

Article

A Quick Search Dynamic Vector-Evaluated Particle Swarm Optimization Algorithm Based on Fitness Distance

Suyu Wang , Dengcheng Ma and Miao Wu

School of Mechanical Electronic & Information Engineering, China University of Mining and Technology-Beijing, Beijing 100083, China; mdc@cumtb.edu.cn (D.M.); wum@cumtb.edu.cn (M.W.)

* Correspondence: wsy@cumtb.edu.cn; Tel.: +86-010-62331083

Abstract: A quick search dynamic vector-evaluated particle swarm optimization algorithm based on fitness distance (DVEPSO/FD) is proposed according to the fact that some dynamic multi-objective optimization methods, such as the DVEPSO, cannot achieve a very accurate Pareto optimal front (POF) tracked after each objective changes, although they exhibit advantages in multi-objective optimization. Featuring a repository update mechanism using the fitness distance together with a quick search mechanism, the DVEPSO/FD is capable of obtaining the optimal values that are closer to the real POF. The fitness distance is used to streamline the repository to improve the distribution of nondominant solutions, and the flight parameters of the particles are adjusted dynamically to improve the search speed. Groups of the standard benchmark experiments are conducted and the results show that, compared with the DVEPSO method, from the figures generated by the test functions, DVEPSO/FD achieves a higher accuracy and clearness with the POF dynamically changing; from the values of performance indexes, the DVEPSO/FD effectively improves the accuracy of the tracked POF without destroying the stability. The proposed DVEPSO/FD method shows a good dynamic change adaptability and solving set ability of the dynamic multi-objective optimization problem.

Keywords: dynamic multi-objective optimization; DVEPSO/FD; fitness distance; quick search mechanism

MSC: 90C29



Citation: Wang, S.; Ma, D.; Wu, M. A Quick Search Dynamic Vector-Evaluated Particle Swarm Optimization Algorithm Based on Fitness Distance. *Mathematics* **2022**, *10*, 1587. <https://doi.org/10.3390/math10091587>

Academic Editors: Árpád Bűrmen and Tadej Tuma

Received: 12 April 2022

Accepted: 5 May 2022

Published: 7 May 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 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

The multi-objective optimization algorithms [1,2] are mainly used for optimizing static multi-objective optimization problems (MOPs) [3], but in the real world, the objective functions of MOPs often conflict with each other, and at least one objective function is dynamically changing with time, which becomes the dynamic multi-objective optimization problem (DMOPs) [4,5]. At this time, the static multi-objective optimization algorithms are not effective or even ineffective when solving such problems. With regard to this, more attention has been received on dynamic multi-objective optimization algorithms (DMOAs) [6,7]. The DMOAs should detect environmental changes and respond, accurately obtain the evolution direction of the population, and continuously find the dynamically changing Pareto optimal front (POF) [8], so as to achieve the goal of solving DMOPs.

There are already many different categories of DMOAs, and the main categories are Evolutionary Algorithm (EA) [9,10], Ant Colony Optimization (ACO) [11], Immune-based Algorithm (IBA) [12,13], Particle Swarm Optimization (PSO) [14,15], etc. The dynamic EA is called the Dynamic Multi-Objective Evolutionary Algorithm (DMOEA) [16,17]. In [9], the use of diploid representations and dominance operators was investigated in EA to improve the performance in environments that vary with time. Simulation results showed that a diploid EA with an evolving dominance map adapts quickly to the sudden changes in this environment problem. In [11], the dynamic Traveling Salesperson Problem (TSP) was studied. Several strategies were proposed to make ACO better adaptive to the dynamic

changes of optimization problems. In [12], the main problem of biologically inspired algorithms (such as EA or PSO) when applied to dynamic optimization was believed to be forcing their readiness for continuous optimizations in changing locations. IBA, an instance of an algorithm that adapts by innovation, seemed to be a perfect candidate for continuous exploration of a search space. Various implementations of the immune principles were described and these instantiations on complex environments were compared. In [14], it was analyzed whether the cooperative system rules they used for static optimization problems make sense when applied to DMOPs. Two control rules for updating the former were proposed and compared. The test results proved that the proposed cooperative system and rules based on the fuzzy set were more suitable for dynamically changing environments. In addition, the more commonly used algorithms that can solve dynamic multi-objective optimization problems are DMSGAI [18], including DMSGAI-A and DMSGAI-B.

Because of its simple principle, few rules, wide application, and fast and accurate tracking of the POF, the PSO algorithm has certain advantages over other optimization algorithms. Therefore, it is more practical to study DMOAs based on the PSO. In addition to the solution proposed by Pelta above [14], there are still several effective solutions. In [19], the new variants of PSO were explored and designed specifically for working in a dynamic environment. The main idea is to split the population of particles into a group of interacting groups that interact locally through an exclusion parameter and globally through a new anti-convergence operator. In [20], a new algorithm based on hierarchical particle swarm optimization (H-PSO) was proposed, namely Partitioned Hierarchical particle swarm optimization (PH-PSO). The algorithm maintained a particle hierarchy that was divided into subgroups within a limited number of generations after the environment had changed. In [21], the application of vector-evaluated particle swarm optimization (VEPSO) in solving DMOPs was introduced. VEPSO [22,23] was first proposed by Parsopoulos et al., inspired by vector-evaluated Genetic Algorithms (VEGA) [24]. The results showed that VEPSO can solve the DMOPs with a discontinuous POF. Their papers have become the source of DVEPSO and are widely used and contrasted by many researchers.

The above literature indicates that DVEPSO is representative in the algorithms of solving DMOPs, and it is discussed by many researchers. However, the POF tracked after each objective changes is not very accurate; therefore, the effect of DVEPSO is not the best and still needs to be improved in this aspect. The improvement scheme of DVEPSO is discussed in this paper, and a new quick search DVEPSO based on fitness distance, which is called DVEPSO/FD, is proposed. It features a repository update mechanism using the fitness distance together with a quick search mechanism. The fitness distance is introduced, which is used to further streamline the repository, thus improving the distribution of nondominant solutions, and the POF tracked after each objective changes is closer to the real POF. At the same time, in order to quickly find the POF before and after the first change in the environment, the flight parameters of the particles are adjusted dynamically to improve the search speed. The simulation experiments on the standard test functions prove that DVEPSO/FD achieves a higher accuracy and stability with the POF dynamically changing, which shows a good dynamic change adaptability and solving set ability of the dynamic multi-objective optimization problem.

