applied sciences

MDPI

Article

Hybrid Particle Swarm Optimization for High-Dimensional Latin Hypercube Design Problem

Zhixin Xu [1], Dongqin Xia [2], Nuo Yong [2], Jinkai Wang [1], Jian Lin [1], Feipeng Wang [2,*], Song Xu [2,3] and Daochuan Ge [2]

[1] State Key Laboratory of Nuclear Power Safety Monitoring Technology and Equipment, China Nuclear Power Engineering Co., Ltd., Shenzhen 518172, China; xuzhixin@cgnpc.com.cn (Z.X.); wangjinkai@cgnpc.com.cn (J.W.); lin_jian@cgnpc.com.cn (J.L.)
[2] Institute of Nuclear Energy Safety Technology, China Hefei Institutes of Physical Science, Chinese Academy of Sciences, Hefei 230031, China; dongqin.xia@inest.cas.cn (D.X.); nuo.yong@inest.cas.cn (N.Y.); ba22168199@mail.ustc.edu.cn (S.X.); daochuan.ge@inest.cas.cn (D.G.)
[3] Science Island Branch, Graduate School of USTC, Hefei 230026, China
* Correspondence: feipeng.wang@inest.cas.cn

Abstract: Latin Hypercube Design (LHD) is widely used in computer simulation to solve large-scale, complex, nonlinear problems. The high-dimensional LHD (HLHD) problem is one of the crucial issues and has been a large concern in the long run. This paper proposes an improved Hybrid Particle Swarm Optimization (IHPSO) algorithm to find the near-optimal HLHD by increasing the particle evolution speed and strengthening the local search. In the proposed algorithm, firstly, the diversity of the population is ensured through comprehensive learning. Secondly, the Minimum Point Distance (MPD) method is adopted to solve the oscillation problem of the PSO algorithm. Thirdly, the Ranked Ordered Value (ROV) rule is used to realize the discretization of the PSO algorithm. Finally, local and global searches are executed to find the near-optimal HLHD. The comparisons show the superiority of the proposed method compared with the existing algorithms in obtaining the near-optimal HLHD.

Keywords: adaptive particle swarm optimization; high-dimensional Latin hypercube design; space-filling quality; design points

check for updates

Citation: Xu, Z.; Xia, D.; Yong, N.; Wang, J.; Lin, J.; Wang, F.; Xu, S.; Ge, D. Hybrid Particle Swarm Optimization for High-Dimensional Latin Hypercube Design Problem. *Appl. Sci.* **2023**, *13*, 7066. https://doi.org/10.3390/app13127066

Academic Editor: Vincent A. Cicirello

Received: 27 April 2023
Revised: 2 June 2023
Accepted: 7 June 2023
Published: 12 June 2023

Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Computer models in the real world usually have the characteristics of complex modeling, large-scale parameters, and nonlinear response surfaces [1–3], which makes it time-consuming to obtain high-precision experimental results. Therefore, reducing the number of experiments is significant in improving Design of Experiments efficiency (DOE) [4,5]. A good DOE should include two properties: the design's unfolding [6] and good space-filling [7].

Latin Hypercube Design (LHD) is a widely used unfolding method [8,9]. Still, this method can only be guaranteed to unfold in one-dimensional projections. Moreover, it cannot guarantee an effectively filled input parameter space. Various criteria for measuring the quality of space filling are provided in the literature, for example, the maximum distance criterion [6,10], the Audze–Eglais (AE) criterion [11–13], the φp criterion [6,14–17], central L2 difference [18], and the entropy criterion [15]. To efficiently find the optimal LHD, some authors have proposed algorithms that can improve the quality of LHD space filling. Liefvendahl and Stocki [12] proposed an algorithm called Columnwise Pairwise (CP), which is a simple local search algorithm. The algorithm obtains a better LHD by swapping two elements on the same column in the LHD matrix. The algorithm performs well on small-scale problems but cannot handle large-scale optimization problems. Morris and Mitchell [14] proposed a simulated annealing search algorithm and pointed out that some optimal LHDs have symmetric structures. One stochastic local search method, proposed by

Jin [16], is named the Enhanced Stochastic Evolution (ESE) algorithm. The contribution of this work is to obtain better solutions through the cooperation of the inner and outer loops. The optimization problem also has limitations, and it is usually challenging to obtain the optimal solution. Doerr [19] and Rainville [20] used a genetic algorithm (GA) to generate low-discrepancy sequences with a high degree of concordance, which was limited in local search ability and easily trapped in locally optimal solutions. Bates et al. [21] proposed a population algorithm based on a GA, using a permutation genetic algorithm (PermGA) combined with local search and global search to construct the optimal LHD. The algorithm can achieve better results in the optimization of medium-scale problems. However, due to the limitations of the local search algorithm, it is not easy to find and achieve the optimal solution for the large-scale problem. In addition, some other optimized LHD algorithms include Iterative Local Search (ILS) [6], Sequential Local Enumeration (SLE) [22], Sliced Latin Hypercube Design (SLHD) [23], Simulation-based Sequencing Optimization for Annealing (SOBSA) [24], and Inflation, Expansion, and Stacking (IES) algorithms [25].

In addition to the above optimization algorithms, iterative heuristic algorithms are designed to optimize different types of problems in various fields because of their easy implementation and ability to optimize complex issues effectively. Particle Swarm Optimization (PSO) [26–29] is one famous algorithm proposed by Eberhart and Kennedy in 1995. The idea of this algorithm comes from a simplified model of bird flocks' food-seeking behavior [26]. The respective positions are updated through information exchange between individual birds, combining the current best position with the individual's history to find the optimal value in the range. Due to its simplicity and high efficiency, it is widely used in various fields in the real world [30–34]. PSO can solve high-dimensional optimization problems with multiple optimal solutions, optimized and applied to the HLHD problem. Chen et al. [35] proposed a LaPSO to optimize LHD by reducing the Hamming distance to obtain the optimal LHD. Mahdi Aziz et al. [36] proposed the Adaptive Memetic Particle Swarm (AMPSO) algorithm, which excels in solving large-scale LHD problems. The efficiency of finding the optimal LHD combined with the PSO can still be improved. The above works are novel and can serve as a valuable reference. However, there are still some problems that need to be addressed in applying PSO to LHD optimization:

(1) Since the PSO algorithm's evolution method is not suitable for discrete issues in LHD, subsequent discretization operations are required when using the solution obtained by PSO.

(2) PSO is highly dependent on the dynamic equation of its evolution. The value of the control parameters in the equation determines the convergence speed and the pros and cons of evolution. There may be insufficient space exploration, and the optimal solution cannot be obtained [31].

(3) Each particle in the PSO learns its personal best particle and global best particle simultaneously. The inappropriate selection of parameters in the dynamic equation will make the population change less frequently during the evolution process, which may lead to the phenomenon of "oscillation" [37,38] and the premature convergence of the population [39]. It will reduce the search efficiency of the algorithm, be unable to fully explore the research space, and fall into the local optimal solution [31,40].

(4) Although PSO can quickly find the optimal solution when dealing with low-dimensional problems, optimizing high-dimensional issues makes it more prone to premature convergence [41]. In practical applications, as the dimension increases, the PSO algorithm is more prone to boundary violation problems during the search process, which affects the evolution effect of particles.

