Hindawi
Mathematical Problems in Engineering
Volume 2020, Article ID 8598543, 7 pages
https://doi.org/10.1155/2020/8598543

Research Article

A Genetic Optimization Algorithm Based on Adaptive Dimensionality Reduction

Tai Kuang,[1] **Zhongyi Hu**,[2] **and Minghai Xu**[2]

[1]*Department of Information Engineering, Zhejiang College of Security Technology, Wenzhou 325016, China*
[2]*Institute of Big Data and Information Technology, Wenzhou University, Wenzhou 325000, China*

Correspondence should be addressed to Tai Kuang; kuangtaikt@qq.com

Received 19 March 2020; Revised 17 April 2020; Accepted 20 April 2020; Published 11 May 2020

Guest Editor: Weicun Zhang

With the rise of big data in cloud computing, many optimization problems have gradually developed into high-dimensional large-scale optimization problems. In order to address the problem of dimensionality in optimization for genetic algorithms, an adaptive dimensionality reduction genetic optimization algorithm (ADRGA) is proposed. An adaptive vector angle factor is introduced in the algorithm. When the angle of an individual's adjacent dimension is less than the angle factor, the value of the smaller dimension is marked as 0. Then, the angle between each individual dimension is calculated separately, and the number of zeros in the population is updated. When the number of zeros of all individuals in a population exceeds a given constant in a certain dimension, the dimension is considered to have no more information and deleted. Eight high-dimensional test functions are used to verify the proposed adaptive dimensionality reduction genetic optimization algorithm. The experimental results show that the convergence, accuracy, and speed of the proposed algorithm are better than those of the standard genetic algorithm (GA), the hybrid genetic and simulated annealing algorithm (HGSA), and the adaptive genetic algorithm (AGA).

1. Introduction

High-dimensional optimization problems have become increasingly prevalent in many fields, for example, radar waveform optimization [1, 2] and water quality monitoring [3]. Recently, major breakthroughs have been made with respect to solving such problems. Tuo et al. [4] proposed a global optimization algorithm that used the membrane calculation principle to solve high-dimensional functions in 2011. Their algorithm, by implementing high-dimensional segmentation, achieved the segmentation of a high-dimensional space into a low-dimensional one, thereby improving the performance. Also addressing the problem of high-dimensional space processing, Chen et al. [5] described an approach that established a sparse regression model by utilizing the geometric structural features of the approximate solution set, mapping the high-dimensional target space into a low-dimensional one. In 2015, Chen et al. [6] designed a congestion control strategy based on the concept of open angles and compared it against the indication-based evolutionary algorithm (IBEA) [7], NSGA III (non-dominated sorting genetic algorithm III) [8], and the grid-based evolutionary algorithm (GrEA) [9]. The results indicated a significant improvement by the proposed algorithm. In the same year, Zheng et al. [10] proposed a high-dimensional multiobjective evolutionary algorithm based on information separation. Although this algorithm decomposed the high-dimensional space into a low-dimensional one, it did not remove the excess dimensions. In order to maintain a balance between the convergence and distributed features of the objective evolutionary algorithms, Bi and Wang [11] proposed a high-dimensional objective evolutionary algorithm based on an angle penalty. In 2018, He et al. [12] combined a dimensionality reduction and differential evolution algorithm to solve the knapsack problem. The experimental results showed that this algorithm achieved good accuracy and stability and is suited for solving large-scale problems. Liang et al. [13] first conducted an analysis of the features of large-scale high-dimensional problems and developed a coevolutionary dynamic particle

