ITC 2/55
Information Technology and Control
Vol. 55 / No. 2/ 2026
pp. 929-952
DOI 10.5755/j01.itc.55.2.44748

CQSGA: A Chaotic and Quantum-Enhanced Snow Geese Algorithm for Complex Optimisation Problems

Received 2026/04/09 | Accepted after revision 2026/06/26

HOW TO CITE: Pei, P. (2026) CQSGA: A Chaotic and Quantum-Enhanced Snow Geese Algorithm for Complex Optimisation Problems. *Information Technology and Control,* 55(2), 929-952. https://doi.org/10.5755/j01.itc.55.2.44748

CQSGA: A Chaotic and Quantum-Enhanced Snow Geese Algorithm for Complex Optimisation Problems

Panhu Pei
School of Internet of Things, Nanjing University of Posts and Telecommunications, Nanjing 210003, China

Corresponding author: B23080511@njupt.edu.cn

This paper presents CQSGA, an enhanced Snow Geese Algorithm that fundamentally differs from existing chaos-enhanced or quantum-inspired methods by synergistically embedding an Exponential Discrete Memristor chaotic map and a quantum rotation gate into SGA's innate dual-phase structure. Unlike approaches that merely inject chaotic perturbations or adopt simplistic qubit encoding, CQSGA leverages the EDM system's superior ergodicity and Lyapunov properties for global exploration, while the quantum rotation gate utilises superposition-driven rotational updates to enable intelligent directional local exploitation. Both mechanisms execute in alternating periodic phases to achieve dynamic exploration-exploitation balance. Rigorous statistical validation via Friedman test ($p < 0.001$) and Nemenyi post-hoc analysis ($\alpha = 0.05$) across 30 independent runs confirms that CQSGA achieves statistically significant improvements over the 14 compared algorithms on the majority of test functions, with competitive performance observed on the remaining cases. CQSGA is further applied to Physics-Informed Neural Network architecture search for solving Burgers equation, achieving sub-millesimal loss (9.9873e-4) and demonstrating superior practical efficacy.

KEYWORDS: Snow Geese Algorithm, Chaotic maps, Quantum rotation gate, PINN architecture search.

1. Introduction

Heuristic algorithms have emerged as indispensable tools for addressing complex optimisation challenges that are often intractable for traditional mathematical methods [31]. Drawing inspiration from biological behaviours [14], physical rules [29], or natural phenomena [16], these algorithms provide robust and flexible frameworks to find satisfactory solutions in domains such as aerodynamic airfoil design [18], structural optimisation of civil engineering trusses [1], and advanced composite material design [22]. The primary appeal of these methods lies in their robustness in exploring intricate search spaces, often discovering effective solutions for complex optimisation problems where traditional methods, such as gradient-based optimisers [20], struggle to locate a global optimum.

This fundamental appeal has led to a diverse family of heuristic algorithms, which are broadly categorised by their foundational source of inspiration. Evolutionary Algorithms (EAs), such as Genetic Algorithms [12], simulate biological evolution through operations like selection, crossover, and mutation. Physics-based algorithms, such as simulated annealing [16] and the gravitational search algorithm [29], are based on the principles of physical processes. Swarm Intelligence (SI) algorithms, including Particle Swarm Optimisation [14] and Ant Colony Optimisation [8], mimic the collective and decentralised intelligence of social organisms, such as birds, ants, or fish. Among these diverse optimisers, the Snow Geese Algorithm (SGA) represents a relatively novel and intriguing one that emulates the distinctive migratory patterns of snow goose flocks. Its core innovation lies in structurally differentiating between two flight modes: the energy-efficient "herringbone" formation for cooperative and broad exploration of the search space, and the straightforward "straight-line" formation for rapid and targeted exploitation of the best-found areas.

SGA has established its competence in handling a range of optimisation challenges. Acknowledging the algorithm's capacity for improvement, researchers have endeavoured to refine its fundamental mechanics using diverse approaches. For instance, to enhance population diversity and mitigate premature convergence, Zhao et al. [37] incorporated adaptive switching and stochastic mechanisms (ESGA), while Bian et al. [4] introduced bio-inspired mechanisms, such as leader rotation and honk guidance (ISGA). Elsewhere, to enhance local search capabilities and adaptability, hybridisation with other computational intelligence techniques, such as deep reinforcement learning, has been explored (RLSGA [19]). Applications have also extended to complex domains, such as power grid optimisation [15] and hyperparameter tuning [30], demonstrating SGA's practical utility. Despite these valuable and concerted efforts, a critical examination reveals two persistent core challenges. The first is the inherent difficulty in achieving a robust and dynamic balance between global exploration and local exploitation. Most existing modifications, while achieving performance gains in specific scenarios, still rely on rule-based adjustments or incorporate additional stochastic components [31]. These approaches often amount to a more sophisticated form of trial-and-error, lacking a theoretically-grounded mechanism to self-adapt the exploration-exploitation trade-off throughout the search process. The second challenge lies in enhancing the algorithm's capability to reliably solve complex real-world problems governed by underlying physical laws, where solution accuracy and convergence robustness are paramount. This fundamental limitation hinders performance and generalizability across diverse and computationally demanding problem landscapes.

To overcome these limitations, this paper proposes a substantially enhanced SGA, called CQSGA, which integrates chaotic mapping and quantum rotation gate strategies to directly tackle the core challenge of balancing global exploration and local exploitation. Unlike existing chaos-enhanced algorithms that merely inject chaotic perturbations without theoretical grounding, or quantum-inspired algorithms that rely on simplistic qubit encoding without exploiting the full potential of quantum state evolution, CQSGA synergistically embeds chaotic mapping and the quantum rotation gate into SGA's innate dual-phase structure. This integration ensures that exploration and exploitation are dynamically adapted throughout the optimisation process, rather than being treated as independent add-ons. Firstly, chaotic mapping is introduced to generate the initial population. Its ergodicity and randomness ensure superior diversity, establishing a solid foundation for global exploration. Secondly, the quantum rotation gate mechanism guides individual position updates. By leveraging quantum superposition principles, it directs local search probabilistically towards elite solutions, replacing random wandering with intelligent, directional exploitation. These strategies are embedded within the main loop of the SGA to work synergistically and periodically, seamlessly aligning with its innate dual-phase structure. The chaotic mapping is applied during the "Herringbone" formation phase in SGA to bolster exploration, and the quantum rotation gate is activated during the "Straight-Line" formation phase in SGA to refine exploitation, ensuring a dynamic balance between exploring new regions and refining promising ones. To validate the performance of CQSGA in complex scenarios, this study conducts experiments across standard benchmark functions and applies it to the automated architecture search for PINNs, a problem involving high-dimensional and discrete search spaces. Extensive experiments on various benchmark functions demonstrate that CQSGA achieves superior performance in terms of solution accuracy. In the practical application of PINN, it also demonstrates improved robustness and effectiveness, confirming its general superiority and practical utility.

The primary contributions of this study are summarised as follows:

1. Propose CQSGA, an enhanced optimizer that effectively balances global exploration and local exploitation.
2. Integrate chaotic mapping to enhance population diversity and quantum rotation gate to guide the population toward optimal solutions.
3. Demonstrate CQSGA's superiority over original and advanced algorithms on CEC 2017/2022 benchmarks.
4. Discretising the proposed CQSGA and applying it to PINN architecture search, yielding models with higher accuracy and robustness.

2. Related Works

2.1. Snow Geese Algorithm

The SGA is a nature-inspired metaheuristic optimiser that mimics the migratory behaviour of snow goose flocks. Its core innovation lies in structurally differentiating between two flight modes: the energy-efficient "herringbone" formation for cooperative and broad exploration of the search space, and the straightforward "straight-line" formation for rapid and targeted exploitation of the best-found areas. Recent years have witnessed significant advancements in SGA through its application across diverse fields. It has been utilised for hyperparameter optimisation in deep learning models for gesture recognition, achieving exceptional accuracy [30], for parameter optimisation in bearing fault diagnosis [41], for power grid optimisation [15], for multi-objective logistics planning using surrogate models (GP-SAMOSGA [36]), and for solving complex economic dispatch problems [27].

In response to the algorithm's inherent limitations, various improvements to the core SGA have been proposed. For instance, to mitigate the sensitivity to initial conditions and enhance diversity, Zhao et al. [37] proposed an ESGA incorporating adaptive switching and stochastic strategies, while Bian et al. [4] introduced bio-inspired mechanisms, such as leader rotation and honk guidance, in their ISGA. Elsewhere, to boost local search capabilities and convergence precision, hybridisation with other computational intelligence techniques has been explored. Notably, Liu et al. [19] fused SGA with deep reinforcement learning (RLSGA) to guide agent behaviour in robot calibration, and Wu et al. [36] employed surrogate models (GP-SAMOSGA) for complex logistics planning. Meanwhile, Fu et al. [9] set a high-performance benchmark with their novel RBMO algorithm, against which subsequent optimisers are often compared.

Despite these advancements, SGA and its variants face several persistent challenges. The algorithm exhibits significant sensitivity to initial conditions, where suboptimal population initialisation has a severe impact on performance. Furthermore, its basic structure lacks refined local search mechanisms, limiting precision in complex search spaces. Existing improvements also introduce new issues: adaptive strategies often rely on simplistic parameter tuning, heuristic diversity mechanisms lack a mathematical foundation, and hybrid approaches substantially increase computational complexity. The fundamental challenge remains achieving an effective balance between global exploration and local exploitation.

2.2. Physics-Informed Neural Network

PINNs have arisen as a formidable paradigm for solving PDEs, achieved by directly encoding the governing physical laws into the neural network's loss function. This inherent capability to seamlessly integrate data and mathematical models has sparked substantial research interest, leading to numerous enhancements aimed at improving their accuracy and applicability across various scientific domains.

Significant research efforts have yielded notable improvements and applications. Leung et al. [17] proposed the Neural Homogenization-based PINN (NH-PINN), a three-step method that leverages mathematical homogenization theory to tackle multiscale problems, significantly improving accuracy. Chiu et al. [7] introduced the coupled-automatic-numerical-differentiation PINN (can-PINN), which unifies automatic differentiation and numerical differentiation to compute loss functions, enabling robust training with sparse data. Meng et al. [23] combined PINN with the First-Order Reliability Method to create PINN-FORM, a black-box solver for structural reliability analysis with implicit PDEs. In terms of application, Raissi et al. [28] employed multiple coupled deep neural networks to solve challenging inverse fluid-structure interaction problems, accurately inferring parameters from minimal data. A comprehensive review by Cai et al. [6] further demonstrated the effectiveness of PINNs for inverse problems in various complex flows.

Despite their ingenuity in addressing specific challenges, such as multiscale features, derivative computation, and inverse problems, a critical standard limitation pervades these works: a pronounced neglect of neural architecture search (NAS) for the PINNs themselves. The improvements proposed by Leung et al. [17], Meng et al. [23], and Raissi et al. [28] are primarily algorithmic and mathematical, focusing on pre-processing, hybrid loss definitions, or multi-network coupling. They operate under the implicit assumption that a standard, often arbitrarily chosen, fully-connected network architecture is sufficient. Similarly, Chiu et al. 's [7] can-PINN focuses on loss calculation but does not question architectural optimality. The review by Cai et al. [6] also does not identify architecture selection as a key bottleneck. This oversight is significant because a suboptimal architecture can severely hinder the network's ability to learn complex physical phenomena, regardless of how sophisticated the training strategy or loss function may be. Consequently, these methods often require inefficient manual hyperparameter tuning and may fail to achieve complete potential accuracy. This work addresses this notable gap by introducing a neural architecture search method for PINNs, which automates the optimisation of network structure to better align with physical laws.

3. The Proposed CQSGA

This section provides a comprehensive elaboration of our proposed CQSGA. It begins with a concise review of the core principles of the original SGA, which serves as the foundation for our enhancements. Subsequently, we detail two key mechanisms: the chaotic mapping, introduced to enhance the initialisation quality and achieve compelling global exploration, and the quantum rotation gate operation, employed to refine the algorithm's local exploitation. Following that, these two mechanisms are intricately integrated with the SGA algorithm, presenting the complete steps of the proposed CQSGA. Then, a computational complexity analysis is conducted. Finally, we apply the CQSGA to a practical scenario, namely the architecture search for PINN.

