processes

Article

Quadratic Interpolation Based Simultaneous Heat Transfer Search Algorithm and Its Application to Chemical Dynamic System Optimization

Ebrahim Alnahari, Hongbo Shi * and Khalil Alkebsi

Key Laboratory of Advanced Control and Optimization for Chemical Processes, Ministry of Education, East China University of Science and Technology, Shanghai 200237, China; barhooma97@hotmail.com (E.A.); kalkebsi@hotmail.com (K.A.)

* Correspondence: hbshi@ecust.edu.cn; Tel.: +86-21-64252189

Received: 14 March 2020; Accepted: 14 April 2020; Published: 19 April 2020

Abstract: Dynamic optimization problems (DOPs) are widely encountered in complex chemical engineering processes. However, due to the existence of highly constrained, nonlinear, and nonsmooth environment in chemical processes, which usually causes nonconvexity, multimodality and discontinuity, handling DOPs is not a straightforward task. Heat transfer search (HTS) algorithm is a relative novel metaheuristic approach inspired by the natural law of thermodynamics and heat transfer. In order to solve DOPs efficiently, a new variant of HTS algorithm named quadratic interpolation based simultaneous heat transfer search (QISHTS) algorithm is proposed in this paper. The QISHTS algorithm introduces three modifications into the original HTS algorithm, namely the effect of simultaneous heat transfer search, quadratic interpolation method, and population regeneration mechanism. These three modifications are employed to provide lower computational complexity, as well as to enhance the exploration and exploitation capabilities. Therefore, the ensemble of these modifications can provide a more efficient optimization algorithm with well-balanced exploration and exploitation capabilities. The proposed variant is firstly investigated by well-defined benchmark problems and then applied to solve four chemical DOPs. Moreover, it is compared with different well-established methods existing in the literature. The results demonstrate that QISHTS algorithm has the greatest robustness and precision than other competitors.

Keywords: dynamic system optimization; chemical engineering processes; heat transfer search algorithm; quadratic interpolation; global optimization

1. Introduction

Many considerable chemical processes depend greatly on dynamic optimization, which aims at optimizing the performance index of the whole process by implementing optimization control on the established dynamic model [1,2], and these models generally contain dynamic variables with values that vary with time [3]. The optimization problems that involve dynamic variables are called dynamic optimization problems (DOPs); their dynamic systems are described by a set of algebraic and differential equations. There are several main fields involving DOPs, such as parameter estimation and optimal control for dynamic models. DOPs are also inherent in the reactor network synthesis, where the differential modeling of chemical reactors produces the dynamic models [4]. Due to the existence of highly constrained, nonlinear, and nonsmooth environments in chemical processes, which usually causes nonconvexity, utilizing new global optimization approaches is required to find appropriate solutions for solving DOPs efficiently.

The solution methods for DOPs can be simply classified into three methods, dynamic programming (DP), indirect, and direct methods. The DP is an effective approach to solve DOPs. It relies on

Bellman's principle of optimality [5]. However, this method suffers from the issues of dimensionality. Thus, in order to avoid such drawbacks, Luus proposed iterative dynamic programming (IDP) [6]. The IDP is an effective global optimization approach that can achieve the global optimum for multimodal optimization problems and can reduce the dimension expansion. However, the computational time of this approach is relatively large, since both control and state variables are required to be segmented [7]; therefore, this approach is merely appropriate for small-scale problems. The original DOP was expanded in the indirect method to a Hamiltonian system using the Pontryagin maximum theory [8], and this method is considered as the most precise approach to solve optimal control problems [9,10], but the application of this method in practical DOPs is complex. Two different parametrization methods are included in the direct methods, and they are the complete parametrization (CP) method and control vector parametrization (CVP) method. The CP method is also named simultaneous method, and the basic principle of this method is to segment both state and control variables simultaneously, while the CVP method only segments the control variables. The CP method can preferably deal with path constraint problems, but it results in a large-scale nonlinear programming (NLP) problem, and certain techniques are needed to solve the NLP efficiently. The CVP method is used to translate the basic DOP into an NLP problem; the dimension range of NLP problems in this method is much smaller than that in the CP method; therefore, the CVP method has been incorporated with various optimization algorithms to handle DOPs [11,12]. A number of deterministic methods have also been proposed for handling DOPs; the gradient-based methods [13,14], such as the sequence quadratic program and Newton's algorithms, are considered as deterministic methods that can converge rapidly. However, they may converge to local optima according to the degree of initial guess and nonlinearity [15]. Other types of the deterministic methods are branch and bound algorithms [16,17]; they need rigorous bounds on the parameters required or strict values for these parameters, which might lead to difficulty in realization [15]. For detailed information about deterministic methods, the reader can refer to the literature [15].

Meta-heuristic algorithms (MHAs) are inspired by varieties of natural phenomena; they mimic some element of biology, ethology, physics, and swarm intelligence [18]. Comparing with other deterministic methods, MHAs possess superior global perspective and make few assumptions about the optimization problems. Due to the importance of dynamic optimization in chemical processes, various MHAs have been presented for handling chemical DOPs in recent years, such as genetic algorithm (GA) [19–21], differential evolution (DE) [22–25], ant colony optimization (ACO) [26,27], simulated annealing (SA) [28,29], particle swarm optimization (PSO) [30–33], artificial bee colony (ABC) [34], scatter search (SS) [35–37], line-up competition algorithm (LCA) [38], cuckoo search (CS) [39], biogeography-based optimization (BBO) [40], and teaching-learning based optimization (TLBO) [41].

Heat transfer search (HTS) algorithm is a relative new MHA presented by Patel and Savsani [42]; it simulates the natural laws of heat transfer and thermodynamics. The main framework of this algorithm basically consists of three basic phases: conduction phase, convection phase, and radiation phase. The HTS algorithm has emerged as one of the most efficient methods, and it has been empirically shown to be superior to other MHAs by its authors [42]. HTS and its modified variants have been applied for solving various optimization problems [43–49]. However, as an efficient optimization method, the report of the applications of HTS algorithm in engineering process optimization is still rarely seen, especially in the areas of chemical process optimization. It is worth mentioning that HTS-based algorithms have not been used to handle chemical DOPs. Thus, the main aim of this work is to apply the HTS algorithm to solve real-world chemical DOPs and, hence, extend the practical applications of this algorithm to the chemical dynamic optimization.

The exploration and exploitation capabilities of an MHA are generally contradictory to each other, and it is important to utilize suitable techniques to keep these two abilities well-balanced. Thus, to provide a more efficient HTS method with well-balanced exploration and exploitation capabilities, three modifications are introduced into the original HTS algorithm, namely the effect of simultaneous heat transfer search (SHTS), quadratic interpolation (QI) method, and population regeneration mechanism. Hence, a new version of HTS algorithm named QISHTS is proposed to solve real-world chemical DOPs efficiently. In the QISHTS algorithm, the effect of SHTS is employed to provide superior performance with lower computational complexity; the QI method is adopted to improve the exploitation capability, whereas the population regeneration mechanism is used to alleviate the probability of local optimum stagnation and, hence, enhance the exploration capability.

The main contributions of this work can be summarized as follows:

(1) A new variant of HTS algorithm named the QISHTS algorithm is presented by integrating the effect of SHTS, QI method, and population regeneration mechanism. The ensemble of these three modifications can provide a more efficient HTS method with well-balanced exploration and exploitation capabilities.
(2) The performance of the QISHTS algorithm is investigated by a set of 18 well-defined benchmark functions, and the obtained results are compared with those of other well-established MHAs.
(3) The proposed QISHTS algorithm is applied for solving four real-world chemical DOPs, including two dynamic parameter estimation problems and two optimal control problems. To the best of our knowledge, HTS-based algorithms have not been used for handling chemical DOPs, and our work is the first attempt to utilize it for solving such problems.
(4) The effectiveness of the QISHTS algorithm in solving chemical DOPs is compared with those of twelve well-established MHAs existing in the literature.

The rest of this paper is organized as follows. In Section 2, the basic formulation of DOPs is defined. In Section 3, the basic theoretical principle of HTS algorithm is explained. In Section 4, the proposed QISHTS algorithm is presented in detail. Some comparisons and investigation are shown in Section 5. In Section 6, the QISHTS algorithm is applied to solve real-word DOPs. Finally, conclusions are described in Section 7.

2. Dynamic Optimization Problems Formulation

In this study, the dynamic system of DOPs is addressed with algebraic and differential constraints, and its mathematical formula can be given as follow [35]:

$$\begin{array}{l}\min obj = \psi(x(t),\ u(t),\ \rho)\\ \text{Subjeted to :}\\ \begin{cases} f\big(\dot{x}(t), x(t), u(t), \rho\big) = 0\\ x(t_0) = x_0\\ h_i(x(t), u(t), \rho) = 0\ , i = 1, \ldots, m\\ g_i(x(t), u(t), \rho) \le 0\ , i = 1, \ldots, m\\ u^L \le u(t) \le u^U\\ \rho^L \le \rho \le \rho^U\\ t \in \big[t_0, t_f\big]\end{cases}\end{array} \quad (1)$$

where $\psi(x(t),\ u(t),\ \rho)$ represents the objective function, f indicates the set of differential equations that exhibits the description of system dynamics, $x(t)$ represents the vector of time-dependent state variables, $u(t)$ indicate the vector of time-dependent control variables, ρ are defined as the undetermined parameters, and g and h represent the inequality and equality constraints, respectively.

In general, the optimization variables of DOPs in chemical processes contain both the time-independent undetermined parameters ρ and time-dependent control variables $u(t)$. In this paper, two main types of DOPs are defined depending on the type of optimization variables, and these two types are as follows:

(1) Dynamic parameter estimation problems: In these problems, the optimization variables only contain undetermined parameters, ρ [37].
(2) Optimal control problems: In these problems, the optimization variables only include control variables, $u(t)$ [13].

The detailed procedure of solving a generalized DOP can be summarized as follow: Firstly, determine the type of DOP depending on its optimization variables. In other words, if the time-independent undetermined parameters ρ exist, then it is a parameter estimation problem which can be directly translated into an NLP problem. While, if the time-dependent control variables $u(t)$ exist, this problem is an optimal control problem which cannot be directly transformed into an NLP problem; thus, a discretization approach such as control vector parametrization (CVP) method [11,12] is required to translate it into an NLP problem. The time interval is divided by the CVP method into a number of stages (N), wherein the control variables $u(t)$ in every sup-interval are approximated by means of basis functions, such as linear functions, constant functions, and wavelet-based functions. Secondly, apply the optimization methods such as MHAs to optimize the undetermined parameters that only exist after the transforming process. Then, utilize an ordinary differential equation integrator for acquiring the objective function value (OFV), since the optimization parameters are required to be integrated into Equation (1) during the solution process. Subsequent to these procedures, the solution of a DOP can be finally output.

3. Heat Transfer Search (HTS) Algorithm

The HTS algorithm is a population-based optimization method; it is based on the natural laws of heat transfer and thermodynamics, which states that "Any system always tries to achieve equilibrium state with its surroundings" [42]. This algorithm considers three heat transfer phases (conduction phase, convection phase, and radiation phase) to simulate the thermal equilibrium behavior of the systems [42]. These three phases have important roles for setting the thermal equilibrium and reaching an equilibrium state. The HTS algorithm begins with a randomly generated population (NP), which is considered as a group of molecules, and these molecules try to achieve an equilibrium state with its surroundings by interacting among themselves and their environment through the three phases of heat transfer. The population is updated in each iteration by one of the three heat transfer phases, where a uniformly distributed random number (R_n) is generated in the interval [0, 1] to decide which phase should be activated to update the solutions in the particular iteration. In other words, if $R_n \in [0,\ 0.3333]$, the population is performed by the conduction phase, while if $R_n \in [0.3333,\ 0.6666]$, the population undergoes the radiation phase; whereas, if $R_n \in [0.6666,\ 1]$, the population is performed by the convection phase. In the HTS algorithm, the selection procedure of the updated solutions is carried out by the greedy selection technique, which accepts the new solution only if it has a superior objective value. After that, the inferior solutions are substituted by the elite solutions. Thus, the superior solution can be obtained by performing the difference between the elite solutions and the current solution. The whole search process of HTS algorithm is carried out through the essential operations of the conduction, convection, and radiation phases, which are briefly explained in the following subsections.

3.1. Conduction Phase

The basic principle of this phase is that heat transfer occurs due to the conduction effect among molecules of the substance. Thus, the molecules with higher energy transfer heat to the molecules with lower energy for the sake of reaching a state of the thermal equilibrium, where the system makes an attempt to neutralize the thermal imbalance. During the conduction phase of the algorithm, a new solution is updated according to the subsequent equations:

$$X_{j,i}^{new} = \left\{ \begin{array}{l} X_{k,i} + \left(-R_n^2 * X_{k,i}\right), if\ f(X_j) > f(X_k) \\ X_{j,i} + \left(-R_n^2 * X_{j,i}\right),\ if\ f(X_j) < f(X_k) \end{array} \right\} if\ FE \leq \frac{FE_{\max}}{CDF} \tag{2}$$

$$X_{j,i}^{new} = \left\{ \begin{array}{l} X_{k,i} + \left(-r_i * X_{k,i}\right), if\ f(X_j) > f(X_k) \\ X_{j,i} + \left(-r_i * X_{j,i}\right),\ if\ f(X_j) < f(X_k) \end{array} \right\} if\ FE \geq \frac{FE_{\max}}{CDF} \tag{3}$$

where $X_{j,i}^{new}$ represents the new updated solution; $j = 1, 2, \ldots, n$; k denotes a randomly chosen solution; $j \neq k$ and $k \in (1, 2, .., n)$; i indicates a randomly chosen decision variable where $i \in (1, 2, .., m)$; R_n denotes the random variable—it varies between 0 and 0.3333; r_i is a random number in the range [0, 1]; R_n^2 and r_i correspond to the conductance parameters of the Fourier's law of heat conduction [42]; *FE* represents the function evaluations; and *CDF* is the conduction factor—it is set to a value of 2 to balance the exploration and exploitation in the conduction phase [42].

3.2. Convection Phase

The basic principle of this phase is that heat transfer takes place because of convection between the system and the surrounding temperature. Thus, the mean temperature of the system interacts with the surrounding temperature for the sake of establishing a state of the thermal equilibrium. In this phase, the mean of the population members (denoted by X_{ms}) is regarded as the mean temperature, whereas the best solution (denoted by X_s) is regarded as the surrounding. In this part of the algorithm, a new solution is updated utilizing the subsequent formula:

$$X_{j,i}^{new} = X_{j,i} + R_n(X_s - X_{ms} * TCF) \tag{4}$$

$$TCF = \left\{ \begin{array}{l} abs(R_n - r_i),\quad if\ FE \leq FE_{\max}/COF \\ round(1 + r_i),\ if\ FE \geq FE_{\max}/COF \end{array} \right. \tag{5}$$

where $X_{j,i}^{new}$ is the new updated solution; $j = 1, 2, \ldots, n$; $i = 1, 2, \ldots, m$; R_n is the random variable—it varies between 0.6666 and 1; r_i is a random number in the interval [0, 1]; R_n and r_i correspond to the convection parameters of Newton's law of cooling [42]; *FE* represents the function evaluations; *TCF* is the temperature change factor; and *COF* is the convection factor—it is set to a value of 10 to balance the exploration and exploitation in the convection phase [42].