swarm optimization algorithm, dividing the whole swarm into different groups. Unlike the improvement targeting the number of dimensions, Xia et al. [14] proposed a fitness-based multiplayer swarm optimization algorithm (FMPSO), with the addition of a new component into the particle velocity update rules, called the "subspace learning" component. During the evolutionary process, two adjustment operators were introduced to adjust the roles and number of objective dimensions of the particles. Xu et al. [15] proposed a multidimensional learning strategy based on the experience of the best individuals, which was used to discover and integrate valuable information from the best swarm solution. In their experimentation, 16 classical benchmark functions, 30 CEC 2014 testing functions, and one actual optimization problem were used. This algorithm demonstrated higher convergence, accuracy, and speed. In 2015, Wu et al. [16] described a nondominated sorting genetic algorithm utilizing fractal dimensions and an elitist strategy for feature selection. This algorithm could successfully reduce the number of dimensions for the objective problems. This algorithm, however, only improved the classification accuracy as compared with the standard genetic algorithms to a limited extent. He and Yen [17] used the minimum included angle between two vectors to look for similar individuals, among which the individuals with poor convergence were deleted. However, it was still difficult to set the difference threshold. Xiang et al. [18] implemented priority selection based on the maximum included angle between two vectors, thereby ensuring the diversity of the swarm. However, a good strategy for determining the angle threshold was still not provided. Furthermore, the U-model methodology, which is a generic systematic approach to convert a nonlinear polynomial model into a controller output $u(t-1)$ -based time-varying polynomial model, has been studied for facilitating a nonlinear control system design over the last decade [19–22]. Given the aforementioned, we define the concept of an adaptive vector angle factor, which changes with the swarm size and number of dimensions. In this paper, the adaptive vector angle factor is used for realizing dimensionality reduction with a GA. At the same time, it also provides a reference for U-model approaches in algorithm implementation.

2. The Proposed ADRGA Algorithm

2.1. Vector Angle Factor. Dimensionality reduction is a good approach for solving large-scale problems. However, most dimensionality reduction algorithms are random processes; those dimensions carrying important information or those crucial for objective problem solving may be deleted. Consider a 2D vector coordinate (a, b) in Figure 1. When the included angle θ between this vector and the x-axis (y-axis) is infinitely small, the vector coordinate is approximated to $(a, 0)$ or $(0, b)$. In this way, the 2D vector is approximately mapped into a one-dimensional one, as illustrated in Figure 2. Similarly, a 3D vector can be approximately mapped into a 2D or one-dimensional one, depending on the set threshold for the included angle between the two vectors.

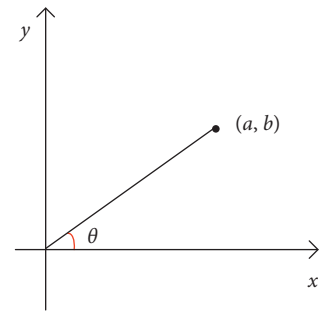

Figure 1: 2D vector representation.

Apparently, setting an appropriate threshold for the included angle between the two vectors is very important.

The dimension describes the precision of the required solution target characteristics. For example, when describing a pen in one dimension, only its length or width can be described, and so it is impossible to perform dimensionality reduction through mapping. In this case, the threshold for the included angle θ between the two vectors must be close to 0. When a pen has two dimensions, both its length and width can be described. At this point, if the included angle between the two vectors is smaller than a certain threshold, the dimension where the smaller values are located can be neglected. In this case, the threshold for the included angle θ between the two vectors may be close to a value above 0. The threshold range can be set according to an individual problem. When three dimensions are used for the description, not only the pen's length and width but also its cross-sectional radius can be described. Therefore, the threshold for the included angle θ between the two vectors is larger than that for the 2D situation. As the number of dimensions increases, the threshold range of the included angle θ between the two vectors increases as well. Experiments have shown that the threshold range cannot be infinitely large. Whatever the number of dimensions, the threshold must be maintained within the interval $(0, \pi/4)$. In this paper, the vector included angle factor is represented by the *sigma* function in equation (1), which is a monotonically increasing function and also a bounded function:

$$sigma(x) = \frac{1}{1 + e^{-x}}, \tag{1}$$

where $x \in (-\infty, +\infty)$, $sigma \in (0, 1)$.

The vector included angle factor proposed in this study must be within the interval $(0, \pi/4)$. An adaptive vector angle factor can be obtained by transformation of the *sigma* function, as shown in equation (2), where a is the control parameter, $a = 4.3926$, b is the adjustment parameter, $b = 3.6072$, and D is the number of dimensions. When the tangent of the included angle between the two vectors in adjacent dimensions $\tan\theta < \tan sigma_*$, the dimension where the smaller value is located is denoted as 0 ($\theta = x_{mi}/x_{mj}$, x_{mi} is the $i^{\text{-th}}$ dimension of the $m^{\text{-th}}$ individual; x_{mj} is the $j^{\text{-th}}$ dimension of the $m^{\text{-th}}$ individual).