2. Related Work

This section introduces the DMOP, PSO, and DVEPSO in basic terms.

2.1. DMOP

The characteristic of DMOPs is that the objective changes with time and can be defined as Equation (1) [5].

$$\min F(x, t) = \langle f_1(x, t), f_2(x, t), \dots, f_M(x, t) \rangle \quad (1)$$

$s.t \ x \in \Omega$

where $x = \langle x_1, x_2, \dots, x_n \rangle$ is the decision vector, t is time or an environment variable, $f_i(x, t) : \Omega \rightarrow R (i = 1, \dots, M)$, $\Omega = [L_1, U_1] \times [L_2, U_2] \times \dots \times [L_n, U_n]$, and $L_i, U_i \in R$ are the lower and upper bounds of the i th decision variable, respectively.

DMOPs are some MOPs in which at least one objective is dynamically changing with time. When the objective changes, in order to find the optimal nondominant solution, it is necessary to continuously track the POF changing with time. There are three basic ideas for solving DMOPs: converting to a series of stable static multi-objective optimization problems; using weights to combine multi-objectives into one dynamic objective; decomposing multi-objectives into many single dynamic objectives and, simultaneously, their multi-threaded optimization with information sharing. The algorithm proposed in this paper is based on information sharing to solve DMOPs.

2.2. Basic PSO

Every potential solution can be called a “particle”, and PSO has a swarm that is constructed by particles. The initial swarm is created with each individual having an initial position and velocity, both of which are randomly generated. The flight of particles is mainly influenced by two parameters: one is the personal best, the best position where each particle is found according to its own experience during the flight; and the other is the global best, the best position where the entire swarm is currently found. The particles constantly adjust their flight through these two parameters.

Suppose there are I particles in the swarm, and the algorithm iterates a total of R times. The position of each particle at the r -th iteration is recorded as X_i^r , and the velocity as V_i^r , $i = 1, 2, \dots, I$, $r = 1, 2, \dots, R$. In the search space of D dimensions, $X_i^r = (x_{i1}^r, x_{i2}^r, \dots, x_{iD}^r)$ and $V_i^r = (v_{i1}^r, v_{i2}^r, \dots, v_{iD}^r)$. Particles in the algorithm update velocity and position according to the following two Equations (2) and (3) (Liang and Kang, 2016):

$$v_{iD}^{r+1} = wv_{iD}^r + c_1r_1(pbest^r - x_{iD}^r) + c_2r_2(gbest^r - x_{iD}^r) \quad (2)$$

$$x_{iD}^{r+1} = x_{iD}^r + v_{iD}^{r+1} \quad (3)$$

where w is the inertia weight, c_1 and c_2 are learning factors, r_1 and r_2 are the random numbers between 0 and 1, $pbest$ is the personal best, and $gbest$ is the global best.

2.3. Basic DVEPSO

For DVEPSO, each subgroup solves only one objective optimization problem, which shares their information with each other by taking advantage of the global best position in particle speed updates. The core structure of the DVEPSO algorithm can be summarized as follows: information sharing mechanism, environmental monitoring and response mechanism, and repository update mechanism. Each particle has a local optimal and a global optimal to guide its search in the search space. The local optimum of a particle is its personal best, $pbest$, which is the best position where the particle is currently found. The global guide to particles, $gbest$, is selected from one of the subgroups through a knowledge sharing mechanism. The particles used to detect the changes in environment are called sentinel particles. When the sentinel particles detect changes in the environment, the subgroup corresponding to the change will randomly re-initialize a certain proportion of the partial particles, which is usually 30%. After re-initialization, each particle's objective fitness value and its optimal position are re-evaluated and the repository is updated. At the same time, the size of the repository is also limited. If the repository reaches the upper limit, the excess nondominant solution is deleted from the crowded area based on the crowding distance.

3. Quick Search DVEPSO Based on Fitness Distance (DVEPSO/FD)

This section is divided into three main parts: first, the introduction about the composition of the system is shown; then, the explanation of the repository update mechanism,

quick search mechanism, and other modules included in the system is shown; and finally, the pseudo-code of the algorithm is presented.

3.1. System Composition

Figure 1 shows the composition of the DVEPSO/FD system, which overcomes the shortcomings of the DVEPSO method that cannot obtain a very accurate POF tracked after each objective changes. The repository update mechanism based on the fitness distance and quick search mechanism are designed to respond quickly when environmental changes are monitored.

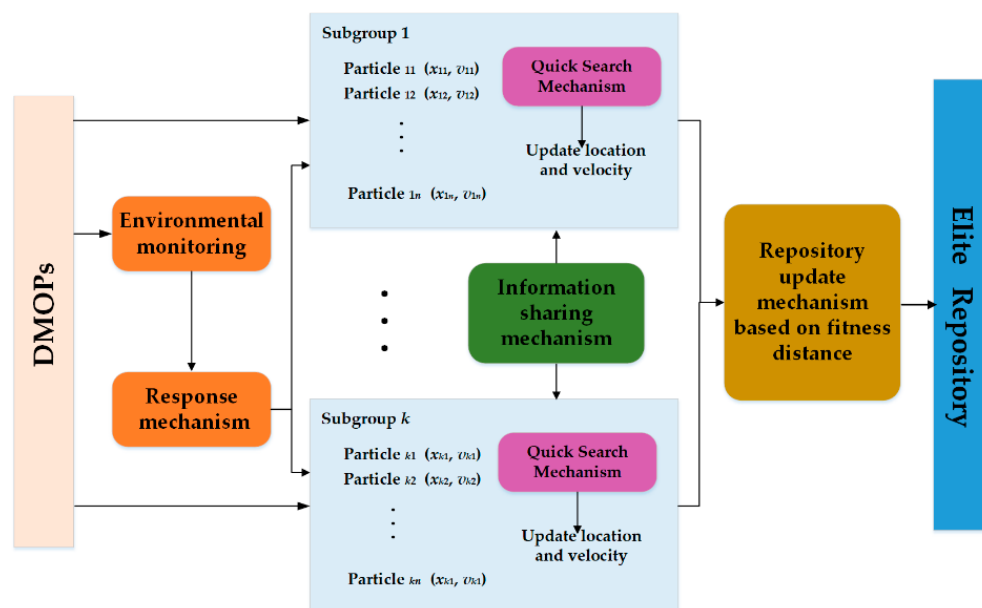


Figure 1. DVEPSO/FD system.