3.3. Radiation Phase

The basic principle of this phase is that the system attempts to neutralize the thermal imbalance through interacting with the surrounding temperature (i.e., best solution) or within the system (i.e., other solution). Here, the system tries to reach a state of the thermal balance. In this part of the algorithm, a new solution is updated according to the subsequent equations:

$$X_{j,i}^{new} = \left\{ \begin{array}{l} X_{j,i} + R_n\left(X_{k,i} - X_{j,i}\right), if\ f(X_j) > f(X_k) \\ X_{j,i} + R_n\left(X_{j,i} - X_{k,i}\right), if\ f(X_j) < f(X_k) \end{array} \right\} if\ FE \leq \frac{FE_{\max}}{RDF} \tag{6}$$

$$X_{j,i}^{new} = \left\{ \begin{array}{l} X_{j,i} + r_i\left(X_{k,i} - X_{j,i}\right), if\ f(X_j) > f(X_k) \\ X_{j,i} + r_i\left(X_{j,i} - X_{k,i}\right), if\ f(X_j) < f(X_k) \end{array} \right\} if\ FE \geq \frac{FE_{\max}}{RDF} \tag{7}$$

where $X_{j,i}^{new}$ represents the new updated solution; $j = 1, 2, \ldots, n$; $i = 1, 2, \ldots, m$; k indicates randomly chosen solutions; $j \neq k$ where $k \in (1, 2, \ldots, n)$; R_n is the random variable—it varies between 0.3333 and 0.6666; r_i is a random number in the range [0, 1]; R_n and r_i correspond to the radiation elements of the Stefan–Boltzmann law [42]; *FE* represents the function evaluations; and *RDF* is the radiation factor—it is set to a value of 2 to balance the exploration and exploitation in the radiation phase [42].

The search process of HTS algorithm is carried out through the three aforementioned phases, and the global optimum of a given problem can be estimated with these phases. In the HTS algorithm, every phase is divided into two sub-phases controlled by different factors of heat transfer phases and function evaluations. As the three phases of heat transfer almost have equivalent probability to transfer heat, the values of design variables fluctuate gradually or abruptly. The exploration and exploitation of a search space can be represented by the large or small changes of the design variables. The overall flowchart of the original HTS algorithm is illustrated in Figure 1.

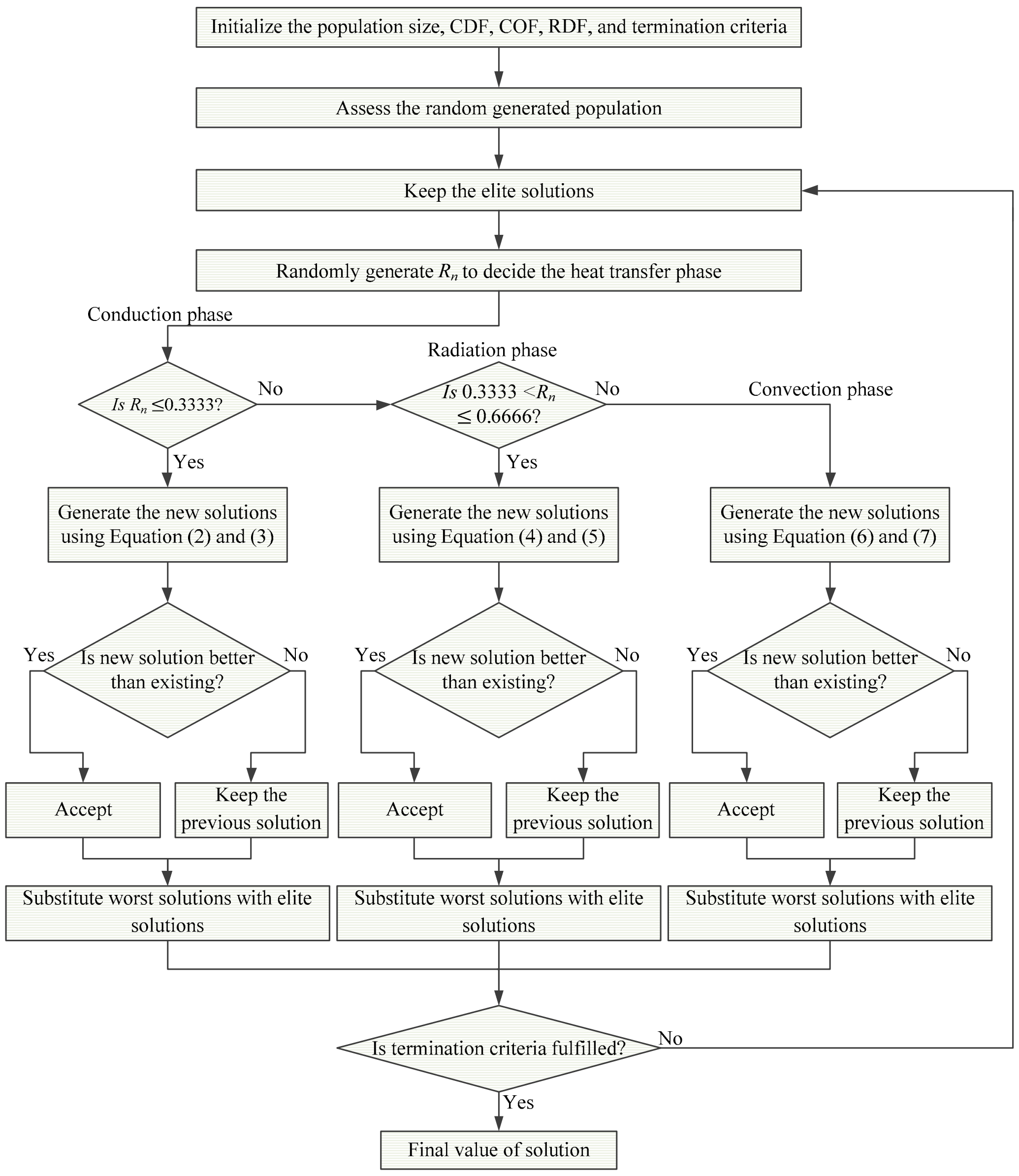

Initialize the population size, CDF, COF, RDF, and termination criteria
Assess the random generated population
Keep the elite solutions
Randomly generate R_n to decide the heat transfer phase
Conduction phase
Radiation phase
Convection phase
Is $R_n \leq 0.3333$?
Is $0.3333 < R_n \leq 0.6666$?
No
Yes
Generate the new solutions using Equation (2) and (3)
Generate the new solutions using Equation (4) and (5)
Generate the new solutions using Equation (6) and (7)
Is new solution better than existing?
Accept
Keep the previous solution
Substitute worst solutions with elite solutions
Is termination criteria fulfilled?
Final value of solution

Figure 1. Flowchart of the heat transfer search (HTS) algorithm. CDF; conduction factor, COF: convection factor, RDF: radiation factor, and R_n: random variable.

4. Quadratic Interpolation Based Simultaneous Heat Transfer Search (QISHTS) Algorithm

In this section, aiming at providing a more efficient HTS method with well-balanced exploration and exploitation capabilities, three modifications are introduced into the original HTS algorithm, namely the effect of simultaneous heat transfer search (SHTS), quadratic interpolation (QI) method, and population regeneration mechanism. Thus, a new version of HTS algorithm named the quadratic interpolation-based simultaneous heat transfer search (QISHTS) algorithm is proposed. In the QISHTS algorithm, the effect of a SHTS is employed to provide superior performance with lower computational complexity; the QI method is adopted to improve the exploitation capability, and the population regeneration mechanism is used to enhance the exploration capability. These improvements are described in detail in the subsequent subsections.

4.1. Simultaneous Heat Transfer Search (SHTS)

In the original HTS algorithm, the search mechanism is carried out through one of the three phases (conduction, convection, and radiation phase) in each iteration, where a random number (R_n) is generated between 0 and 1 in each iteration to determine which one of the three phases should be activated to update the new solutions. Thus, the population members are performed by either conduction phase, convection phase, or radiation phase in each generation, and the overall procedure is repeated until the stopping criterion is met. However, it should be obvious that randomly generating the probability R_n in each iteration will lead to increase the computational complexity; meanwhile, it does not guarantee a better solution. Besides, if any phase of the three phases gets more chances during the optimization process, it will remarkably affect the overall performance of the algorithm. Therefore, in order to overcome these drawbacks, the idea of a simultaneous heat transfer search (SHTS) is incorporated into the original HTS algorithm. It is worth mentioning that all the three phases are needed to be performed during the optimization process to get efficient functioning on the HTS [42]. Thus, in the SHTS, the process of generating the probability R_n in each iteration is cancelled, all the three phases are performed simultaneously, and the population members will undergo all the three phases. In other words, the entire population (NP) is divided into three sub-populations in each generation, and every sub-population is specified for one of the three phases of HTS algorithm. Let N_{CD}, N_{CV} and N_{RD} be the set of population members for conduction phase, convection phase, and radiation phase, respectively, where $N_{CD} + N_{CV} + N_{RD} = NP$. Hence, three new best solutions ($f(x_{CD})$, $f(x_{CV})$ and $f(x_{RD})$) will be generated by the three phases in each iteration. The simultaneous operation of performing the three phases in each iteration not only can help to provide superior performance; it can also help to decrease the computational time. Therefore, the introduction of SHTS into the basic HTS algorithm is beneficial to provide superior performance with lower computational complexity. More details about SHTS can be found in the literature [47].

4.2. Quadratic Interpolation Method

The quadratic interpolation (QI) is a nonlinear one-dimensional optimization operator, which utilizes a parabola for fitting the shape of the objective function close to the optima. This operator can reach the minimum in the initial space. This method was firstly applied in the improved controlled random search [50]. Subsequently, it has been integrated into various MHAs, such as differential evolution [51], genetic algorithms [52], and the Alopex-based evolutionary algorithm [53] for solving complex continuous optimization problems. Moreover, it was also combined with teaching-learning-based optimization [41] for handling dynamic optimization problems. The main principle of a QI operator is to use three known points (e.g., two randomly chosen members and the most outstanding member in the whole population) for the sake of forming a quadratic curve, which is used for approximating the shape of the objective function. Hence, the extreme point of the quadratic function can approximate the optima of the objective function [53]. Assume that $x_1 = \left(x_1^1, x_1^2, \ldots, x_1^D\right)$, $x_2 = \left(x_2^1, x_2^2, \ldots, x_2^D\right)$, and $x_3 = \left(x_3^1, x_3^2, \ldots, x_3^D\right)$ are the three selected members that use the QI method to generate a new member $\left(x_{QI}\right)$, then it can be calculated as follows:

$$x_{QI} = \left(x_{QI}^1, x_{QI}^2, \ldots., x_{QI}^D\right) \tag{8}$$

$$x_{QI}^d = 0.5\frac{[(x_3^d - x_2^d)^2 f(x_1)] + [(x_1^d - x_3^d)^2 f(x_2)] + [(x_2^d - x_1^d)^2 f(x_3)]}{[(x_3^d - x_2^d) f(x_1)] + [(x_1^d - x_3^d) f(x_2)] + [(x_2^d - x_1^d) f(x_3)]} \quad d = 1, 2, \ldots, D \tag{9}$$

where $f(x_1)$, $f(x_2)$ and $f(x_3)$ represent the fitness values of the three selected members, respectively.

In the proposed method, the three members (X_{CD}, X_{CV} and X_{RD}) obtained by the conduction phase, convection phase, and radiation phase in each iteration are selected to execute the QI method to generate a new member (X_{QI}). The new generated X_{QI} is evaluated; if it is superior to the existing one, then it will be accepted and used to replace the X_{worst}. In this method, the quadratic interpolation among different individual members can help to enhance the exploitation capability.

4.3. Population Regeneration Mechanism

In the proposed method, the solution extremely relies on the initial population members, which do not often contain a global optimum solution. Moreover, the solutions may tend to quickly group around local optima during the optimization process, which will lead to reduce the exploration capability of the algorithm. As a challenging task in various MHAs is how to alleviate the probability of local optima stagnation, therefore, the population regeneration mechanism is introduced into the proposed method to avoid local optima and to search for superior solutions. In the proposed mechanism, the population regenerator is utilized to regenerate the population when the fittest solution keeps identical without significant changes for a predefined number of function evaluations $(PDNFES)$. This regeneration mechanism is carried out by flip population way, and its mathematical formula is given as:

$$\mathrm{X}_{j,i}^{flip} = (L_{j,i} + U_{j,i}) - X_{j,i};\, if\ R_i \le PV \tag{10}$$

where $X_{j,i}^{flip}$ represents an opposite point of the population member $X_{j,i}$—it is produced by observing symmetry about the midpoint of its bound of ith design variable and jth population, respectively; L and U indicate the lower and upper bounds, respectively. R_i is a random number in the range [0, 1], and PV is the population flip probability factor—it is a specific value between 0 and 1.

The population regeneration process is applied to all the three phases; it can help to alleviate the probability of local optima stagnation and improve the exploration capability. In this mechanism, two important controlling parameters ($PDNFES$ and PV) play an important role in the performance of the proposed algorithm. Thus, the setting values of these two parameters are set differently among the optimization problems.

4.4. The Overall Process of QISHTS Algorithm

By simultaneously integrating the SHTS, QI method, and population regeneration mechanism to the original HTS algorithm, we present a new version of HTS algorithm called the QISHTS. The overall flowchart of the QISHTS algorithm is illustrated in Figure 2, and the overall process of the QISHTS algorithm is described in detail below:

- **Step 1.** Define the population size (NP), the function optimization goal, the maximum number of function evaluations $(FE_{\max})$, the setting values of two controlling parameters ($PDNFES$ and PV), the current function evaluations $CFES = 0$, and the stopping criteria.
- **Step 2.** Randomly generate the main population $x_i, i = 1, 2, \ldots, NP$ within the lower and upper boundaries $[L, U]$ of the decision variables and calculate the objective function values $f(x_i), i = 1, 2, \ldots, NP$.
- **Step 3.** Determine the best individual (X_{best}, f_{best}) in the population according to the fitness.
- **Step 4.** Randomly divide the whole population members into three sub-populations (N_{CD}, N_{CV} and N_{RD}) and assign every sub-population to one of the three phases, where N_{CD}, N_{CV} and N_{RD} are the set of population members for the conduction, convection, and radiation phase, respectively.
- **Step 5.** Generate the new solutions by the conduction phase, convection, and radiation phase.
  Step 6. Select the three best members (X_{CD}, X_{CV} and X_{RD}) obtained by the conduction phase, convection phase, and radiation phase to generate a new member (X_{QI}) by the QI method according to Equations (8) and (9).
- **Step 7.** Bound the new solution to the search domain.
- **Step 8.** Evaluate the new solution; if the new obtained solution is superior to the existing one, then accept it and use it to replace the worst one.
- **Step 9.** If the stopping criteria is met, then output the final solution. Otherwise, return to step 3.