In this work, we proposed a high-dimensional LHD design framework based on the improved hybrid PSO (IHPSO). First, the diversity of the population is ensured by using comprehensive learning. Second, the oscillation problem of the PSO is solved by the minimum point distance (MPD), accelerating the calculation process. Third, the ranked ordered value (ROV) rule is adopted to realize the discretization of the PSO. Finally, local and global searches are executed to search for the near-optimal HLHD.

The remainder of this paper is organized as follows: Section 2 introduces the technical background, including the LHD definition, the PSO algorithm, and the space-filling criteria. In Section 3, the composition of the proposed PSO algorithm for the fast construction of optimal LHD is presented. Numerical results and discussion are shown in Section 4. Finally, Section 5 concludes this paper and proposes an outlook for improving the performance of the proposed algorithm.

2. Technical Background

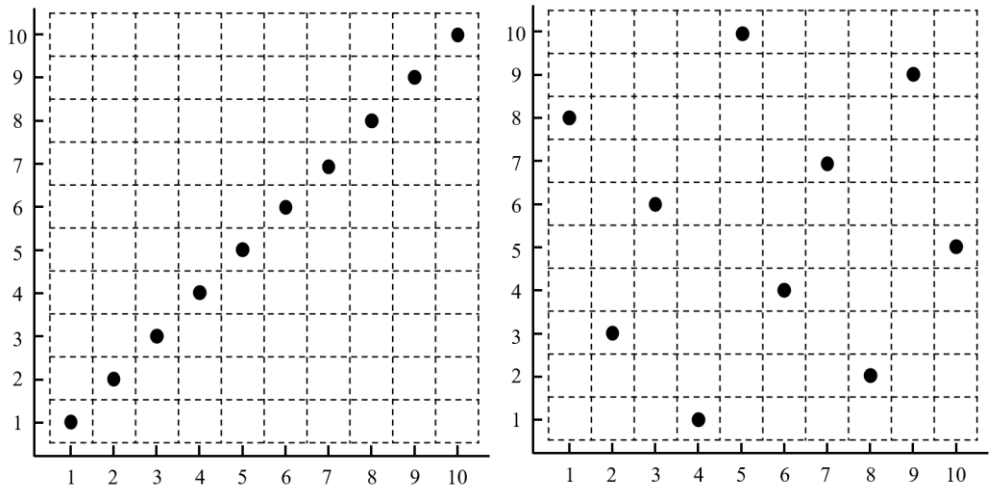
2.1. Latin Hypercube Design

For an LHD matrix with n design points and m dimensions, each matrix column represents a dimension. Each row is a design point. The method of constructing the LHD matrix is shown in Equation (1) [42]:

$$x_{i,j} = \frac{H(i,j) - \varepsilon(0,1)}{n} \tag{1}$$

where $H(i,j)$ is the column function of the LHD, the value is [1, n] and does not repeat; $\varepsilon(i,j)$ is a uniformly distributed random independent variable of [0, 1], whose value is set as 0.5 for the convenience of calculation.

The non-space-filling and space-filling LHD are shown in Figure 1. An LHD with n samples and m factors is built. Each factor is divided into n levels, and the arrangement of the points makes each level precisely one point. Therefore, LHD has a one-dimensional projected non-collapsing property [43], indicating that projected n-point onto any factor will lead to exactly n different factors. The LHD of the n-th order m factor can be expressed as an $n \times m$ matrix. Each row of the matrix corresponds to a design point, and each column is a factor. The total number of LHD designs with n design points and m factors is $(n!)^{m-1}$, which increases rapidly with the increase in n or m. Although the LHD is guaranteed not to collapse, the randomly generated LHD may not always fill the space.



(**a**) A non-space-filling LHD (**b**) A space-filling LHD

Figure 1. Two LHD based examples where n = 10, m = 2.

2.2. Particle Swarm Optimization and Comprehensive PSO

PSO is a population-based, heuristic method to solve high-dimensional optimization with optimal solutions. Since the proposal of PSO [26], most of the variants in PSO have been developed to focus on the optimization problems in different scenarios [30,33,34,38,44–48]. After PSO initializes the position of each particle, it will determine the speed of each particle based on the self-cognition and social learning factors of each particle. The position is then updated, and the evaluation value at this new position is assessed. In the whole process, all particles seek the optimal value through social interaction and cooperation [35].

The detailed updated processes of speed and position of particles are shown as:

$$v_i^{t+1} = \omega v_i^t + c_1 r_1^t (pBest_i^t - x_i^t) + c_2 r_2^t (gBest^t - x_i^t) \quad (2)$$

$$x_i^{t+1} = x_i^t + v_i^{t+1} \quad (3)$$

where ω is the weight of the inertia, constants c_1 and c_2 are acceleration coefficients, and r_1^t and r_2^t are two uniformly distributed random numbers generated independently in the range of [0, 1]. In Equation (2), $pBest_i^t$ is the best adaptive position of the i-th particle found thus far, and $gBest_i^t$ is currently the best overall position. c_1 represents self-cognition factors, and c_2 represents social learning factors. c_1 and c_2 decide the learning degree in which they balance the individual and global best position, searching for the optimal particle.

The search environment of LHD is complex, and a large number of the local optimal solutions are assembled in it. Searching the LHD by the PSO can easily cause particles to be attracted to the *gBest* zone and fall into the local optimal solution. The current particles may find the corresponding regions of the optimal solution in some dimensions, but they may have poor fitness due to poor performance in other dimensions. This paper proposed an improved hybrid method to use the different dimensions of learning different particles to improve the searching ability of PSO.

For each dimension of each particle, a random number is generated. If it is greater than the set learning probability *Pi*, the *pBest* of the corresponding dimension will be learned. Otherwise, the *pBest* of other particles will be further learned. The method of selecting other particles is to randomly select two particles from the population and choose the current dimension of the particle with better applicability for learning. The speed update equation of the comprehensive learning strategy is shown in Equation (4):

$$v_i^d = \omega \times v_i^d + c \times rand_i^d \times (pBest_{fi(d)}^d - x_i^d) \quad (4)$$

and the flow of the comprehensive learning strategy is shown in Figure 2.

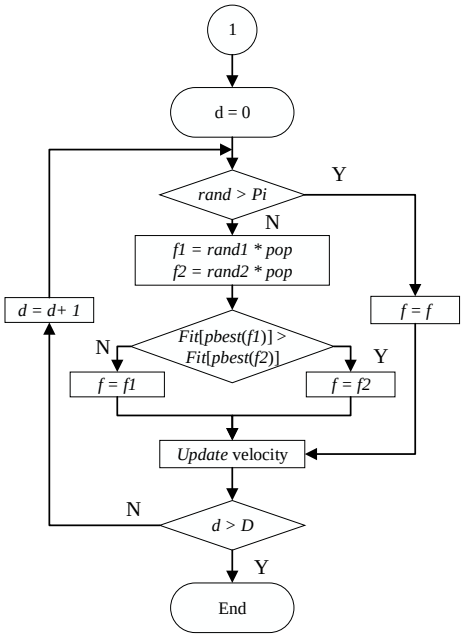


Figure 2. Comprehensive particle learning strategy flowchart.

2.3. ϕp Criterion

The commonly used index to measure the space-filling quality of experimental design is ϕp, which is the widespread criterion in the literature of optimal LHD [6,14–17,35]. The

criterion points out that the better the space-filling of the design, the smaller the calculated ϕp value. The function equation of ϕp is:

$$\phi_p = [\sum_{i=1}^{n} \sum_{j=i+1}^{n} d_{ij}^{-p}]^{\frac{1}{p}} \quad (5)$$

where p is a positive integer and n is the number of points. d_{ij} represents the distance between points. x_i, x_j, and d_{ij} can be expressed as:

$$d_{ij} = [\sum_{k=1}^{m} \left| x_{ik} - x_{jk} \right|^t]^{1/t} \quad (6)$$

where x_{ik} and x_{jk} represent the k-th component of point i and point j, respectively. In general, set $p = 50$ and $t = 1$ [16]. In this paper, the ϕp criterion is used to evaluate the optimization performance of the LHD.