The core structures of the DVEPSO/FD algorithm can be summarized as follows: information sharing mechanism, environmental monitoring and response mechanism, quick search mechanism, and repository update mechanism. A swarm of particles are used to solve DMOPs, and it is divided into k subgroups. Each particle has its own location and velocity, which are updated in each iteration to obtain $pbest$ and $gbest$. The information sharing mechanism is adapted to change information between subgroups. The system monitors changes in the environment in real time. Once it changes, the response mechanism will pick out part of the particles to make corresponding adjustments, and the quick search mechanism will be activated at the first time. The repository is used to store Pareto solutions and it is updated based on fitness distance to obtain an elite repository. The $gbest$ is picked up from the elite repository to better guide the particles.

Because the quality of the repository update mechanism largely determines whether the algorithm can filter out the good nondominant solutions, which will affect the accuracy of the POF found by the algorithm, an innovation is first made in the repository update mechanism. The repository update mechanism based on fitness distance is generally based on the crowding distance, which may delete the better solution, and is not conducive to the diversity of MOPSO. While the fitness distance is defined and introduced in DVEPSO/FD to streamline the repository, calculate the average of the distance between the nondominant solutions with respect to each fitness function, use its average characteristics to streamline the repository more than once, and eliminate nondominant solutions with invalid or poor performance. It could be more effective at guiding the selection of the global optimal solution, thus achieving a better choice of nondominant solutions and maintaining a more efficient repository.

In addition, in order to improve the efficiency of the algorithm and quickly find the POF, this paper proposed the addition of a quick search mechanism, that is, dynamically adjusting the flight parameters before and after the first change of the environment. Compared with the DVEPSO method, the repository update mechanism based on fitness distance could improve the distribution of nondominant solutions, and the quick search mechanism could adjust dynamically to improve the search speed.

3.2. Module Design

3.2.1. Repository Update Mechanism Based on Fitness Distance

In the past, the repository update mechanism was usually based on crowding distance, and the nondominant solution beyond the repository size was eliminated directly by calculating the crowding distance, while the fitness distance was used in DVEPSO/FD to streamline the repository. The definitions of the crowding distance and fitness distance are as follows.

Definition 1 ([25]). Crowding distance is defined in Equation (4). Suppose there are a total of m sub-objectives, and the initial crowding distance of all individuals is $I(i)_d = 0$.

$$L_i = \sum_{k=1}^m (|f_{i+1,k} - f_{i-1,k}|) \quad (4)$$

where L_i is the crowding distance of particle i and m is the number of fitness functions, where the fitness function is the specific function model of the each objective optimization problem. $f_{i,k}$ is the k -th fitness function value of particle i .

Definition 2. The definition and equation of fitness distance is shown in Equation (5).

$$S(i)_{fd}^m = \frac{(S(i+1)_m - S(i-1)_m)}{f_m^{\max} - f_m^{\min}} \quad (5)$$

where $S(i)_{fd}^m$ is the fitness distance of the individual i relative to the objective m , and $S(i)_m$ is the fitness function value of the individual i relative to the objective m . It is important to point out that if there are M objectives, then each particle has and totally has M fitness distances.

Figure 2 shows a schematic diagram of the repository update mechanism based on fitness distance. The optimization problem could be divided into several independent fitness functions. For each fitness function, each particle can calculate a corresponding fitness distance. The fitness distance of the nondominant solution is calculated and compared with the set threshold α_F to determine whether the nondominant solution is retained or not. All of the fitness distances are used to streamline the repository independently. Compared to the crowding distance, the repository update mechanism based on the crowding distance is only streamlined once, but the one based on the fitness distance will be streamlined m times according to the number of fitness functions. The advantage of this design is that it can retain its optimal solution for each fitness function. Especially in the case of environmental changes, the response of each objective is different, so the changes of each fitness function are different. The repository update mechanism based on the crowding distance may miss the optimal solution, while the proposed method is more conducive to solve dynamic optimization problems.

After completing all the streamlining actions, the remaining nondominant solutions form the elite repository. The elite repository also has a certain capacity, so the number of nondominant solutions in the elite repository should also be limited, and the elimination rule is also based on crowding distance when out of range.

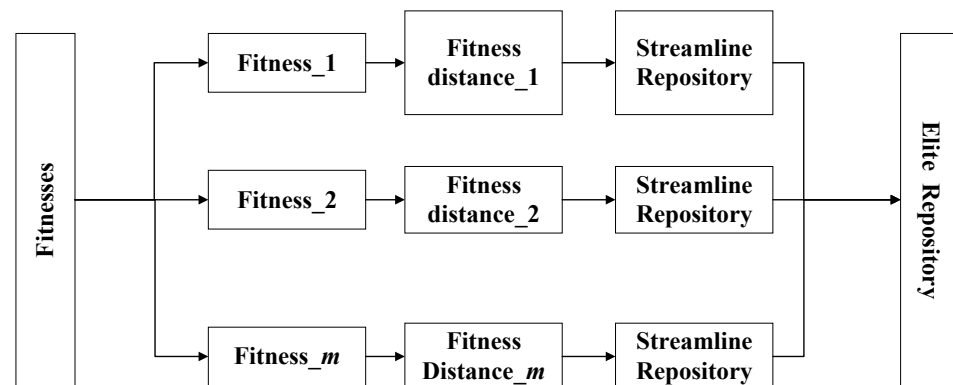


Figure 2. Schematic diagram of repository update mechanism based on fitness distance.

3.2.2. Quick Search Mechanism

The Pareto optimal frontier of the dynamic algorithm also changes according to the change in the objective function. The Pareto optimal frontier of the first generation is difficult to find. The Pareto optimal frontier of the offspring is changed on the basis of the previous generation, so it will be easier to find than the first generation.

In order to improve particles' ability of finding the initial POF quickly in the early stage, the quick search mechanism is proposed. Iterations before the first time environmental change can be named the Quickly Search stage (QS stage); therefore, the value of w , c_1 , and c_2 in the QS stage should be dynamically adjusted. After the QS stage, the next POFs are usually easily found on the basis of the existing data of the previous POF, and this stage can be named the Non-QS stage. The parameters do not need to be dynamically adjusted in the Non-QS stage, which does not contribute much to save the runtime.