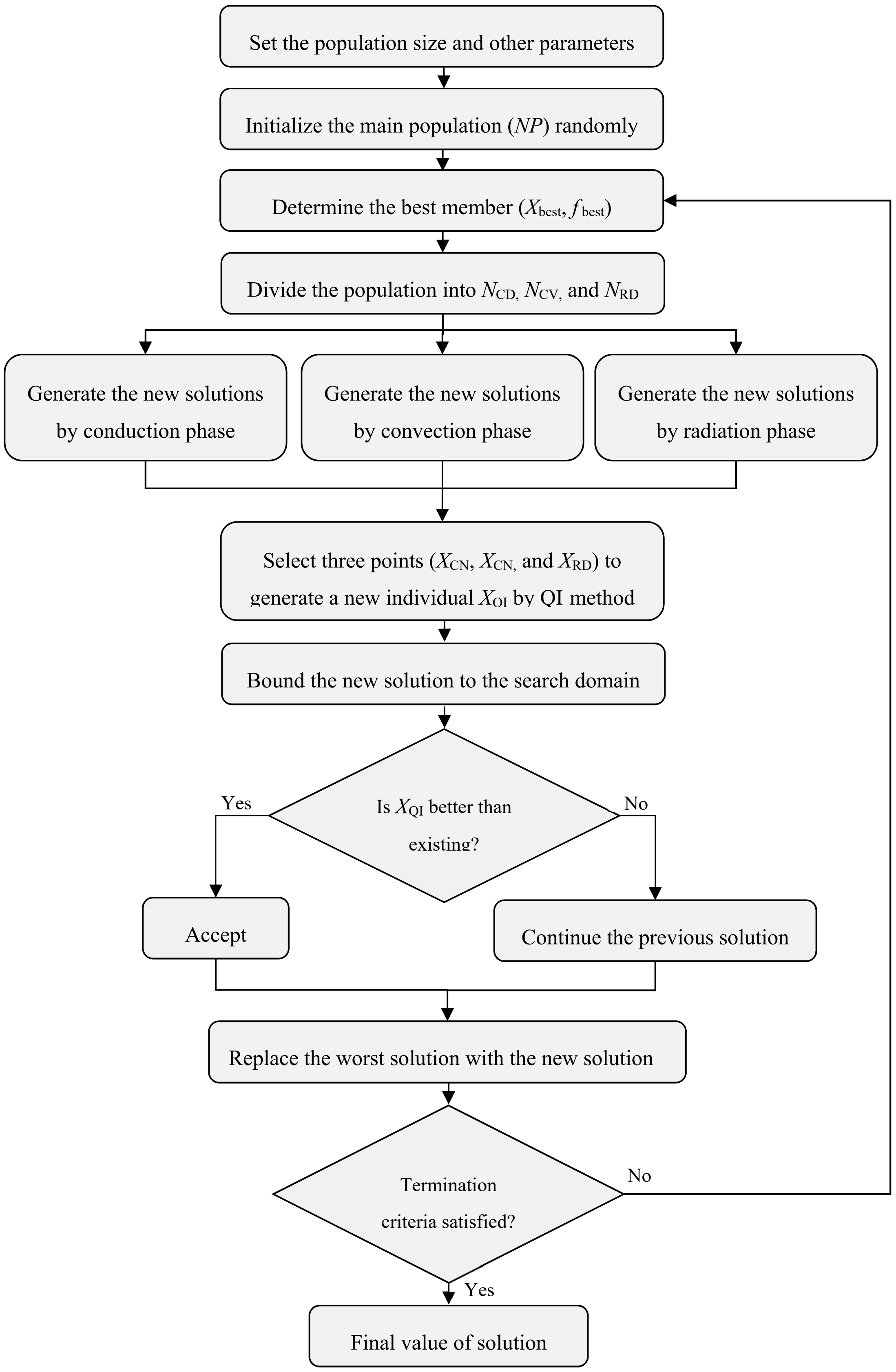

Set the population size and other parameters
Initialize the main population (NP) randomly
Determine the best member ($X_{\text{best}}, f_{\text{best}}$)
Divide the population into N_{CD}, N_{CV}, and N_{RD}
Generate the new solutions by conduction phase
Generate the new solutions by convection phase
Generate the new solutions by radiation phase
Select three points (X_{CN}, X_{CN}, and X_{RD}) to generate a new individual X_{OI} by QI method
Bound the new solution to the search domain
Is X_{QI} better than existing?
Yes
No
Accept
Continue the previous solution
Replace the worst solution with the new solution
Termination criteria satisfied?
No
Yes
Final value of solution

Figure 2. Flowchart of the quadratic interpolation based simultaneous heat transfer search (QISHTS) algorithm.

5. Numerical Experiments and Discussions

In this section, in order to investigate the performance of the QISHTS algorithm, it is experimented by a set of well-defined traditional benchmark problems, and the computational results obtained by this algorithm are compared with those of other well-established MHAs. Moreover, to further evaluate the overall effectiveness of the new variant, comparisons between QISHTS algorithm and several state-of-the-art DEs are also conducted. The competitors were tested on these benchmark problems previously, and these problems offer challenge to an MHA. Hence, these benchmark problems are employed for the experimentation, and the above-mentioned algorithms are considered for the comparison with the proposed variant. The detailed mathematical formulation of each function with its dimension, range value, and optimum value are illustrated in Table 1. These benchmark problems involve different characteristics and can be classified into three types: $f_1 - f_6$ are unimodal functions, $f_7 - f_{11}$ are multi-modal high-dimensional functions, and $f_{12} - f_{18}$ are multimodal low-dimensional functions. The considered benchmark functions are appropriate for testing the exploration and exploitation capability of an MHA.

Table 1. The traditional benchmark problems used in investigation.

Test Function	Dimension	Range	Optimum
$f_1 = \sum_{i=1}^{n} x_i^2$	30	[−100, 100]	0
$f_2 = \sum_{i=1}^{n} \lvert x_i \rvert + \prod_{i=1}^{n} \lvert x_i \rvert$	30	[−10, 10]	0
$f_3 = \sum_{i=1}^{n} \left(\sum_{j=1}^{i} x_j\right)^2$	30	[−100, 100]	0
$f_4 = \max_i \{\lvert x_i \rvert, 1 \le i \le n\}$	30	[−100, 100]	0
$f_5 = \sum_{i=1}^{n} (\lfloor x_i + 0.5 \rfloor)^2$	30	[−100, 100]	0
$f_6 = \sum_{i=1}^{n} i x_i^4 + \text{random}[0, 1]$	30	[−1.28, 1.28]	0
$f_7 = \sum_{i=1}^{n-1} \left[100\left(x_{i+1} - x_i^2\right)^2 + (x_i - 1)^2\right]$	30	[−30, 30]	0
$f_8 = \sum_{i=1}^{n} -x_i \sin\left(\sqrt{\lvert x_i \rvert}\right)$	30	[−500, 500]	-418.98×10^5
$f_9 = \sum_{i=1}^{n} \left[x_i^2 - 10\cos(2\pi x_i) + 10\right]$	30	[−5.12, 5.12]	0
$f_{10} = -20\exp\left(-0.2\sqrt{\frac{1}{n}\sum_{i=1}^{n} x_i^2}\right) - \exp\left(\frac{1}{n}\sum_{i=1}^{n} \cos 2\pi x_i\right) + 20 + e$	30	[−32, 32]	0
$f_{11} = \frac{1}{4000}\sum_{i=1}^{n} x_i^2 - \prod_{i=1}^{n} \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$	30	[−600, 600]	0
$f_{12} = \left[\frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2} (x_i - a_{ij})^6}\right]^{-1}$	2	[−65, 65]	9.9800×10^{-1}
$f_{13} = \sum_{i=1}^{11} \left[a_i - \frac{x_i\left(b_i^2 + b_i x_2\right)}{b_i^2 + b_i x_3 + x_4}\right]^2$	4	[−5, 5]	3.0000×10^{-4}
$f_{14} = 4x_1^2 - 2.1x_1^4 + x_1^6/3 + x_1 x_2 - 4x_2^2 + 4x_2^4$	2	[−5, 5]	−1.0316
$f_{15} = \left(x_2 - \frac{5.1}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 + 10\left(1 - \frac{1}{8\pi}\right)\cos x_1 + 10$	2	[−5, 5]	3.9800×10^{-1}
$f_{16} = \left[1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)\right]$ $\times \left[30 + (2x_1 - 3x_2)^2(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)\right]$	2	[−2, 2]	3
$f_{17} = -\sum_{i=1}^{4} c_i \exp\left[-\sum_{j=1}^{3} a_{ij}(x_j - p_{ij})^2\right]$	3	[1, 3]	−3.86
$f_{18} = -\sum_{i=1}^{4} c_i \exp\left[-\sum_{j=1}^{6} a_{ij}(x_j - p_{ij})^2\right]$	6	[0, 1]	−3.322

5.1. Comparison of QISHTS with MHAs

In this section, to investigate the performance of the QISHTS algorithm, the results obtained by QISHTS algorithm are compared with the results of other MHAs, such as cuckoo search (CS) [54], gravitational search algorithm (GSA) [55], artificial bee colony (ABC) [56], animal migration optimization (AMO) [57], heat transfer search (HTS) [42], and improved heat transfer search (IHTS) [44]. To conduct a fair comparison with the considered competitors, a common experimental platform is needed. Thus, the maximum number of function evaluations (*FEmax*) is set as follows: 150,000 for functions f_1, f_5, and f_{10}; 200,000 for functions f_2 and f_{11}; 500,000 for functions f_3, f_4, and f_7; 300,000 for functions f_6, f_8, and f_9; 10,000 for functions f_{12}, f_{14}, f_{15}, and f_{17}; 40,000 for function f_{13}; 3000 for function f_{16}; and 20,000 for function f_{18}. Furthermore, the population size (*NP*) is set to 50, and each method is tested 25 times independently for each benchmark problem. The experimental parameters settings and results of the competitors are derived from the literature [44]. The controlling parameters of the QISHTS algorithm are set as follows: the population flip probability factor (*PV*) is set to 0.1 for all the test functions; the predefined number of function evaluations (*PDNFES*) is set as 1000 for the test functions whose overall function evaluations (*FEs*) are larger than 100,000, and 150 for the test functions whose overall *FEs* are less than 50,000. The best results, such as the best *Mean* and *Std* obtained by the compared algorithms, are bolded. Besides, each algorithm is ranked between 1 and 7 in accordance to its corresponding results; the algorithm which obtains the perfect result is ranked 1, whereas the algorithm with the worst result is ranked 7.

Table 2 displays the comparative results of f_1 to f_6 for the QISHTS algorithm with other algorithms; it can be seen from the results table that the QISHTS, IHTS, and the basic HTS algorithm has achieved the greatest results on functions f_1 and f_2, followed by the other algorithms. Furthermore, the HTS and QISHTS algorithms can obtain the best solution on f_3. On functions f_4 and f_6, the QISHTS performs significantly superior to the other competitive methods. On function f_5, the QISHTS and AMO algorithms achieve global mean values. Comparing with IHTS and the original HTS algorithm, the performance of QISHTS is improved on f_3, f_4, f_5, and f_6. From the mean rank and final rank of all competitive methods, it can be seen that the QISHTS algorithm shows the best overall effectiveness on unimodal functions, followed by the IHTS, HTS, and the rest of the algorithms. Hence, it can be observed from the results that the exploitation capability of the QISHTS algorithm is enhanced by the proposed improvements.

Table 2. Computational results of f_1–f_6 for the QISHTS algorithm with other meta-heuristic algorithms (MHAs) in Investigation 1. (The results of the competitors are adapted from the literature [44]). CS: cuckoo search, GSA: gravitational search algorithm, AMO: animal migration optimization, ABC: artificial bee colony, HTS: heat transfer search, IHTS: improved heat transfer search, and QISHTS: quadratic interpolation based simultaneous heat transfer search.

Function		CS	GSA	ABC	AMO	HTS	IHTS	QISHTS
f_1	Mean	5.6565×10^{-6}	3.3748×10^{-18}	2.9860×10^{-20}	8.6464×10^{-40}	**0.0000**	**0.0000**	**0.0000**
	STD	2.8611×10^{-6}	8.0862×10^{-19}	2.1455×10^{-20}	1.0435×10^{-39}	**0.0000**	**0.0000**	**0.0000**
	Rank	7	6	5	4	1	1	1
f_2	Mean	2.0000×10^{-3}	8.9212×10^{-9}	1.4213×10^{-15}	8.2334×10^{-32}	**0.0000**	**0.0000**	**0.0000**
	STD	8.0959×10^{-4}	1.3340×10^{-9}	5.5340×10^{-16}	3.4120×10^{-32}	**0.0000**	**0.0000**	**0.0000**
	Rank	7	6	5	4	1	1	1
f_3	Mean	1.1400×10^{-3}	1.1260×10^{-1}	2.4027×10^{3}	8.8904×10^{-4}	**0.0000**	2.5658×10^{-36}	**0.0000**
	STD	6.0987×10^{-4}	1.2660×10^{-1}	6.5696×10^{2}	8.7256×10^{-4}	**0.0000**	7.3407×10^{-36}	**0.0000**
	Rank	5	6	7	4	1	3	1
f_4	Mean	3.2388	9.9302×10^{-10}	1.8523×10^{1}	2.8622×10^{-5}	7.8547×10^{-43}	2.3219×10^{-33}	**0.0000**
	STD	6.6440×10^{-1}	1.1899×10^{-10}	4.2477	2.3468×10^{-5}	2.1944×10^{-42}	4.9053×10^{-33}	**0.0000**
	Rank	6	4	7	5	2	3	1
f_5	Mean	5.4332×10^{-6}	3.3385×10^{-18}	3.0884×10^{-20}	**0.0000**	1.3920×10^{-1}	1.3906×10^{-7}	**0.0000**
	STD	2.2446×10^{-6}	5.6830×10^{-19}	4.0131×10^{-20}	**0.0000**	4.5100×10^{-1}	2.7467×10^{-7}	**0.0000**
	Rank	6	4	3	1	7	5	1
f_6	Mean	9.6000×10^{-3}	3.900×10^{-3}	3.2400×10^{-2}	1.7000×10^{-3}	1.7370×10^{-3}	1.5430×10^{-3}	$\mathbf{1.0744 \times 10^{-4}}$
	STD	2.8000×10^{-3}	1.3000×10^{-3}	5.9000×10^{-3}	4.7058×10^{-4}	7.5904×10^{-4}	6.3685×10^{-4}	$\mathbf{8.9991 \times 10^{-5}}$
	Rank	6	5	7	3	4	2	1
Mean Rank		6.16	5.16	5.66	3.50	2.66	2.50	1.00
Final Rank		7	5	6	4	3	2	1

Table 3 shows the comparative results of f_7 to f_{11} for the QISHTS algorithm with the other algorithms; it can be observed from the results table that, on function f_7, the ABC algorithm can find the best results, followed by other algorithms, while the IHTS and the HTS algorithms do not achieve the best results, whereas the QISHTS algorithm has shown more success on this function. Moreover, the QISHTS, IHTS, HTS, AMO, and ABC algorithms can find the best solution on functions f_9 and f_{11}. On functions f_8 and f_{10}, the QISHTS performs significantly superior to the other competitive methods. Comparing with IHTS and the original HTS algorithm, the performance of the QISHTS is improved on f_7, f_8, and f_{10}. From the mean rank and final rank of each competitor, it can be seen that the QISHTS shows the greatest overall effectiveness, followed by the AMO, IHTS, ABC, and the rest of the algorithms. Thus, it can be observed from the results that the exploration ability of the QISHTS algorithm is enhanced by the proposed improvements.