$$sigma_*(D) = a * \frac{1}{1 + e^{-D}} - b. \tag{2}$$

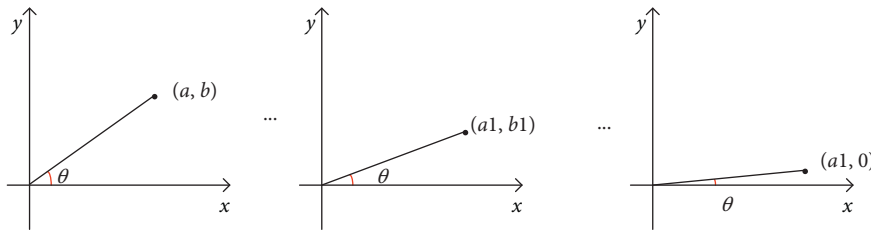

Figure 2: Changes of the included angle between the two vectors.

2.2. ADRGA Workflow. ADRGA targets high-dimensional swarms. First, the tangent value of the included angle between the two vectors in adjacent dimensions is calculated for each individual and is compared against the adaptive vector angle factor $sigma_*$. If it is smaller than the latter, the dimension where the smaller value of the individual is located is denoted as 0. Then, using the same method, the tangent values of the included angle between the two vectors in adjacent dimensions for all individuals are compared against $sigma_*$. Finally, the number of 0 elements in each dimension in the updated swarm pop is determined; if it is above the critical value Q, then this dimension is deleted.

Adaptive dimensionality reduction is feasible following the principle mentioned above. The pseudocode for ADRGA is shown in Algorithm 1 (P_c is the probability of crossover, P_m is the probability of mutation, N is the swarm size, G is the number of generations upon termination of evolution, T is the number of tests, Q is the critical value).

3. Experimental Results

Simulation was carried out using MATLAB 2014b under Windows 7 (also applicable to a higher version). A comparison was made between ADRGA, the standard GA [1], the hybrid genetic and simulated annealing algorithm (HGSA) [23], and the adaptive genetic algorithm (AGA) [24].

In this study, 8 standard composite testing functions were used, namely, F_1~F_8, as shown in Table 1. F_1~F_3 are high-dimensional unimodal functions, which only have one global best solution; F_4~F_8 are high-dimensional multimodal functions, which have several local best solutions, but only one global best solution. The latter are generally hard to optimize. The global best solutions of all testing functions were 0. Since some testing functions have several optimal solutions, the algorithm is very likely to be trapped in a local optimum. This aspect may be challenging for the proposed ADRGA algorithm.

In the experimentation, the value of the fitness function f was the function value at the current position. The parameters of the GA were configured as follows: crossover probability $P_c = 0.70$, mutation probability $P_m = 0.05$, number of iterations FEs = 1000, swarm size $sizepop = 50$, number of dimensions $D = 1000$, and the critical value $Q = 1$.

Each of the four algorithms was run 20 times on the 8 testing functions, and the means and standard deviations are shown in Table 2. Table 3 presents the improvement percentages of the optimal solutions using ADRGA as compared with the other three algorithms, calculated as follows: (mean of other algorithm−mean of ADRGA)/(mean of other algorithm). When optimizing these 8 classical testing functions, ADRGA could obtain a solution closer to the global best solution compared to the other algorithms, as shown in Table 2. This was especially true for the testing functions F_3, F_4, and F_6. But on the other testing functions, although a better solution was obtained using ADRGA, it failed to find the global optimum and was still far from achieving this goal. As shown in Table 3, the higher the improvement percentage, the greater the amplitude of performance improvement of ADRGA would be, and vice versa. Although a solution closer to the global optimum was obtained for F_4 in Table 2, its improvement percentage was smaller than for the other 7 functions, indicating that F_4 was less influenced by the number of dimensions.

The number of dimensions after dimensionality reduction and the average number of dimensions after running each algorithm 20 times are shown in Table 4. The number of residual dimensions varied after each run, indicating that the dimensionality reduction with the proposed algorithm changed with the initialized swarm in each run. Therefore, valuable information could be preserved when reducing the number of dimensions. Under the similar minimum included angle between the two vectors, the larger the included angle for the newly generated swarm, the smaller the number of dimensions deleted would be, and vice versa. As shown in Table 3, the maximum number of dimensions deleted was 40%, and the minimum was 34%. The maximum number of dimensions deleted was not above 50%, but it was ensured that the dimensions were not deleted randomly. Table 4 provides a better proof of the adaptive features of the proposed algorithm.

