. (2022). An adaptive differential evolution algorithm based on belief space and generalized opposition-based learning for resource allocation. *Applied Soft Computing*, *127*, Article 109419.

Derrac, J., García, S., Molina, D., & Herrera, F. (2011). A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. *Swarm and Evolutionary Computation*, *1*(1), 3–18.

Dhar, J., Shukla, A., Kumar, M., & Gupta, P. (2020). A weighted mutual k-nearest neighbour for classification mining. ArXiv preprint arXiv:2005.08640.

Ding, Z., Li, J., Hao, H., & Lu, Z.-R. (2019). Nonlinear hysteretic parameter identification using an improved tree-seed algorithm. *Swarm and Evolutionary Computation*, *46*, 69–83.

Dorigo, M., Birattari, M., & Stützle, T. (2006). Ant colony optimization. *IEEE Computational Intelligence Magazine*, *1*(4), 28–39.

Droste, S., Jansen, T., & Wegener, I. (2002). Optimization with randomized search heuristics—the (A)NFL theorem, realistic scenarios, and difficult functions. *Theoretical Computer Science*, *287*(1), 131–144.

Eswaran, M., kumar Inkulu, A., Tamilarasan, K., Bahubalendruni, M. R., Jaideep, R., Faris, M. S., & Jacob, N. (2024). Optimal layout planning for human robot collaborative assembly systems and visualization through immersive technologies. *Expert Systems with Applications*, *241*, Article 122465.

Fix, E. (1985). *Discriminatory analysis: nonparametric discrimination, consistency properties (Vol. 1)*. USAF School of Aviation Medicine.

Gharehchopogh, F. S. (2022). Advances in tree seed algorithm: a comprehensive survey. *Archives of Computational Methods in Engineering*, *29*(5), 3281–3304.

Goli, A., Tirkolaee, E. B., & Aydın, N. S. (2021). Fuzzy integrated cell formation and production scheduling considering automated guided vehicles and human factors. *IEEE Transactions on Fuzzy Systems*, *29*(12), 3686–3695.

Gou, J., Ma, H., Ou, W., Zeng, S., Rao, Y., & Yang, H. (2019). A generalized mean distance-based k-nearest neighbor classifier. *Expert Systems with Applications*, *115*, 356–372.

Han, E.-H., Karypis, G., & Kumar, V. (2001). Text categorization using weight adjusted k-nearest neighbor classification. In *Advances in knowledge discovery and data mining: 5th Pacific-Asia conference, PAKDD 2001 Hong kong, China, April 16–18, 2001 proceedings 5* (pp. 53–65). Springer.

Hassanat, A. B., Abbadi, M. A., Altarawneh, G. A., & Alhasanat, A. A. (2014). Solving the problem of the k parameter in the KNN classifier using an ensemble learning approach. ArXiv preprint arXiv:1409.0919.

Holland, J. H., & Reitman, J. S. (1977). Cognitive systems based on adaptive algorithms. *ACM Sigart Bulletin*, (63), 49.

Jiang, J., Han, R., Meng, X., & Li, K. (2020). TSASC: tree–seed algorithm with sine–cosine enhancement for continuous optimization problems. *Soft Computing*, *24*(24), 18627–18646.

Jiang, J., Jiang, S., Meng, X., & Qiu, C. (2019). EST-TSA: an effective search tendency based to tree seed algorithm. *Physica A. Statistical Mechanics and its Applications*, *534*, Article 122323.

Jiang, J., Liu, Y., & Zhao, Z. (2021). TriTSA: triple tree-seed algorithm for dimensional continuous optimization and constrained engineering problems. *Engineering Applications of Artificial Intelligence*, *104*, Article 104303.

Jiang, J., Meng, X., Chen, Y., Qiu, C., Liu, Y., & Li, K. (2020). Enhancing tree-seed algorithm via feed-back mechanism for optimizing continuous problems. *Applied Soft Computing*, *92*, Article 106314.