For the adaptive PSO, the values of w and learning factor c_1 are required to be large in the early iteration to enhance the search ability of the local optimal value. When close to the POF, the search ability of this local optimal value should be reduced, so the later value is small. Conversely, the value of learning factor c_2 is required to change from small to large. In the later iteration, the influence of the global optimal value is enhanced to make more particles close to the POF. The specific adjustment equations of these three parameters in the QS stage are shown in (6)–(8).

$$w(r) = w_{\max} - \frac{(w_{\max} - w_{\min})}{(1 + \exp(5 - 0.07 * r))} \quad (6)$$

where $w_{\max} = 0.9$, $w_{\min} = 0.4$, and r is the current number of iterations.

$$c_1(r) = 2 - 1/(0.98 + \exp(10 - 0.1 * r)) \quad (7)$$

$$c_2(r) = 1 + 8/(8 + \exp(-13 + 1500/r)) \quad (8)$$

3.2.3. Other Structures

(1) Information sharing mechanism

Each particle has a local optimum and a global optimal to guide its search in the search space. The local optimum of a particle is its personal best, $pbest$, which is the best position where the particle is currently found. When there are no changes in the environment, the global guide of particles, $gbest$, is selected from one of the subgroups through a knowledge sharing mechanism. In this way, the $pbest$ of each population may become $gbest$, that is, the information of its own population is transmitted through $pbest$, and then other populations are affected by $gbest$, thereby achieving information sharing between the populations [21].

There are many mechanisms for information sharing, such as the circular loop strategy and roulette selection. Among them, the circular loop strategy is relatively simple. The selection method of subgroup s is shown in Equation (9).

$$s = \begin{cases} k & \text{for } j = 1 \\ j - 1 & \text{for } j = 2, \dots, k \end{cases} \quad (9)$$

(2) Environmental monitoring and response mechanism

After each iteration, randomly select some particles as sentinel particles, and evaluate the environment before the start of the next iteration. If the differences in the fitness values of the sentinel particles between current and previous are all greater than a certain value, the environment is considered changed. If a change in the environment is detected, the particles of the corresponding subgroup are reinitialized at a certain ratio, which is usually 30%.

3.3. The Pseudo-Code of the Algorithm

The repository update mechanism based on fitness distance possesses the elite repository to improve the distribution of nondominant solutions, and the quick search mechanism adjusts the flight parameters dynamically, so the DVEPSO/FD responds quickly when environmental changes are monitored. The pseudo-code of the DVEPSO/FD is shown in Table 1.

Table 1. The pseudo-code of the DVEPSO/FD method.

Pseudo-Code	
1.	Randomly initialize each particle (x_{ki}, v_{ki}) in the swarm and divide the swarm into K subgroups, set w_0 , c_{10} , c_{20} , and clear the repository Re
2.	For R from 1 to N , iterating Randomly select some particles, calculate the fitness $f_m^r(x_i)$ If $f_m^r(x_i) - f_m^{r-1}(x_i) < \alpha_T$ (a) Using $v_{iD}^{r+1} = wv_{iD}^r + c_1r_1(pbest_i^r - x_{iD}^r) + c_2r_2(gbest^r - x_{iD}^r)$, $x_{iD}^{r+1} = x_{iD}^r + v_{iD}^{r+1}$ to update the position and velocity (b) Determine $pbest_i^r$ by non-dominant relationship, $gbest^r$ from subgroup based on information sharing mechanism Else (a) Initiate response mechanism, randomly pick out 30% particles of corresponding subgroup, initialize the positions and velocities of these particles. (b) Initiate quick search mechanism, update the position and velocity of the rest particles using $w(r) = w_{max} - \frac{w_{max} - w_{min}}{1 + \exp(5 - 0.07 * r)}$, $c_1(r) = 2 - 1 / (0.98 + \exp(10 - 0.1r))$, $c_2(r) = 1 + 8 / (8 + \exp(-13 + 1500/r))$ instead of fixed parameters (c) Determine $pbest_i^r$ by non-dominant relationship, $gbest^r$ from elite repository End if Calculate all the fitness distances $S(i)_{fd}^m = \frac{S(i+1)_m - S(i-1)_m}{f_{max}^m - f_{min}^m}$, streamline the repository to form elite repository and update the Re If Re out of range Limited elite repository based on crowding distance End if End for

4. Experiments and Results

4.1. Standard Benchmarks

The changes based on the POF and Pareto optimal solutions (POs) [26] can be classified as four categories. The dynamic multi-objective optimization standard test problems are mainly FDA [27] (FDA5-iso [28]) series and nonlinear-related DMOP [29] series. The specific classification [8] and their correspondences with the test functions are shown in Table 2, and the formulas are defined in Table 3.

Table 2. Classification of standard benchmarks.

POF	POS	
	No Change	Change
No change	Type IV Problem changes	Type I FDA1; FDA4
Change	Type III FDA2; DMOP1	Type II FDA3; FDA5; DMOP2

Table 3. The formula definition of standard benchmarks.