Table 3. Computational results of f_7 to f_{11} for the QISHTS algorithm with other MHAs in Investigation 1. (The results of the competitors are adapted from the literature [44]).

Function		CS	GSA	ABC	AMO	HTS	IHTS	QISHTS
f_7	Mean	8.0092	2.0082×10^{1}	$\mathbf{4.4100 \times 10^{-2}}$	4.1817	2.6156×10^{1}	2.2876×10^{1}	2.0024×10^{1}
	STD	1.9188	1.7220×10^{-1}	$\mathbf{7.0700 \times 10^{-2}}$	2.1618	3.0311	4.3741	8.1310×10^{-1}
	Rank	3	5	1	2	7	6	4
f_8	Mean	-9.1492×10^{3}	-3.0499×10^{3}	-1.2507×10^{4}	-1.2569×10^{4}	-1.2569×10^{4}	-1.2569×10^{4}	$\mathbf{-5.9959 \times 10^{4}}$
	STD	2.5314×10^{2}	3.3886×10^{2}	6.1118×10^{1}	1.2384×10^{-7}	5.7799×10^{-1}	$\mathbf{3.7130 \times 10^{-12}}$	1.1102
	Rank	6	7	5	4	3	2	1
f_9	Mean	5.1220×10^{1}	7.2831	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
	STD	8.1069	1.8991	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
	Rank	7	6	1	1	1	1	1
f_{10}	Mean	2.3750	1.4717×10^{-9}	1.1946×10^{-9}	4.4409×10^{-15}	2.8741×10^{-14}	2.8599×10^{-14}	$\mathbf{8.8818 \times 10^{-16}}$
	STD	1.1238	1.4449×10^{-10}	5.0065×10^{-10}	**0.0000**	5.6806×10^{-15}	4.3511×10^{-15}	**0.0000**
	Rank	7	6	5	2	4	3	1
f_{11}	Mean	4.4900×10^{-5}	1.265×10^{-2}	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
	STD	8.9551×10^{-5}	2.1600×10^{-2}	**0.0000**	**0.0000**	**0.0000**	**0.0000**	**0.0000**
	Rank	6	7	1	1	1	1	1
Mean Rank		5.80	6.20	2.60	2.00	3.20	2.60	1.60
Final Rank		6	7	3	2	5	3	1

Table 4 displays the comparative results of f_{12} to f_{18} for the QISHTS algorithm with the other algorithms; it is observed that, the QISHTS, IHTS, HTS, AMO, and ABC algorithms can find the best solution on function f_{12}. On function f_{13}, the QISHTS ranks first, followed by the rest of the algorithms, while the IHTS and the HTS algorithms do not achieve the best results, whereas the QISHTS algorithm has shown more success on this function. Moreover, the QISHTS, IHTS, HTS, AMO, and ABC algorithms can find the best solution on functions f_9 and f_{11}. On functions f_8 and f_{10}, the QISHTS performs significantly superior to the other competitive methods. Comparing with IHTS and the original HTS algorithm, the performance of the QISHTS is improved on f_7, f_8, and f_{10}. From the mean rank and final rank of each competitor, it can be seen that the IHTS shows the greatest overall effectiveness, followed by the QISHTS, HTS, and the rest of the algorithms. Hence, it can be observed from the results that the exploration ability of the QISHTS algorithm is enhanced by the proposed improvements.

Table 4. Computational results of f_{14}–f_{23} for the QISHTS algorithm with other MHAs in Investigation 1. (The results of the competitors are adapted from the literature [44]).

Function		CS	GSA	ABC	AMO	HTS	IHTS	QISHTS
f_{12}	Mean	9.9810×10^{-1}	5.9533	$\mathbf{9.9800 \times 10^{-1}}$	$\mathbf{9.9800 \times 10^{-1}}$	$\mathbf{9.9800 \times 10^{-1}}$	$\mathbf{9.9800 \times 10^{-1}}$	$\mathbf{9.9800 \times 10^{-1}}$
	STD	4.8277×10^{-4}	3.4819	3.7921×10^{-16}	3.3858×10^{-12}	$\mathbf{3.3993 \times 10^{-16}}$	$\mathbf{3.3993 \times 10^{-16}}$	$\mathbf{3.3993 \times 10^{-16}}$
	Rank	6	7	4	5	1	1	1
f_{13}	Mean	5.0310×10^{-4}	4.8000×10^{-3}	7.4715×10^{-4}	3.9738×10^{-4}	5.3397×10^{-4}	3.4416×10^{-4}	$\mathbf{3.1988 \times 10^{-4}}$
	STD	1.1180×10^{-4}	3.3000×10^{-3}	2.1481×10^{-4}	4.4503×10^{-5}	3.9577×10^{-4}	1.8313×10^{-4}	$\mathbf{3.0497 \times 10^{-5}}$
	Rank	4	7	6	3	5	2	1
f_{14}	Mean	**−1.03163**	**−1.03163**	−1.0316	−1.0316	**−1.03163**	**−1.03163**	**−1.03163**
	STD	1.5821×10^{-7}	4.7536×10^{-16}	1.1269×10^{-14}	5.2006×10^{-11}	$\mathbf{4.5325 \times 10^{-16}}$	$\mathbf{4.5325 \times 10^{-16}}$	6.5323×10^{-8}
	Rank	7	3	4	5	1	1	6
f_{15}	Mean	$\mathbf{3.9790 \times 10^{-1}}$	$\mathbf{3.9790 \times 10^{-1}}$	$\mathbf{3.9790 \times 10^{-1}}$	$\mathbf{3.9790 \times 10^{-1}}$	3.9789×10^{-1}	3.9789×10^{-1}	$\mathbf{3.9790 \times 10^{-1}}$
	STD	3.2449×10^{-6}	**0.0000**	5.3819×10^{-8}	**0.0000**	5.6656×10^{-17}	5.6656×10^{-17}	5.5872×10^{-17}
	Rank	7	1	6	1	**4**	**4**	3
f_{16}	Mean	3.0013	3.7403	**3.0000**	3.0018	**3.0000**	**3.0000**	**3.0000**
	STD	2.6000×10^{-3}	1.6055	2.6164×10^{-5}	5.5000×10^{-3}	**0.0000**	**0.0000**	3.7456×10^{-4}
	Rank	5	7	3	6	1	1	4
f_{17}	Mean	**−3.8628**	−3.8625	**−3.8628**	**−3.8628**	**−3.8628**	**−3.8628**	**−3.8628**
	STD	1.4043×10^{-5}	3.8767×10^{-4}	1.3654×10^{-10}	1.3669×10^{-15}	9.0649×10^{-16}	9.0649×10^{-16}	$\mathbf{1.6997 \times 10^{-16}}$
	Rank	6	7	5	4	2	2	1
f_{18}	Mean	−3.321	**−3.3220**	**−3.3220**	**−3.3220**	−3.2837	−3.2935	−3.3228
	STD	7.5186×10^{-4}	$\mathbf{4.7967 \times 10^{-16}}$	8.5733×10^{-10}	5.085×10^{-6}	5.6553×10^{-2}	5.1824×10^{-2}	4.8300×10^{-2}
	Rank	4	1	2	3	7	6	5
Mean Rank		5.57	4.71	4.28	3.85	3.00	2.42	3.00
Final Rank		7	6	5	4	2	1	2

5.2. Comparison of QISHTS with State-Of-The-Art DEs

In order to further observe the overall effectiveness of the proposed variant, another comparison between the QISHTS algorithm with well-established state-of-the-art DEs is conducted. These methods, such as opposition-based differential evolution (ODE) [58], self-adaptive differential evolution (SaDE) [59], modified differential evolution (MoDE) [60], composite differential evolution (CoDE) [61], and modified differential evolution with *p*-best crossover (MDE-*p*BX) [62], have efficient robustness and global search ability. The experimental results of the competitors are acquired from [63]. To be consistent with the reference, a common experimental platform is required. Thus, the population size (*NP*) is set at 100, the dimension (*D*) of each benchmark problem is set at 30, as well as each method is repeated 25 times independently for all test problems. Moreover, the maximum number of function evaluations (*maxFEs*) is set differently for each function, as given in Table 5. The controlling parameters of the QISHTS algorithm after conducting many trials are set as: *PDNFES* = 10^3 and *PV* = 0.1. The best mean (*Mean*) and standard deviations (*Std*) results obtained by the compared methods are bolded.

Table 5. Computational results for the QISHTS algorithm with the state-of-the art different evolutions (DEs) in Investigation 2. (The results of the competitors are adapted from the literature [63]). FEs: function evaluations, ODE: opposition-based differential evolution, SaDE: self-adaptive differential evolution, MoDE: modified differential evolution, CoDE: composite differential evolution, MDE-*p*BX: modified differential evolution with *p*-best crossover, and QISHTS: quadratic interpolation based simultaneous heat transfer search.

F	FEs		ODE	SaDE	MoDE	CoDE	MDE-*p*BX	QISHTS
f_1	150,000	Mean	2.24×10^{-27}	1.62×10^{-48}	6.51×10^{-4}	2.28×10^{-7}	1.24×10^{-3}	**0.00**
		STD	2.72×10^{-27}	4.46×10^{-48}	2.81×10^{-3}	8.29×10^{-8}	4.83×10^{-3}	**0.00**
		Rank	3	2	5	4	6	1
f_2	150,000	Mean	1.52×10^{-11}	6.11×10^{-45}	2.89×10^{-3}	1.16×10^{-7}	4.17×10^{-8}	**0.00**
		STD	1.06×10^{-11}	1.54×10^{-44}	1.23×10^{-2}	3.79×10^{-8}	1.85×10^{-7}	**0.00**
		Rank	3	2	6	5	4	1
f_5	150,000	Mean	2.32×10^{4}	1.00×10^{4}	6.05×10^{2}	6.25×10^{4}	8.12×10^{3}	**0.00**
		STD	1.76×10^{3}	1.65×10^{3}	5.43×10^{2}	1.91×10^{3}	1.18×10^{4}	**0.00**
		Rank	5	4	2	6	3	1
f_7	500,000	Mean	2.52×10^{1}	1.28×10^{1}	4.51	3.33	4.70×10^{1}	**0.00**
		STD	1.1000	5.86	5.30	1.96	3.15×10^{1}	**0.00**
		Rank	5	4	3	2	6	1
f_8	300,000	Mean	6.99×10^{3}	3.55×10^{1}	4.14×10^{3}	$\mathbf{1.21 \times 10^{-12}}$	1.22×10^{3}	3.74×10^{-4}
		STD	3.08×10^{2}	6.34×10^{1}	6.51×10^{2}	8.72×10^{-13}	4.34×10^{2}	**0.00**
		Rank	6	3	5	1	4	**2**
f_9	300,000	Mean	3.60×10^{1}	1.43	6.01×10^{1}	1.02×10^{-11}	7.80	**0.00**
		STD	1.91×10^{1}	1.13	1.41×10^{1}	1.73×10^{-11}	2.25	**0.00**
		Rank	5	3	6	2	4	1
f_{10}	150,000	Mean	3.53×10^{-14}	4.02×10^{-15}	7.70	1.31×10^{-4}	1.77×10^{-2}	$\mathbf{8.88 \times 10^{-16}}$
		STD	2.04×10^{-14}	6.49×10^{-16}	1.82	3.74×10^{-5}	9.30×10^{-2}	**0.00**
		Rank	3	2	6	4	5	1
f_{11}	200,000	Mean	2.47×10^{-4}	2.79×10^{-3}	2.49×10^{-1}	4.19×10^{-9}	4.92×10^{-3}	**0.00**
		STD	1.35×10^{-3}	6.61×10^{-3}	2.33×10^{-1}	3.72×10^{-9}	1.24×10^{-2}	**0.00**
		Rank	3	4	6	2	5	1
	Mean Rank		4.12	3	4.87	3.25	4.62	1.12
	Final Rank		4	2	6	3	5	1

It can be seen from Table 5 that, although the SaDE algorithm can find the greatest result among DEs on functions f_1, f_2, and f_{10}, the QISHTS algorithm shows more success and ranks first on all these benchmark functions. The MoDE algorithm can obtain the best results among the DEs on function f_5, but it wins the second place after the QISHTS algorithm, which ranks first on this function. On functions f_7, f_9, and f_{11}, CoDE can find the best results among the DEs, but the QISHTS shows more success and wins first on these functions. From the mean rank and final rank of each competitor, it can be observed that the QISHTS ranks first on the whole benchmark functions; it performs significantly better than other competitive DEs, which prove that the proposed algorithm possesses efficient robustness for handling optimization problems.