3.1. An Overview of SGA

This section briefly outlines the original SGA [33], a metaheuristic algorithm derived from natural phenomena, which emulates the long-distance migratory habits of snow goose flocks. The algorithm achieves optimisation search by simulating two energy-efficient flight formations adopted by the flock: the "Herringbone" formation and the "Straight Line" formation. These behavioural modes adaptively switch based on a dynamic angle parameter tied to the iteration count. The algorithm begins by generating the positions and velocities of the snow goose population within the search space. The fitness value of each candidate solution goose is evaluated to identify the initial global best solution. During optimisation, the algorithm switches between two modes based on the phase angle. In the "Herringbone" exploration phase, a fitness-rank-based stratified update strategy is employed: the top 20% of best-performing individuals move directly toward the global leader, the middle 60% are influenced by a combination of the leader and the population center, while the worst-performing 20% additionally incorporate an avoidance mechanism towards the worst solution, thereby enhancing diversity and preventing premature convergence. Upon entering the "Straight Line" exploitation phase, the algorithm randomly selects one of two strategies with a 50% probability: one simulates individuals following a randomly chosen stronger companion, and the other introduces Brownian motion, a type of random walk, to apply stochastic perturbations to the current positions, which helps the algorithm escape local optima.

Despite the commendable design of SGA, it still faces challenges in maintaining a robust balance between its search strategies. To address this fundamental limitation, we introduce two key mechanisms: chaotic mapping and the quantum rotation gate.

3.2. Chaotic Maps

The original SGA suffers from poor initialisation quality and inadequate global exploration capability. To overcome these limitations, we introduce a chaotic mapping strategy based on the Exponential Discrete Memristor (EDM) system, a nonlinear dynamical system derived from memristor circuit theory. A controlled experiment was conducted to quantitatively assess the contribution of chaotic initialisation: SGA (without the quantum rotation gate) was tested with EDM, Logistic map, Tent map, and uniform random initialisation on six standard 50D benchmark functions over 30 independent runs. As shown in Table 1, EDM achieves the best average fitness with a Friedman rank of 1, significantly outperforming the Logistic map (rank 3) and Tent map (rank 2), with the Friedman test confirming statistical significance ($p = 0.00044$). This mechanism enhances both the population initialisation process and global exploration capability by replacing conventional random generation with deterministic chaotic sequences. It transforms the initialisation from a simple stochastic process into a systematic exploration of the search space through ergodic chaotic dynamics. The core principle involves generating chaotic sequences using Equation 1:

$$\begin{aligned} x_{n+1} &= k\cdot(exp(-cos(\pi\cdot y_n))-1)\cdot x_n \\ y_{n+1} &= y_n + x_n \end{aligned} \tag{1}$$

where k=2.66 is the control parameter of the EDM system, x_n and y_n represent the current state variables, x_{n+1} and y_{n+1} denote the updated state variables. For population initialisation, the chaotic sequences are mapped to the actual search space through linear transformation, as shown in Equation (2):

$$P_i = L + (U - L)\cdot\frac{x_{chaos}+1}{2}, \tag{2}$$

where P_i represents the initial position of individual i, L and U define the lower and upper bounds of the search space, x_{chaos} is the final value from the chaotic sequence after multiple iterations.

During the optimisation process, chaotic perturbations are dynamically injected to maintain diversity, as shown in Equation (3):

$$chaos_{term} = \frac{x_{chaos}-L}{U-L}\cdot(up_{chaos}-low_{chaos})+low_{chaos}, \tag{3}$$

where $chaos_{term}$ represents the chaotic perturbation vector added to the position update of individuals, up_{chaos} and $low_{chaos}\in R^D$ are the upper-bound and lower-bound chaotic pattern vectors, respectively. These vectors are generated by mapping chaotic sequences to the search space bounds as follows:

$$\begin{aligned} up_{chaos} &= X_{max}\odot\varkappa_{upper} \\ low_{chaos} &= X_{min}\odot\varkappa_{lower} \end{aligned}$$

where $X_{max}, X_{min}\in R^D$ denote the upper and lower bounds of the search space, respectively. $\varkappa_{upper}$, $\varkappa_{lower}\in[0,1]^D$ are D-dimensional chaotic sequences generated by the logistic map $z_{d+1}=4z_d(1-z_d)$ with different initial seeds $z_0^{up}=0.7$ and $z_0^{low}=0.3$, and $\odot$ denotes the element-wise product.

The alternating application of these two bound vectors creates diverse perturbation directions across different dimensions.

The chaotic mapping mechanism effectively enhances both the initialisation quality and global exploration capability of SGA. Ensuring comprehensive coverage of the search space through ergodic chaotic sequences, it provides systematic diversity maintenance throughout the optimisation process, preventing premature convergence while enhancing the algorithm's performance in complex optimisation landscapes.

Table 1
Comparison of chaotic initialization methods.

Fun	Item	EDM	Logistic	Tent	Random
Sphere	AVG	1.60E+00	4.78E+02	4.01E+02	9.55E+02
	STD	1.54E+00	3.62E+02	3.74E+02	7.02E+02
Rastrigin	AVG	9.67E-01	7.35E+01	5.83E+01	8.17E+01
	STD	1.15E+00	2.53E+01	2.20E+01	3.12E+01
Ackley	AVG	6.59E-01	9.84E+00	9.27E+00	1.18E+01
	STD	5.71E-01	1.91E+00	1.62E+00	2.17E+00
Griewank	AVG	6.83E-01	5.52E+00	4.45E+00	8.09E+00
	STD	2.71E-01	3.15E+00	3.32E+00	6.75E+00
Rosenbrock	AVG	4.94E+01	1.08E+05	9.03E+04	3.78E+05
	STD	1.85E+01	1.97E+05	1.97E+05	8.54E+05
Zakharov	AVG	8.62E-02	2.65E+01	1.96E+01	5.95E+01
	STD	9.84E-02	2.37E+01	1.21E+01	5.45E+01
AVG		8.90E+00	1.81E+04	1.51E+04	6.32E+04
Rank		1	3	2	4

3.3. Quantum Rotation Gate

The original SGA exhibits inherent limitations in its local exploitation capability. To overcome this drawback, we introduce a quantum rotation gate strategy. This mechanism refines the local search behaviour by replacing the stochastic updates with a series of precise, quantum-inspired operations. It transforms the solution update process from a random perturbation into a guided, rotational adjustment toward elite regions.

The quantum rotation gate strategy establishes a rigorous theoretical connection between quantum state evolution and search space exploration. In this framework, each individual solution is represented as a quantum bit (qubit) in Hilbert space, analogous to a quantum superposition state $|\psi\rangle = cos\theta|0\rangle + sin\theta|1\rangle$, where $|0\rangle$ and $|1\rangle$ are the computational basis states and $\theta\in[0,\pi/2]$ is the phase angle encoding the solution information. The rotation gate performs a unitary transformation $R(\Delta\theta)$ that rotates the quantum state toward the optimal direction. This rotation mimics the quantum mechanical process of state evolution, where the rotation angle $\Delta\theta$ (defined in Equation (5)) controls the magnitude of state change: the adaptive scaling factor $s(\alpha)$ decreases the step size as iterations progress (analogous to quantum adiabatic evolution), while the stochastic component $\Delta\varphi$ enables probabilistic tunnelling through local optima. The position update in Equation (6) then corresponds to wave-function collapse guided by the global best solution, where the term $(cos\theta - sin\theta) \in [-1, 1]$ acts as a signed contraction factor controlling the update direction and magnitude. This formulation ensures that local exploitation is guided probabilistically toward elite regions while maintaining population diversity through the superposition principle.

The core principle involves representing the position of an individual solution as a quantum bit in Hilbert space, as defined in Equation (4):

$$|\psi_i^t\rangle = cos\,\theta_i^t|0\rangle + sin\,\theta_i^t|1\rangle, \tag{4}$$

where $|\psi_i^t\rangle$ is the quantum state of individual i at iteration t, $|0\rangle$ and $|1\rangle$ are the computational basis states of the qubit, $\theta_i^t \in [0,\pi/2]$ is the phase angle that encodes the solution information.

The rotation gate dynamically adjusts quantum states through unitary transformation, as shown in Equation (5):

$$\begin{bmatrix} cos\,\theta_i^{t+1} \\ sin\,\theta_i^{t+1} \end{bmatrix} = \begin{bmatrix} cos\,\Delta\theta & -sin\,\Delta\theta \\ sin\,\Delta\theta & cos\,\Delta\theta \end{bmatrix} \begin{bmatrix} cos\,\theta_i^{t} \\ sin\,\theta_i^{t} \end{bmatrix}, \tag{5}$$

where θ_i^t is the current phase angle of the i-th individual at iteration t, ranging in $[0,\pi/2]$, θ_i^{t+1} is the updated phase angle of the i-th individual at iteration $t+1$. The rotation angle is computed as $\Delta\theta = s(\alpha)\cdot\Delta\phi$, where $\Delta\phi = (\pi/4)\cdot N(0,1)$ is the basic rotation magnitude with $N(0,1)$ being a random number drawn from the standard normal distribution, and $s(\alpha) = 0.1\cdot(1 - t/T)\cdot\alpha$ is the adaptive scaling factor that linearly decreases the step size as iterations progress. Here, T is the maximum iteration number, and $\alpha = 1.0$ is the control parameter that modulates the exploration-exploitation trade-off.

Finally, positions are updated through quantum state collapse, as shown in Equation (6):

$$P_i^{t+1} = P_b^t + \beta\cdot(cos\,\theta_i^{t+1} - sin\,\theta_i^{t+1}), \tag{6}$$

where β=0.5 is the contraction factor, P_i^{t+1} is the updated position of individual i at iteration $t+1$, P_b^t is the global best position found up to iteration t. The term $cos\theta_i^{t+1} - sin\theta_i^{t+1}$ acts as a signed direction indicator: when $\theta_i^{t+1} < \pi/4$ the individual moves toward P_b (exploitation), when $\theta_i^{t+1} > \pi/4$ it explores away from P_b, and when $\theta_i^{t+1} = \pi/4$ the update is zero, allowing the population to stabilize around promising regions.

The quantum rotation gate mechanism effectively enhances SGA's local search precision by incorporating principles of quantum superposition. It enables fine-grained directional updates that significantly improve convergence accuracy while preserving the global search capabilities of the original algorithm.

3.4. CQSGA

Having detailed the two core enhancement mechanisms in the preceding sections, this section now provides a comprehensive elaboration of the complete CQSGA workflow. CQSGA is an enhanced nature-inspired metaheuristic optimisation algorithm that builds on SGA by integrating chaotic mapping and QRG mechanisms. Unlike existing approaches that apply chaotic mapping and quantum operators as independent post-hoc modifications, CQSGA's two mechanisms are intrinsically coupled with SGA's dual-phase structure: chaotic perturbations are active during the herringbone formation (exploration), while quantum rotations refine solutions during the straight-line phase (exploitation). This phase-locked synergy ensures that exploration and exploitation are not merely balanced but dynamically co-adapted throughout the search process. The algorithm operates through three core phases: chaotic initialisation of snow geese positions and velocities, dynamic position updates with chaotic perturbations, and quantum-inspired refinement via rotational transformations. The complete algorithm workflow is summarised in Algorithm 1. The CQSGA flowchart is intuitively illustrated in Figure 1.