Figure 3 shows the comparison of convergence for the eight testing functions F_1~F_8. The x-axis is the number of fitness calculations, and the y-axis is the mean fitness value. As shown in Figure 3, ADRGA had a faster convergence speed and accuracy on the functions F_1, F_2, F_5, F_6, and F_7, though it was more likely to be trapped in a local optimum. For the testing function F_3, the convergence curves of ADRGA and GA nearly coincided. This was because the value range of the independent variable in function F_3 was [−1, 1]. Therefore, when the numbers of dimensions were 600 and 1000, the difference was negligibly small. This also demonstrated that the impact on F_3 was small if the number of dimensions did not change significantly. For the multimodal function F_4, the convergence curve of ADRGA not only had a faster convergence speed but also showed a more

Step 1: initialize parameters P_c, P_m, N, G, T, and Q and randomly generate the first swarm pop
Step 2: for $i < popsize$
Step 3: calculate the tangent value $\tan(x_{in}/x_{in+1})$ of the included angle between the two vectors in adjacent dimensions for each individual $pop(i)$
Step 4: if $\tan(x_{in}/x_{in+1}) < 4.3926(1/(1+e^{-D})) - 3.6072$, then update the value of the $n^{\text{-th}}$ dimension of the $i^{\text{-th}}$ individual to 0; otherwise, do not update the value
Step 5: return to step 2
Step 6: calculate the number of 0 elements in each dimension for the updated swarm pop; if it is above the critical value Q, then delete this dimension
Step 7: obtain the updated swarm pop
Step 8: calculate the fitness value $F(i)$ of each individual in the swarm pop
Step 9: initialize the new swarm $newpop$
Step 10: select two individuals from the swarm pop according to the fitness using the proportional selection algorithm
Step 11: if $random(0,1) < P_c$, then move on to step 12; otherwise, implement step 13
Step 12: apply the crossover operator according to the crossover probability P_c on the two individuals
Step 13: if $random(0,1) < P_m$, then move on to step 14
Step 14: apply the mutation operator according to the mutation probability P_m on the two individuals
Step 15: add the two new individuals into the swarm $newpop$
Step 16: repeat this process until the $N^{\text{-th}}$ generation is produced; otherwise, return to step 4
Step 17: replace pop with $newpop$
Step 18: repeat this process until the number of generations exceeds G; otherwise, return to step 8
Step 19: end

Algorithm 1: ADRGA.

Table 1: Benchmark testing functions.

Testing function	Number of dimensions	Feasible solution space
$F_1 = \sum_{i=1}^{D} x_i^2$	1000	[−100, 100]
$F_2 = (x_1 - 1)^2 + \sum_{i=1}^{D} i * (2x_i^2 - x_{i-1})^2$	1000	[−10, 10]
$F_3 = \sum_{i=1}^{D} \lvert x_i \rvert^{i+1}$	1000	[−1, 1]
$F_4 = -a * \exp(-b\sqrt{(1/D)\sum_{i=1}^{D} x_i^2}) - \exp((1/D)\sum_{i=1}^{D} \cos(cx_i)) + a + \exp(1)$, $a = 20, b = 0.2, c = 2\pi$	1000	[−32.768, 32.768]
$F_5 = \sum_{i=1}^{D} (x_i^2 - 10 * \cos(2*\pi* x_i) + 10)$	1000	[−5.12, 5.12]
$F_6 = \sum_{i=1}^{D} (x_i^2/4000) - \prod_{i=1}^{D} \cos(x_i/\sqrt{i}) + 1$	1000	[−600, 600]
$F_7 = \sin^2(\pi\omega_1) + \sum_{i=1}^{D-1} (\omega_i - 1)^2 [1 + 10\sin^2(\pi\omega_i + 1)] + (\omega_D - 1)^2 [1 + \sin^2(2\pi\omega_D)]$ $\omega_i = 1 + ((x_i - 1)/4)$	1000	[−10, 10]
$F_8 = 418.9829\, D - \sum_{i=1}^{D} x_i \sin(\sqrt{\lvert x_i \rvert})$	1000	[−500, 500]

Table 2: Optimization results of the different algorithms on 8 standard composite testing functions.