Jiang, J., Meng, X., Qian, L., & Wang, H. (2022). Enhance tree-seed algorithm using hierarchy mechanism for constrained optimization problems. *Expert Systems with Applications*, *209*, Article 118311.

Jiang, J., Wu, J., Luo, J., Yang, X., & Huang, Z. (2024). MOBCA: Multi-Objective Besiege and Conquer Algorithm. *Biomimetics*, *9*(6), 316.

Jiang, J., Xu, M., Meng, X., & Li, K. (2020). STSA: a sine tree-seed algorithm for complex continuous optimization problems. *Physica A. Statistical Mechanics and its Applications*, *537*, Article 122802.

Jiang, J., Yang, X., Li, M., & Chen, T. (2023). ATSA: an adaptive tree seed algorithm based on double-layer framework with tree migration and seed intelligent generation. *Knowledge-Based Systems*, Article 110940.

Karaboga, D., & Ozturk, C. (2011). A novel clustering approach: artificial bee colony (ABC) algorithm. *Applied Soft Computing*, *11*(1), 652–657.

Keller, J. M., Gray, M. R., & Givens, J. A. (1985). A fuzzy k-nearest neighbor algorithm. *IEEE Transactions on Systems, Man and Cybernetics*, (4), 580–585.

Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. In *Proceedings of international conference on neural networks (Vol. 4)* (pp. 1942–1948). IEEE.

Kılıç, F., Kaya, Y., & Yildirim, S. (2021). A novel multi population based particle swarm optimization for feature selection. *Knowledge-Based Systems*, *219*, Article 106894.

Kiran, M. S. (2015). TSA: tree-seed algorithm for continuous optimization. *Expert Systems with Applications*, *42*(19), 6686–6698.

Kiran, M. S., & Hakli, H. (2021). A tree–seed algorithm based on intelligent search mechanisms for continuous optimization. *Applied Soft Computing*, *98*, Article 106938.

Kumar, A., Wu, G., Ali, M. Z., Mallipeddi, R., Suganthan, P. N., & Das, S. (2020). A test-suite of non-convex constrained optimization problems from the real-world and some baseline results. *Swarm and Evolutionary Computation*, *56*, Article 100693.

Kunakote, T., Sabangban, N., Kumar, S., Tejani, G. G., Panagant, N., Pholdee, N., Bureerat, S., & Yildiz, A. R. (2022). Comparative performance of twelve metaheuristics for wind farm layout optimisation. *Archives of Computational Methods in Engineering*, 1–14.

Li, L., Yang, W., Bai, S., & Ma, Z. (2024). KNN-GNN: A powerful graph neural network enhanced by aggregating K-nearest neighbors in common subspace. *Expert Systems with Applications*, Article 124217.

Li, W., Zhang, T., Wang, R., Huang, S., & Liang, J. (2023). Multimodal multi-objective optimization: Comparative study of the state-of-the-art. *Swarm and Evolutionary Computation*, *77*, Article 101253.

Liang, Y., Sun, C., Jiang, J., Liu, X., He, H., & Xie, Y. (2020). An efficiency-improved clustering algorithm based on KNN under ultra-dense network. *IEEE Access*, *8*, 43796–43805.

Lin, L., & Gen, M. (2009). Auto-tuning strategy for evolutionary algorithms: balancing between exploration and exploitation. *Soft Computing*, *13*(2), 157–168.

Liu, G., Zhao, H., Fan, F., Liu, G., Xu, Q., & Nazir, S. (2022). An enhanced intrusion detection model based on improved KNN in WSNs. *Sensors*, *22*(4), 1407.

Mirjalili, S. (2016). SCA: a sine cosine algorithm for solving optimization problems. *Knowledge-Based Systems*, *96*, 120–133.

Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey wolf optimizer. *Advances in Engineering Software*, *69*, 46–61.

Pan, Z., Wang, Y., & Pan, Y. (2020). A new locally adaptive k-nearest neighbor algorithm based on discrimination class. *Knowledge-Based Systems*, *204*, Article 106185.