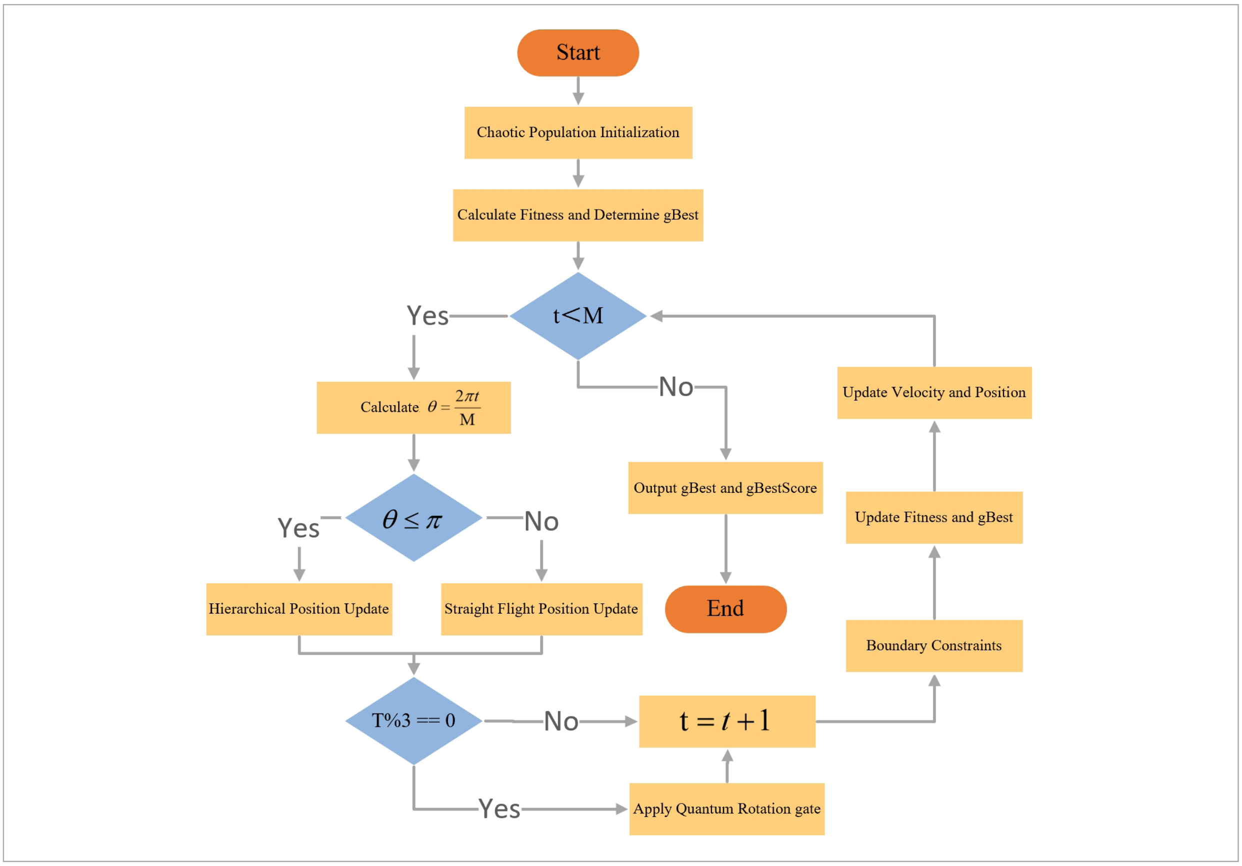
Algorithm 1 Pseudo-code of CQSGA

Require: Population size n, Max iterations M, Problem dimension d, Search bounds x_{min}, x_{max}

Ensure: Global best solution $gBest$, Best fitness $fitness_{best}$

```
P, V ← ChaoticInitialization(n, d, x_min, x_max)  ▷ P is the position matrix, V is the velocity matrix
 fitness ← EvaluateFitness(P)
 gBest ← P[argmin(fitness)], fitness_best ← min(fitness)
 For t = 1 to M do              ▷ t is the current iteration
     θ ← 2πt/M                  ▷ θ is the phase angle
     if θ < π then
         P ← HerringbonePhase(P, V, gBest)
     else
         P ← StraightLinePhase(P, gBest, t, M)
     end if
 fitness ← EvaluateFitness(P)
 Update gBest and fitness_best if improved
 if t mod 3==0 then
     P ← QuantumRotationGate(P, gBest, t, M)
 endif
endfor
```

Figure 1
The flowchart of CQSGA.



The algorithm employs chaotic mapping detailed in Section 3.2 to generate and maintain the population's positions P and velocities V, replacing the random initialization of the original SGA, as represented in Equation (7):

$$P = \begin{bmatrix} p_{1,1} & p_{1,2} & \cdots & p_{1,d} \\ p_{2,1} & p_{2,2} & \cdots & p_{2,d} \\ \vdots & \vdots & \ddots & \vdots \\ p_{n,1} & p_{n,2} & \cdots & p_{n,d} \end{bmatrix}, \quad V = \begin{bmatrix} v_{1,1} & v_{1,2} & \cdots & v_{1,d} \\ v_{2,1} & v_{2,2} & \cdots & v_{2,d} \\ \vdots & \vdots & \ddots & \vdots \\ v_{n,1} & v_{n,2} & \cdots & v_{n,d} \end{bmatrix} \tag{7}$$

Here, $P\{i,j\}$ denotes the j-th dimensional coordinate of individual i's position, and $V\{i,j\}$ denotes the corresponding velocity component used during the herringbone phase. Both matrices are of size $n \times d$, where n is the population size and d is the problem dimension.

During the herringbone phase, geese adopt a V-shaped formation where position updates are enhanced with chaotic perturbations and stratified by fitness rank. The update rules for different ranks are as follows: The top 20% follow the global leader with additional chaotic guidance, the middle 60% balance attraction to the leader and repulsion from the group centre through chaotic crossover operations, and the weakest 20% incorporate velocity-based updates with chaotic elements. The position updates are detailed in Section 3.2. This rank-driven strategy, enhanced with chaotic dynamics, significantly improves diversity and prevents premature convergence. During the straight-line phase, when the transition angle $\theta = \frac{2\pi t}{M}$ exceeds π, geese switch to linear formation. With 50% probability, they either follow strong peers with chaotic perturbations or perform quantum rotation gate operations. The update of the quantum rotation gate is governed by Equations (5)-(6) detailed in Section 3.3.

To further bolster exploitation capabilities, the quantum rotation gate operation is periodically applied every three iterations across the entire population, irrespective of the current phase. It ensures continuous refinement of solutions throughout the optimisation process.

3.5. Computational Complexity Analysis

The computational complexity of CQSGA is analyzed theoretically and empirically. In terms of time complexity, the main components include population initialization, fitness evaluation, position updates, and periodic quantum rotation operations. Initialization using chaotic mapping requires $O(N{\cdot}d{\cdot}C)$ operations, where N is the population size, d is the problem dimension, and C is a small number of chaotic iterations. Each iteration performs either the herringbone phase or the straight-line phase, both with complexity $O(N{\cdot}d)$. The quantum rotation gate, applied every three iterations, adds an additional $O(N{\cdot}d{\cdot}Q)$ overhead, where Q is the cost of computing the rotation matrix (a constant factor). The chaotic perturbation term adds $O(N{\cdot}d)$ per iteration. Therefore, the overall time complexity per iteration remains $O(N{\cdot}d)$, and the total complexity over T iterations is $O(T{\cdot}N{\cdot}d)$, which is of the same order as the original SGA. The constant factors introduced by chaotic mapping and quantum rotation operations increase the practical runtime by approximately 24.3% compared to the original SGA, as confirmed by the runtime measurements in Section 4, but this moderate increase is well justified by the significant performance gains achieved.

3.6. PINN Architecture Search

To further demonstrate the practical utility of CQSGA, we apply it as an optimiser to the process of PINN architecture search. PINN [6] have established itself as a formidable methodology for solving PDEs, achieved by directly encoding the governing physical laws into the loss function of a neural network. While significant progress has been made, research into PINN variants has highlighted the critical role of network architecture design in enhancing their accuracy, convergence speed, and robustness for complex problems. PINN architecture search is a discrete optimisation problem where neural network structural parameters must be represented as discrete values. However, CQSGA was initially designed for continuous space optimisation. To apply CQSGA to PINN architecture search problems, a transformation mechanism is required to discretise the continuous search space.

The continuous chaotic map is not suitable for discrete optimisation. We employ a discrete tent map for population initialisation and local perturbation, which is defined in Equation (8):

$$x_{chaos} = \begin{cases} 2\cdot x_{int} & if\ x_{int} < \lfloor 0.5\cdot M \rfloor \\ 2\cdot(M - x_{int}) & otherwise \end{cases}, \tag{8}$$

where b is the bit width, $M=2^b–1$ is the maximum integer value, x_{chaos} is clipped to the range [0,M] and x_{int} represents the current integer-encoded position value.

The continuous rotation operation is redefined as a probabilistic bit-flip, acting on the binary representation of integer-encoded solutions. The flip probability is defined in Equation (9):

$$P_{flip} = 1 - e^{-10\cdot\varphi}, \quad \varphi = 0.1\cdot(1 - t/T), \tag{9}$$

where P_{flip} is the probability of flipping a single bit in the binary encoding, φ is a dynamic control parameter that decreases linearly with iterations, analogous to an annealing temperature.

We employ an integer encoding scheme that maps the continuous position of a snow goose to a discrete integer using Equation (10):

$$x_{int} = \left\lfloor \frac{x_{real} - L}{U - L}\cdot(2^b - 1) \right\rfloor, \tag{10}$$

where x_{real} is a coordinate in the continuous search space, L and U are the lower and upper bounds of that dimension, b is the bit width.

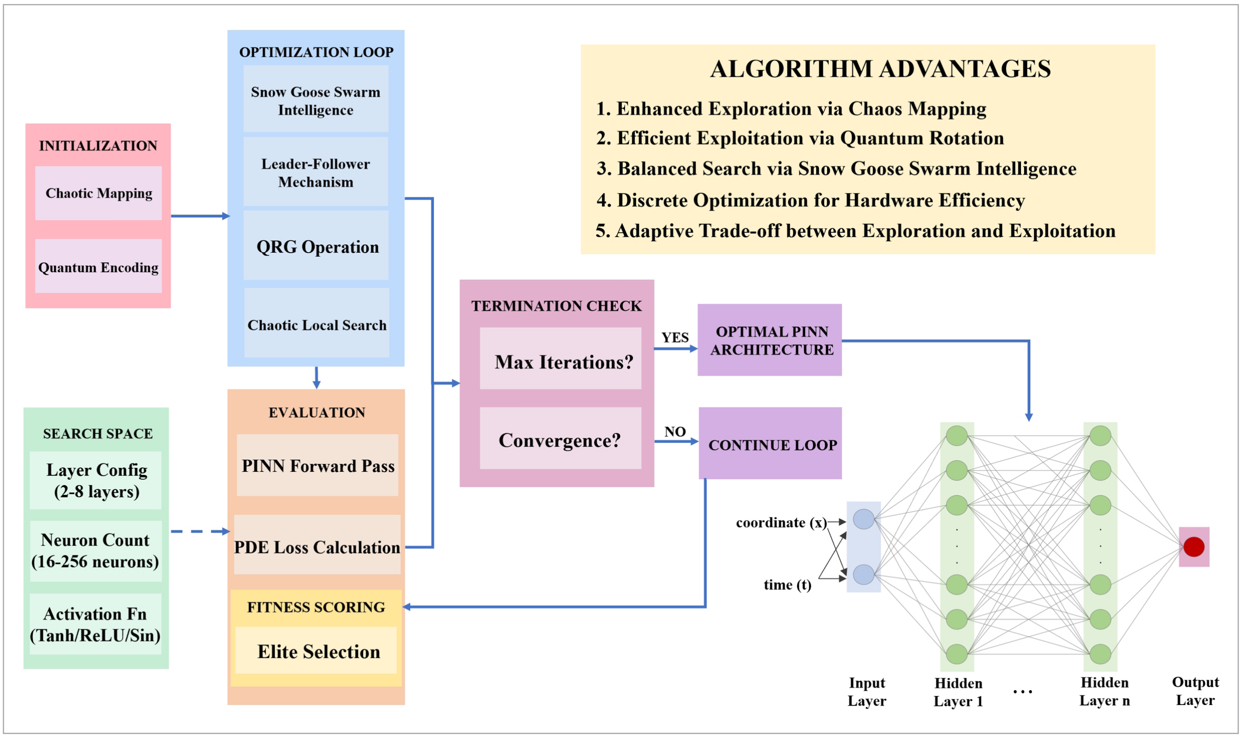
The fitness function for PINN is defined in Equation (11):

$$F = \omega L_{PDE} + (1-\omega) L_{BC}, \tag{11}$$

where L_{PDE} denotes the PDE residual loss, L_{BC} represents the boundary condition loss, and ω is a balancing weight.