Benchmarks	Definition	Benchmarks	Definition
FDA1	$f_1(X_I) = x_1$ $f_2(X_{II}) = g \cdot h$ $g(X_{II}) = 1 + \sum_{i=2}^m (x_i - G(t))^2$ $h(f_1, g) = 1 - \sqrt{\frac{f_1}{g}}$ $G(t) = \sin(0.5\pi \cdot t)$ $t = \frac{1}{n_t} \left\lfloor \frac{\tau}{\tau_i} \right\rfloor$ $where X_{II} = 9, X_I \in [0, 1], X_{II} \in [-1, 1]$	FDA2	$f_1(X_I) = x_1$ $f_2(X_{II}) = g \cdot h$ $g(X_{II}) = 1 + \sum_{i \in X_{II}} (x_i)^2$ $h(X_{III}, f_1, g) = 1 - \left(\frac{f_1}{g}\right)^{H(t) + \sum_{x_i \in X_{III}} (x_i - H(t))^2}$ $H(t) = 0.75 + 0.7 \cdot \sin(0.5\pi \cdot t)$ $t = \frac{1}{n_t} \left\lfloor \frac{\tau}{\tau_i} \right\rfloor$ $where X_{II} = X_{III} = 15, X_I \in [0, 1], X_{II}, X_{III} \in [0, 1]$
FDA3	$f_1(X_I) = \sum_{x_i \in X_I} x_i^{F(t)}$ $f_2(X_{II}) = g \cdot h$ $g(X_{II}) = 1 + G(t) + \sum_{x_i \in X_{II}} (x_i - G(t))^2$ $h(f_1, g) = 1 - \sqrt{\frac{f_1}{g}}$ $G(t) = \sin(0.5\pi \cdot t) $ $F(t) = 10^{2 \sin(0.5\pi \cdot t)}$ $t = \frac{1}{n_t} \left\lfloor \frac{\tau}{\tau_i} \right\rfloor$ $where X_I = 5, X_{II} = 25, X_I \in [0, 1], X_{II} \in [-1, 1]$	FDA 4	$f_1(X) = (1 + g(X_{II})) \prod_{i=1}^{M-1} \cos\left(\frac{x_i \pi}{2}\right)$ $f_k(X) = (1 + g(X_{II})) \prod_{i=1}^{M-1} \left(\cos\left(\frac{x_i \pi}{2}\right)\right) \sin\left(\frac{y_{M-k+1} \pi}{2}\right)$ $f_M(X) = (1 + g(X_{II})) \prod_{i=1}^{M-1} \sin\left(\frac{x_i \pi}{2}\right)$ $g(X_{II}) = \sum_{x_i \in X_{II}} (x_i - G(t))^2$ $G(t) = \sin(0.5\pi \cdot t) $ $t = \frac{1}{n_t} \left\lfloor \frac{\tau}{\tau_i} \right\rfloor$ $where X \in [0, 1]$
FDA5	$f_1(X) = (1 + g(X_{II})) \prod_{i=1}^{M-1} \cos\left(\frac{y_i \pi}{2}\right)$ $f_k(X) = (1 + g(X_{II})) \prod_{i=1}^{M-1} \left(\cos\left(\frac{y_i \pi}{2}\right)\right) \sin\left(\frac{y_{M-k+1} \pi}{2}\right)$ $f_M(X) = (1 + g(X_{II})) \prod_{i=1}^{M-1} \sin\left(\frac{y_i \pi}{2}\right)$ $g(X_{II}) = G(t) + \sum_{x_i \in X_{II}} (y_i - G(t))^2$ $y_i = x_i^{F(t)}$ $G(t) = \sin(0.5\pi \cdot t) $ $F(t) = 1 + 100 \sin^4(0.5\pi \cdot t)$ $t = \frac{1}{n_t} \left\lfloor \frac{\tau}{\tau_i} \right\rfloor$ $where X \in [0, 1]$	FDA5-iso	$f_1(X) = (1 + g(X_{II})) \prod_{i=1}^{M-1} \cos\left(\frac{y_i \pi}{2}\right)$ $f_k(X) = (1 + g(X_{II})) \prod_{i=1}^{M-1} \left(\cos\left(\frac{y_i \pi}{2}\right)\right) \sin\left(\frac{y_{M-k+1} \pi}{2}\right)$ $f_M(X) = (1 + g(X_{II})) \prod_{i=1}^{M-1} \sin\left(\frac{y_i \pi}{2}\right)$ $g(X_{II}) = \sum_{x_i \in X_{II}} (y_i - G(t))^2$ $y_i = x_i^{F(t)}$ $G(t) = \sin(0.5\pi \cdot t) $ $F(t) = 1 + 100 \sin^4(0.5\pi \cdot t)$ $t = \frac{1}{n_t} \left\lfloor \frac{\tau}{\tau_i} \right\rfloor$ $where X \in [0, 1], X_{II} = (x_M, \dots, x_n)$
DMOP1	$f_1(x_1) = x_1$ $f_2(x_2, \dots, x_m) = g \cdot h$ $g(x_2, \dots, x_m) = 1 + 9 \cdot \sum_{i=2}^m x_i^2$ $h(f_1, g) = 1 - \left(\frac{f_1}{g}\right)^{H(t)}$ $H(t) = 0.75 \cdot \sin(0.5\pi \cdot t) + 1.25$ $where m = 10, x_i \in [0, 1]$	DMOP2	$f_1(x_1) = x_1$ $f_2(x_2, \dots, x_m) = g \cdot h$ $g(x_2, \dots, x_m) = 1 + \sum_{i=2}^m (x_i - G(t))^2$ $h(f_1, g) = 1 - \left(\frac{f_1}{g}\right)^{H(t)}$ $H(t) = 0.75 \cdot \sin(0.5\pi \cdot t) + 1.25$ $G(t) = \sin(0.5\pi \cdot t)$ $where m = 10, x_i \in [0, 1]$

4.2. Performance Metrics

Weicker proposed that it is necessary to consider the metrics he describes when analyzing and comparing algorithms for dynamic problems. They are accuracy and stability [30].

(1) Accuracy

The accuracy measures how close the best solution is found to the actual one. It usually takes a value between 0 and 1, where 1 is the best precision value. The specific definition is shown in Equation (10).

$$accuracy_{F,EA}(t) = \frac{F(best_{EA}(t)) - \min_F(t)}{\max_F(t) - \min_F(t)} \quad (10)$$

where $best_{EA}(t)$ is the best solution found in the population at time t , the maximum and minimum fitness values in the search space are represented by $\max_F(t)$ and $\min_F(t)$, and F is the fitness function of the objective problem.

(2) Stability

The stability measures the stability of dynamic algorithms, which would be called stable algorithms when they do not seriously affect the accuracy of optimization. Stability is an important issue for optimization in dynamic environments. Its value ranges from 0 to 1, where a value close to 0 indicates higher stability. The definition of stability is as in Equation (11).

$$stab_{F,EA}(t) = \max\{0, accuracy_{F,EA}(t) - accuracy_{F,EA}(t-1)\} \quad (11)$$

4.3. Parameters' Settings

There are six subgroups and each swarm has 50 particles. The size of the repository is set to 100, and 30% of the particles are reinitialized after an environmental change is detected. In addition, other parameters are shown in Table 4, where τ_t is the environment change frequency.

Table 4. Parameters' settings.

Parameters	n_t	τ_t	w_0	c_{10}	c_{20}	R
Values	15 (FDA2: 2.5)	100	0.72 (Non-QS stage)	1.49 (Non-QS stage)	1.49 (Non-QS stage)	1000

4.4. Experiments

The experiments are carried out on the standard benchmarks that have been mentioned in Table 3 separately. The specific results of each test function are shown in Figure 3a–n. There are two pictures shown in each test function. The first one is the Pareto front found by the basic algorithm (DVEPSO), and the second one is the Pareto front found by the proposed algorithm (DEVPSO/FD). Different colors in the pictures represent different changed POFs.

