Deletion Scheduling for JADE Search Points with a Limited Number of Searches

Tomoya Matsuki[1], Akira Notsu[2], Seiki Ubukata[1], Katsuhiro Honda[1]

[1]Graduate School of Informatics, Osaka Metropolitan University, Sakai, Osaka 599-8531, Japan
[2]Graduate School of Sustainable System Sciences, Osaka Metropolitan University, Sakai, Osaka 599-8531, Japan
E-MAIL: sc24930d@st.omu.ac.jp, notsu@omu.ac.jp

Abstract:

This research proposes a more general framework for controlling the search points of JADE using a sigmoid function in optimization problems with a limited number of searches. The proposed method integrates effective conditions from previous studies with a flexible reduction scheme that adapts to search progress. Its key feature is continuous population size control based on generation progress, ensuring a balance between exploration and exploitation. Experiments on 16 benchmark functions demonstrate improved performance over conventional JADE in both 2D and 10D problems, particularly under limited numbers of function evaluations. This generalized approach enhances optimization efficiency by dynamically adjusting the number of search points to suit different problem characteristics.

Keywords:

black-box optimization problems, DE, JADE

1. Introduction

In real-world optimization problems, it is often difficult to derive general equations that directly yield solutions. Therefore, optimization algorithms that seek near-optimal solutions through multiple trials are widely used, and among them, Differential Evolution (DE) has garnered particular attention due to its efficiency and simplicity of implementation [1], [2].

The population size of search points is an important parameter that determines the performance of differential evolution algorithms. A large population size improves the global search capability but increases the computational cost, while a small population size improves the efficiency of local search but is prone to local solutions. In particular, in real problems using computationally expensive evaluation functions, such as the finite element method, the number of available function evaluations may be limited to 50 to 100 times, making it important to control the number of efficient search points [3], [4]. Under these constraints, methods to dynamically control the population size as the search progresses have been studied [5].

Previous research proposed introducing three effective conditions for controlling population size into JADE [6]. Specifically, these conditions are triggered when i) the solution. the solution update success rate exceeds a threshold (0.3), ii) the population the population size is larger than the problem dimension plus one, and iii) the current generation number exceeds the value obtained by subtracting the population size from the maximum number of generations. These control conditions have achieved improved solution accuracy in the latter stages of the search.

However, the generation-based control in this method still has potential for extension into a more general framework. The control is based on a discrete scheme with simple threshold judgments, making it difficult to flexibly adapt to problem characteristics and search situations. In particular, there is a need for continuous control mechanisms that respond to search progress and consider the scale of the maximum number of generations.

This research extends the control method proposed in previous studies into a more general framework. Specifically, we introduce a continuous control mechanism based on the sigmoid function, combining it with the "solution update success rate" and "population size" conditions that have proven effective in previous research [7]. This enables flexible population size control that adapts to problem characteristics and search situations.

The proposed method constructs a more general framework for population size control by utilizing the characteristics of the sigmoid function [8]. Through adaptive control based on search progress, it maintains sufficient population size in the early stages of search while efficiently ad-

justing the population size as the search progresses. This generalized control mechanism is expected to enable flexible application to various optimization problems.

2. Background: Differential Evolution and JADE

Differential Evolution is a distinctive method that generates new search points using differential information between individuals and is widely used as a direct search method that does not require gradient information. It enables fast and high-precision search for nonlinear and multimodal problems using evolutionary operations like mutation, crossover, and selection [9]. DE has been improved in various ways, such as adding random perturbations to search points and introducing statistical methods. Maintaining balance between population diversity and accuracy is important in its improvement [10], [11], [12].

Search points are updated through mutation using difference vectors. A representative strategy is DE/rand/1/bin, where 'rand' indicates a random base vector, '1' the number of difference vectors, and 'bin' binomial crossover. This strategy generates a mutation vector as follows:

$$\mathbf{v}_i = \mathbf{x}_{r1} + F \cdot (\mathbf{x}_{r2} - \mathbf{x}_{r3}) \tag{1}$$

where $\mathbf{x}_{r1}$, $\mathbf{x}_{r2}$, $\mathbf{x}_{r3}$ are distinct individuals randomly selected from the population, and F is the mutation factor. The use of these difference vectors enables efficient exploration by utilizing the local structure of the search space.