Figure 2

Workflow of CQSGA for PINNs.



Algorithm 2 Discrete CQSGA for PINN Architecture Search

Input: PDE, Domain, boundary conditions BC, Population size N, Max iterations T
Output: Optimized PINN parameters θ^*
Procedure Discrete CQSGA
  Initialize population P of N individuals with discrete chaotic mapping
  best $\leftarrow \varnothing$ $best_{fit} \leftarrow \infty$ ▷ best is the best fitness value, best_fit is the best PINN architecture
  for $t = 1$ to T **do**
    $\omega \leftarrow 0.5(1+\cos(\pi t/T))$ ▷ ω is a balancing weight for PDE and BC losses
    $new_{pop} \leftarrow \varnothing$ ▷ new_{pop} is an updated population with new candidates
    for each individual x_i in P **do**
      if exploitation phase **then**
        $x_{new} \leftarrow$ *Chaos mapping*(x_i) ▷ x_new is a new candidate architecture
      elseif crossover phase **then**
        $x_{new} \leftarrow$ *Differential evolution* (x_i, P)
    else
        $x_{new} \leftarrow$ *Lévy jump*(x_i)
    endif
      $new_{pop} \leftarrow new_{pop} \cup x_{new}$
  endfor
  if t *mod rotation interval*$=0$ **then**

$new_{pop} \leftarrow Quantum\ rotation\ (new_{pop})$
endif
Evaluate fitness of $P \cup new_{pop}$ via PDE residual loss
$P \leftarrow Select\ top\ (P \cup new_{pop}, N)$
Update *best* and $best_{fit}$
endfor
return Decode(best)
end procedure

Although the representations and core operators are discretised, the overarching iterative process of CQSGA remains consistent with its continuous counterpart. The algorithm retains its fundamental dual-phase structure, fitness-based stratified updates, and adaptive switching mechanism. The pseudo-code for the PINN search is presented in Algorithm 2. Figure 2 delineates the PINN architecture search workflow, comprising five key components. The initialisation phase employs chaotic mapping and quantum encoding. The optimisation loop executes snow goose swarm operations, including leader-follower updates, QRG fine-tuning, and chaotic local search. The search space defines architectural parameters. The evaluation phase computes PDE loss through forward passes and fitness scoring. The termination check outputs the final architecture when the criteria are met.

4. Experimental Results and Analysis

This section presents comprehensive experiments designed to validate the effectiveness and demonstrate the superiority of CQSGA. Specifically, Section 4.1 introduces the experimental setup. Section 4.2 compares the performance of CQSGA against multiple original and advanced algorithms on the IEEE CEC 2017 [35] and CEC 2022 [21] benchmark suites. Necessary significance tests and runtime analysis have been conducted. Section 4.3 conducts an ablation study to dissect the individual contributions of the key components (chaotic mapping and QRG) within CQSGA. Section 4.4 investigates the sensitivity of CQSGA to its critical parameters. Finally, Section 4.5 applies CQSGA to the PINN architecture search for solving the Burgers equation, showcasing its practical efficacy in a complex real-world problem. The entire suite of experiments was coded in Python and performed on a workstation utilising the Windows 11 operating system, which featured an Intel Core i7 CPU and 32 GB RAM.

4.1. Experimental Setup

Benchmark Functions: This section employs IEEE CEC 2017 and CEC 2022 benchmark function sets. CEC 2017 includes 30 functions: F1-F3 (unimodal with single global optimum), F4-F10 (basic multimodal with multiple local optima), F11-F20 (hybrid compositions), and F21-F30 (nonlinear composite functions). CEC 2022 features 12 functions: F1 (unimodal), F2-F5 (multimodal), F6-F8 (hybrid), and F9-F12 (composition) with enhanced challenges like asymmetric search spaces and variable linkages.

Compared Algorithms: To rigorously evaluate the core optimisation capability of our proposed algorithm, CQSGA, comparative experiments were conducted against established metaheuristics. This includes the original Snow Geese Algorithm (SGA), seven original algorithms (PSO [14], GA [11], DE [32], GWO [24], HHO [10], WOA [26], SA [16]), and five advanced algorithms (AEO [39], Augmented-AEO [38], MRFO [40], ES [2], SCA [25]).

Experimental Configuration: Evaluations were performed on the CEC 2017 benchmark at a dimension of 50 and the CEC 2022 benchmark at a dimension of 20. The parameter configuration for CQSGA was fixed as follows: population size = 50, maximum iterations = 1500, and 30 independent runs. All experiments were conducted with fixed random seeds (1–30) to ensure reproducibility. The parameter settings for all comparison algorithms are provided in Tables 2-3 to maintain consistency. They follow the default values recommended in their original publications. No additional tuning was performed to ensure a fair comparison.

Table 2

Basic Algorithm Parameter Settings.

Algorithm	Parameter Values
SGA	CR=0.7
WOA	b=1.0
PSO	c_1=2, c_2=2, w=0.9-0.5×$\frac{t}{T}$
DE	F=0.8, CR=0.9
GA	Pc=0.95, Pm=0.05
GWO	a=2-2×$\frac{t}{T}$
HHO	E=1.5-1.5×$\frac{t}{T}$
SA	T_0=1000, α=0.95

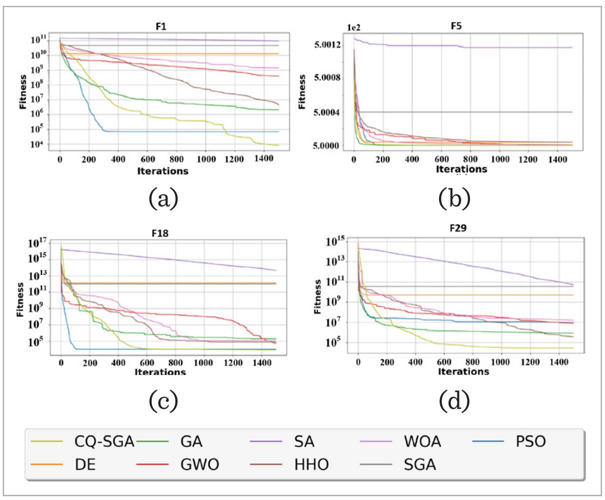
Table 3

Advanced Algorithm Parameter Settings.

Algorithm	Parameter Values
CQSGA	$chaos_N$=3, $base_{angle}$=0.006, CR=0.95×(0.5+0.5×$\frac{t}{T}$)
AEO	δ=0.7, λ=1.5
Aug-AEO	θ=0.85, γ=2.0
MRFO	α=0.8, z_{value}=0.02
ES	γ=0.85, strategy=1
SCA	β=0.3, w=1.2

Figure 3

Convergence curves of CQSGA and original algorithms on specific CEC 2017(50D) functions: (a) F1, (b) F5, (c) F18, (d) F29.



4.2. Comparative Algorithm Performance Analysis

To provide a comprehensive assessment of CQSGA, this section presents a comparative performance analysis on high-dimensional benchmark suites. To rigorously evaluate scalability, we test CQSGA on CEC 2017 at 50 dimensions and CEC 2022 at 20 dimensions.

Comparison with Original Algorithms: Table 4 and Figure 3 present experimental results and the fitness convergence curves of several specific functions, respectively. CQSGA exhibited competitive performance in 50D problems. On F18, its optimal solution (7.44E+03) surpassed suboptimal PSO (7.73E+03) by 3.8%. For F29, it led with 1.69E+05, exceeding PSO (3.02E+05) by 44.0%. Although matching PSO in mean value (5.00E+02) for F1, its STD (5.84E-05) was 1.8% lower than suboptimal GA (5.95E-05). Synthetically, CQSGA's overall average (3.66E+04) was 97.4% lower than that of the runner-up GA (1.41E+06), while GA's average was significantly higher. Table 5 and Figure 4 present experimental results and the fitness convergence curves of several specific functions, respectively. In 20D tests, CQSGA achieved the optimal solution for F2 (4.11 × 10^2), surpassing the suboptimal PSO (4.19 × 10^2) by 1.9%. For F7, it led with 1.82E+03, exceeding PSO (1.99E+03) by 8.5%. Although matching GA's mean for F11 (2.60E+03), its STD (1.86E-02) was 56.7% better than GA (4.30E-02). The overall average (3.62E+03) was 37.9% below runner-up GWO (5.83E+03), while GWO's average was 61.2% higher.

Figure 4

Convergence curves of CQSGA and original algorithms on specific CEC 2022(20D) functions: (a) F1, (b) F2, (c) F7, (d) F11.

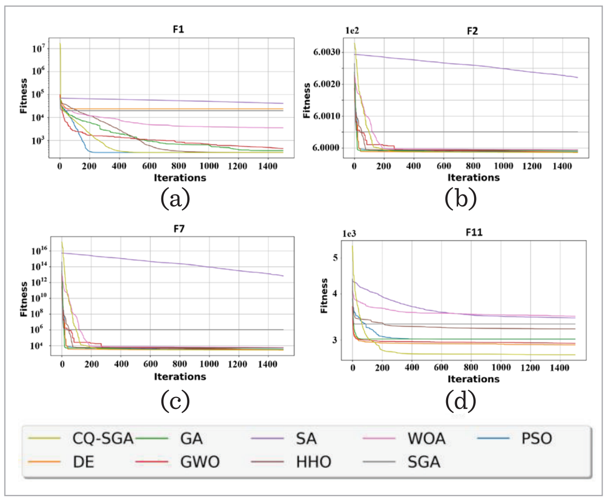


Table 4
The AVG and STD of CQSGA and original algorithms on CEC 2017 specific functions(50D).

Fun	Item	CQSGA	SGA	PSO	GA	DE	GWO	HHO	WOA	SA
F1	AVG	1.33E+04	1.07E+11	2.32E+08	3.27E+07	6.70E+10	3.79E+09	7.36E+07	1.63E+10	1.90E+11
	STD	1.25E+03	1.05E+10	1.98E+08	8.50E+06	2.27E+10	2.41E+09	2.06E+07	2.90E+09	1.77E+10
F5	AVG	5.00E+02	5.00E+02	5.00E+02	5.00E+02	5.00E+02	5.00E+02	5.00E+02	5.00E+02	5.00E+02
	STD	5.84E-05	1.56E-02	7.78E-04	5.95E-05	2.85E-03	1.20E-04	1.55E-03	2.85E-03	2.31E-02
F18	AVG	7.44E+03	7.95E+14	7.73E+03	3.41E+05	9.06E+12	4.96E+09	1.86E+06	3.63E+07	1.24E+15
	STD	1.87E+03	6.00E+14	1.93E+03	1.76E+05	1.32E+13	4.77E+09	2.06E+06	5.23E+07	1.47E+15
F29	AVG	1.69E+05	4.11E+14	3.02E+05	1.43E+06	2.76E+11	7.61E+08	1.62E+08	1.93E+09	2.16E+14
	STD	6.59E+04	2.37E+14	1.95E+05	1.85E+05	1.59E+11	1.25E+09	1.87E+08	1.60E+09	1.73E+14
AVG		3.66E+04	4.10E+13	8.30E+06	1.41E+06	3.21E+11	3.24E+08	1.50E+07	6.64E+08	5.48E+13
Rank		1	8	3	2	7	5	4	6	9
%		-	1.12E+09	2.26E+02	3.73E+01	8.75E+06	8.85E+03	4.08E+02	1.81E+04	1.50E+09

Table 5
The AVG and STD of CQSGA and original algorithms on CEC 2022 specific functions(20D).