Function	Mean and standard deviation			
	GA	HGSA	AGA	ADRGA
F_1	$1.78e+05 \pm 6.12e+03$	$8.44e+05 \pm 2.02e+05$	$1.30e+06 \pm 2.07e+05$	$9.46e+04 \pm 2.65e+03$
F_2	$3.65e+07 \pm 1.85e+06$	$4.08e+08 \pm 1.10e+08$	$8.04e+08 \pm 2.36e+08$	$9.97e+06 \pm 8.74e+05$
F_3	$1.11e-07 \pm 1.84e-07$	0.1793 ± 0.0885	$3.95e-06 \pm 5.84e-06$	$7.22e-08 \pm 1.77e-07$
F_4	13.2492 ± 0.1323	18.3028 ± 0.7001	19.9960 ± 0.0226	12.6016 ± 0.2194
F_5	$9.68e+03 \pm 86.5390$	$1.21e+04 \pm 512.6435$	$1.31e+04 \pm 369.5460$	$5.84e+03 \pm 147.5333$
F_6	$1.61e+03 \pm 64.0915$	$7.37e+03 \pm 1.56e+03$	$1.17e+04 \pm 1.76e+03$	874.5617 ± 40.2598
F_7	$8.62e+03 \pm 366.1671$	$2.83e+04 \pm 4.72e+03$	$4.01e+04 \pm 5.71e+03$	$4.52e+03 \pm 280.3275$
F_8	$3.70e+05 \pm 2.42e+03$	$4.17e+05 \pm 757.5530$	$3.98e+05 \pm 2.56e+03$	$2.27e+05 \pm 5.62e+03$

apparent decreasing trend as compared with the other six testing functions (except for F_8). This suggested that ADRGA was not trapped in the local optimum on F_4. For the testing function F_8, ADRGA outperformed all the other algorithms. There was a dramatic improvement in the convergence accuracy and speed with ADRGA, and the

Table 3: Improvement percentage of optimal solutions for ADRGA.

Function	Algorithm		
	GA (%)	HGSA (%)	AGA (%)
F_1	46.85	88.79	92.72
F_2	72.68	97.56	98.76
F_3	34.95	100.00	98.17
F_4	4.89	31.15	36.98
F_5	39.67	51.74	55.42
F_6	45.68	88.13	92.53
F_7	47.56	84.03	88.73
F_8	38.65	45.56	42.96

Table 4: Number of residual dimensions and means after each cycle of dimensionality reduction using each algorithm.

Function	Experiment													
	1	2	3	4	⋯	10	11	⋯	16	17	18	19	20	Mean
F_1	642	637	635	631	⋯	614	656	⋯	621	633	625	631	641	636.65
F_2	620	638	654	622	⋯	656	641	⋯	634	635	649	660	634	639.25
F_3	635	657	652	646	⋯	633	637	⋯	620	633	643	612	649	638.45
F_4	642	637	635	631	⋯	614	656	⋯	621	633	625	631	641	636.65
F_5	620	638	654	622	⋯	656	641	⋯	634	635	649	660	634	639.25
F_6	635	657	652	646	⋯	633	637	⋯	620	633	643	612	649	638.45
F_7	653	632	638	648	⋯	650	639	⋯	612	661	641	647	619	639.05
F_8	646	625	619	641	⋯	649	641	⋯	638	660	647	600	631	638