3. Proposed Method

The PSO can rapidly find the optimal solution when dealing with optimization problems with lower dimensions. However, it is prone to appear premature convergence when faced with higher dimension problems. It falls into the predicament of the local optimal solution [31,40].

In this section, we proposed an optimization framework based on the following procedures:

(1) The ROV rule is firstly used [49] to transform continuous space into discrete space in the PSO evolution.

(2) The MPD is adopted to solve the oscillation problem.

(3) Four local search strategies are used to make the global best particle *gBest* go to a better position.

(4) An improved hybrid method is adopted based on the PSO to preserve the diversity of the population and solve the problem of premature convergence.

(5) The boundary reflection strategy is adopted to address the problem of exceeding the boundary in the evolution of high-dimensional particle swarms.

The pseudo-codewords of the IHPSO algorithm are shown in Algorithm 1.

Algorithm 1. Pseudo-codewords of proposed IHPSO algorithm

```
Initialize particle position and velocity
Evaluate the function value of different particles
Initialize the pBest
Initialize the gBest data
 while t < MI (Maximum Iterations)
    if (flag < m)
      Use Equation (1) for speed update
    else
      Use Equation (2) for speed update
    Update the position of each particle
    Discretize particle position data using ROV rules
    Update the personal best particle pBest
    Optimize gBest using the local search algorithm; if it is not better than gBest, randomly
    replace a current particle
    Update the global optimal particle gBest
    t = t + 1
 end
return gBest
end
```

3.1. Ranked Ordered Value Rule

The continuous solutions must be converted into discrete solutions to make the PSO algorithm suitable for optimizing discrete problems. To this end, the ROV rule with random key representation [49] is used to convert the continuous space of the particle position in the PSO into an arrangement of design points. ROV sorts and transforms the position of the updated particles and assigns the minimum value in the updated position dimension to the minimum value set in the LHD. Separate each column of the updated matrix in a similar manner. Convert the *i*-th largest value of the matrix to the *i*-th largest discrete value set in the LHD; an example of the conversion is shown in Figure 3. It shows an LHD particle after the position is updated, and the corresponding form of the particle is Y after the transformation. In Y, each row represents a design point, and each column represents the position of the design point in each dimension. Obtain an LHD particle by executing the ROV rule on each column.

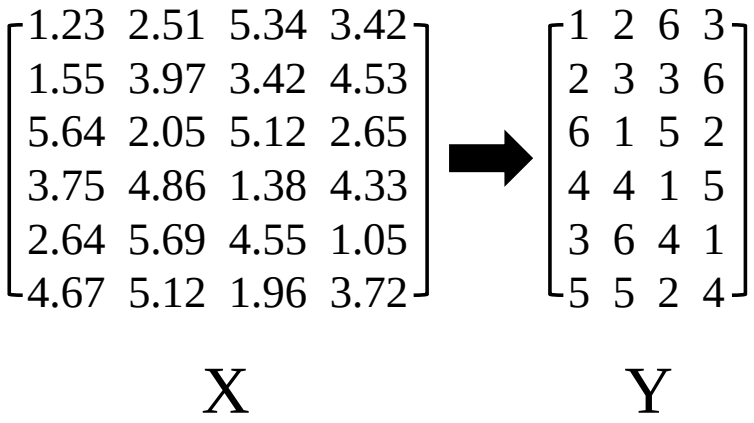


Figure 3. Use the ROV rule to convert the position value of particle X to the arrangement of design point Y.

3.2. Minimum Point Distance Method

According to Equation (2), each particle in PSO will learn its personal best particle *pBest* and global best particle *gBest* at the same time. When the personal best particle *pBest* and the global best particle *gBest* are in opposite directions from the current position x_i and the positions of *pBest* and *gBest* may be quite different, one particle may linger between *pBest* and *gBest*, which will cause "oscillation" [37,38] and reduce the search efficiency of the algorithm.

The MPD method is proposed to avoid the oscillation problem of LHD particles. By rearranging the rows of the personal best particle *pBest* of the LHD particles, the distance between the rows of the LHD particle and the global best particle *gBest* is minimized. The Euclidean distance is used to calculate the distance between points, and the calculation equation is:

$$d_{ij} = [\sum_{k=1}^{m} (x_{ik} - x_{jk})^2]^{1/2} \quad (7)$$

By calculating the row spacing between each row of the personal best particle *pBest* and each row of the global best particle *gBest*, the rows of the current particles are rearranged, and then the particle speed and position are updated. The specific operation is shown in Figure 4. In this way, the personal best particle *pBest* and the global best particle *gBest* can be in the same direction as the current particle, reducing the "oscillation" problem caused by the learning strategy.

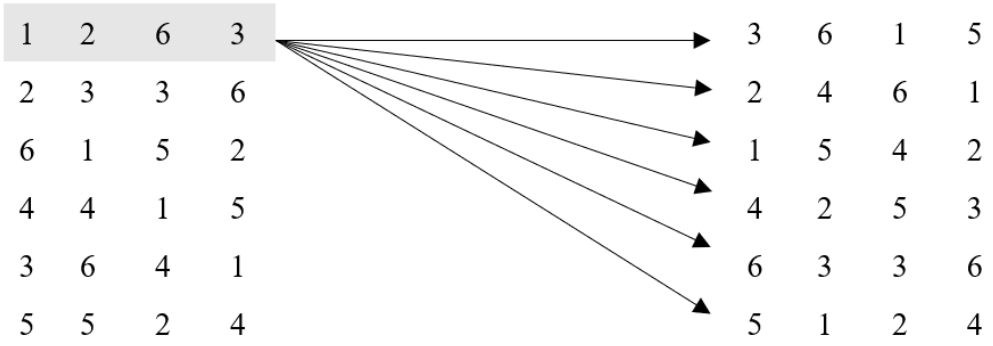


Figure 4. Schematic diagram of the line transformation between the global best position (**left**) and the personal best position (**right**) (The row spacing between the first row on the left and the right is 7.0, 3.0, 3.741, 3.606, 6.633, and 5.831, so the data in the first row and the second row need to be exchanged).

3.3. Local Search Algorithm for gBest

The global best particle *gBest* is different from other particles, and there is no example to refer to evolution. This paper uses four local search strategies to improve local search capabilities.

Inversion: Randomly select a specific area of one column in the matrix and invert the data in the area.

Clipping: Randomly select a specific area of one column in the matrix, randomly select a position in the area, and exchange the data in the area at and above the position with the data in the area below the position.

Center symmetry: Randomly select a certain section of a column in the matrix and convert the symmetrical area with the center into data symmetrical to the data section of the position.

Exchange: randomly select two positions in the matrix to exchange.

After improving the global best particle by the local search algorithm, the original global best particle will be replaced only when the value of ϕp is less than the current *gBest*. Otherwise, a new position will be used to replace a particle in the population randomly.

A schematic diagram of the four local search strategies is shown in Figure 5.