Fun	Item	CQSGA	SGA	PSO	GA	DE	GWO	HHO	WOA	SA
F1	AVG	3.00E+02	1.98E+04	3.00E+02	3.60E+02	2.38E+04	4.49E+02	3.00E+02	3.55E+03	4.10E+04
	STD	1.38E-09	2.92E+03	1.81E-09	5.22E+01	9.08E+03	1.65E+02	1.81E-01	2.92E+03	2.05E+03
F2	AVG	4.11E+02	1.75E+03	4.19E+02	4.62E+02	7.03E+02	4.59E+02	4.57E+02	5.06E+02	3.12E+03
	STD	3.68E+00	8.64E+02	2.66E+01	2.49E+01	7.45E+01	7.87E+00	1.02E+01	3.17E+01	8.66E+02
F7	AVG	1.82E+03	4.08E+03	1.99E+03	2.03E+03	2.12E+03	2.00E+03	2.45E+03	3.30E+03	7.18E+03
	STD	6.92E+01	2.94E+02	1.07E+02	7.82E+00	1.81E+02	8.74E+01	5.43E+02	2.95E+02	1.06E+03
F11	AVG	2.60E+03	7.34E+03	2.60E+03	2.60E+03	3.47E+03	2.60E+03	2.65E+03	2.79E+03	1.04E+04
	STD	1.86E-02	1.05E+03	2.29E+00	4.30E-02	1.38E+03	1.59E+00	5.80E+01	2.74E+02	3.39E+03
AVG		3.62E+03	2.93E+08	6.18E+03	1.01E+04	1.19E+06	5.83E+03	6.44E+03	6.61E+03	5.75E+11
Rank		1	8	3	6	7	2	4	5	9
%		-	8.11E+04	7.08E-01	1.78E+00	3.27E+02	6.12E-01	7.81E-01	8.28E-01	1.59E+08

Comparison with Advanced Algorithms: Table 6 and Figure 5 present experimental results and the convergence curves of fitness for several specific functions, respectively. CQSGA achieved better results relative to the advanced algorithms under comparison in 50D. On F11, its optimal solution (1.74E+05) outperformed suboptimal Aug-AEO (3.78E+05) by 54.0%. For F25, it matched ES in mean (3.89E+03) but exhibited a 30.7% better standard deviation (9.36E+01 vs. 1.35E+02). Synthetically, the average (4.47E+04) was 53.3% below Aug-AEO (9.57E+04), while Aug-AEO's average was 114%

Table 6

The AVG and STD of CQSGA and advanced algorithms on CEC 2017 specific functions(50D).

Fun	Item	CQSGA	AEO	Aug-AEO	MRFO	ES	SCA
F3	AVG	4.52E+02	1.73E+03	4.53E+02	1.25E+04	4.41E+03	2.18E+04
	STD	4.84E+01	5.13E+02	6.32E+01	1.24E+04	1.31E+01	1.02E+04
F5	AVG	5.00E+02	5.00E+02	5.00E+02	5.00E+02	5.00E+02	5.00E+02
	STD	3.95E-04	4.37E-03	4.49E-04	5.06E-04	3.48E-03	3.31E-03
F11	AVG	1.74E+05	2.96E+08	3.78E+05	7.77E+09	1.50E+07	1.15E+10
	STD	2.84E+04	1.78E+08	2.59E+05	2.14E+09	1.11E+07	5.72E+08
F25	AVG	3.89E+03	1.74E+04	4.14E+03	4.74E+04	3.89E+03	1.77E+04
	STD	9.36E+01	8.65E+03	1.06E+01	2.09E+04	1.35E+02	2.72E+03
AVG		4.47E+04	7.40E+07	9.57E+04	1.94E+09	3.75E+06	2.87E+09
Rank		1	4	2	5	3	6
%		-	1.65E+03	1.14E+00	4.34E+04	8.29E+01	6.43E+04

Table 7

The AVG and STD of CQSGA and advanced algorithms on CEC 2022 specific functions(20D).

Fun	Item	CQSGA	AEO	Aug-AEO	MRFO	ES	SCA
F1	AVG	3.00E+02	1.27E+03	3.00E+02	3.00E+02	3.00E+02	9.63E+03
	STD	5.28E-08	5.16E+02	5.81E-08	1.53E-06	7.83E-06	9.36E+02
F3	AVG	6.00E+02	6.00E+02	6.00E+02	6.00E+02	6.00E+02	6.01E+02
	STD	2.49E-15	4.49E-03	1.14E-13	4.19E-01	3.62E-10	2.15E-01
F6	AVG	2.63E+04	3.54E+05	4.89E+04	1.08E+09	8.30E+04	3.30E+08
	STD	2.92E+03	1.93E+05	3.35E+04	9.88E+08	5.88E+03	3.09E+08
F12	AVG	2.98E+03	3.14E+03	3.00E+03	3.46E+03	3.16E+03	3.16E+03
	STD	1.44E+02	6.50E+01	2.05E+01	1.31E+02	8.08E+01	1.10E+02
AVG		7.54E+03	8.97E+04	1.32E+04	2.70E+08	2.17E+04	8.25E+07
Rank		1	4	2	6	3	5
%		-	1.08E+01	7.49E-01	3.58E+04	1.88E+00	1.09E+04

higher. Table 7 and Figure 6 present experimental results and the fitness convergence curves of several specific functions, respectively. In 20D advanced benchmarks, CQSGA secured the optimal solution for F6 (2.63E+04), surpassing suboptimal Aug-AEO (4.89E+04) by 46.2%. On F12, it led with 2.98E+03, exceeding Aug-AEO (3.00E+03) by 0.7%. For F1/F3, it matched Aug-AEO in means but showed a superior standard deviation (e.g., F1: 5.28E-08 vs. 5.81E-08). The overall average (7.54E+03) was 42.9% below Aug-AEO (1.32E+04), while Aug-AEO's average was 74.9% higher.

Figure 5

Convergence curves of CQSGA and advanced algorithms on specific CEC 2017(50D) functions: (a) F3, (b) F5, (c) F11, (d) F25.

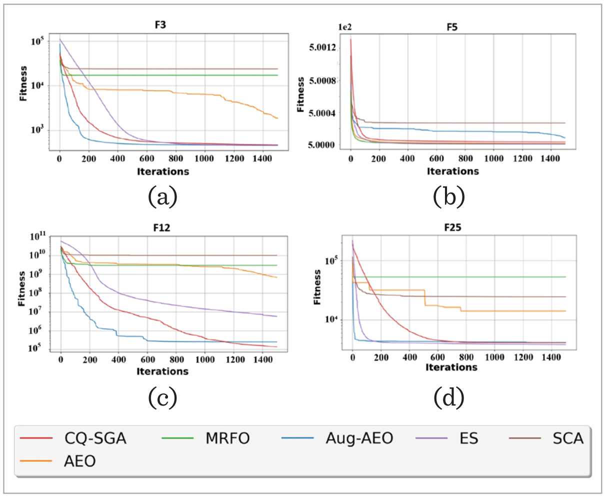
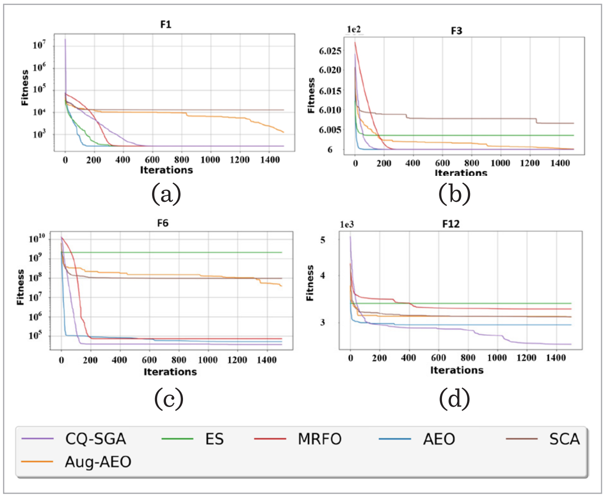


Figure 6

Convergence curves of CQSGA and advanced algorithms on specific CEC 2022(20D) functions: (a) F1, (b) F3, (c) F6, (d) F12.



Significance test is a statistical method used to determine whether the observed differences among algorithms are statistically meaningful rather than due to random chance. To further verify the statistical significance of the performance improvements, the Friedman test was conducted separately for the original algorithms and the advanced algorithms across CEC 2017 and CEC 2022 benchmark suites. Table 8 and 9 present the average rankings from the Friedman test for both groups with $p<0.001$. Table 10 and 11 present the Nemenyi post-hoc test at the 0.05 significance level. For the original algorithms, CQSGA achieved the lowest average ranks, followed by GA and PSO. The Nemenyi post-hoc test confirmed that CQSGA significantly outperforms all other original algorithms. For the advanced algorithms, CQSGA also achieved the lowest average ranks, followed by Aug-AEO and ES. The Nemenyi test indicated significant superiority over all advanced algorithms.

In addition to the Friedman test and Nemenyi post-hoc test, we computed Kendall's W as a measure of effect size. The obtained $W = 0.87$ indicates a large effect size, confirming that the performance differences among algorithms are not only statistically significant but also practically meaningful.

These results consistently demonstrate that the advantage of CQSGA is statistically robust across different algorithm categories.

Table 8

Friedman test for CQSGA and original algorithms.

Algorithm	CEC 2017 50D	CEC 2022 20D	Average Rank	Overall Rank
CQSGA	2.28	2.92	2.60	1
GA	2.48	2.91	2.69	2
PSO	2.68	2.91	2.79	3
GWO	4.24	3.67	3.95	4
HHO	4.68	4.75	4.71	5
WOA	5.41	5.08	5.24	6
DE	6.38	6.25	6.31	7
SGA	8.2	8.08	8.14	8
SA	8.62	8.42	8.52	9

Table 9

Friedman test for CQSGA and advanced algorithms.

Algorithm	CEC 2017 50D	CEC 2022 20D	Average Rank	Overall Rank
CQSGA	1.62	1.67	1.64	1
Aug-AEO	2.10	2.67	2.38	2
ES	2.51	3.08	2.79	3
AEO	4.10	4.00	4.05	4
MRFO	5.27	4.75	5.01	5
SCA	5.37	4.83	5.10	6

Table 10
Nemenyi post-hoc test results for CQSGA and original algorithms.

Algorithm	CQSGA	GA	PSO	GWO	HHO	WOA	DE	SGA	SA
CEC 2017 50D	NaN	2.28E-04	2.15E-06	2.59E-02	4.51E-02	2.55E-03	5.55E-16	1.51E-03	6.23E-14
CEC 2022 20D	NaN	2.98E-02	4.08E-02	1.99E-02	7.82E-03	6.51E-04	3.06E-05	4.87E-02	1.32E-04
Significantly Better	NaN	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE

Table 11
Nemenyi post-hoc test results for CQSGA and advanced algorithms.

Algorithm	CQSGA	Aug-AEO	ES	AEO	MRFO	SCA
CEC 2017 50D	NaN	9.23E-03	6.69E-04	9.59E-03	3.90E-10	1.60E-09
CEC 2022 20D	NaN	7.80E-03	4.01E-03	9.94E-03	2.98E-02	3.18E-02
Significantly Better	NaN	TRUE	TRUE	TRUE	TRUE	TRUE

Runtime is an important indicator of practical feasibility, reflecting the computational cost required to achieve a given solution. Comparing runtime alongside solution accuracy allows us to determine whether the performance gains of an algorithm come at an excessive computational expense. Table 12 presents the average runtime per run for all algorithms on the CEC benchmark suites. As shown, CQSGA requires more time than the original SGA and lightweight algorithms such as PSO and DE, which is expected due to its chaotic mapping and quantum rotation gate operations. However, its runtime remains substantially lower than that of more complex methods like SCA and MRFO. Given that CQSGA also achieves the best solution accuracy and statistical significance, the moderate increase in computational cost is well justified and does not compromise its practical applicability.

CQSGA demonstrates competitive and consistent performance across CEC 2017/2022 benchmarks, significantly outperforming mainstream methods in solution quality and stability. It consistently achieves the lowest Friedman average rank, with higher dimensions further amplifying its advantage. The integrated chaos-quantum mechanisms effectively suppress parametric oscillations, resulting in a standard deviation approximately 30% lower than that of the runner-up algorithms. The Friedman test yielded p-values below 0.001 for all benchmark suites, and the Nemenyi post-hoc test confirmed that CQSGA significantly outperforms all other original and advanced algorithms at the 0.05 significance level.