In the subsequent crossover operation, genetic information is exchanged between the mutation vector and the current solution. In binomial crossover, trial vectors are generated according to the following equation:

$$u_{i,j} = \begin{cases} v_{i,j}, & \text{if } rand \leq CR \text{ or } j = j_{rand} \\ x_{i,j}, & \text{otherwise} \end{cases} \tag{2}$$

where CR is the crossover rate, $rand$ is a uniform random number in [0,1], and j_{rand} is a randomly selected dimension. This operation maintains diversity in the search.

Finally, in the selection operation, the trial vector is compared with the current solution, and the better solution is retained for the next generation:

$$\mathbf{x}_i^{new} = \begin{cases} \mathbf{u}_i & \text{if } f(\mathbf{u}_i) < f(\mathbf{x}_i) \\ \mathbf{x}_i & \text{otherwise} \end{cases} \tag{3}$$

The performance of Differential Evolution depends heavily on the mutation factor F and the crossover rate CR, whose optimal values vary across problems and search stages. As a result, fixed parameter settings can reduce search efficiency. To address this, methods such as jDE [13] have introduced self-adaptive control, allowing individuals to adjust parameters during evolution. Population size also plays a critical role in balancing global and local search, and the effective control of these parameters remains a central challenge in DE research.

Many variants of DE have been developed to improve its performance. One notable example is JADE, which introduced adaptive parameter control and a new mutation strategy. In JADE, the mutation strategy "DE/current-to-pbest" uses the current individual and a randomly selected individual from the top p% best solutions to enhance convergence:

$$\mathbf{v}_{i,g} = \mathbf{x}_{i,g} + F_i \cdot (\mathbf{x}_{best,g}^p - \mathbf{x}_{i,g}) + F_i \cdot (\mathbf{x}_{r1,g} - \mathbf{x}_{r2,g}) \tag{4}$$

Here, g denotes the generation number, indicating that all variables are updated each generation. In particular, $\mathbf{x}_{best,g}^p$ is selected from the top p% solutions in generation g.

JADE also generates the control parameters F_i and CR_i from probability distributions and updates them based on successful trials. These mechanisms improve performance by adapting to the problem and search progress. However, like standard DE, JADE maintains a fixed population size throughout the search, which may limit flexibility in problems with a limited number of function evaluations. This motivates our proposed approach, which introduces a dynamic population control mechanism into the JADE framework.

3 Proposed method

This research proposes a new approach that extends the population size control method from previous studies into a more general framework, applying a flexible deletion schedule using the sigmoid function. Based on JADE, we introduce a population reduction mechanism that enables flexible and adaptive adjustment of population size according to search progress. This method supports efficient and stable search by providing control suited to problem characteristics and search situations. This method is applicable to many DE systems. However, in the case of optimization with a small number of generations, there is no significant difference in performance between recent methods. Therefore, JADE was adopted as a baseline. Furthermore, since practical application is emphasized, careful attention

to convergence is necessary when the method is used in real-world scenarios requiring convergence.

In population size reduction, control is performed according to search progress using the sigmoid function. Specifically, we calculate the search progress as a normalized value and input this to the sigmoid function to adjust the population size. The normalized progress is expressed by the following equation:

$$\gamma = \frac{I - MAXITER}{MAXITER} \tag{5}$$

where I represents the current generation number and $MAXITER$ denotes the maximum number of generations. This enables population size reduction according to search progress.