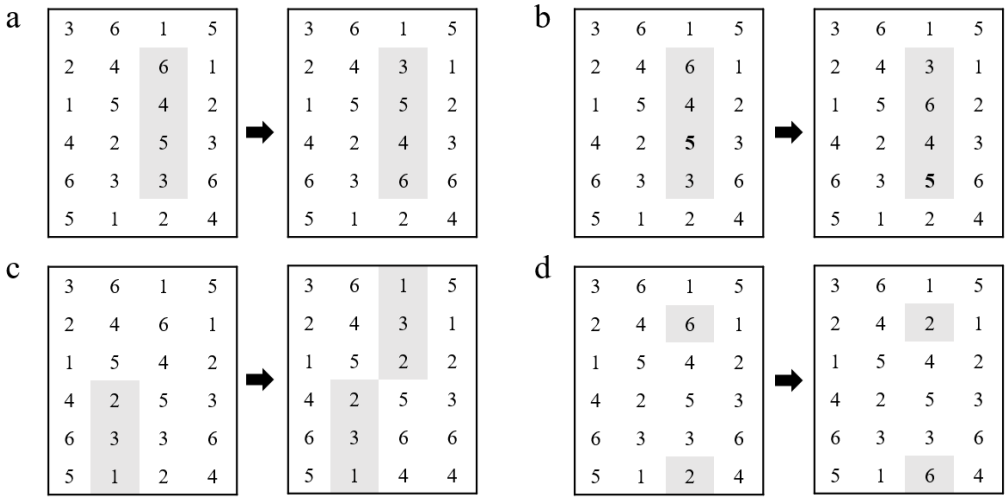


Figure 5. Four different local search strategies, showing a 4-dimensional LH with six design points (specific operations such as dark shaded area display) (**a**) Inversion, (**b**) Clipping, (**c**) Center symmetry, (**d**) Exchange.

3.4. Boundary Processing Scheme

In the PSO algorithm, the particles exceeding the boundary during the movement process will significantly be increased [50,51]. An appropriate boundary processing scheme will improve the optimization capabilities of PSO when optimizing high-dimensional problems. This article solves the boundary problem by using the boundary reflection processing scheme. A schematic diagram of the boundary reflection scheme is shown in Figure 6.

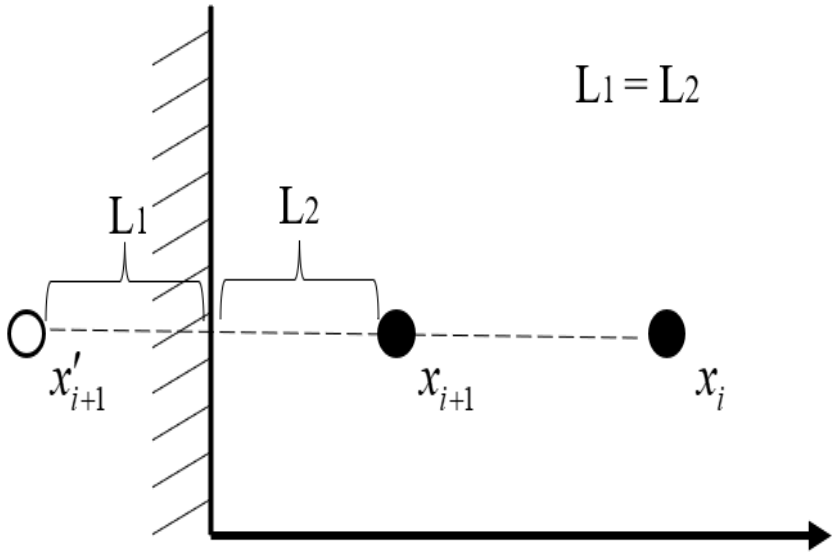


Figure 6. Schematic diagram of the boundary processing scheme. x_i is the current position, x'_{i+1} prime is the position after the position is updated, and x_{i+1} is the final position.

4. Numerical Results and Discussion

In this section, parameter research and numerical comparison of the proposed algorithm will be carried out. After scaling the LHD to $[0, 1]^k$, calculate the ϕp value. We set the population size to 50 and iterate 1000 times. All the results are the average after multiple independent runs.

4.1. Evolutionary Strategy

The evolutionary strategy with the MPD method is tested. In the dimension of 14×60, running 30 times independently with and without this algorithm, the average value obtained is shown in Figure 7. Numerical results highlight the performance of the proposed evolutionary algorithm. The particles will evolve slowly when this algorithm is not used, indicating that the proposed method can slightly suppress the oscillation phenomenon and speed up the evolution of particles. However, the difference is not significant, so the improvement is limited.

4.2. Comprehensive Learning Strategy

In this section, we examine the role of comprehensive learning strategies. The refresh interval m is set to 7, the learning probability Pi is $0.05 + 0.45 \times gen/max_gen$, and the inertia factor is $0.2 + 0.2 \times gen/max_gen$. The algorithm works best, among which gen is the current number of iterations and max_gen is the maximum number of iterations. In the 14×60 dimension, each refresh interval runs 30 times independently, and the average value obtained is shown in Figure 8. Since the numerical result is as small as possible, we can find that setting the refresh interval to 7 is the most consistent with the proposed algorithm.

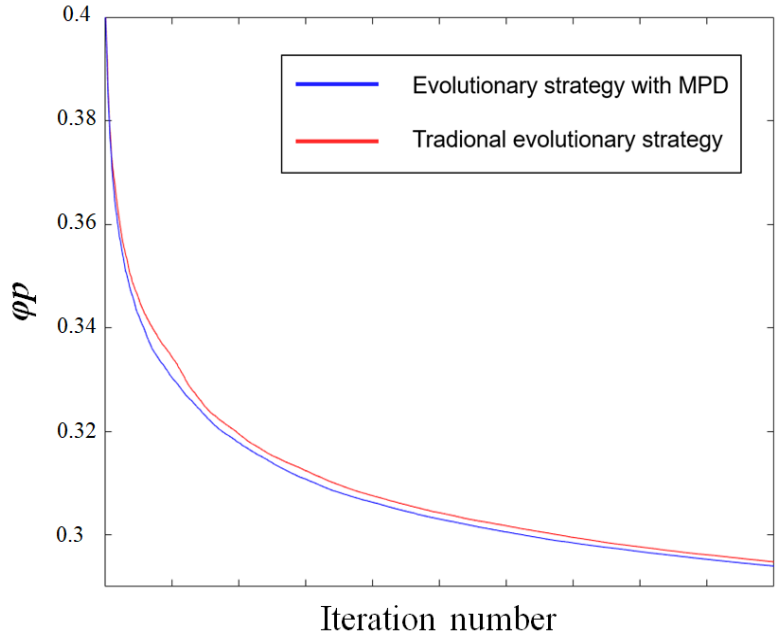


Figure 7. The variation of φp with MPD based evolutionary strategy.