Table 12
Runtime of CQSGA and other algorithms on CEC benchmark suites.

Algorithm	CEC 2017 50D	CEC 2022 20D
CQSGA	64.06	35.09
GA	57.61	28.46
PSO	46.40	24.46
GWO	50.15	27.93
HHO	51.49	27.19
WOA	55.39	29.55
DE	48.42	25.21
SGA	52.08	28.23
SA	58.99	30.17
Aug-AEO	101.48	54.70
AEO	104.98	56.05
ES	99.13	49.03
SCA	242.43	92.77
MRFO	150.26	84.26

4.3. Ablation Study

This ablation study quantifies the individual and combined contributions of chaotic mapping (C) and quantum rotation gate (Q) components in SGA, using an experimental setup identical to that described in Section 4.1. The evaluation spans two benchmark datasets: CEC 2017, with 50-dimensional functions, and CEC 2022, with 20-dimensional functions, measuring mean error values across all functions in each benchmark. The study examines four algorithm variants: the original SGA without enhancements, C-SGA, which integrates C for population initialisation, Q-SGA, which incorporates Q for solution updates, and CQSGA, the full hybrid combining both C and Q.

Figure 7 reveals a clear performance hierarchy across both benchmarks. The original SGA performed worst, with error values of 6.364×10^8 on CEC 2017 and 1.674×10^8 on CEC 2022, demonstrating fundamental limitations in population diversity and convergence behaviour. Q-SGA showed moderate improvement with 3.751×10^8 on CEC 2017 (41.1% reduction vs SGA) and 5.555×10^7 on CEC 2022 (66.8% reduction), though quantum operations alone proved insufficient for effective initialisation in higher dimensions. Significantly better results came from C-SGA, achieving 2.384×10^8 on CEC 2017 (62.5% reduction) and 2.027×10^7 on CEC 2022 (87.9% reduction), highlighting chaotic mapping's critical role in enhancing exploration capabilities. The full hybrid CQSGA delivered exceptional performance with 8.117×10^7 on CEC 2017 (87.2% reduction vs SGA) and 6.416×10^6 on CEC 2022 (96.2% reduction), representing a 6.60×10^7 improvement over C-SGA and 2.939×10^8 improvement over Q-SGA on CEC 2017, confirming the synergistic value of combining both components.

Figure 8 provides additional insights by categorising results across function types. Consistently across all categories, multimodal, hybrid, unimodal and composition functions, the performance hierarchy remained SGA > Q-SGA > C-SGA > CQSGA. This pattern confirms the robustness of our findings, demonstrating that chaotic mapping offers significant benefits for multimodal and composition functions where exploration is crucial, while quantum rotation gates excel in unimodal functions that require precise convergence. Most significantly, CQSGA maintained its superiority in every category, demonstrating the hybrid approach's adaptability to diverse optimisation challenges.

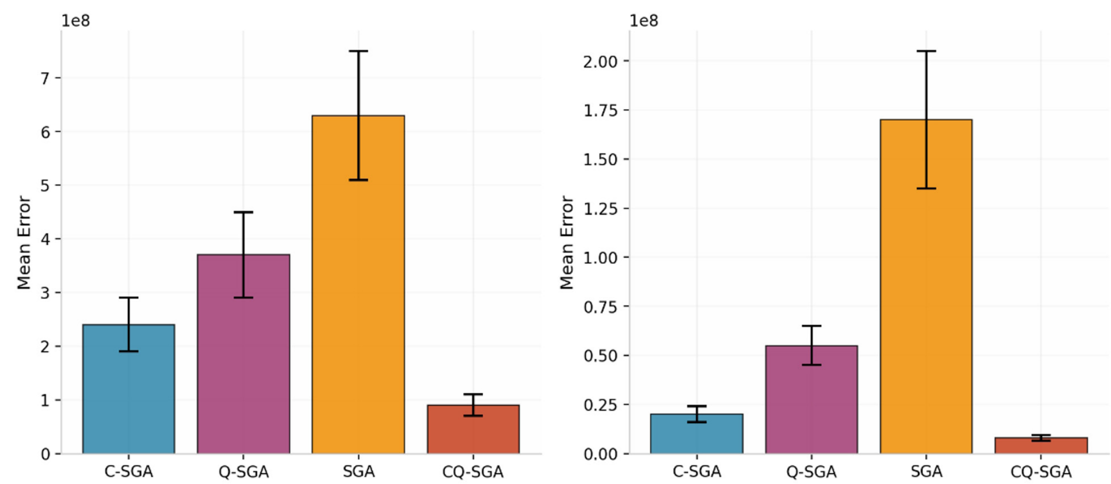
This ablation study conclusively demonstrates that both chaotic mapping and QRG significantly enhance SGA performance. The results underscore the crucial role of component interactions in hybrid algorithms, where synergistic combinations consistently outperform individual enhancements.

4.4 Sensitivity Analysis

Sensitivity analysis evaluates how variations in critical parameters affect algorithm performance, identifying optimal configurations and robustness boundaries. This analysis is conducted exclusively on the CEC 2017 50D benchmark suite, where pa-

Figure 7

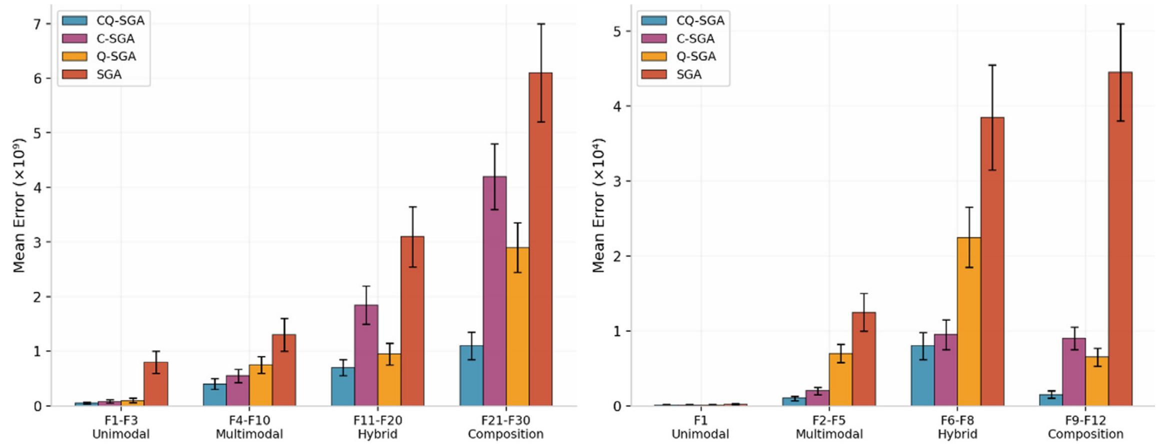
The mean error values of SGA variants on CEC 2017 and CEC 2022.



(a) CEC 2017 (b) CEC 2022

Figure 8

Categorising results of SGA variants across CEC 2017 and CEC 2022.



(a) CEC 2017 (b) CEC 2022

rameter effects on algorithm behaviour are most pronounced. When analysing one parameter, all other parameters are held constant at their preset values from Section 3. The experiment systematically varies four key parameters: chaos mapping iterations (Nchaos) controlling chaotic sequence complexity, rotation angle (θ) determining quantum state update magnitude, population size (N), and maximum iteration count (T). Parameter configurations are tested with $N_{chaos} \in \{1, 3, 5, 7, 9\}$, $\theta \in \{0.002, 0.004, 0.006, 0.008, 0.010\}$, $N \in \{10, 30, 50, 70, 90\}$, and $T \in \{500, 1000, 1500, 2000, 2500\}$, with the average best fitness across all CEC 2017 functions as the performance metric. Figure 9 illustrates the results.

4.4.1. Analysis of N_{chaos}

The sensitivity analysis for N_chaos reveals a pronounced U-shaped performance curve. At N_chaos = 3, the algorithm achieves its minimum average fitness of 8.78×10^8, which is 49.3% lower than the fitness at N_chaos = 1 and 51.8% lower than at N_chaos = 5. This indicates that while insufficient chaos (N_chaos < 3) limits exploration capability, excessive iterations (N_chaos > 5) introduce destructive randomness that disrupts convergence, confirming N_chaos = 3 as the optimal balance between ergodicity and stability.

4.4.2 Analysis of θ

For the θ parameter, the fitness landscape shows a clear minimum at $\theta = 0.006$ (1.638×10^9), which is 26.9% lower than the worst-performing $\theta = 0.004$ (2.242×10^9) and 9.5% lower than $\theta = 0.010$. This confirms that moderate rotation angles effectively balance quantum tunnelling for escaping local optima with stable convergence, consistent with the preset value of 0.006 in Table 3.

4.4.3. Analysis of N

The sensitivity analysis for N reveals a steep improvement followed by saturation. Performance improves dramatically from $N = 10$ (4.881×10^9) to $N = 50$ (2.016×10^9), but essentially plateaus beyond this point ($N = 70$: 2.015×10^9; $N = 90$: 2.013×10^9). This confirms $N = 50$ as the optimal trade-off between population diversity and computational cost.

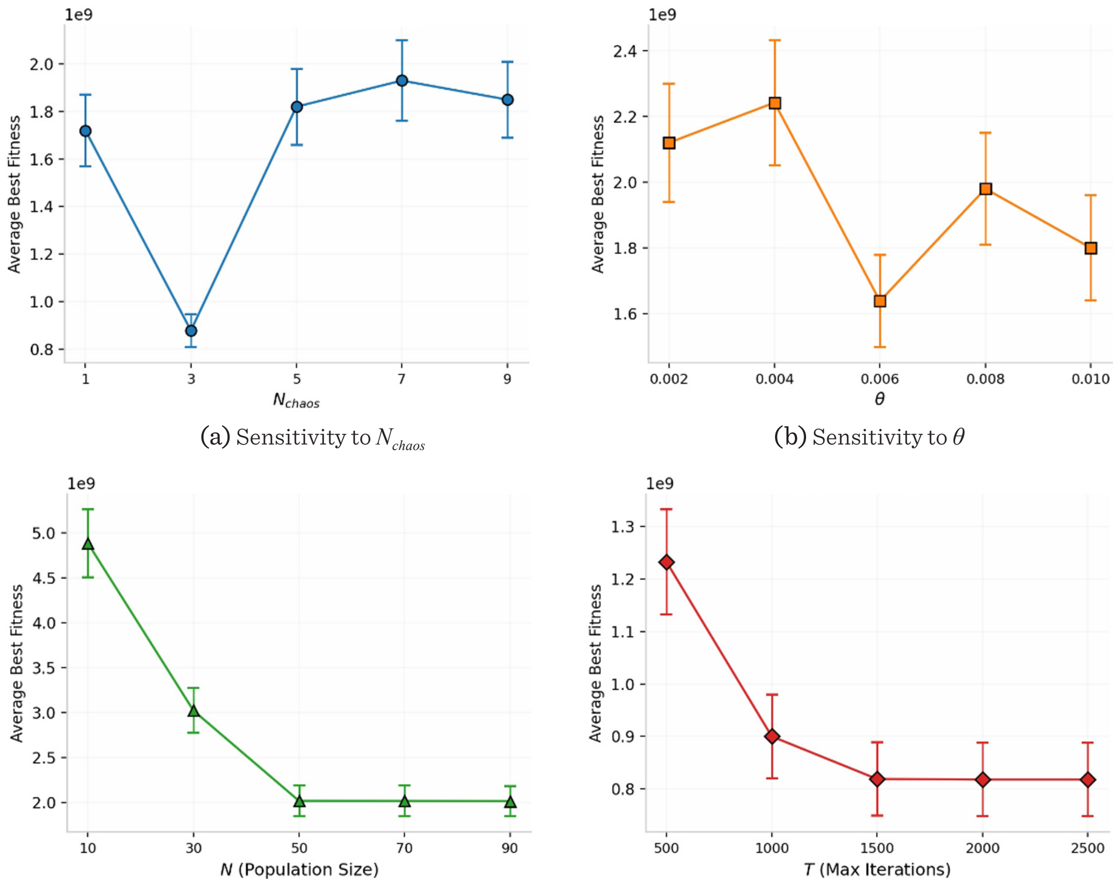
4.4.4. Analysis of T

The analysis of T exhibits rapid convergence followed by a flat plateau. Performance improves significantly from $T = 500$ (1.232×10^9) to $T = 1500$ (0.819×10^9), but stabilises entirely beyond this point (T = 2000 and T = 2500 both achieve 0.818×10^9, less than 0.15% improvement). This demonstrates that CQSGA has sufficient convergence capacity within 1500 iterations.