Qi, L., & Jie, S. (1993). A nonsmooth version of Newton's method. *Mathematical Programming*, *58*(1–3), 353–367.

Rao, R. V., Savsani, V. J., & Vakharia, D. (2011). Teaching–learning-based optimization: a novel method for constrained mechanical design optimization problems. *Computer-Aided Design*, *43*(3), 303–315.

Salgotra, R., Singh, S., Singh, U., Mirjalili, S., & Gandomi, A. H. (2023). Marine predator inspired naked mole-rat algorithm for global optimization. *Expert Systems with Applications*, *212*, Article 118822.

Savsani, V. J., Tejani, G. G., Patel, V. K., & Savsani, P. (2017). Modified meta-heuristics using random mutation for truss topology optimization with static and dynamic constraints. *Journal of Computational Design and Engineering*, *4*(2), 106–130.

Shehadeh, A., Alshboul, O., Al-Shboul, K. F., & Tatari, O. (2024). An expert system for highway construction: Multi-objective optimization using enhanced particle swarm for optimal equipment management. *Expert Systems with Applications*, *249*, Article 123621.

Sigmund, O. (2001). A 99 line topology optimization code written in Matlab. *Structural and Multidisciplinary Optimization*, *21*, 120–127.

Smith, W. A., & Randall, R. B. (2015). Rolling element bearing diagnostics using the case western reserve university data: a benchmark study. *Mechanical Systems and Signal Processing*, *64*, 100–131.

Sun, S., & Huang, R. (2010). An adaptive k-nearest neighbor algorithm. In *2010 seventh international conference on fuzzy systems and knowledge discovery (Vol. 1)* (pp. 91–94). IEEE.

Suykens, J. A. K., & Vandewalle, J. (1999). Least squares support vector machine classifiers. *Neural Processing Letters*, *9*(3), 293–300.

Tanabe, R., & Fukunaga, A. S. (2014). Improving the search performance of SHADE using linear population size reduction. In *2014 IEEE congress on evolutionary computation* (pp. 1658–1665). IEEE.

Uddin, S., Haque, I., Lu, H., Moni, M. A., & Gide, E. (2022). Comparative performance analysis of K-nearest neighbour (KNN) algorithm and its different variants for disease prediction. *Scientific Reports*, *12*(1), 6256.

Urooj, S., Arunachalam, R., Alawad, M. A., Tripathi, K. N., Sukumaran, D., & Ilango, P. (2024). An effective model for network selection and resource allocation in 5G heterogeneous network using hybrid heuristic-assisted multi-objective function. *Expert Systems with Applications*, *248*, Article 123307.

Wang, Y., Liu, H., Long, H., Zhang, Z., & Yang, S. (2017). Differential evolution with a new encoding mechanism for optimizing wind farm layout. *IEEE Transactions on Industrial Informatics*, *14*(3), 1040–1054.

Yang, X., & Deb, S. (2014). Cuckoo search: recent advances and applications. *Neural Computing and Applications*, *24*(1), 169–174.

Yang, X. S., & He, X. (2013). Bat Algorithm: literature review and applications. *International Journal of Bio-Inspired Computation*, *5*(3), 141.

Yang, X. S., & Hossein, A. (2012). Bat algorithm: a novel approach for global engineering optimization. *Engineering Computations*, *29*(5), 464–483.

Zhang, K., Deng, M., & Chen, T. (2002). A dynamic programming algorithm for haplotype block partitioning. *Proceedings of the National Academy of Sciences of the United States of America*, *99*(11), 7335–7339.

Zhao, W., Wang, L., Zhang, Z., Fan, H., Zhang, J., Mirjalili, S., Khodadadi, N., & Cao, Q. (2024). Electric eel foraging optimization: A new bio-inspired optimizer for engineering applications. *Expert Systems with Applications*, *238*, Article 122200.

Zhou, J., Xiahou, T., & Liu, Y. (2021). Multi-objective optimization-based TOPSIS method for sustainable product design under epistemic uncertainty. *Applied Soft Computing*, *98*, Article 106850.