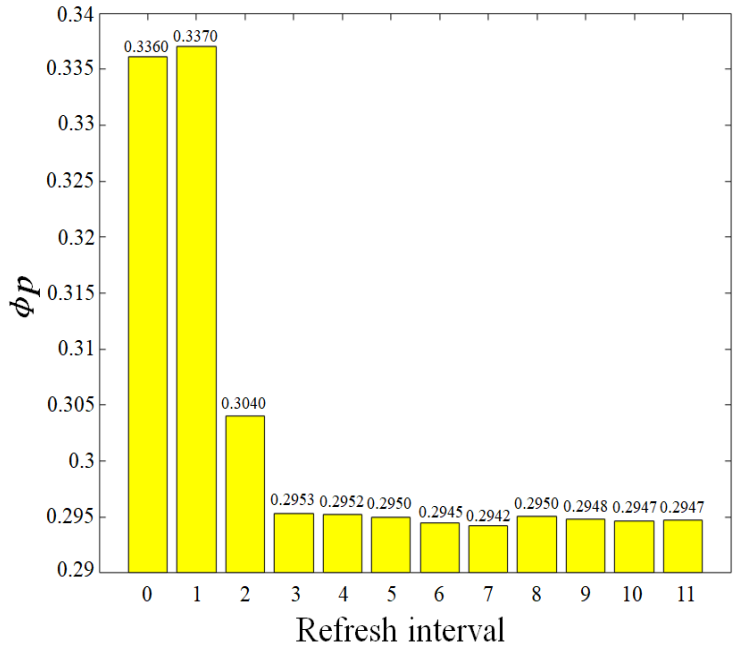


Figure 8. Numerical results of different refresh intervals.

4.3. Comparison Results

Several state-of-the-art methods are compared with our proposed method in this section, including Comprehensive Learning PSO (CLPSO) [52,53], Adaptive Memetic PSO (AMPSO) [36], traditional PSO [26,28], Enhanced Stochastic Evolutionary Algorithm (ESE) [16], Genetic Algorithm (GA) [12], and LaPSO [35]; other algorithms perform performance comparison, after scaling LHD to $[0, 1]^k$, and calculate the value of ϕp.

Tables 1 and 2 show 20 sets of the value of ϕp with dimensions m = 2, 6, 10, 14, 18 and n = 20, 40, 60, 80. The best results are shown in bold. Meanwhile, we selected five dimensions of 2 × 60, 6 × 60, 10 × 60, 14 × 60, and 18 × 60 to draw the population evolution diagram, shown in Figure 9. Furthermore, a box plot of 18 × 60 between the proposed algorithm and the compared algorithm was drawn, as shown in Figure 10.

Table 1. Comparison of the IHPSO algorithm with CLPSO, AMPSO, PSO, ESE, GA, and LaPSO.

Dimension	Algorithm	Measure	$n = 20$	$n = 40$	$n = 60$	$n = 80$
$m = 2$	CLPSO	Mean	4.27695	8.03165	11.7084	15.1155
		Std	0.268134	0.314826	0.694002	1.082523
	AMPSO	Mean	4.082846	6.430663	8.770171	11.02583
		Std	0.082087	0.338895	0.419517	0.576738
	PSO	Mean	4.055958	6.051447	8.306216	17.43493
		Std	8.88×10^{-16}	0.315769	1.198285	1.608196
	ESE	Mean	5.070727	9.852305	14.73661	19.39307
		Std	0.042239	0.558617	0.912121	1.359123
	GA	Mean	4.062401	5.852419	7.473188	8.786267
		Std	0.112058	0.149028	0.340156	0.418199
	LaPSO	Mean	4.118622	6.628554	8.190774	11.61435
		Std	0.087171	0.339539	0.39903	0.261372
	IHPSO	Mean	**3.981064**	**5.825657**	**7.370558**	**8.743883**
		Std	**0.188432**	**0.123855**	**0.27801**	**0.316564**
$m = 6$	CLPSO	Mean	0.755957	0.999181	1.182143	1.325422
		Std	0.013096	0.014428	0.018933	0.020479
	AMPSO	Mean	0.690927	0.882016	1.015902	1.118163
		Std	0.009545	0.014175	0.010908	0.018196
	PSO	Mean	0.695391	0.964931	1.154155	1.300088
		Std	0.011854	0.019534	0.023778	0.034823
	ESE	Mean	0.843597	1.130684	1.338438	1.505415
		Std	0.010928	0.020177	0.021391	0.022172
	GA	Mean	0.659701	0.798081	0.895581	0.971953
		Std	0.008956	0.007465	0.007773	0.005263
	LaPSO	Mean	0.667212	0.828105	0.936295	1.027103
		Std	0.009503	0.009213	0.010532	0.010696
	IHPSO	Mean	**0.666972**	**0.806242**	**0.906449**	**0.984813**
		Std	**0.009571**	**0.009701**	**0.014546**	**0.020373**
$m = 10$	CLPSO	Mean	0.400005	0.490980	0.550981	0.595821
		Std	0.004002	0.004836	0.004982	0.005118
	AMPSO	Mean	0.372430	0.444516	0.492998	0.528137
		Std	0.003088	0.00411	0.00387	0.004777
	PSO	Mean	0.370697	0.473244	0.537667	0.584522
		Std	0.005284	0.007759	0.009553	0.010786
	ESE	Mean	0.434070	0.539433	0.605248	0.653991
		Std	0.004917	0.005437	0.007293	0.00531
	GA	Mean	0.355487	0.416147	0.45421	0.484463
		Std	0.002606	0.002666	0.00242	0.002496
	LaPSO	Mean	0.357563	0.419152	0.459798	0.491216
		Std	0.002862	0.003152	0.003987	0.003844
	IHPSO	Mean	**0.355233**	**0.409927**	**0.447199**	**0.475609**
		Std	**0.002995**	**0.002826**	**0.004459**	**0.004361**

Table 2. Comparison of the proposed IHPSO with CLPSO, AMPSO, PSO, ESE, GA, and LaPSO.

Dimension	Algorithm	Measure	$n = 20$	$n = 40$	$n = 60$	$n = 80$
$m = 14$	CLPSO	Mean	0.269425	0.31932	0.350386	0.373451
		Std	0.001927	0.002999	0.00284	0.00256
	AMPSO	Mean	0.253795	0.294597	0.321671	0.34096
		Std	0.001683	0.002171	0.001937	0.002072
	PSO	Mean	0.252575	0.310313	0.343301	0.366767
		Std	0.002471	0.004441	0.004086	0.004428
	ESE	Mean	0.28967	0.344736	0.379043	0.403719
		Std	0.002499	0.003196	0.003377	0.003723
	GA	Mean	0.246427	0.282642	0.305583	0.322375
		Std	0.001653	0.001479	0.001741	0.001401
	LaPSO	Mean	0.244589	0.279448	0.301728	0.318759
		Std	0.001495	0.001272	0.001639	0.001395
	IHPSO	Mean	**0.243145**	**0.275425**	**0.29421**	**0.31047**
		Std	**0.001327**	**0.00225**	**0.001569**	**0.001791**

Table 2. *Cont.*

Dimension	Algorithm	Measure	n = 20	n = 40	n = 60	n = 80
m = 18	CLPSO	Mean	0.202025	0.235049	0.254872	0.269295
		Std	0.001344	0.001654	0.001584	0.001466
	AMPSO	Mean	0.192168	0.219966	0.236842	0.24933
		Std	0.001495	0.001027	0.001175	0.001658
	PSO	Mean	0.191344	0.227283	0.250179	0.265241
		Std	0.001357	0.003413	0.002731	0.003079
	ESE	Mean	0.216271	0.25211	0.273186	0.28809
		Std	0.001382	0.001373	0.001873	0.001753
	GA	Mean	0.188378	0.213831	0.228716	0.240114
		Std	0.00119	0.001074	0.000933	0.000972
	LaPSO	Mean	0.185635	0.209297	0.224207	0.234901
		Std	0.000784	0.00094	0.000766	0.000898
	IHPSO	Mean	**0.184472**	**0.20569**	**0.218815**	**0.230125**
		Std	**0.000891**	**0.000842**	**0.000889**	**0.001343**