6. Application to Chemical Dynamic System Optimization

In this section, in order to further evaluate the effectiveness of the QISHTS method in solving real-world optimization problems, the proposed algorithm is used to handle four chemical DOPs (two dynamic parameter estimation problems and two optimal control problems), involving different levels of difficulty. Moreover, the results obtained by the QISHTS algorithm are compared with those of twelve well-established algorithms, such as linearly decreasing inertia weight particle swarm optimization (LDWPSO) [64], comprehensive learning particle swarm optimization (CLPSO) [65], differential evolution with self-adapting control parameters (jDE) [66], artificial bee colony (ABC) [56], self-adaptive differential evolution (SaDE) [59], real-coded biogeography-based optimization (RCBBO) [67], hybrid biogeography-based optimization with differential evolution (DEBBO) [68], teaching-learning-based optimization (TLBO) [69], nonlinear inertia weighted teaching-learning-based optimization (NIWTLBO) [70], biogeography-based learning particle swarm optimization (BLPSO) [71], generalized oppositional teaching-learning-based optimization (GOTLBO) [72], and quadratic interpolation based teaching-learning-based optimization (QITLBO) [41]. These methods are very well-known due to their good performance, as well as they were examined on these DOPs previously in the literature [41]; thus, they were selected for comparison with the proposed method. The experimental parameter settings of the first twelve algorithms refer to corresponding literatures. To conduct a fair comparison, the population size (*NP*) is set to 50, and each algorithm is tested 30 times independently for the all problems. To be consistent with the references, the maximum number of function evaluations (maxFEs), as well as the acceptable precision (*Ap*), are used for recording the performance criteria, and these parameter values are set differently among the DOPs, as listed in Table 6. Moreover, the controlling parameters of the QISHTS algorithm (*PDNFES* and *PV*) after conducting many trials are also set differently for each problem, as shown in Table 6. Some performance criterions are used for the effectiveness comparisons; these criterions, such as the best, mean, and worst objective function values and also the standard deviations (*Std*) obtained by the competitive algorithms on 30 independent runs, are listed. Besides, the success rate (*SR*) is given to show the ability of the algorithm in obtaining an acceptable solution with precision *Ap* before the maximum number of function evaluations is reached. Furthermore, the execution time (T) is also given to display the runtime complexity of all competitors. In order to facilitate the comparisons, the greatest results among the competitive methods are bolded. Moreover, each algorithm is ranked between 1 and 13 in accordance to its corresponding results; the algorithm which obtains the perfect result is ranked 1, whereas the algorithm with the worst result is ranked 13.

Table 6. Parameter setting values for recording the performance criterion. maxFEs: maximum number of function evaluations, *Ap*: acceptable precision, *PDNFES*: predefined number of function evaluations, *PV*: population flip probability factor, and CSTR: continuous stirred tank reactor.

Dynamic Optimization Problem	***maxFEs***	***Ap***	***PDNFES***	***PV***
First-order reversible series reaction problem	1000	0.000002	10	0.1
Catalytic cracking of gas oil problem	1000	0.0030	10	0.1
Multi-modal CSTR problem	10,000	0.1400	100	0.3
Six-plate gas absorption tower problem	10,000	0.1125	100	0.3

6.1. Dynamic Parameter Estimation Problems

In this subsection, the proposed QISHTS algorithm is applied for solving two dynamic parameter estimation problems. Moreover, the computational results are compared with those of twelve well-established algorithms.

6.1.1. First-Order Reversible Series Reaction Problem

This problem indicates the first-order reversible chain reaction [16,73]. In this reaction system ($A \overset{K1}{\rightarrow} B \overset{K2}{\rightarrow} C$), three parts (A, B, and C) are included, since only the concentrations of the first two parts (A with B) were measured, while the last part (C) is not available in the model used for estimation. The mathematical equation of this model takes the form as follows:

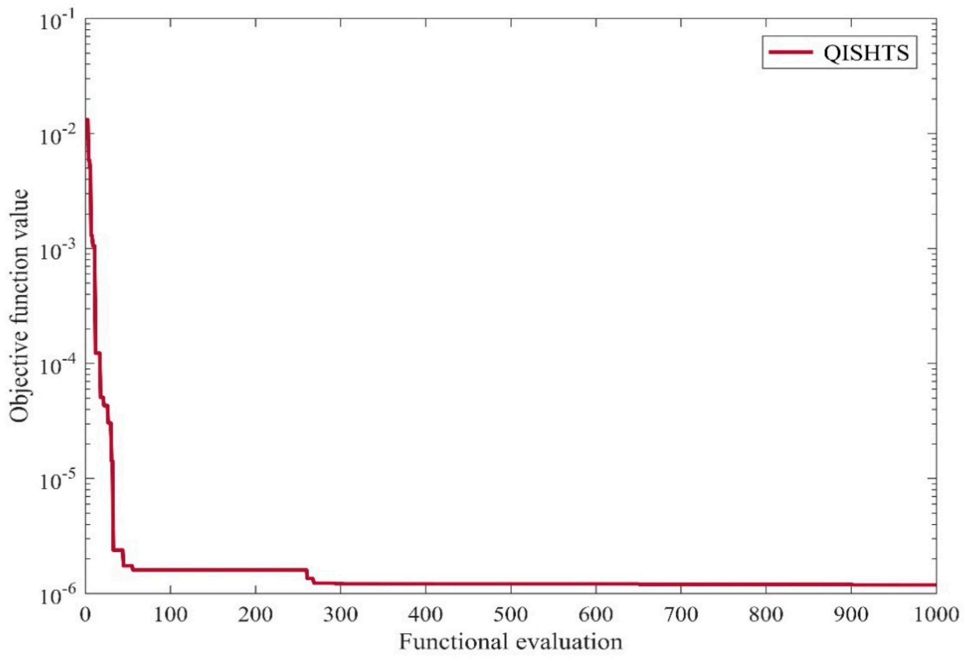
$$\min_{\theta} \sum_{\mu=1}^{10} \sum_{i=1}^{2} \left(\hat{z}_{\mu,i} - \widetilde{z}_{\mu,i}\right)^2$$
$$S.t. \begin{cases} \dot{z}_1 = -\theta_1 z_1 \\ \dot{z}_2 = \theta_1 z_1 - \theta_2 z_2 \\ z_0 = (1,0),\ t \in [0,1] \\ [0,1] \le \theta \le [10,10] \end{cases} \tag{11}$$

where $\hat{z}_\mu$ and $\widetilde{z}_\mu$ are defined as the vectors of the fitted point and experimental values, respectively, z_1 and z_2 represent the mole fractions of the first two parts (A and B), respectively, and θ_1 and θ_2 indicate the parameter vectors of the first and second reactions, respectively. The measurement data of this problem are derived from [4].

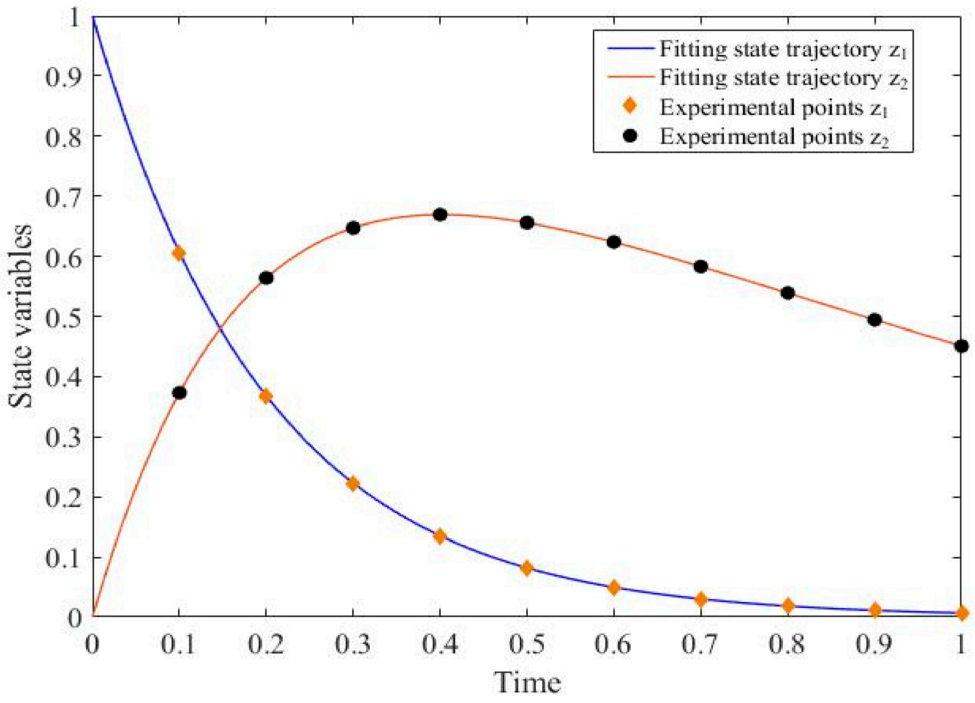
Table 7 displays the comparative results obtained by the QISHTS algorithm and other competitors for this problem. It can be seen from results table that the QISHTS algorithm can find the best results in terms of best, mean, std, and worst objective function value among all the comparative algorithms, followed by the QITLBO, NIWTLBO, ABC, and other competitors. While Figure 3a displays the convergence graph of the proposed algorithm, from Figure 3a and Table 7, it is obvious that the QISHTS algorithm has the best robustness, precision, and stability in performance. Figure 3b displays the fitting state trajectories with experimental points, which implies that a good agreement is obtained between the fitting state trajectories and the experimental points for this problem. Referring to the success rate, the QISHTS algorithm achieves a value of 100%, which means that it can successfully find trusted solutions in all the total runs. Considering the execution time, GOTLBO ranks first, followed by DEBBO and JDE, while the QISHTS takes 17.26 s to obtain the corresponding results.

Table 7. Results of the compared algorithms for the first-order reversible series reaction model. Best: the best objective function value, Mean: the mean objective function value, STD: the standard deviation, Worst: the worst objective function value, SR: success rate, and T(S): time (second).

Method	Best	Mean	STD	Worst	SR	T(S)	Rank
LDWPSO	1.18620×10^{-6}	1.25679×10^{-6}	8.30596×10^{-8}	1.47490×10^{-6}	**100%**	18.03	11
CLPSO	3.73979×10^{-6}	1.98549×10^{-4}	1.54809×10^{-4}	5.19505×10^{-4}	0.00%	17.97	12
jDE	1.18585×10^{-6}	1.18586×10^{-6}	1.43989×10^{-11}	1.18589×10^{-6}	**100%**	15.60	5
ABC	$\mathbf{1.18584 \times 10^{-6}}$	**1.18585** $\times 10^{-6}$	2.52200×10^{-11}	1.18597×10^{-6}	**100%**	17.52	4
SaDE	1.18585×10^{-6}	1.18643×10^{-6}	1.11590×10^{-9}	1.19024×10^{-6}	**100%**	17.90	8
RCBBO	9.85416×10^{-6}	1.68216×10^{-3}	2.87773×10^{-3}	9.61824×10^{-3}	0.00%	18.23	13
DEBBO	1.18585×10^{-6}	1.18664×10^{-6}	1.55322×10^{-9}	1.19254×10^{-6}	**100%**	15.46	10
TLBO	1.18585×10^{-6}	1.18598×10^{-6}	2.84734×10^{-10}	1.18703×10^{-6}	**100%**	15.62	6
NIWTLBO	$\mathbf{1.18584 \times 10^{-6}}$	$\mathbf{1.18585 \times 10^{-6}}$	5.14840×10^{-11}	1.18613×10^{-6}	**100%**	18.92	3
BLPSO	1.18585×10^{-6}	1.18650×10^{-6}	1.13912×10^{-9}	1.19090×10^{-6}	**100%**	18.06	9
GOTLBO	1.18585×10^{-6}	1.18611×10^{-6}	5.20662×10^{-10}	1.18805×10^{-6}	**100%**	**15.29**	7
QITLBO	$\mathbf{1.18584 \times 10^{-6}}$	1.18585×10^{-6}	2.06093×10^{-12}	1.18586×10^{-6}	**100%**	17.48	2
QISHTS	$\mathbf{1.18584 \times 10^{-6}}$	$\mathbf{1.18584 \times 10^{-6}}$	**0.00000**	$\mathbf{1.18585 \times 10^{-6}}$	**100%**	17.26	1



(**a**)



(**b**)

Figure 3. (**a**) Convergence graph of the QISHTS algorithm for the first-order reversible series reaction model. (**b**) Fitting state trajectories and experimental points for the first-order reversible series reaction model.

6.1.2. Catalytic Cracking of Gas Oil Problem

This problem indicates the catalytic cracking process of gas oil to gasoline and other miscellaneous products. It involves three components (A, Q, and S), which represent the gas oil, gasoline, and other products, respectively. Only the concentrations of the first two components (A with Q) are available in this model, and the reaction design contains nonlinear reaction kinetics. This problem is formulated as below:

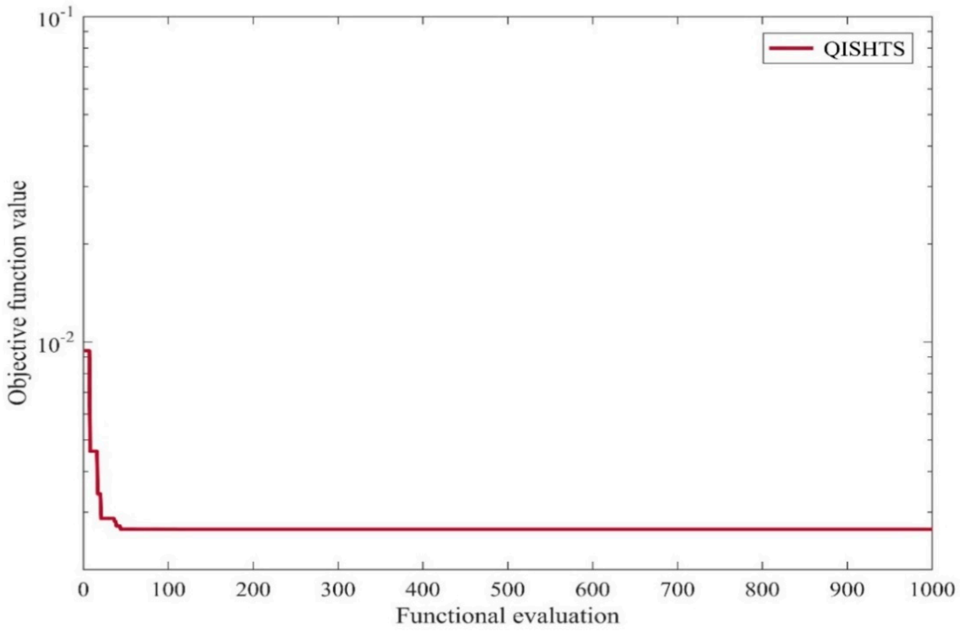
$$\min_{\theta} \sum_{\mu=1}^{10} \sum_{i=1}^{2} \left(\hat{z}_{\mu,i} - \widetilde{z}_{\mu,i}\right)^2$$
$$S.t. \begin{cases} \dot{z}_1 = -(\theta_1 + \theta_3) z_1^2 \\ \dot{z}_2 = \theta_1 z_1^2 - \theta_2 z_2 \\ z_0 = (1, 0) \\ t \in [0, 0.95] \\ [0,0,0] \le \theta \le [20,20,20] \end{cases} \tag{12}$$

where θ_1, θ_2, and θ_3 indicate the rate constants of the respective reactions, and z_1 and z_2 represent the mole fractions of the first two components (A with Q), respectively. The measurement data of this problem are acquired by utilizing parameter values [12,8,2], with a small amount of accidental error added, and are available in [4].