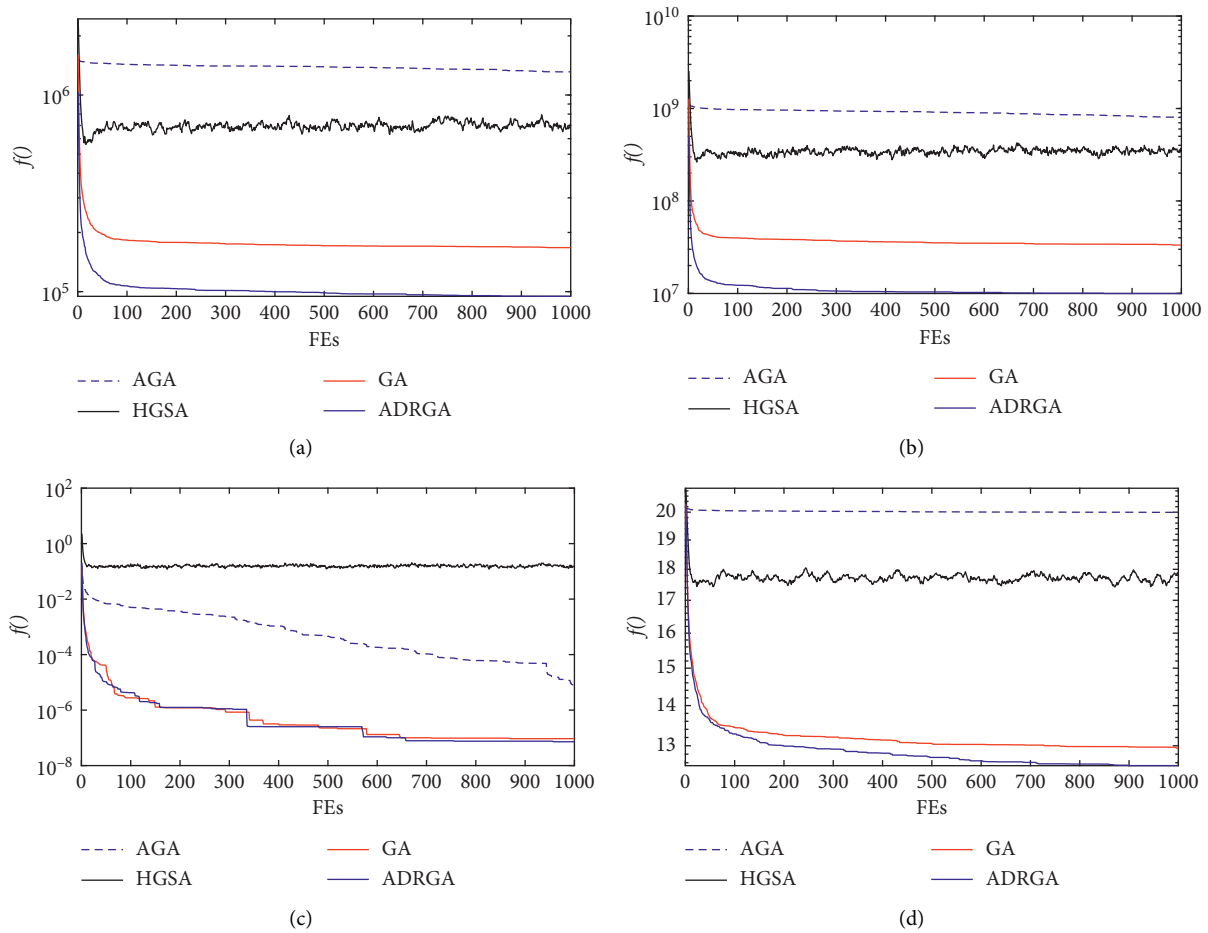

Figure 3: Continued.