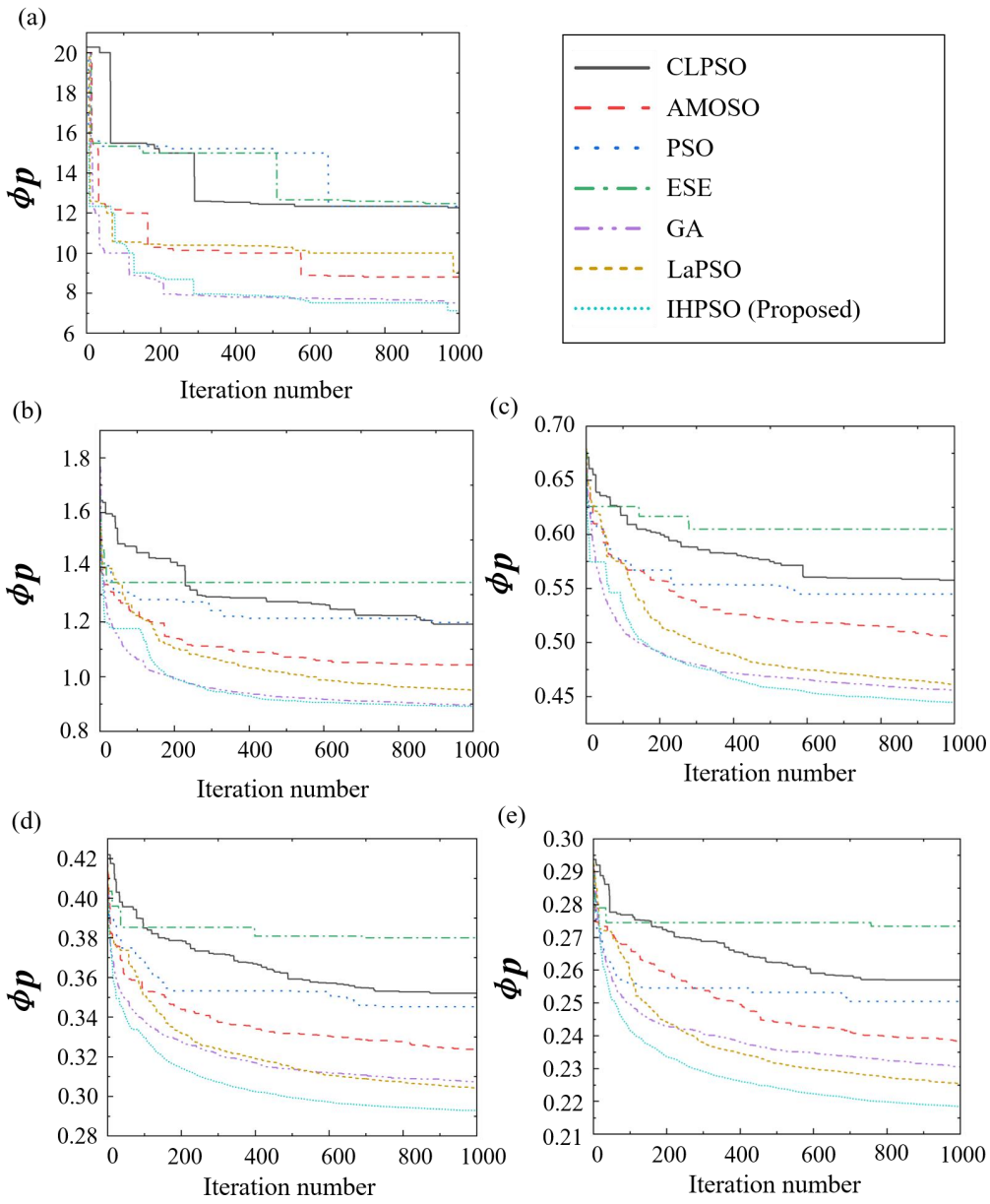


Figure 9. Performance trends of different LHD algorithms: (**a**) 2 × 60, (**b**) 6 × 60, (**c**) 10 × 60, (**d**) 14 × 60, and (**e**) 18 × 60.

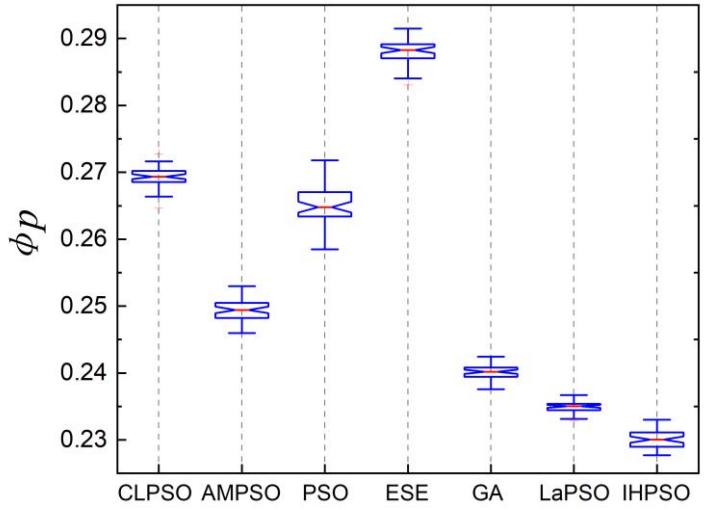


Figure 10. Boxplots of different algorithms at 18 × 80 LHD problem.

As shown in Tables 1 and 2, the proposed algorithm outperforms all the compared algorithms in most cases of numerical tests. The numerical results in six dimensions are also close to the optimal data in the compared algorithms. The proposed algorithm is superior to the compared algorithms in high dimensions after parameter adjustment and has greater stability. Experiments find that the initial values of c_1, c_2, and ω in Equation (2) are 0.5, 2.7, and 0.25, respectively. When the global optimal particle *gBest* is updated, the parameters in Equation (2) change into the initial value. With the increase in iterations, c_1 gradually increases and c_2 decreases. The method is to enhance the search capabilities of particle swarms, shown as:

$$|c_i(t+1) - c_i(t)| \leq \delta,\ i = 1,2 \tag{8}$$

The increment or decrement between the two generations of particles should not be too large. After testing, when the change δ is set as a random number in the interval [0.05, 0.1], the experimental result is appropriate. The changes in c_1 and c_2 are limited, where the upper limit of c_1 is 1.8 and the lower limit of c_2 is 1.9.

From the population evolution trend shown in Figure 9, it can be found that the proposed algorithm has a very high evolution efficiency. The reduction rate of the φp value is faster than other algorithms. There is no evolutionary stagnation, fast convergence, and excellent numerical results. In the case of low dimensionality, the proposed algorithm can evolve rapidly. After approximately 200 generations of evolution, its performance is better than almost all compared algorithms. In the case of high dimensionality, after approximately 100 generations of evolution, its performance can be better than all the compared algorithms. The numerical results of the proposed algorithm perform better in all dimensions.

From the box plot in Figure 10, it can be found that in the high dimension of 18 × 80, the proposed algorithm exhibits good performance compared with the other algorithms. On the one hand, the value of the IHPSO algorithm is smaller, which indicates that the proposed algorithm can fill the space completely. On the other hand, the IHPSO algorithm is numerically concentrated, indicating that the performance is relatively stable. Furthermore, the maximum value of the IHPSO algorithm is still smaller than the minimum result among other algorithms, which indicates that the proposed IHPSO has a good performance in optimizing HLHD and can obtain better results than other algorithms.