Table 5 records the performance metrics of the experiments above, which are accuracy, stability, and their respective runtimes of DVEPSO and DVEPSO/FD. In Table 5, Mean is the mean, Std is the standard deviation, and Best is the best value.

Table 5. The accuracy and stability metrics on benchmarks.

Benchmarks		Accuracy		Stability		Runtime	
		DVEPSO	DVEPSO/FD	DVEPSO	DVEPSO/FD	DVEPSO	DVEPSO/FD
FDA1	Mean	0.4292	0.4236	0.0223	0.0209	111.1841	222.6885
	Std	0.0015	0.0004	0.0012	0.0007		
	Best	0.4308	0.4239	0.0237	0.0213		
FDA2	Mean	0.5621	0.5712	0.0399	0.0323	139.2537	141.3722
	Std	0.0088	0.0065	0.0010	0.0021		
	Best	0.5722	0.5741	0.0411	0.0338		
FDA3	Mean	0.6846	0.6849	0.0412	0.0290	119.9075	125.3159
	Std	0.0012	0.0043	0.0054	0.0009		
	Best	0.6859	0.6916	0.0472	0.0297		
FDA4	Mean	0.2423	0.2482	0.0284	0.0297	161.0242	675.8527
	Std	0.0012	0.0023	0.0008	0.0013		
	Best	0.2432	0.2499	0.0293	0.0310		
FDA5	Mean	0.2269	0.2342	0.0256	0.0249	168.8640	492.0685
	Std	0.0015	0.0023	0.0010	0.0003		
	Best	0.2284	0.2368	0.0263	0.0251		
DMOP1	Mean	0.6194	0.6228	0.0204	0.0165	159.8208	1213.6000
	Std	0.0054	0.0030	0.0020	0.0046		
	Best	0.6255	0.6296	0.0227	0.0218		
DMOP2	Mean	0.5680	0.5691	0.0136	0.0139	118.3666	163.2108
	Std	0.0005	0.0011	0.0003	0.0006		
	Best	0.5683	0.5694	0.0139	0.0146		

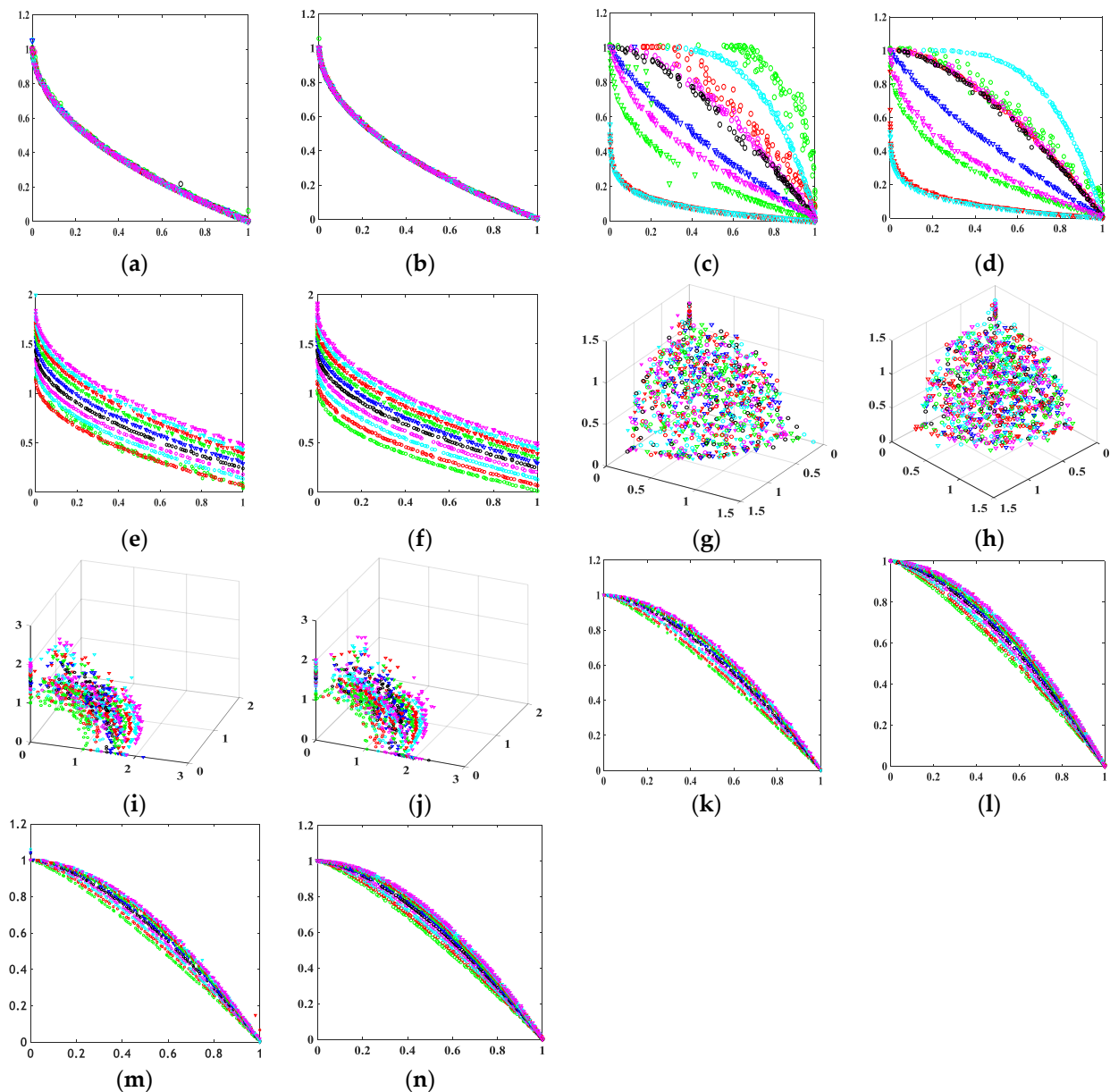


Figure 3. Simulation results on benchmarks: (a) POF of DVEPSO on FDA1, (b) POF of DEVPSO/FD on FDA1, (c) POF of DVEPSO on FDA2, (d) POF of DEVPSO/FD on FDA2, (e) POF of DVEPSO on FDA3, (f) POF of DEVPSO/FD on FDA3, (g) POF of DVEPSO on FDA4, (h) POF of DEVPSO/FD on FDA4, (i) POF of DVEPSO on FDA5, (j) POF of DEVPSO/FD on FDA5, (k) POF of DVEPSO on DMOP1, (l) POF of DEVPSO/FD on DMOP1, (m) POF of DVEPSO on DMOP2, and (n) POF of DEVPSO/FD on DMOP2.