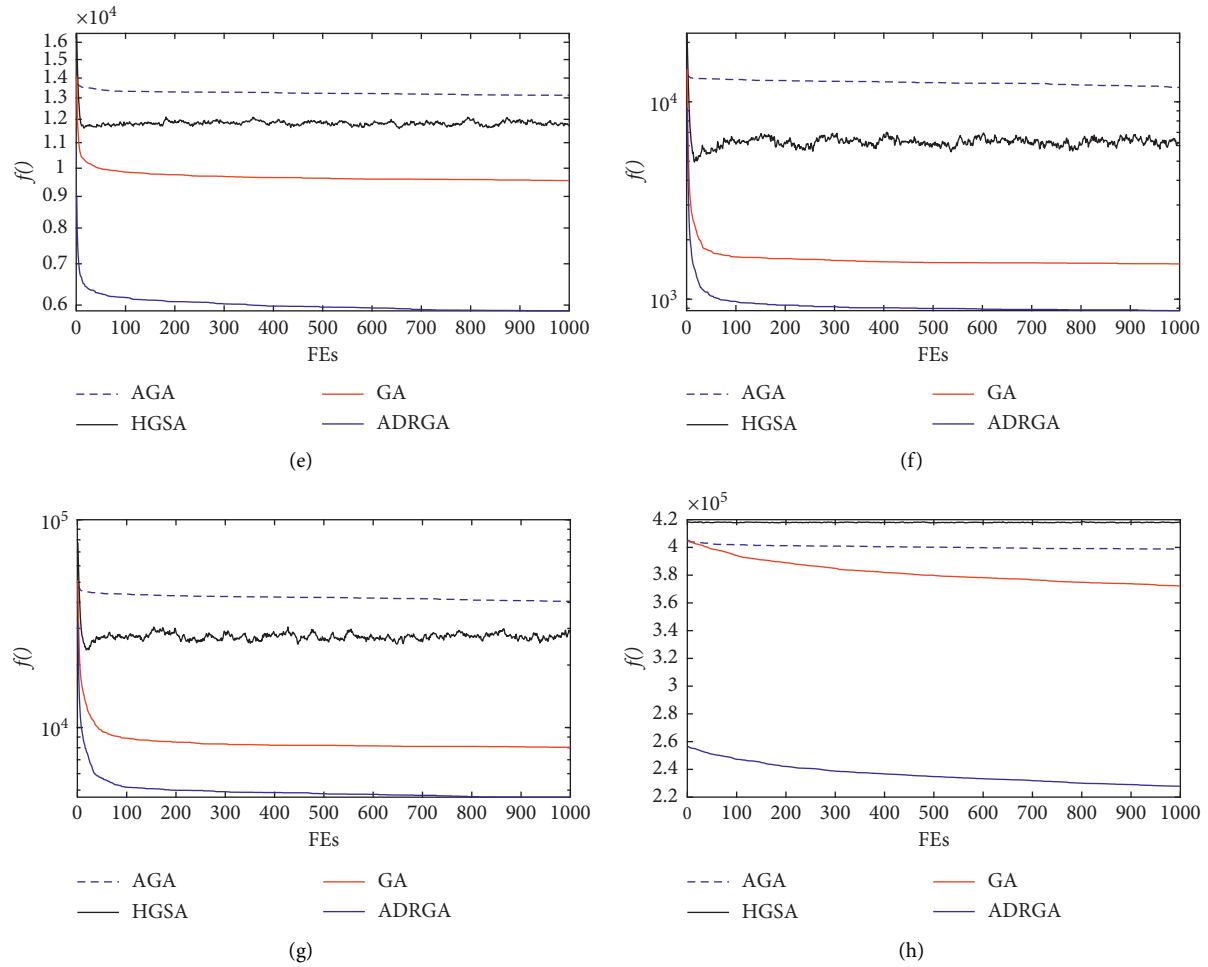

Figure 3: Convergence curve of different algorithms compared with the standard composite test function (including AGA, GA, HGSA, and ADRGA proposed in this paper). (a) Function F_1. (b) Function F_2. (c) Function F_3. (d) Function F_4. (e) Function F_5. (f) Function F_6. (g) Function F_7. (h) Function F_8.

convergence curve displayed a more apparent decreasing trend. For the five high-dimensional multimodal functions, ADRGA achieved better results on two testing functions, indicating that this algorithm was more appropriate for solving high-dimensional multimodal targets.

4. Conclusion

This paper aims to find the optimal solutions for high-dimensional testing functions based on GAs. Similarity, the minimum included angle between the two vectors is utilized, and a GA approach for adaptive dimensionality reduction is proposed. The specific steps for dimensionality reduction are provided. The critical value Q is defined to ensure the diversity of the swarm information. It was then demonstrated via eight testing functions that ADRGA displays faster convergence speed and higher accuracy than other algorithms. However, a major problem with ADRGA is that the critical value Q is prespecified instead of changing with the swarm size and number of dimensions in real-time. In the future, the critical value Q will be the focus in order to achieve better control of the number of dimensions in the swarm and more accurate dimensionality reduction.

Data Availability

The data used to support the study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant U1809209 and the Zhejiang Provincial Natural Science Foundation (LZ15F030002) and the Zhejiang Provincial Natural Science Foundation under Grant LY16F020022.

References

[1] J. D. Bagley, "The behavior of adaptive systems which employ genetic and correlation algorithms," *Dissertation Abstracts International*, vol. 28, no. 12, 1967.

[2] E. J. Hughes, "Radar waveform optimisation as a many objective application benchmark," in *International Conference on Evolutionary Multi-Criterion Optimization*, pp. 700–714, Springer-Verlag, Berlin, Germany, 2007.

[3] P. M. Reed and J. B. Kollat, "Save now, pay later? Multi-period many-objective groundwater monitoring design given systematic model errors and uncertainty," *Advances in Water Resources*, vol. 35, pp. 55–68, 2012.

[4] S. H. Tuo, F. A. Deng, and T. Zhou, "A global optimization algorithm using membrane calculation to solve high-dimensional functions," *Computer Engineering and Applications*, vol. 47, no. 19, pp. 27–30, 2011.