5. Conclusions

This paper proposes an improved hybrid PSO (IHPSO) to optimize the High-dimensional LHD (HLHD), which combines local and global search strategies. First, the minimum point

spacing method is proposed to solve the oscillation problem in the PSO. Second, the ranked ordered value (ROV) is adopted to achieve the discretization of PSO. Third, a comprehensive learning strategy is used to ensure the diversity of the population in the evolution process. The superiority of the proposed algorithm is verified by numerical comparison with six other state-of-the-art algorithms. The advantages of the proposed method are shown as follows:

(1) The proposed IHPSO can obtain a better HLHD than the compared algorithm at the same number of iterations. It performs better in high dimensions, and the numerical results are significantly better than the compared algorithms.

(2) IHPSO can effectively improve the efficiency of obtaining the near-optimal HLHD. Compared with the compared algorithm, IHPSO converges faster and performs better in optimizing medium and large LHDs. IHPSO shows better local search capabilities and global search performance.

(3) IHPSO effectively promotes convergence in the initial optimization stage and maintains the convergence performance until the end of the iteration. Even in the case of a small number of iterations, IHPSO still exhibits excellent performance. Under the same number of iterations, IHPSO can obtain HLHD with better space-filling.

(4) IHPSO balances the local search and global search capabilities by adaptively adjusting the parameters of the PSO equation, ensuring the diversity of the population, and can quickly obtain the near-optimal HLHD.

Although our proposed IHPSO algorithm is efficient and stable and can search for the optimal solution, it needs further optimization. IHPSO has many parameters that need to be discussed. These parameters significantly impact the algorithm's performance and will be further researched in future work.

Author Contributions: Conceptualization, Z.X. and F.W.; methodology, software, validation, Z.X. and D.X.; formal analysis, J.W.; resources, J.L.; writing-original draft preparation, Z.X.; writing-review and editing, N.Y.; visualization, S.X.; supervision, D.G. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by the National Key R&D Program of China (2018YFB1900301) National Natural Science Foundation of China (71901203 and 72204246). At the same time, the authors sincerely thank the editor and anonymous reviewers for their insightful comments that help us improve the quality of the article.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: All relevant data are included in the article.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Kianifar, M.R.; Campean, F.; Wood, A. Application of permutation genetic algorithm for sequential model building-model validation design of experiments. *Soft Comput.* **2016**, *20*, 3023–3044. [CrossRef]
2. Han, Z.H.; Zhang, Y.; Song, C.X.; Zhang, K.S. Weighted Gradient-Enhanced Kriging for High-Dimensional Surrogate Modeling and Design Optimization. *AIAA J.* **2017**, *55*, 4330–4346. [CrossRef]
3. Yao, Y.T.; Wang, J.Y.; Xie, M. Adaptive residual CNN-based fault detection and diagnosis system of small modular reactors. *Appl. Soft Comput.* **2022**, *114*, 108064. [CrossRef]
4. Wu, Z.P.; Wang, D.H.; Okolo, P.N.; Zhao, K.; Zhang, W.H. Efficient space-filling and near-orthogonality sequential Latin hypercube for computer experiments. *Comput. Methods Appl. Mech. Eng.* **2017**, *324*, 348–365. [CrossRef]
5. Garud, S.S.; Karimi, I.A.; Kraft, M. Design of computer experiments: A review. *Comput. Chem. Eng.* **2017**, *106*, 71–95. [CrossRef]
6. Grosso, A.; Jamali, A.; Locatelli, M. Finding maximin latin hypercube designs by Iterated Local Search heuristics. *Eur. J. Oper. Res.* **2009**, *197*, 541–547. [CrossRef]
7. Crombecq, K.; Laermans, E.; Dhaene, T. Efficient space-filling and non-collapsing sequential design strategies for simulation-based modeling. *Eur. J. Oper. Res.* **2011**, *214*, 683–696. [CrossRef]
8. Ma, Y.M.; Xiao, Y.; Wang, J.; Zhou, L.B. Multicriteria Optimal Latin Hypercube Design-Based Surrogate-Assisted Design Optimization for a Permanent-Magnet Vernier Machine. *IEEE Trans. Magn.* **2022**, *58*, 1–5. [CrossRef]