4.5. Results

From the POF figures of FDA1~FDA5 and DMOP1~DMOP2, it can be seen that the algorithm that used the fitness distance to streamline the repository, that is, DVEPSO/FD, has clearer POFs and is more accurate than DVEPSO. FD2, FD3, FD5, DMOP1, and DMOP1 are all classified as the POF changed. When the POF is dynamically changing, it can be seen from the figures that the lines between those changed POFs are also clearer, which indicates that DVEPSO/FD can find better solutions. In particular, the POF of the DVEPSO/FD can converge faster in the early stage of the optimization objective environment change, which indicates that the quick search mechanism proposed by the algorithm has also played an important role.

Accuracy represents the precision of the solutions, and the closer its value to 1, the higher the accuracy. It can be seen from Table 5 that the accuracy of FDA4 with the POS changed obtained by DVEPSO/FD is 2.43% higher than that of DVEPSO; the accuracy of FDA3, FDA5, and DMOP2 with both the POF and POS changed can be improved by 3.22%; and the accuracy of FDA2 and DMOP1 with the POF changed can be improved by 1.62%, while the accuracy of FDA1 obtained by DVEPSO/FD is lower than that of DVEPSO. As FDA1 has two objective functions and no change in the POF, DVEPSO/FD does not show its advantages. Although the accuracy data of FDA1 are slightly lower, a more accurate POF and accurate solutions can still be found by DVEPSO/FD. On the other hand, DVEPSO/FD shows good performance on the complex problems, especially when the POF and POS both change on three objectives.

Stability represents the solid state of the algorithm, and the closer its value to 0, the better the stability. As can be seen from Table 5, the stability values of DVEPSO are all between 0.01 and 0.05, which are still very close to 0. It indicates that the stability is very good under such a condition. Similarly, the stability values of DVEPSO/FD are all between 0.01 and 0.04, which are even better than DVEPSO, improved at most by 19.12%. The results of the stability index show that DVEPSO/FD does not destabilize the algorithm and maintain the original high stability.

As for the runtime, due to the dynamic adjustment of the flight parameters of the particles, the particles can quickly find the POF of the first generation, thereby saving a certain time. However, as the repository update mechanism using the fitness distance has more steps than the crowding distance condition, the running time of the overall algorithm of DVEPSO/FD is longer than that of DVEPSO. It can be seen from Table 5 that, for dual objective functions and slightly simpler nonlinear problems, such as FDA1, FDA2, FDA3, and DMOP1, the runtimes of DVEPSO/FD do not increase by much, while for three objective functions and complex nonlinear problems, such as FDA4, FDA5, and DMOP2, the runtimes increase by a factor of three or more. In addition to extremely complex problems, the test functions selected in this paper can basically represent the complexity of dynamic multi-objective optimization problems. Therefore, in order to obtain a better POF, although the running time is increased, it is still within an acceptable range.

In summary, DVEPSO/FD achieves a higher accuracy and stability with the POF dynamically changing, which shows a good dynamic change adaptability and solving the set ability of the dynamic multi-objective optimization problem.

5. Conclusions

In this paper, a quick search dynamic vector-evaluated particle swarm optimization algorithm based on fitness distance (DVEPSO/FD) is proposed. Taking the DVEPSO method as a foundation, the repository update mechanism using the fitness distance and quick search mechanism are designed aiming for a good dynamic change adaptability and solving set ability of the dynamic multi-objective optimization problem. The fitness distance is used to streamline the repository to achieve better optimal solutions, and the flight parameters of the particles are adjusted dynamically to improve the search speed. Both from the figures generated by the experiments of test functions and the values of performance indexes, the DVEPSO/FD has a better accuracy of the POF than the basic DVEPSO, obtains a better POS, and maintains the same strong stability.

From the perspective of the overall algorithm structure, the update mechanism of the repository has a great influence on whether the optimal value could be found closer to the real POF or not. DVEPSO/FD verifies this more clearly by improving this mechanism. Of course, other structures of the algorithm are also important, such as the environmental monitoring and response mechanism and information sharing mechanism. They will influence whether the algorithm can respond quickly and accurately to changes in the environment and correctly grasp the general direction of all population evolutions. This is also the direction of the author's future research.

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

Funding: This research was supported by the National Natural Science Foundation of China with grant No. 62003350, and the Fundamental Research Funds for the Central Universities of China with grant No.2022YQJD16.

Data Availability Statement: The data presented in this study are available upon reasonable request from the corresponding author.