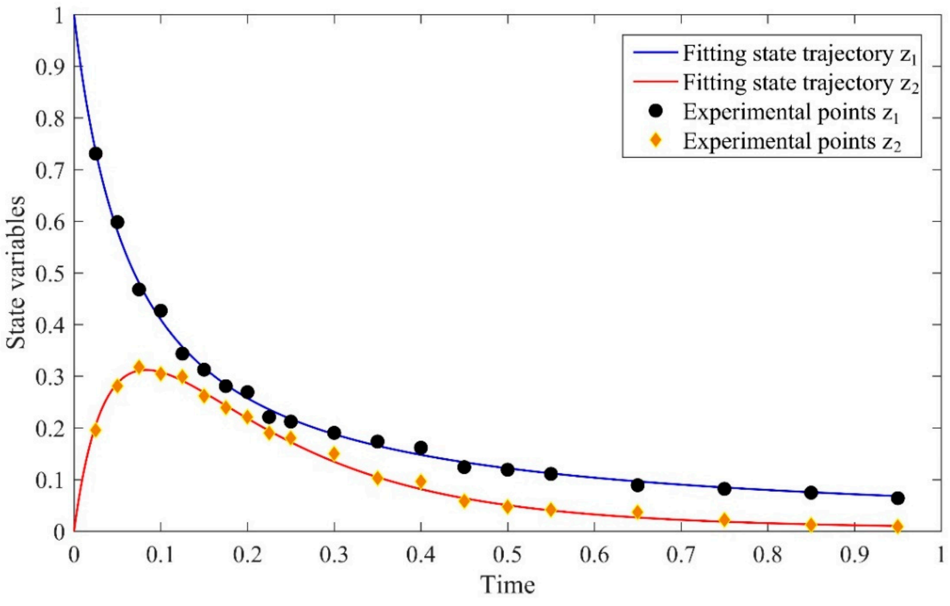
Table 8 displays the comparative results obtained by the QISHTS algorithm and other competitors for this problem. While Figure 4a displays the convergence graph of the proposed algorithm, Figure 4b displays the fitting state trajectories with experimental points. From Figure 4a and Table 8, it can be seen that the QISHTS algorithm can find the greatest results in terms of best, mean, std, and worst objective function values among all the comparative algorithms, followed by the QITLBO, TLBO, SaDE, and other competitors, meaning that the QISHTS algorithm has the best robustness and precision. Moreover, it can be observed from Figure 4b that a good agreement is obtained between the fitting state trajectories and the experimental points for this problem. Considering the success rate, the QISHTS algorithm achieves a value of 100%, which means that it can successfully find trusted solutions in all the total runs. Referring to the execution time, the DEBBO algorithm ranks the first, followed by the jDE, GOTLBO, and other competitors, while the QISHTS algorithm takes 32.14 s to obtain these results.

Table 8. Results of the compared algorithms for the catalytic cracking of gas oil model.

Method	Best	Mean	STD	Worst	SR	T(S)	Rank
LDWPSO	2.65569×10^{-3}	2.65663×10^{-3}	1.15296×10^{-6}	2.66061×10^{-3}	**100%**	32.10	6
CLPSO	2.71167×10^{-3}	3.54925×10^{-3}	7.98382×10^{-4}	6.06821×10^{-3}	37%	35.17	13
jDE	$\mathbf{2.65567 \times 10^{-3}}$	2.65702×10^{-3}	2.73064×10^{-6}	2.66749×10^{-3}	**100%**	30.18	7
ABC	2.65674×10^{-3}	3.13821×10^{-3}	1.33151×10^{-3}	8.51756×10^{-3}	83%	32.53	11
SaDE	$\mathbf{2.65567 \times 10^{-3}}$	2.65587×10^{-3}	2.91411×10^{-7}	2.65698×10^{-3}	**100%**	32.41	4
RCBBO	2.68559×10^{-3}	3.47463×10^{-3}	1.00151×10^{-3}	6.97048×10^{-3}	53%	32.35	12
DEBBO	2.65582×10^{-3}	2.78029×10^{-3}	4.26505×10^{-4}	4.85929×10^{-3}	90%	**29.82**	10
TLBO	$\mathbf{2.65567 \times 10^{-3}}$	2.65578×10^{-3}	1.32328×10^{-7}	2.65626×10^{-3}	**100%**	30.59	3
NIWTLBO	$\mathbf{2.65567 \times 10^{-3}}$	2.73768×10^{-3}	4.49100×10^{-4}	5.11551×10^{-3}	97%	36.61	9
BLPSO	2.65568×10^{-3}	2.65901×10^{-3}	7.55110×10^{-6}	2.68741×10^{-3}	**100%**	34.17	8
GOTLBO	2.65568×10^{-3}	2.65649×10^{-3}	1.54447×10^{-6}	2.66225×10^{-3}	**100%**	30.22	5
QITLBO	$\mathbf{2.65567 \times 10^{-3}}$	2.65570×10^{-3}	8.30733×10^{-8}	2.65607×10^{-3}	**100%**	34.31	2
QISHTS	$\mathbf{2.65567 \times 10^{-3}}$	$\mathbf{2.65568} \times 10^{-3}$	$\mathbf{1.81904 \times 10^{-9}}$	$\mathbf{2.65599 \times 10^{-3}}$	**100%**	32.14	1



(**a**)



(**b**)

Figure 4. (**a**) Convergence graph of QISHTS algorithm for the catalytic cracking of gas oil model. (**b**) Fitting state trajectories and experimental points for the catalytic cracking of gas oil model.

6.2. Optimal Control Problems

In this subsection, the proposed QISHTS algorithm is applied for solving two optimal control problems. Moreover, the computational results are compared with those of twelve well-established methods. Due to the existence of time-dependent control variables $u(t)$ in these problems, a discretization approach such as the control vector parametrization (CVP) method [11,12] is utilized to discretize these problems. To be consistent with the reference, the time interval is divided by the CVP method into 20 stages with equal length, wherein the control variables $u(t)$ in every sup-interval are approximated by piece-linear functions.

6.2.1. Multi-Modal Continuous Stirred Tank Reactor (CSTR) Problem

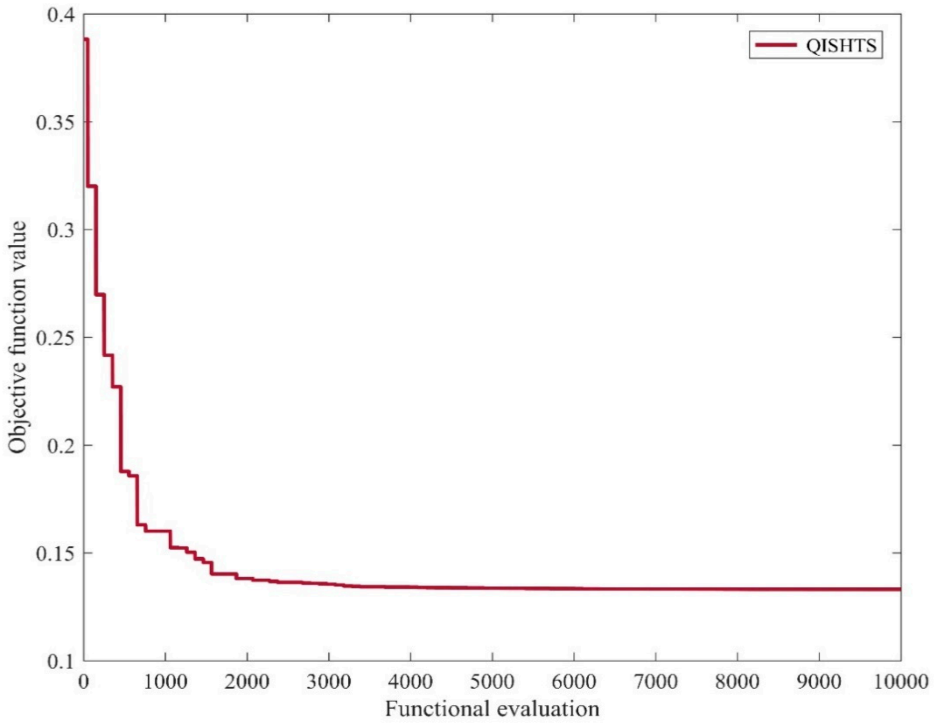
This model is defined as a multi-modal problem; it basically represents the optimal control of a first-order reversible chemical reaction performed in a continuous stirred tank reactor (CSTR) [12]. This problem offers a challenge to the optimization methods, since it mainly involves two local optimums. The gradient-based optimization methods usually fall into the inferior local optimum. The mathematical equation of this problem can be formulated as below:

$$\min_u J = \int_0^{t_f} \left(x_1^2(t) + x_2^2(t) + 0.1 * u^2(t)\right)dt$$
$$S.t.\begin{cases} \dot{x}_1 = -(2+u)(x_1+0.25) + (x_2+0.5)\exp\left(\frac{25x_1}{x_1+2}\right) \\ \dot{x}_2 = 0.5 - x_2 - (x_2+0.5)\exp\left(\frac{25x_1}{x_1+2}\right) \\ x(0) = [0.09, .\, 0.9] \\ 0 \le u(t) \le 5.0,\ \mathrm{t}_f = 0.78 \end{cases} \tag{13}$$

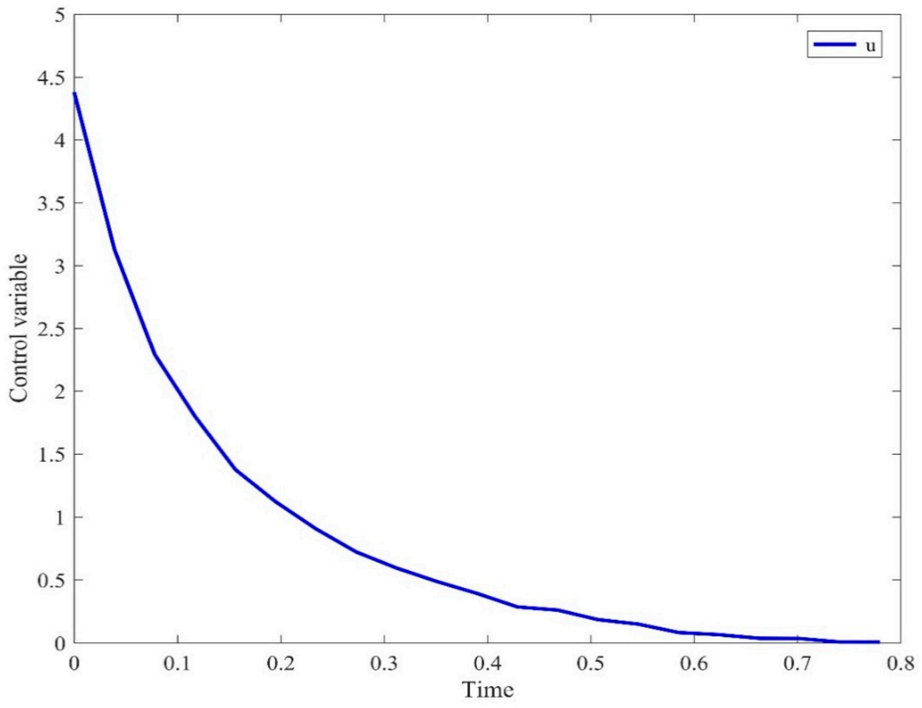
Table 9 displays the comparative results obtained by the QISHTS algorithm and other competitors for this problem. It can be seen from results table that the BLPSO algorithm gets the greatest results in terms of best, mean, std, and worst objective function values among all the comparative algorithms, while the QISHTS gets the second-best results, followed by the QITLBO, LDWPSO, DEBBO, and the rest of the competitors. Figure 5a shows the convergence graph of the proposed algorithm, and Figure 5b displays the optimal control trajectory for this problem. It is obvious from Figure 5 and Table 9 that, although the QISHTS algorithm is worse than BLPSO, its performance is clearly superior to that of other competitors. Referring to the success rate, the QISHTS algorithm achieves a value of 100%, which means that it can successfully find the acceptable accuracy in all the total runs. Considering the execution time, the NIWTLBO ranks first, followed by the BLPSO and other algorithms, while the QISHTS algorithm spends 455.86 s to obtain these results.

Table 9. Results of the compared algorithms for the multi-modal CSTR model.

Method	**Best**	**Mean**	**STD**	**Worst**	**SR**	**T(S)**	**Rank**
LDWPSO	1.3311×10^{-1}	1.3317×10^{-1}	3.89×10^{-5}	1.3329×10^{-1}	**100%**	458.99	4
CLPSO	1.4920×10^{-1}	1.6569×10^{-1}	8.35×10^{-3}	1.7957×10^{-1}	0.00%	478.32	12
jDE	1.3315×10^{-1}	1.3327×10^{-1}	6.96×10^{-5}	1.3341×10^{-1}	**100%**	456.75	6
ABC	1.3968×10^{-1}	1.4788×10^{-1}	4.51×10^{-3}	1.5592×10^{-1}	3%	464.76	11
SaDE	1.3315×10^{-1}	1.3338×10^{-1}	2.32×10^{-4}	1.3421×10^{-1}	**100%**	458.89	7
RCBBO	1.3346×10^{-1}	1.3550×10^{-1}	2.07×10^{-3}	1.4314×10^{-1}	97%	463.42	9
DEBBO	1.3317×10^{-1}	1.3324×10^{-1}	4.37×10^{-5}	1.3333×10^{-1}	**100%**	457.52	5
TLBO	1.3311×10^{-1}	1.3700×10^{-1}	2.03×10^{-2}	2.4445×10^{-1}	97%	466.63	10
NIWTLBO	1.3383×10^{-1}	2.3744×10^{-1}	2.75×10^{-2}	2.4515×10^{-1}	7%	**419.09**	13
BLPSO	$\mathbf{1.3310 \times 10^{-1}}$	$\mathbf{1.3311 \times 10^{-1}}$	$\mathbf{1.03 \times 10^{-5}}$	$\mathbf{1.3314 \times 10^{-1}}$	**100%**	448.59	1
GOTLBO	1.3311×10^{-1}	1.3392×10^{-1}	4.30×10^{-3}	1.5669×10^{-1}	97%	460.76	8
QITLBO	1.3311×10^{-1}	1.3314×10^{-1}	2.39×10^{-5}	1.3320×10^{-1}	**100%**	460.91	3
QISHTS	$\mathbf{1.3310 \times 10^{-1}}$	1.3312×10^{-1}	1.98×10^{-5}	1.3318×10^{-1}	**100%**	455.86	2



(**a**)



(**b**)

Figure 5. (**a**) Convergence graph of the QISHTS algorithm for the multi-modal CSTR model. (**b**) Optimal control trajectory for the multi-modal CSTR model. *u* portrays the optimal control variable of this problem.