Based on the comprehensive sensitivity analysis, we establish the following optimal parameter selections: $N_{chaos} = 3$, $\theta = 0.006$, $N = 50$, and $T = 1500$. This configuration represents the optimal operational point for CQSGA, where the algorithm achieves the best balance between exploration diversity and exploitation precision while maintaining computational efficiency.

Figure 9

Sensitivity analysis of N_{chaos}, θ, N and T.



(a) Sensitivity to N_{chaos}

(b) Sensitivity to θ

(c) Sensitivity to N

(d) Sensitivity to T

4.5. Application of CQSGA

The Burgers' equation stands as a cornerstone benchmark in fluid mechanics, governing nonlinear phenomena ranging from shock waves to turbulence modelling [3]. Its challenging combination of nonlinear advection and viscous diffusion renders high-fidelity solutions essential for advancing physical understanding [5]. While high-accuracy numerical schemes exist, they often require sophisticated spatial approximations and careful handling of time integration to resolve sharp gradients, especially at high Reynolds numbers [13]. Furthermore, obtaining multiple-front solutions analytically remains a complex task, underscoring the equation's inherent nonlinearity [33]. To rigorously assess the capability of the CQSGA-optimised PINN framework in capturing such complex physics, we deploy it on this canonical problem.

In this study, the one-dimensional Burgers' equation is considered with viscosity coefficient ν over the spatial-temporal domain $x\in[-1,1]$ and $t\in[0,1]$. Its strong form reads: $\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} - \nu\frac{\partial^2 u}{\partial x^2} = 0$. The problem is subject to the initial condition $u(x,0) = -sin(\pi x)$ and the Dirichlet boundary conditions $u(-1,t) = u(1,t) = 0$. Let $u_0(x,t)$ denote the output of the PINN with trainable parameters θ. The PDE residual operator is then defined as:

$$F(u_\theta) = \frac{\partial u_\theta}{\partial t} + u_\theta\frac{\partial u_\theta}{\partial x} - \nu\frac{\partial^2 u_\theta}{\partial x^2}. \tag{12}$$

Given N_f interior collocation points $\{(x^i f, t^i f)\}_{i=1}^{N_f}$ randomly sampled within the domain, the PDE residual loss is computed via the mean squared error:

$$L_{PDE} = \frac{1}{N_f}\sum_{i=1}^{N_f}\left|F(u_0(x^i f, t^i f))\right|^2 . \quad (13)$$

The boundary condition loss L_{BC} comprises the initial condition loss L_{IC} and the spatial boundary loss L_{BC_x}, evaluated over N_{ic} initial points and N_{bc} boundary points, respectively:

$$L_{IC} = \frac{1}{N_{ic}}\sum_{i=1}^{N_{ic}}\left|u_\theta(x_{ic}^i, 0) + sin\,(\pi x_{ic}^i)\right|^2 \quad (14)$$

$$L_{BC_x} = \frac{1}{N_{bc}}\sum_{i=1}^{N_{bc}}\left[\left|u_\theta(-1, t_{bc}^i)\right|^2 + \left|u_\theta(1, t_{bc}^i)\right|^2\right] \quad (15)$$

$$L_{BC} = L_{IC} + L_{BC_x} . \quad (16)$$

Consequently, the total fitness function follows Equation (11).

Having defined the governing equation and loss components, the experiment performs a comparative analysis on CQSGA and five original algorithms, namely PSO, GA, GWO, DE, and HHO. The parameter configuration for CQSGA was fixed as follows: population size = 50, 40 coarse-search iterations, 1000 fine-search iterations and 30 independent runs. All baseline algorithms have their parameter settings provided in Table 2. All the algorithms were integrated into the DeepXDE-based PINN solver by function discretisation. The search space, summarized in Table 13, includes the number of hidden layers (2–5), the number of neurons per hidden layer (16–256, step 16), the activation function (tanh, relu, sigmoid, swish), and the learning rate (1e-4–1e-1 on a logarithmic scale). The network is a fully connected MLP with a fixed input layer of 2 neurons (space and time) and an output layer of 1 neuron (the solution field). The Adam optimizer is used with a constant learning rate.

Table 13

Search space for PINN architecture optimization.

Hyperparameter	Type	Encoding	Range / Options
Number of hidden layers	Integer	2 bit	[2, 5]
Neurons per hidden layer	Integer	4 bit	16, 32, 48, ..., 256
Activation function	Categorical	2 bit	tanh, relu, sigmoid, swish
Learning rate	Continuous (log)	3 bit	[1e-4, 1e-1]

Figure 10

Fine-search and coarse-search loss curves of PINN architectures.

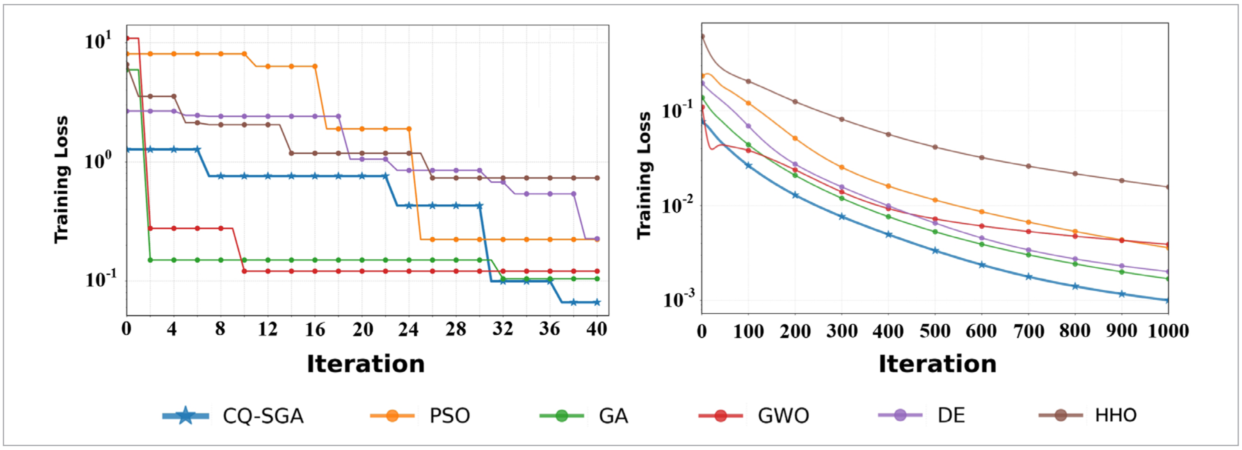
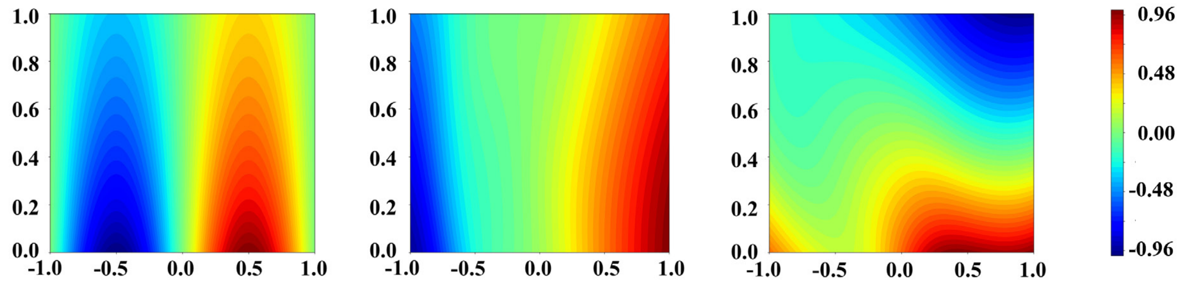


Figure 11

Comparison of Burgers equation solution fields: (a) true solution, (b) CQSGA reconstruction, and (c) PSO reconstruction.



(a) True Solution (b) CQ-SGA (c) PSO

Figure 10(a) illustrates the results of the coarse-search phase, where CQSGA achieved the lowest loss of 6.528e-2, outperforming the second-best algorithm, GA (1.012e-1), by 35.5%, demonstrating exceptional initial optimisation capability. Figure 11 delivers a critical visual validation: the CQSGA-PINN solution (b) exhibits close agreement with the ground truth solution (a), particularly in preserving boundary layer integrity. In stark contrast, the PSO-optimised solution (c) exhibits unphysical oscillations near boundaries, indicative of inadequate representation capacity or optimisation entrapment. It visually underscores CQSGA's success in encoding physically consistent behaviour. Figure 10 (b) illustrates the results of the fine-search phase. After 1,000 fine-search iterations, CQSGA attained a final loss of 9.9873e-4 at epoch 1000, representing a 98.47% reduction from its coarse-search result, which validates the efficacy of the two-phase optimisation strategy. Comparative analysis reveals CQSGA's performance advantage, as it surpasses all baselines by 40.7% (1.6859e-3) compared to GA and by 93.6% compared to the weakest performer, HHO (1.5687e-2). Crucially, CQSGA is the sole algorithm breaking the 1e-3 loss barrier, achieving sub-millesimal accuracy (9.9873e-4), while PSO and GWO remain at millesimal levels. These results confirm CQSGA's dual strengths in architecture design and optimisation stability for solving Burgers' equation, providing a competitive optimization approach for training PINNs.

Table 14 reports the best architectures found by each algorithm. CQSGA achieved the lowest training loss (9.98e-4) and the lowest L2 relative error (9.99e-2), with a seven-layer network ($2 \rightarrow 192 \rightarrow 32 \rightarrow 256 \rightarrow 16 \rightarrow 16 \rightarrow 1$), sigmoid activation, and a learning rate of 1.73e-2. The L2 error ranking is fully consistent with the training loss ranking, confirming effective generalisation. CQSGA's L2 error is 15.8% lower than that of the second-best algorithm GA (1.19e-1), achieved with a substantially more compact architecture (19,601 vs. 87,105 parameters). In contrast, the sec-

Table 14

Best architectures discovered by each algorithm.

Algorithm	Best Loss	L2 Error	Layers (Input→Hidden→Output)	Activation	Learning Rate	Total Parameters	Time (h)
CQSGA	9.98e-4	9.989e-02	2 → 192 → 32 → 256 → 16 → 16 → 1	sigmoid	1.73e-2	19,601	3.5
PSO	5.11e-3	1.329e-01	2 → 256 → 16 → 1	relu	1.43e-2	4,897	2.5
GA	3.34e-3	1.186e-01	2 → 80 → 192 → 128 → 144 → 192 → 1	sigmoid	1.77e-2	87,105	2.7
GWO	5.13e-3	1.341e-01	2 → 176 → 176 → 176 → 176 → 176 → 1	sigmoid	3.72e-4	125,313	3.0
DE	3.48e-3	1.230e-01	2 → 64 → 128 → 160 → 16 → 144 → 1	relu	1.43e-3	34,321	2.9
HHO	3.90e-3	1.317e-01	2 → 128 → 128 → 128 → 16 → 1	relu	1.57e-2	40,089	3.2

ond-best algorithm GA obtained a loss of 3.34e-3 with a different seven-layer structure. Notably, the CQSGA architecture balances model complexity (19,601 parameters) with high predictive accuracy, outperforming both deeper (GWO, 125,313 parameters) and shallower (PSO, 4,897 parameters) networks. Although CQSGA has slightly higher time consumption due to the additional computations from chaotic mapping and quantum rotation gates, the quality of the architectures obtained through search is higher.

These findings validate CQSGA's dual advantages in architecture design and optimisation stability, which are crucial for solving the Burgers' equation, thereby providing an effective baseline for future comparisons in PINN optimization.