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

References

1. Coello, A.C.; Lechuga, M.S. MOPSO: A Proposal for Multiple Objective Particle Swarm. In Proceedings of the 2002 Congress on Evolutionary Computation, Honolulu, HI, USA, 12–17 May 2002; Volume 2, pp. 1051–1056.
2. Coello, C.A.C.; Pulido, G.T.; Lechuga, M.S. Handling Multiple Objectives with Particle Swarm Optimization. *IEEE Trans. Evol. Comput.* **2004**, *8*, 256–279. [\[CrossRef\]](#)
3. Yao, S.; Dong, Z.; Wang, X.; Ren, L. A Multiobjective multifactorial optimization algorithm based on decomposition and dynamic resource allocation strategy. *Inf. Sci.* **2020**, *511*, 18–35. [\[CrossRef\]](#)
4. Helbig, M.; Engelbrecht, A.P. Population-based metaheuristics for continuous boundary-constrained dynamic multi-objective optimisation problems. *Swarm Evol. Comput.* **2014**, *14*, 31–47. [\[CrossRef\]](#)
5. Jiang, M.; Member, S.; Huang, Z.; Qiu, L.; Huang, W.; Yen, G.G. Transfer Learning-Based Dynamic Multiobjective Optimization Algorithms. *IEEE Trans. Evol. Comput.* **2018**, *22*, 501–514. [\[CrossRef\]](#)
6. Helbig, M.; Engelbrecht, A. Dynamic Vector-evaluated PSO with Guaranteed Convergence in the Sub-swarms. In Proceedings of the 2015 IEEE Symposium Series on Computational Intelligence, Cape Town, South Africa, 7–10 December 2015; pp. 1286–1293.
7. Helbig, M.; Engelbrecht, A.P. Using Headless Chicken Crossover for Local Guide Selection When Solving Dynamic Multiobjective Optimization. In *Advances in Nature and Biologically Inspired Computing*; Springer: Cham, Switzerland, 2016; p. 419.
8. Ou, J.; Zheng, J.; Ruan, G.; Hu, Y.; Zou, J.; Li, M. A pareto-based evolutionary algorithm using decomposition and truncation for dynamic multi-objective optimization. *Appl. Soft Comput. J.* **2019**, *85*, 105673. [\[CrossRef\]](#)
9. Goldberg, D.E.; Smith, R.E. Nonstationary function optimization using genetic algorithm with dominance and diploidy. In Proceedings of the Second International Conference on Genetic Algorithms and Their Application, Cambridge, MA, USA, 28–31 July 1987; Grefenstette, J.J., Ed.; Lawrence Erlbaum Associates Inc.: Mahwah, NJ, USA, 1987; pp. 59–68.
10. Ortega, J.; Toro, F.; De Mario, C. A single front genetic algorithm for parallel multi-objective optimization in dynamic environments. *Neurocomputing* **2009**, *72*, 3570–3579.
11. Guntsch, M.; Middendorf, M.; Schmeck, H. An ant colony optimization approach to dynamic TSP. In Proceedings of the Genetic and Evolutionary Computation Conference, San Francisco, CA, USA, 7–11 July 2001; Morgan Kaufmann: Burlington, MA, USA, 2001; pp. 860–867.
12. Trojanowski, K. Immune-based algorithms for dynamic optimization. *Inf. Sci. J.* **2009**, *179*, 1495–1515. [\[CrossRef\]](#)
13. Zhang, Z. Multiobjective optimization immune algorithm in dynamic environments and its application to greenhouse control. *Appl. Soft Comput.* **2008**, *8*, 959–971. [\[CrossRef\]](#)
14. Pelta, D.; Cruz, C.; Verdegay, J.L. Simple control rules in a cooperative system for dynamic optimisation problems. *Int. J. Gen. Syst.* **2009**, *38*, 701–717. [\[CrossRef\]](#)
15. Urade, H.S.; Patel, R. Dynamic Particle Swarm Optimization to Solve Multi-objective Optimization Problem. *Procedia Technol.* **2012**, *6*, 283–290. [\[CrossRef\]](#)
16. Optimization, D.M.; Zhou, A.; Jin, Y.; Member, S.; Zhang, Q.; Member, S. A Population Prediction Strategy for Evolutionary. *IEEE Trans. Cybern.* **2014**, *44*, 40–53.
17. Azzouz, R.; Bechikh, S.; Ben, L. A dynamic multi-objective evolutionary algorithm using a change severity-based adaptive population management strategy. *Soft Comput.* **2017**, *21*, 885–906. [\[CrossRef\]](#)
18. Deb, K.; Rao, N.U.B.; Karthik, S. Dynamic multi-objective optimization and decision-making using modied NSGA-II: A case study on hydro-thermal power scheduling. In Proceedings of the International Conference on Evolutionary Multi-criterion Optimization, Matsushima, Japan, 5–8 March 2007; pp. 803–817.
19. Branke, J.; Blackwell, T. Multiswarms, exclusion, and anti-convergence in dynamic environments. *IEEE Trans. Evol. Comput.* **2006**, *10*, 459–472.
20. Janson, S.; Middendorf, M. A hierarchical particle swarm optimizer for noisy and dynamic environments. *Genet. Program. Evolvable Mach.* **2006**, *7*, 329–354. [\[CrossRef\]](#)

21. Greeff, M.; Engelbrecht, A.P. Solving dynamic multi-objective problems with vector evaluated particle swarm optimisation. In Proceedings of the 2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence), Hong Kong, China, 1–6 June 2008; pp. 2917–2924.
22. Parsopoulos, K.E.; Vrahatis, M.N. Recent approaches to global optimization problems through Particle Swarm Optimization. *Nat. Comput.* **2002**, *1*, 235–306. [[CrossRef](#)]
23. Helbig, M.; Engelbrecht, A.P. Analyses of Guide Update Approaches for Vector Evaluated Particle Swarm Optimisation on Dynamic Multi-Objective Optimisation Problems. In Proceedings of the 2012 IEEE Congress on Evolutionary Computation, Brisbane, QLD, Australia, 10–15 June 2012.
24. Schaffer, J. Multiple objective optimization with vector evaluated genetic algorithms. In Proceedings of the 1st International Conference on Genetic Algorithms, Sheffield, UK, 12–14 September 1985; pp. 93–100.
25. Peng, G.; Fang, Y.W.; Peng, W.S.; Chai, D.; Xu, Y. Multi-objective particle optimization algorithm based on sharing-learning and dynamic crowding distance. *Optik* **2016**, *127*, 5013–5020. [[CrossRef](#)]
26. Saremi, S.; Mirjalili, S.; Lewis, A.; Liew, A.W.C.; Dong, J.S. Enhanced multi-objective particle swarm optimisation for estimating hand postures. *Knowl.-Based Syst.* **2018**, *158*, 175–195. [[CrossRef](#)]
27. Farina, M.; Deb, K.; Amato, P. Dynamic Multiobjective Optimization Problems: Test Cases, Approximations, and Applications. *IEEE Trans. Evol. Comput.* **2004**, *8*, 425–442. [[CrossRef](#)]
28. Helbig, M.; Engelbrecht, A.P. Benchmarks for Dynamic Multi-objective Optimisation. In Proceedings of the 2013 IEEE Symposium on Computational Intelligence in Dynamic and Uncertain Environments (CIDUE), Singapore, 16–19 April 2013; pp. 84–91.
29. Goh, C.; Tan, K.C. A Competitive-Cooperative Coevolutionary Paradigm for Dynamic Multiobjective Optimization. *IEEE Trans. Evol. Comput.* **2009**, *13*, 103–127.
30. Weicker, K. Performance Measures for Dynamic Environments. In *International Conference on Parallel Problem Solving from Nature*; Springer: Berlin/Heidelberg, Germany, 2002; pp. 64–76.