6.2.2. Six-Plate Gas Absorption Tower Problem

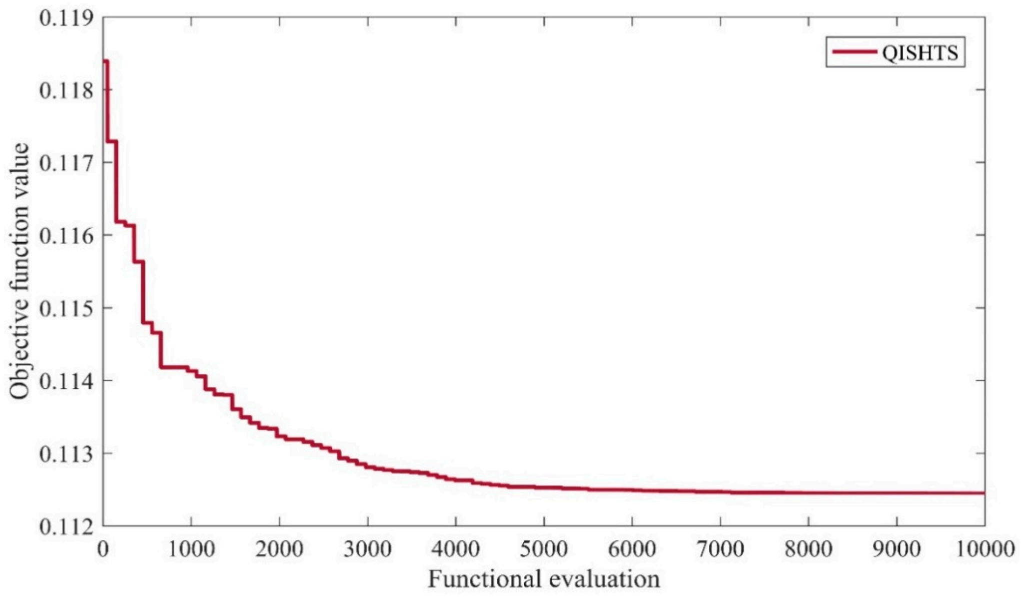
This model mainly includes the determination of two optimal control variables (u_1 and u_2) in a nonlinear six-plate gas absorption tower [74]. The mathematical equation of this problem can be given as follows:

$$\min J = \int_0^{t_f} \left(\sum_{i=1}^{6} x_i^2 + \sum_{i=1}^{2} u_i^2 \right) dt$$
$$S.t. \begin{cases} \dot{x}_1 = \{-[40.8 + 66.7(v_1 + 0.08x_1)]x_1 + 66.7(v_2 + 0.08x_2)x_2 + 40.8u_1\}/w_1 \\ \dot{x}_2 = \{40.8x_1 - [40.8 + 66.7(v_2 + 0.08x_2)]x_2 + 66.7(v_3 + 0.08x_3)x_3\}/w_2 \\ \dot{x}_3 = \{40.8x_2 - [40.8 + 66.7(v_3 + 0.08x_3)]x_3 + 66.7(v_4 + 0.08x_4)x_4\}/w_3 \\ \dot{x}_4 = \{40.8x_3 - [40.8 + 66.7(v_4 + 0.08x_4)]x_4 + 66.7(v_5 + 0.08x_5)x_5\}/w_4 \\ \dot{x}_5 = \{40.8x_4 - [40.8 + 66.7(v_5 + 0.08x_5)]x_5 + 66.7(v_6 + 0.08x_6)x_6\}/w_5 \\ \dot{x}_6 = \{40.8x_5 - [40.8 + 66.7(v_6 + 0.08x_6)]x_6 + 66.7(v_7 + 0.08x_7)x_7\}/w_6 \\ w_i = v_i + 0.16x_i + 75,\ i = 1, \ldots, 6 \\ v = [0.7358, 0.7488, 0.7593, 0.7593, 0.7677, 0.7744, 0.7797, 0.7838]^T \\ x(0) = [-0.0342, -0.0619, -0.0837, -0.1004, -0.1131, -0.1224]^T \\ 0 \le u_1 \le 0.04,\ 0 \le u_2 \le 0.09,\ \mathrm{t}_f = 5 \end{cases} \quad (14)$$

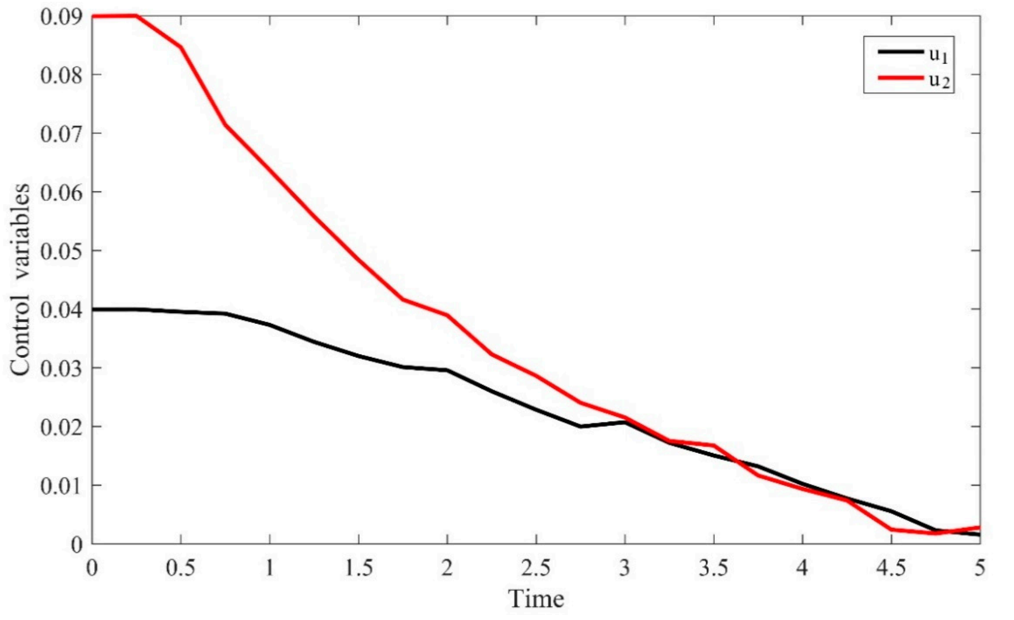
Table 10 displays the computational results obtained by the QISHTS algorithm and other competitors for this problem. Figure 6a displays the convergence graph of the proposed algorithm, and Figure 6b shows the optimal control trajectories for this problem. From Figure 6a and Table 10, it can be seen that the QISHTS algorithm can find the greatest results in terms of best, mean, std, and worst objective function values among all the comparative algorithms, followed by the jDE, BLPSO, QITLBO, and other competitors, meaning that the QISHTS algorithm has the best robustness and precision. Considering the success rate, the QISHTS algorithm achieves a value of 100%, which means that it can successfully find trusted solutions in all the total runs. Referring to the execution time, the jDE algorithm ranks the first, followed by the SaDE, NIWTLBO, and the rest of the competitors, while the QISHTS algorithm spends 558.96 s to obtain these results.

Table 10. Results of the compared algorithms for the six-plate gas absorption tower model.

Method	Best	Mean	STD	Worst	SR	T(S)	Rank
LDWPSO	1.1244×10^{-1}	1.1245×10^{-1}	6.86×10^{-6}	1.1246×10^{-1}	**100%**	552.45	6
CLPSO	1.1321×10^{-1}	1.1390×10^{-1}	2.07×10^{-4}	1.1420×10^{-1}	0.00%	554.22	13
jDE	$\mathbf{1.1243 \times 10^{-1}}$	$\mathbf{1.1243 \times 10^{-1}}$	1.14×10^{-6}	1.1244×10^{-1}	**100%**	**548.63**	2
ABC	1.1286×10^{-1}	1.1326×10^{-1}	1.85×10^{-4}	1.1357×10^{-1}	0.00%	588.13	12
SaDE	1.1244×10^{-1}	1.1245×10^{-1}	5.95×10^{-6}	1.1247×10^{-1}	**100%**	549.59	6
RCBBO	1.1244×10^{-1}	1.1247×10^{-1}	1.55×10^{-5}	1.1251×10^{-1}	93%	584.34	8
DEBBO	$\mathbf{1.1243 \times 10^{-1}}$	1.1244×10^{-1}	1.17×10^{-6}	1.1244×10^{-1}	**100%**	583.57	4
TLBO	1.1244×10^{-1}	1.1248×10^{-1}	1.97×10^{-5}	1.1253×10^{-1}	87%	581.01	10
NIWTLBO	1.1257×10^{-1}	1.1280×10^{-1}	1.68×10^{-4}	1.1333×10^{-1}	0.00%	549.69	11
BLPSO	$\mathbf{1.1243 \times 10^{-1}}$	$\mathbf{1.1243 \times 10^{-1}}$	2.96×10^{-8}	$\mathbf{1.1243 \times 10^{-1}}$	**100%**	618.02	2
GOTLBO	1.1244×10^{-1}	1.1247×10^{-1}	2.74×10^{-5}	1.1254×10^{-1}	90%	582.16	8
QITLBO	$\mathbf{1.1243 \times 10^{-1}}$	1.1244×10^{-1}	5.14×10^{-6}	1.1246×10^{-1}	**100%**	561.71	4
QISHTS	$\mathbf{1.1243 \times 10^{-1}}$	$\mathbf{1.1243 \times 10^{-1}}$	$\mathbf{2.45 \times 10^{-8}}$	$\mathbf{1.1243 \times 10^{-1}}$	**100%**	558.96	1



(**a**)



(**b**)

Figure 6. (**a**) Convergence graph of the QISHTS algorithm for the six-plate gas absorption tower model. (**b**) Optimal control trajectories for the six-plate gas absorption tower model. u_1 and u_2 portray the optimal control variables of this problem.

7. Conclusions

Many complex chemical engineering processes depend much on dynamic optimization. Due to the existence of highly constrained, nonlinear, and nonsmooth environments in chemical engineering processes, which usually causes nonconvexity, using efficient global optimization algorithms to reach appropriate solutions for handling DOPs is still a relatively difficult task. The heat transfer search (HTS) algorithm is a relative novel metaheuristic approach inspired by the natural law of thermodynamics and heat transfer. In order to handle the DOPs, a new variant of the HTS algorithm called the quadratic interpolation based simultaneous heat transfer search (QISHTS) algorithm is presented in this paper. The QISHTS algorithm introduces three modifications into the original HTS algorithm, namely the idea of simultaneous heat transfer search (SHTS), quadratic interpolation (QI) method, and population regeneration mechanism. The effect of the SHTS is employed to provide superior performance with lower computational complexity, the QI method is adopted to improve the exploitation capability, and the population regeneration mechanism is used to alleviate the probability of local optimum stagnation, hence enhancing the exploration ability. Thus, the ensemble of these three modifications can provide a more efficient optimization algorithm with well-balanced exploration and exploitation abilities.

To demonstrate the effectiveness of the QISHTS algorithm, it is investigated by a set of 18 well-defined benchmark problems. Moreover, the obtained results are compared with those of other well-established methods. Furthermore, it is applied for solving four real-world chemical DOPs, involving varying levels of difficulty, and the computational results obtained by QISHTS algorithm are compared with those of twelve well-established methods. The results demonstrate that the QISHTS algorithm has the perfect robustness and precision than other the methods tested.

Author Contributions: Conceptualization, E.A.; methodology, E.A.; software, K.A.; validation, K.A.; formal analysis, E.A.; investigation, E.A.; resources, H.S.; data curation, E.A.; writing—original draft preparation, E.A.; writing—review and editing, E.A.; visualization, H.S.; supervision, H.S.; project administration, H.S.; and funding acquisition, H.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research is supported by the National Natural Science Foundation of China under grants 61703161 and 61673173 and the Fundamental Research Funds for the central universities under grant 222201714031.

Conflicts of Interest: The authors declare no conflicts of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

References

1. Srinivasan, B.; Bonvin, D.; Visser, E.; Palanki, S. Dynamic optimization of batch processes: Ii. Role of measurements in handling uncertainty. *Comput. Chem. Eng.* **2003**, *27*, 27–44. [CrossRef]
2. Irizarry, R. A generalized framework for solving dynamic optimization problems using the artificial chemical process paradigm: Applications to particulate processes and discrete dynamic systems. *Chem. Eng. Sci.* **2005**, *60*, 5663–5681. [CrossRef]
3. Jiménez-Hornero, J.E.; Santos-Dueñas, I.M.; García-García, I. Optimization of biotechnological processes. The acetic acid fermentation. Part iii: Dynamic optimization. *Biochem. Eng. J.* **2009**, *45*, 22–29.
4. Floudas, C.A.; Pardalos, P.M.; Adjiman, C.; Esposito, W.R.; Gümüs, Z.H.; Harding, S.T.; Klepeis, J.L.; Meyer, C.A.; Schweiger, C.A. *Handbook of Test Problems in Local and Global Optimization*; Springer Science & Business Media: Berlin, Germany, 2013; Volume 33.
5. Bellman, R. *Dynamic Programming [M]*; Princeton University Press: Princeton, NJ, USA, 2010.
6. Luus, R. *Iterative Dynamic Programming [M]*; CRC Press: Boca Raton, FL, USA, 2010.
7. Sundaralingam, R. Two-step method for dynamic optimization of inequality state constrained systems using iterative dynamic programming. *Ind. Eng. Chem. Res.* **2015**, *54*, 7658–7667. [CrossRef]
8. Bryson, A.E. *Applied Optimal Control: Optimization, Estimation and Control*; Routledge: Abingdon, UK, 2018.
9. Cervantes, A.; Biegler, L.T. Optimization strategies for dynamic systems. *Encycl. Optim.* **2009**, *4*, 216–227.
10. Sarkar, D.; Modak, J.M. Optimization of fed-batch bioreactors using genetic algorithm: Multiple control variables. *Comput. Chem. Eng.* **2004**, *28*, 789–798. [CrossRef]
11. Du, W.; Bao, C.; Chen, X.; Tian, L.; Jiang, D. Dynamic optimization of the tandem acetylene hydrogenation process. *Ind. Eng. Chem. Res.* **2016**, *55*, 11983–11995. [CrossRef]
12. Chen, X.; Du, W.; Tianfield, H.; Qi, R.; He, W.; Qian, F. Dynamic optimization of industrial processes with nonuniform discretization-based control vector parameterization. *IEEE Trans. Autom. Sci. Eng.* **2014**, *11*, 1289–1299. [CrossRef]