[5] X. H. Chen, X. Li, and N. Wang, "Objective dimension reduction with sparse feature selection for multi-objective optimization problem," *Chinese Journal of Electronics*, vol. 43, no. 7, pp. 1300–1307, 2015.

[6] Z. X. Chen, X. H. Yan, K. A. Wu et al., "Multi-objective optimization integrating open angle -based congestion control strategy," *Acta AutomaticaSinica*, vol. 41, no. 6, pp. 1145–1158, 2015.

[7] E. Zitzler and S. Kúnzli, "Indicator-based selection in multiobjective search," in *Proceedings of the 8th International Conference. Parallel Problem Solving from Nature-PPSN VIII*, pp. 832–842, Springer, Birmingham, UK, 2004.

[8] K. Deb and H. Jain, "An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, Part I: solving problems with box constraints," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 4, pp. 577–601, 2014.

[9] S. Yang, M. Li, X. Liu, and J. Zheng, "A grid-based evolutionary algorithm for many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 17, no. 5, pp. 721–736, 2013.

[10] J. H. Zheng, R. M. Shen, M. Q. Li, and J. Zou, "Evolutionary algorithm based on information separation for multi-objective optimization," *Journal of Software*, vol. 26, no. 5, pp. 1013–1036, 2015.

[11] X. J. Bi and C. Wang, "A multi-objective evolutionary algorithm based on angle penalized distance," *Journal of Electronics & Information Technology*, vol. 40, no. 02, pp. 314–322, 2018.

[12] Y. C. He, X. Z. Wang, X. L. Zhang, and H. Z. Li, "Modeling and solving by dimensionality reduction of KPC problem based on discrete differential evolution," *Chinese Journal of Computers*, vol. 41, no. 116, pp. 1–15, 2018.

[13] J. Liang, R. Liu, K. J. Yu et al., "Dynamic multi-swarm particle swarm optimization with cooperative coevolution for large scale global optimization," *Journal of Software*, vol. 29, no. 9, pp. 2595–2605, 2018.

[14] X. Xia, Y. Xing, B. Wei et al., "A fitness-based multi-role particle swarm optimization," *Swarm and Evolutionary Computation*, vol. 44, pp. 349–364, 2019.

[15] G. Xu, Q. Cui, X. Shi et al., "Particle swarm optimization based on dimensional learning strategy," *Swarm and Evolutionary Computation*, vol. 45, pp. 33–51, 2019.

[16] M. Wu, G. R. Zhang, and H. Liu, "Feature selection based on fractal dimension and multi-objective genetic algorithm," *Computer Engineering and Applications*, vol. 51, no. 11, pp. 109–113, 2015.

[17] Z. He and G. G. Yen, "Many-objective evolutionary algorithms based on coordinated selection strategy," *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 2, pp. 220–233, 2017.

[18] Y. Xiang, Y. Zhou, M. Li, and Z. Chen, "A vector angle-based evolutionary algorithm for unconstrained many-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 1, pp. 131–152, 2017.

[19] Q. M. Zhu, D. Y. Zhao, and J. H. Zhang, "A general U-Block model-based design procedure for nonlinear polynomial control systems," *International Journal of Systems Science*, vol. 47, no. 14, pp. 3465–3475, 2016.

[20] X. Geng, Q. Zhu, T. Liu, and J. Na, "U-model based predictive control for nonlinear processes with input delay," *Journal of Process Control*, vol. 75, pp. 156–170, 2019.

[21] Q. Zhu, W. Zhang, J. Zhang, and B. Sun, "U-neural network-enhanced control of nonlinear dynamic systems," *Neurocomputing*, vol. 352, pp. 12–21, 2019.

[22] Q. M. Zhu, L. Liu, W. C. Zhang, and S. Y. Li, "Control of complex nonlinear dynamic rational systems," *Complexity*, vol. 2018, Article ID 8953035, 12 pages, 2018.

[23] C. F. Meng, D. H. Chu, K. Q. Liu et al., "Solving SaaS components optimization placement problem with hybird genetic and simulated annealing algorithm," *Journal of Software*, vol. 27, no. 4, pp. 916–932, 2016.

[24] S. G. Zhao, G. X. Liu, J. N. Wang, and W.-H. Yang, "Automated design approach for analog circuits based on a multistage adaptive genetic algorithm," *Chinese Journal of Electronics*, vol. 32, no. 4, pp. 680–683, 2004.