In population size control, adaptive control based on search progress is crucial. This research proposes a control mechanism that determines population size reduction by comparing a control value controlValue with the squared generation number I to maximize search efficiency. The control value controlValue is defined by the following equation:

$$\text{controlValue} = MAXITER\sqrt{\frac{\exp(-\alpha \cdot \gamma)}{1 + \exp(-\alpha \cdot \gamma)}} \tag{6}$$

This equation enables maintaining high population size in the early stages while efficiently reducing population size in the later stages. The slope of the sigmoid function (steepness: α) is adjusted based on the maximum number of generations (MAXITER), realizing flexible control adapted to search progress. We defined α as follow:

$$\alpha = \frac{MAXITER}{10} \tag{7}$$

Proposed method was adopted as the criterion based on extensive preliminary experiments across various test cases. Through numerous trials with different dimensionalities and function types, we observed that this function effectively aligns with the convergence characteristics of the search process. This control rule demonstrated the ability to maintain sufficient population size in early stages while promoting appropriate reduction as the search progresses. This control mechanism enables adaptive adjustment of population size according to search progress, facilitating efficient search. In particular, it significantly contributes to improving the efficiency of local search in the later stages of exploration.

In this research, we generalized the population size control method proposed in previous research [5] and achieved a balance between search efficiency and solution diversity by combining multiple control conditions that consider solution update success rate and problem dimensionality.

A threshold parameter θ for the solution update rate plays a key role in controlling population size reduction. Through preliminary experiments across various benchmark problems, we determined that values in the range $\theta \in \{0.3, 0.4, 0.5, 0.6, 0.7\}$ provide effective control over the population reduction process. A lower threshold allows more aggressive population reduction when the solution update rate exceeds this value, while a higher threshold maintains population diversity longer by requiring a higher success rate for reduction. The steps of the proposed method are described below.

Step 1 Population Generation:
Generate N initial individuals $\{\boldsymbol{x}_i, i = 1, 2, \ldots, N\}$ randomly in the search space.

Step 2 Exploration in Each Generation:
If the number of searches exceeds the maximum search count (SEARCHMAX), terminate the algorithm. Repeat the following as long as the conditions are satisfied.

2-1 Generation of Next Search Point Population:
Substitute JADE equations.

2-2 Selection of Fitter Individuals:
If the specified number of searches is reached, end Step 2.

2-3 Parameter Update:

- Crossover rate: CR
- Mutation factor: F
- Update rate: $rate = \frac{\text{total improved offspring}}{\text{N}}$
- Equation (5)
- Equation (6)

2-4 Search Point Control:

- $rate > \theta$
- $N >$ Dimension $+ 1$
- $I >$ ControlValue

When all of the above conditions are met, exclude the individual with the worst evaluation from the population.

Step 3 Output of Solutions:
The final solution is the minimum (or maximum) value among the stored individuals.

4. Numerical experiments, results and discussion

In this research, the conventional JADE and the proposed method are applied to a total of 16 benchmark functions, consisting of four representative unimodal functions (F1: Sphere, F2: Rosenbrock, F3: Booth, F4: Matyas) and twelve multimodal functions (F5: Easom, F6: Rastrigin, F7: Ackley, F8: Levi N.13, F9: Bukin N.6, F10: Beale, F11: Goldstein-Price, F12: Schaffer N.2, F13: Five-well Potential, F14: Griewank, F15: Xin-She Yang, F16: Styblinski-Tang), in order to compare the solution accuracy of both approaches.

We discuss the experimental results presented in TABLE 1 to 6. In each table, results that outperform JADE are underlined, and the best solutions are highlighted in bold. For all experiments, the population size was set to $N = D \times 5$. The number of generations and runs was determined based on the dimensionality and the experimental setting (e.g., 10, 20, or 40 generations; 100 to 2000 runs).

TABLE 1. Performance comparison with optimal parameters (100 iterations, 10 generations, $D = 2$)