13. Canto, E.B.; Banga, J.R.; Alonso, A.A.; Vassiliadis, V.S. Restricted second order information for the solution of optimal control problems using control vector parameterization. *J. Process Control* **2002**, *12*, 243–255. [CrossRef]
14. Nocedal, J.; Wright, S. *Numerical Optimization*; Springer Science & Business Media: Berlin, Germany, 2006.
15. Angira, R.; Santosh, A. Optimization of dynamic systems: A trigonometric differential evolution approach. *Comput. Chem. Eng.* **2007**, *31*, 1055–1063. [CrossRef]
16. Papamichail, I.; Adjiman, C.S. Global optimization of dynamic systems. *Comput. Chem. Eng.* **2004**, *28*, 403–415. [CrossRef]
17. Esposito, W.R.; Floudas, C.A. Global optimization for the parameter estimation of differential-algebraic systems. *Ind. Eng. Chem. Res.* **2000**, *39*, 1291–1310. [CrossRef]
18. BoussaïD, I.; Lepagnot, J.; Siarry, P. A survey on optimization metaheuristics. *Inf. Sci.* **2013**, *237*, 82–117. [CrossRef]
19. Patel, N.; Padhiyar, N. Modified genetic algorithm using box complex method: Application to optimal control problems. *J. Process Control* **2015**, *26*, 35–50. [CrossRef]
20. Qian, F.; Sun, F.; Zhong, W.; Luo, N. Dynamic optimization of chemical engineering problems using a control vector parameterization method with an iterative genetic algorithm. *Eng. Optim.* **2013**, *45*, 1129–1146. [CrossRef]
21. Dai, K.; Wang, N. A hybrid DNA based genetic algorithm for parameter estimation of dynamic systems. *Chem. Eng. Res. Des.* **2012**, *90*, 2235–2246. [CrossRef]
22. Babu, B.; Angira, R. Modified differential evolution (MDE) for optimization of non-linear chemical processes. *Comput. Chem. Eng.* **2006**, *30*, 989–1002. [CrossRef]
23. Chen, X.; Du, W.; Qian, F. Multi-objective differential evolution with ranking-based mutation operator and its application in chemical process optimization. *Chemom. Intell. Lab. Syst.* **2014**, *136*, 85–96. [CrossRef]
24. Penas, D.R.; Banga, J.R.; González, P.; Doallo, R. Enhanced parallel differential evolution algorithm for problems in computational systems biology. *Appl. Soft Comput.* **2015**, *33*, 86–99. [CrossRef]
25. Chen, X.; Du, W.; Qian, F. Solving chemical dynamic optimization problems with ranking-based differential evolution algorithms. *Chin. J. Chem. Eng.* **2016**, *24*, 1600–1608. [CrossRef]
26. Zhang, B.; Chen, D.; Zhao, W. Iterative ant-colony algorithm and its application to dynamic optimization of chemical process. *Comput. Chem. Eng.* **2005**, *29*, 2078–2086. [CrossRef]
27. Schluter, M.; Egea, J.A.; Antelo, L.T.; Alonso, A.A.; Banga, J.R. An extended ant colony optimization algorithm for integrated process and control system design. *Ind. Eng. Chem. Res.* **2009**, *48*, 6723–6738. [CrossRef]
28. Shelokar, P.; Jayaraman, V.K.; Kulkarni, B.D. Multicanonical jump walk annealing assisted by tabu for dynamic optimization of chemical engineering processes. *Eur. J. Oper. Res.* **2008**, *185*, 1213–1229. [CrossRef]
29. Faber, R.; Jockenhövel, T.; Tsatsaronis, G. Dynamic optimization with simulated annealing. *Comput. Chem. Eng.* **2005**, *29*, 273–290. [CrossRef]
30. Chen, X.; Du, W.; Qi, R.; Qian, F.; Tianfield, H. Hybrid gradient particle swarm optimization for dynamic optimization problems of chemical processes. *Asia-Pac. J. Chem. Eng.* **2013**, *8*, 708–720. [CrossRef]
31. Zhou, Y.; Liu, X. Control parameterization-based adaptive particle swarm approach for solving chemical dynamic optimization problems. *Chem. Eng. Technol.* **2014**, *37*, 692–702. [CrossRef]
32. Yang, J.; Lu, L.; Ouyang, W.; Gou, Y.; Chen, Y.; Ma, H.; Guo, J.; Fang, F. Estimation of kinetic parameters of an anaerobic digestion model using particle swarm optimization. *Biochem. Eng. J.* **2017**, *120*, 25–32. [CrossRef]
33. Sun, J.; Palade, V.; Cai, Y.; Fang, W.; Wu, X. Biochemical systems identification by a random drift particle swarm optimization approach. *BMC Bioinform.* **2014**, *15*, S1. [CrossRef]
34. Castellani, M.; Pham, Q.T.; Pham, D.T. Dynamic optimisation by a modified bees algorithm. *Proc. Inst. Mech. Eng. Part I J. Syst. Control Eng.* **2012**, *226*, 956–971. [CrossRef]
35. Egea, J.A.; Balsa-Canto, E.; García, M.a.-S.G.; Banga, J.R. Dynamic optimization of nonlinear processes with an enhanced scatter search method. *Ind. Eng. Chem. Res.* **2009**, *48*, 4388–4401. [CrossRef]
36. Villaverde, A.F.; Egea, J.A.; Banga, J.R. A cooperative strategy for parameter estimation in large scale systems biology models. *BMC Syst. Biol.* **2012**, *6*, 75. [CrossRef] [PubMed]
37. Penas, D.R.; González, P.; Egea, J.A.; Doallo, R.; Banga, J.R. Parameter estimation in large-scale systems biology models: A parallel and self-adaptive cooperative strategy. *BMC Bioinform.* **2017**, *18*, 52. [CrossRef] [PubMed]

38. Sun, D.-Y.; Lin, P.-M.; Lin, S.-P. Integrating controlled random search into the line-up competition algorithm to solve unsteady operation problems. *Ind. Eng. Chem. Res.* **2008**, *47*, 8869–8887. [CrossRef]
39. Rakhshani, H.; Dehghanian, E.; Rahati, A. Hierarchy cuckoo search algorithm for parameter estimation in biological systems. *Chemom. Intell. Lab. Syst.* **2016**, *159*, 97–107. [CrossRef]
40. Nikumbh, S.; Ghosh, S.; Jayaraman, V.K. Biogeography-based optimization for dynamic optimization of chemical reactors. In *Applications of Metaheuristics in Process Engineering*; Springer: Berlin, Germany, 2014; pp. 201–216.
41. Chen, X.; Mei, C.; Xu, B.; Yu, K.; Huang, X. Quadratic interpolation-based teaching-learning-based optimization for chemical dynamic system optimization. *Knowl.-Based Syst.* **2018**, *145*, 250–263. [CrossRef]
42. Patel, V.K.; Savsani, V.J. Heat transfer search (hts): A novel optimization algorithm. *Inf. Sci.* **2015**, *324*, 217–246. [CrossRef]
43. Degertekin, S.; Lamberti, L.; Hayalioglu, M. Heat transfer search algorithm for sizing optimization of truss structures. *Lat. Am. J. Solids Struct.* **2017**, *14*, 373–397. [CrossRef]
44. Tejani, G.G.; Savsani, V.J.; Patel, V.K.; Mirjalili, S. An improved heat transfer search algorithm for unconstrained optimization problems. *J. Comput. Des. Eng.* **2019**, *6*, 13–32. [CrossRef]
45. Tejani, G.; Savsani, V.; Patel, V. Modified sub-population based heat transfer search algorithm for structural optimization. *Int. J. Appl. Metaheuristic Comput. (IJAMC)* **2017**, *8*, 1–23. [CrossRef]
46. Tejani, G.G.; Kumar, S.; Gandomi, A.H. Multi-objective heat transfer search algorithm for truss optimization. *Eng. Comput.* **2019**, 1–22. [CrossRef]
47. Maharana, D.; Kotecha, P. Simultaneous heat transfer search for computationally expensive numerical optimization. In Proceedings of the 2016 IEEE Congress on Evolutionary Computation (CEC), Vancouver, BC, Canada, 24–29 July 2016; IEEE: Piscataway, NJ, USA, 2016; pp. 2982–2988.
48. Maharana, D.; Kotecha, P. Simultaneous heat transfer search for single objective real-parameter numerical optimization problem. In Proceedings of the 2016 IEEE Region 10 Conference (TENCON), Singapore, Singapore, 22–25 November 2016; IEEE: Piscataway, NJ, USA, 2016; pp. 2138–2141.
49. Tawhid, M.A.; Savsani, V. $\in$-constraint heat transfer search ($\in$-HTS) algorithm for solving multi-objective engineering design problems. *J. Comput. Des. Eng.* **2018**, *5*, 104–119. [CrossRef]
50. Ali, M.M.; Törn, A.; Viitanen, S. A numerical comparison of some modified controlled random search algorithms. *J. Glob. Optim.* **1997**, *11*, 377–385. [CrossRef]
51. Li, H.; Jiao, Y.-C.; Zhang, L. Hybrid differential evolution with a simplified quadratic approximation for constrained optimization problems. *Eng. Optim.* **2011**, *43*, 115–134. [CrossRef]
52. Deep, K.; Das, K.N. Quadratic approximation based hybrid genetic algorithm for function optimization. *Appl. Math. Comput.* **2008**, *203*, 86–98. [CrossRef]
53. Yang, Y.; Zong, X.; Yao, D.; Li, S. Improved alopex-based evolutionary algorithm (AEA) by quadratic interpolation and its application to kinetic parameter estimations. *Appl. Soft Comput.* **2017**, *51*, 23–38. [CrossRef]
54. Yang, X.-S.; Deb, S. Engineering optimisation by cuckoo search. *arXiv* **2010**, arXiv:1005.2908. [CrossRef]
55. Rashedi, E.; Nezamabadi-Pour, H.; Saryazdi, S. GSA: A gravitational search algorithm. *Inf. Sci.* **2009**, *179*, 2232–2248. [CrossRef]
56. Karaboga, D.; Basturk, B. A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm. *J. Glob. Optim.* **2007**, *39*, 459–471. [CrossRef]
57. Li, X.; Zhang, J.; Yin, M. Animal migration optimization: An optimization algorithm inspired by animal migration behavior. *Neural Comput. Appl.* **2014**, *24*, 1867–1877. [CrossRef]
58. Rahnamayan, S.; Tizhoosh, H.R.; Salama, M.M. Opposition-based differential evolution. *IEEE Trans. Evol. Comput.* **2008**, *12*, 64–79. [CrossRef]
59. Qin, A.K.; Huang, V.L.; Suganthan, P.N. Differential evolution algorithm with strategy adaptation for global numerical optimization. *IEEE Trans. Evol. Comput.* **2009**, *13*, 398–417. [CrossRef]
60. Maulik, U.; Saha, I. Automatic fuzzy clustering using modified differential evolution for image classification. *IEEE Trans. Geosci. Remote Sens.* **2010**, *48*, 3503–3510. [CrossRef]
61. Wang, Y.; Cai, Z.; Zhang, Q. Differential evolution with composite trial vector generation strategies and control parameters. *IEEE Trans. Evol. Comput.* **2011**, *15*, 55–66. [CrossRef]

62. Islam, S.M.; Das, S.; Ghosh, S.; Roy, S.; Suganthan, P.N. An adaptive differential evolution algorithm with novel mutation and crossover strategies for global numerical optimization. *IEEE Trans. Syst. ManCybern. Part B Cybern.* **2012**, *42*, 482–500. [CrossRef]
63. Cai, Y.; Wang, J. Differential evolution with hybrid linkage crossover. *Inf. Sci.* **2015**, *320*, 244–287. [CrossRef]
64. Shi, Y.; Eberhart, R. A modified particle swarm optimizer. In Proceedings of the 1998 IEEE International Conference on Evolutionary Computation Proceedings. IEEE World Congress on Computational Intelligence (Cat. No. 98TH8360), Anchorage, AK, USA, 4–9 May 1998; IEEE: Piscataway, NJ, USA, 1988; pp. 69–73.
65. Liang, J.J.; Qin, A.K.; Suganthan, P.N.; Baskar, S. Comprehensive learning particle swarm optimizer for global optimization of multimodal functions. *IEEE Trans. Evol. Comput.* **2006**, *10*, 281–295. [CrossRef]
66. Brest, J.; Greiner, S.; Boskovic, B.; Mernik, M.; Zumer, V. Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems. *IEEE Trans. Evol. Comput.* **2006**, *10*, 646–657. [CrossRef]
67. Gong, W.; Cai, Z.; Ling, C.X.; Li, H. A real-coded biogeography-based optimization with mutation. *Appl. Math. Comput.* **2010**, *216*, 2749–2758. [CrossRef]
68. Gong, W.; Cai, Z.; Ling, C.X. DE/BBO: A hybrid differential evolution with biogeography-based optimization for global numerical optimization. *Soft Comput.* **2010**, *15*, 645–665. [CrossRef]
69. Rao, R.V.; Savsani, V.J.; Vakharia, D. Teaching–learning-based optimization: An optimization method for continuous non-linear large scale problems. *Inf. Sci.* **2012**, *183*, 1–15. [CrossRef]
70. Wu, Z.-S.; Fu, W.-P.; Xue, R. Nonlinear inertia weighted teaching-learning-based optimization for solving global optimization problem. *Comput. Intell. Neurosci.* **2015**, *2015*. [CrossRef] [PubMed]
71. Chen, X.; Tianfield, H.; Mei, C.; Du, W.; Liu, G. Biogeography-based learning particle swarm optimization. *Soft Comput.* **2017**, *21*, 7519–7541. [CrossRef]
72. Chen, X.; Yu, K.; Du, W.; Zhao, W.; Liu, G. Parameters identification of solar cell models using generalized oppositional teaching learning based optimization. *Energy* **2016**, *99*, 170–180. [CrossRef]
73. Lin, Y.; Stadtherr, M.A. Deterministic global optimization for parameter estimation of dynamic systems. *Ind. Eng. Chem. Res.* **2006**, *45*, 8438–8448. [CrossRef]
74. Wang, F.-S.; Chiou, J.-P. Nonlinear optimal control and optimal parameter selection by a modified reduced gradient method. *Eng. Optim. + A35* **1997**, *28*, 273–298. [CrossRef]