9. Yao, Y.T.; Yang, M.H.; Wang, J.Y.; Xie, M. Multivariate Time-Series Prediction in Industrial Processes via a Deep Hybrid Network Under Data Uncertainty. *IEEE Trans. Ind. Inf.* **2023**, *19*, 1977–1987. [CrossRef]
10. van Dam, E.R.; Husslage, B.; Melissen, H.H. Maximin Latin hypercube designs in two dimensions. *Oper. Res.* **2007**, *55*, 158–169. [CrossRef]
11. Audze, P.; Eglais, V. New approach for planning out of experiments. *Probl. Dyn. Strengths* **1977**, *35*, 104–107.
12. Liefvendahl, M.; Stocki, R. A study on algorithms for optimization of Latin hypercubes. *J. Stat. Plan. Inference* **2006**, *136*, 3231–3247. [CrossRef]
13. Fuerle, F.; Sienz, J. Formulation of the Audze-Eglais uniform Latin hypercube design of experiments for constrained design spaces. *Adv. Eng. Softw.* **2011**, *42*, 680–689. [CrossRef]
14. Morris, M.D.; Mitchell, T.J. Exploratory designs for computational experiments. *J. Stat. Plan. Inference* **1995**, *43*, 381–402. [CrossRef]
15. Ye, K.Q.; Li, W.; Sudjianto, A. Algorithmic construction of optimal symmetric Latin hypercube designs. *J. Stat. Plan. Inference* **2000**, *90*, 145–159. [CrossRef]
16. Jin, R.C.; Chen, W.; Sudjianto, A. An efficient algorithm for constructing optimal design of computer experiments. *J. Stat. Plan. Inference* **2005**, *134*, 268–287. [CrossRef]
17. Viana, F.A.C.; Venter, G.; Balabanov, V. An algorithm for fast optimal Latin hypercube design of experiments. *Int. J. Numer. Meth. Eng.* **2010**, *82*, 135–156. [CrossRef]
18. Fang, K.T.; Ma, C.X.; Winker, P. Centered L-2-discrepancy of random sampling and Latin hypercube design, and construction of uniform designs. *Math. Comput.* **2002**, *71*, 275–296. [CrossRef]
19. Doerr, C.; De Rainville, F.M. Constructing Low Star Discrepancy Point Sets with Genetic Algorithms. In Proceedings of the 15th Genetic and Evolutionary Computation Conference (GECCO), Madrid, Spain, 11–15 July 2013; Association Computing Machinery: Amsterdam, The Netherlands, 2013. [CrossRef]
20. De Rainville, F.M.; Gagne, C.; Teytaud, O.; Laurendeau, D. Evolutionary Optimization of Low-Discrepancy Sequences. *ACM Trans. Model. Comput. Simul.* **2012**, *22*, 1–25. [CrossRef]
21. Bates, S.J.; Sienz, J.; Langley, D.S. Formulation of the Audze-Eglais Uniform Latin Hypercube design of experiments. *Adv. Eng. Softw.* **2003**, *34*, 493–506. [CrossRef]
22. Zhu, H.G.; Liu, L.; Long, T.; Peng, L. A novel algorithm of maximin Latin hypercube design using successive local enumeration. *Eng. Optimiz.* **2012**, *44*, 551–564. [CrossRef]
23. Ba, S.; Myers, W.R.; Brenneman, W.A. Optimal Sliced Latin Hypercube Designs. *Technometrics* **2015**, *57*, 479–487. [CrossRef]
24. Pholdee, N.; Bureerat, S. An efficient optimum Latin hypercube sampling technique based on sequencing optimisation using simulated annealing. *Int. J. Syst. Sci.* **2015**, *46*, 1780–1789. [CrossRef]
25. Guiban, K.L.; Rimmel, A.; Weisser, M.A.; Tomasik, J. The First Approximation Algorithm for the Maximin Latin Hypercube Design Problem. *Oper. Res.* **2018**, *66*, 253–266. [CrossRef]
26. Kennedy, J.; Eberhart, R. Particle swarm optimization. In Proceedings of the International Conference on Neural Networks (ICNN 95), Perth, WA, Australia, 27 November–1 December 1995; IEEE: Perth, Australia, 1995.
27. Houssein, E.H.; Gad, A.G.; Hussain, K.; Suganthan, P.N. Major Advances in Particle Swarm Optimization: Theory, Analysis, and Application. *Swarm Evol. Comput.* **2021**, *63*, 100868. [CrossRef]
28. Kennedy, M.C.; O'Hagan, A. Predicting the output from a complex computer code when fast approximations are available. *Biometrika* **2000**, *87*, 1–13. [CrossRef]
29. Sengupta, S.; Basak, S.; Peters, R.A., II. Particle Swarm Optimization: A Survey of Historical and Recent Developments with Hybridization Perspectives. *Mach. Learn. Knowl. Extr.* **2019**, *1*, 157–191. [CrossRef]
30. Bratton, D.; Kennedy, J. Defining a standard for particle swarm optimization. In Proceedings of the IEEE Swarm Intelligence Symposium, Honolulu, HI, USA, 1–5 April 2007; IEEE: Honolulu, HI, USA, 2007.
31. Jain, M.; Singh, V.; Rani, A. A novel nature-inspired algorithm for optimization: Squirrel search algorithm. *Swarm Evol. Comput.* **2019**, *44*, 148–175. [CrossRef]
32. Clerc, M.; Kennedy, J. The particle swarm—Explosion, stability, and convergence in a multidimensional complex space. *IEEE Trans. Evol. Comput.* **2002**, *6*, 58–73. [CrossRef]
33. Dixit, A.; Mani, A.; Bansal, R. CoV2-Detect-Net: Design of COVID-19 prediction model based on hybrid DE-PSO with SVM using chest X-ray images. *Inform. Sci.* **2021**, *571*, 676–692. [CrossRef] [PubMed]
34. Gheisari, S.; Meybodi, M.R. BNC-PSO: Structure learning of Bayesian networks by Particle Swarm Optimization. *Inform. Sci.* **2016**, *348*, 272–289. [CrossRef]
35. Chen, R.B.; Hsieh, D.N.; Hung, Y.; Wang, W.C. Optimizing Latin hypercube designs by particle swarm. *Stat. Comput.* **2013**, *23*, 663–676. [CrossRef]
36. Aziz, M.; Tayarani-N, M.H. An adaptive memetic Particle Swarm Optimization algorithm for finding large-scale Latin hypercube designs. *Engin. Appl. Artif. Intel.* **2014**, *36*, 222–237. [CrossRef]
37. Van den Bergh, F.; Engelbrecht, A.P. A cooperative approach to particle swarm optimization. *IEEE Trans. Evol. Comput.* **2004**, *8*, 225–239. [CrossRef]
38. Xu, G.P.; Cui, Q.L.; Shi, X.H.; Ge, H.W.; Zhan, Z.H.; Lee, H.P.; Liang, Y.C.; Tai, R.; Wu, C.G. Particle swarm optimization based on dimensional learning strategy. *Swarm Evol. Comput.* **2019**, *45*, 33–51. [CrossRef]

39. Ye, H.T.; Luo, W.G.; Li, Z.Q. Convergence Analysis of Particle Swarm Optimizer and Its Improved Algorithm Based on Velocity Differential Evolution. *Comput. Intel Neurosc.* **2013**, *2013*, 384125. [CrossRef]
40. Bansal, J.C.; Singh, P.K.; Saraswat, M.; Verma, A.; Jadon, S.S.; Abraham, A. Inertia Weight strategies in Particle Swarm Optimization. In Proceedings of the 2011 Third World Congress on Nature and Biologically Inspired Computing, Salamanca, Spain, 19–21 October 2011. [CrossRef]
41. Larsen, R.B.; Jouffroy, J.; Lassen, B. On the premature convergence of particle swarm optimization. In Proceedings of the European Control Conference (ECC), Aalborg, Denmark, 29 June–1 July 2016. [CrossRef]
42. Qian, P. Sliced Latin Hypercube Designs. *J. Am. Stat. Assoc.* **2012**, *107*, 393–399. [CrossRef]
43. Santner, T.J.; Williams, B.J.; Notz, W.I. *The Design and Analysis of Computer Experiments*, 2nd ed.; Springer: Berlin, Germany, 2019.
44. Gong, Y.J.; Zhang, J.; Chung, H.S.H.; Chen, W.N.; Zhan, Z.H.; Li, Y.; Shi, Y.H. An Efficient Resource Allocation Scheme Using Particle Swarm Optimization. *IEEE Trans. Evol. Comput.* **2012**, *16*, 801–816. [CrossRef]
45. Michaloglou, A.; Tsitsas, N.L. Feasible Optimal Solutions of Electromagnetic Cloaking Problems by Chaotic Accelerated Particle Swarm Optimization. *Mathematics* **2021**, *9*, 2725. [CrossRef]
46. Alkayem, N.F.; Cao, M.; Shen, L.; Fu, R.; Sumarac, D. The combined social engineering particle swarm optimization for real-world engineering problems: A case study of model-based structural health monitoring. *Appl. Soft Comput.* **2022**, *123*, 108919. [CrossRef]
47. Michaloglou, A.; Tsitsas, N.L. A Brain Storm and Chaotic Accelerated Particle Swarm Optimization Hybridization. *Algorithms* **2023**, *16*, 208. [CrossRef]
48. Alkayem, N.F.; Shen, L.; Al-hababi, T.; Qian, X.; Cao, M. Inverse Analysis of Structural Damage Based on the Modal Kinetic and Strain Energies with the Novel Oppositional Unified Particle Swarm Gradient-Based Optimizer. *Appl. Sci.* **2022**, *12*, 11689. [CrossRef]
49. Bean, J.C. Genetic algorithms and random keys for sequencing and optimization. *Inform. J. Comput.* **1994**, *6*, 154–160. [CrossRef]
50. Chu, W.; Gao, X.G.; Sorooshian, S. Handling boundary constraints for particle swarm optimization in high-dimensional search space. *Inf. Sci.* **2011**, *181*, 4569–4581. [CrossRef]
51. Helwig, S.; Wanka, R. Particle swarm optimization in high-dimensional bounded search spaces. In Proceedings of the IEEE Swarm Intelligence Symposium, Honolulu, HI, USA, 1–5 April 2007. [CrossRef]
52. 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]
53. Wu, Q.; Zhang, C.J.; Zhang, M.Y.; Yang, F.J.; Gao, L. A modified comprehensive learning particle swarm optimizer and its application in cylindricity error evaluation problem. *Math. Biosci. Eng.* **2019**, *16*, 1190–1209. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.