5. Conclusion

This paper proposed an enhanced version of SGA, called CQSGA, by integrating chaotic mapping and a quantum rotation gate. The chaotic mapping significantly bolsters the algorithm's population diversity and global exploration capability, while the quantum rotation gate precisely refines its local exploitation accuracy. Notably, CQSGA demonstrates strong adaptive performance across different problem dimensions, with its advantages becoming more pronounced in high-dimensional optimisation problems, as evidenced by the experimental results on the CEC 2017 50D and CEC 2022 20D benchmarks, where CQSGA consistently achieved the lowest average error among all compared algorithms. These performance advantages are further substantiated by the Friedman test ($p < 0.001$) and Nemenyi post-hoc analysis ($\alpha = 0.05$). Furthermore, its successful application to the architecture search for PINNs provides strong validation of its applicability in tackling demanding practical problems. Future work could consider extending CQSGA to the multi-objective optimization domain and exploring its application potential in real-time control systems.

References

1. Askarzadeh, A. A Novel Metaheuristic Method for Solving Constrained Engineering Optimization Problems: Crow Search Algorithm. Computers & Structures, 2016, 169, 1-12. https://doi.org/10.1016/j.compstruc.2016.03.001
2. Azizi, M., Aickelin, U., Khorshidi, H. A., Baghalzadeh Shishehgarkhaneh, M. Energy Valley Optimizer: A Novel Metaheuristic Algorithm for Global and Engineering Optimization. Scientific Reports, 2023, 13(1), 226. https://doi.org/10.1038/s41598-022-27344-y
3. Bateman, H. Some Recent Researches on the Motion of Fluids. Monthly Weather Review, 1915, 43(4), 163-170. https://doi.org/10.1175/1520-0493(1915)43<163:SRROTM>2.0.CO;2
4. Bian, H., Li, C., Liu, Y., Tong, Y., Bing, S., Chen, J., Ren, Q., Zhang, Z. Improved Snow Geese Algorithm for Engineering Applications and Clustering Optimization. Scientific Reports, 2025, 15(1), 4506. https://doi.org/10.1038/s41598-025-88080-7
5. Bonkile, M. P., Awasthi, A., Lakshmi, C., Mukundan, V., Aswin, V. A Systematic Literature Review of Burgers' Equation With Recent Advances. Pramana, 2018, 90(6), 69. https://doi.org/10.1007/s12043-018-1559-4
6. Cai, S., Mao, Z., Wang, Z., Yin, M., Karniadakis, G. E. Physics-Informed Neural Networks (PINNs) for Fluid Mechanics: A Review. Acta Mechanica Sinica, 2021, 37(12), 1727-1738. https://doi.org/10.1007/s10409-021-01148-1
7. Chiu, P.-H., Wong, J. C., Ooi, C., Dao, M. H., Ong, Y.-S. CAN-PINN: A Fast Physics-Informed Neural Network Based on Coupled Automatic Numerical Differentiation Method. Computer Methods in Applied Mechanics and Engineering, 2022, 395, 114909. https://doi.org/10.1016/j.cma.2022.114909
8. Dorigo, M., Birattari, M., Stützle, T. Ant Colony Optimization. IEEE Computational Intelligence Magazine, 2006, 1(4), 28-39. https://doi.org/10.1109/MCI.2006.329691
9. Fu, S., Li, K., Huang, H., Ma, C., Fan, Q., Zhu, Y. Red-Billed Blue Magpie Optimizer: A Novel Metaheuristic Algorithm for 2D/3D UAV Path Planning and Engineering Design Problems. Artificial Intelligence Review, 2024, 57(6), 134. https://doi.org/10.1007/s10462-024-10716-3
10. Heidari, A. A., Mirjalili, S., Faris, H., Aljarah, I., Mafarja, M., Chen, H. Harris Hawks Optimization: Algorithm and Applications. Future Generation Computer Sys-

tems, 2019, 97, 849-872. https://doi.org/10.1016/j.future.2019.02.028

11. Holland, J. H. Adaptation in Natural and Artificial Systems: An Introductory Analysis With Applications to Biology, Control, and Artificial Intelligence. MIT Press, 1992. https://doi.org/10.7551/mitpress/1090.001.0001
12. Holland, J. H. Genetic Algorithms. Scientific American, 1992, 267(1), 66-73. https://doi.org/10.1038/scientificamerican0792-66
13. Hon, Y.-C., Mao, X. An Efficient Numerical Scheme for Burgers' Equation. Applied Mathematics and Computation, 1998, 95(1), 37-50. https://doi.org/10.1016/S0096-3003(97)10060-1
14. Kennedy, J., Eberhart, R. Particle Swarm Optimization. Proceedings of the IEEE International Conference on Neural Networks (ICNN'95), 1995, 4, 1942-1948. https://doi.org/10.1109/ICNN.1995.488968
15. Kien, L. C., Loi, T. D., Duong, M. P., Nguyen, T. T. Energy Loss Reduction for Distribution Electric Power Systems With Renewable Power Sources, Reactive Power Compensators, and Electric Vehicle Charge Stations. Sensors, 2025, 25(7), 1997. https://doi.org/10.3390/s25071997
16. Kirkpatrick, S., Gelatt Jr., C. D., Vecchi, M. P. Optimization by Simulated Annealing. Science, 1983, 220(4598), 671-680. https://doi.org/10.1126/science.220.4598.671
17. Leung, W. T., Lin, G., Zhang, Z. NH-PINN: Neural Homogenization-Based Physics-Informed Neural Network for Multiscale Problems. Journal of Computational Physics, 2022, 470, 111539. https://doi.org/10.1016/j.jcp.2022.111539
18. Lim, H., Kim, H. Multi-Objective Airfoil Shape Optimization Using an Adaptive Hybrid Evolutionary Algorithm. Aerospace Science and Technology, 2019, 87, 141-153. https://doi.org/10.1016/j.ast.2019.02.019
19. Liu, J., Deng, Y., Xiao, C., Li, Z. A Deep Reinforcement Learning Enhanced Snow Geese Optimizer for Robot Calibration. Processes, 2025, 13(5), 1407. https://doi.org/10.3390/pr13051407
20. Ruder, S. An Overview of Gradient Descent Optimization Algorithms. arXiv Preprint, 2016, arXiv:1609.04747. DOI: https://arxiv.org/abs/1609.04747
21. Kumar, A., Price, K. V., Mohamed, A. W., Hadi, A. A., Suganthan, P. N. Problem Definitions and Evaluation Criteria for the 2022 Special Session and Competition on Single Objective Bound Constrained Numerical Optimization. Technical Report, Nanyang Technological University, Singapore, 2021. Available at: https://github.com/P-N-Suganthan/2022-SO-BO.
22. Manan, A., Vio, G., Harmin, M., Cooper, J. Optimization of Aeroelastic Composite Structures Using Evolutionary Algorithms. Engineering Optimization, 2010, 42(2), 171-184. https://doi.org/10.1080/03052150903104358
23. Meng, Z., Qian, Q., Xu, M., Yu, B., Yıldız, A. R., Mirjalili, S. PINN-FORM: A New Physics-Informed Neural Network for Reliability Analysis With Partial Differential Equation. Computer Methods in Applied Mechanics and Engineering, 2023, 414, 116172. https://doi.org/10.1016/j.cma.2023.116172
24. Mirjalili, S., Mirjalili, S. M., Lewis, A. Grey Wolf Optimizer. Advances in Engineering Software, 2014, 69, 46-61. https://doi.org/10.1016/j.advengsoft.2013.12.007
25. Mirjalili, S. SCA: A Sine Cosine Algorithm for Solving Optimization Problems. Knowledge-Based Systems, 2016, 96, 120-133. https://doi.org/10.1016/j.knosys.2015.12.022
26. Mirjalili, S., Lewis, A. The Whale Optimization Algorithm. Advances in Engineering Software, 2016, 95, 51-67. https://doi.org/10.1016/j.advengsoft.2016.01.008
27. Qiu, Z., Mo, Y. Economic Dispatch Optimization of Combined Heat and Power System Based on Improved Snow Goose Algorithm. Modern Electronics Technique, 2025, 48(6), 127-135. DOI: https://doi.org/10.16652/j.issn.1004-373x.2025.06.020.
28. Raissi, M., Wang, Z., Triantafyllou, M. S., Karniadakis, G. E. Deep Learning of Vortex-Induced Vibrations. Journal of Fluid Mechanics, 2019, 861, 119-137. https://doi.org/10.1017/jfm.2018.872
29. Rashedi, E., Nezamabadi-Pour, H., Saryazdi, S. GSA: A Gravitational Search Algorithm. Information Sciences, 2009, 179(13), 2232-2248. https://doi.org/10.1016/j.ins.2009.03.004
30. Sahana, B. C. GestProNet: A Hybrid Deep Learning Framework for High-Precision EMG-Based Hand Gesture Classification. American Journal of Psychiatric Rehabilitation, 2025, 28(1), 647-657. Available at: https://ajprui.com/index.php/ajpr/article/view/166.
31. Sörensen, K. Metaheuristics-The Metaphor Exposed. International Transactions in Operational Research, 2015, 22(1), 3-18. https://doi.org/10.1111/itor.12001
32. Storn, R., Price, K. Differential Evolution-A Simple and Efficient Heuristic for Global Optimiza-

tion Over Continuous Spaces. Journal of Global Optimization, 1997, 11(4), 341-359. https://doi.org/10.1023/A:1008202821328

33. Tian, A.-Q., Liu, F.-F., Lv, H.-X. Snow Geese Algorithm: A Novel Migration Inspired Meta-Heuristic Algorithm for Constrained Engineering Optimization Problems. Applied Mathematical Modelling, 2024, 126, 327-347. https://doi.org/10.1016/j.apm.2023.10.045

34. Wazwaz, A.-M. Multiple-Front Solutions for the Burgers Equation and the Coupled Burgers Equations. Applied Mathematics and Computation, 2007, 190(2), 1198-1206. https://doi.org/10.1016/j.amc.2007.02.003

35. Wu, G., Mallipeddi, R., Suganthan, P. N. Problem Definitions and Evaluation Criteria for the CEC 2017 Competition on Constrained Real-Parameter Optimization. Technical Report, National University of Defense Technology, Kyungpook National University, and Nanyang Technological University, 2017. Available at: https://github.com/P-N-Suganthan/CEC2017-BoundContrained.

36. Wu, J., Yang, Z.-J., Du, Z.-G., Tian, A.-Q. Surrogate-Assisted Multi-Objective Snow Goose Algorithm With Gaussian Process for Cold Chain Logistics. IEEE Access, 2025, 13. https://doi.org/10.1109/ACCESS.2025.3566924

37. Zhao, B., Fang, Y., Chen, T. An Enhanced Snow Geese Optimizer Integrating Multiple Strategies for Numerical Optimization. Biomimetics, 2025, 10(6), 388. https://doi.org/10.3390/biomimetics10060388

38. Zhao, W., Zhang, Z., Wang, L. A Novel Atom Search Optimization for Dispersion Coefficient Estimation in Groundwater. Future Generation Computer Systems, 2019, 91, 601-610. https://doi.org/10.1016/j.future.2018.05.037

39. Zhao, W., Wang, L., Zhang, Z. Artificial Ecosystem-Based Optimization: A Novel Nature-Inspired Meta-Heuristic Algorithm. Neural Computing and Applications, 2020, 32(13), 9383-9425. https://doi.org/10.1007/s00521-019-04452-x

40. Zhao, W., Zhang, Z., Wang, L. Manta Ray Foraging Optimization: An Effective Bio-Inspired Optimizer for Engineering Applications. Engineering Applications of Artificial Intelligence, 2020, 87, 103300. https://doi.org/10.1016/j.engappai.2019.103300

41. Zhu, G., Song, Y., Liu, B., Sun, X., Ji, C. Research on Bearing Fault Diagnosis Based on SGA-Optimized ICEEMDAN. Journal of Physics: Conference Series, 2025, 3067(1), 012027. https://doi.org/10.1088/1742-6596/3067/1/012027