		JADE	$\theta = 0.3$	$\theta = 0.4$	$\theta = 0.5$	$\theta = 0.6$	$\theta = 0.7$
Single-peak	F1	0.00087296	0.00078569	0.00084819	0.00077269	0.00078201	0.00072214
	F2	0.94626	0.88261	0.89707	0.92400	0.91501	0.88135
	F3	0.63981	0.61400	0.63797	0.58076	0.51937	0.53000
	F4	0.023978	0.02186	0.021872	0.022326	0.020656	0.019124
Multi-peak	F5	-0.0054949	-0.013344	-0.010755	-0.01053	-0.0095933	-0.0095933
	F6	2.5622	2.4161	2.5390	2.4547	2.4502	2.4248
	F7	3.9286	3.8715	3.8291	3.7562	3.6922	3.6312
	F8	0.37762	0.38281	0.38920	0.37727	0.33756	0.34130
	F9	7.8385	7.7533	7.5043	7.4315	7.4979	7.5933
	F10	0.36521	0.32002	0.34463	0.34628	0.36562	0.35652
	F11	12.2973	11.7883	11.8399	11.7573	11.2228	11.2616
	F12	0.15404	0.15252	0.14786	0.14587	0.14814	0.15217
	F13	-0.70707	-0.69708	-0.69450	-0.69535	-0.71330	-0.71677
	F14	1.0786	1.0709	1.0750	1.0695	1.0708	1.0638
	F15	0.27916	0.25990	0.26127	0.26168	0.26526	0.26057
	F16	-74.6379	-74.9704	-75.2733	-74.9310	-75.2484	-75.1211
better than JADE		-	14	14	15	15	16
equal to JADE		-	0	0	0	0	0
worse than JADE		-	2	2	1	1	0
sign test result		-	$p < 0.01$	$p < 0.01$	$p < 0.01$	$p < 0.01$	$p < 0.01$

The experimental results for two-dimensional problems (TABLE 1 to TABLE 3) demonstrate the basic performance of the proposed method. In the experiment with 10 generations (TABLE 1), performance improvements were confirmed in all 16 functions with $\theta = 0.7$ ($p < 0.01$), particularly in Multi-peak functions. The timing of population reduction is appropriately scaled according to the maximum number of generations, achieving control based on search progress. In the experiment with increased maximum generations to 20 (TABLE 2), performance improvements were achieved in all functions with settings of $\theta = 0.3$ and $\theta = 0.4 (p < 0.01)$, with notable enhancement

TABLE 2. Performance comparison with optimal parameters (200 iterations, 20 generations, $D = 2$)

		JADE	$\theta = 0.3$	$\theta = 0.4$	$\theta = 0.5$	$\theta = 0.6$	$\theta = 0.7$
Single-peak	F1	7.3802e-06	7.138e-06	6.0563e-06	5.388e-06	3.9191e-06	2.7743e-06
	F2	0.27507	0.2631	0.23266	0.24168	0.23846	0.22239
	F3	0.028155	0.020201	0.017325	0.014223	0.011685	0.009116
	F4	0.0016522	0.00099005	0.00078596	0.00077748	0.00064955	0.00062488
Multi-peak	F5	-0.045226	-0.067886	-0.081336	-0.095551	-0.10013	-0.10181
	F6	0.89963	0.77105	0.72922	0.73409	0.70904	0.73663
	F7	0.60466	0.56613	0.49686	0.47627	0.38609	0.33865
	F8	0.025428	0.023922	0.019985	0.017988	0.017298	0.0166
	F9	3.3832	2.8645	2.9881	2.8627	2.8914	2.9305
	F10	0.17347	0.15124	0.17034	0.15319	0.15727	0.14873
	F11	5.9054	5.3349	5.0059	4.7613	5.5775	5.0199
	F12	0.057067	0.046906	0.042911	0.044537	0.041879	0.043524
	F13	-0.92641	-0.92697	-0.93604	-0.92353	-0.94116	-0.92444
	F14	1.0005	1.0005	1.0005	1.0003	1.0006	1.0001
	F15	0.14225	0.12187	0.12669	0.12281	0.11463	0.11188
	F16	-76.9267	-77.0796	-77.0373	-77.0483	-76.828	-77.008
better than JADE		-	16	16	15	14	15
equal to JADE		-	0	0	0	0	0
worse than JADE		-	0	0	1	2	1
sign test result		-	$p < 0.01$	$p < 0.01$	$p < 0.01$	$p < 0.01$	$p < 0.01$

TABLE 3. Performance comparison with optimal parameters (400 iterations, 40 generations, $D = 2$)

		JADE	$\theta = 0.3$	$\theta = 0.4$	$\theta = 0.5$	$\theta = 0.6$	$\theta = 0.7$
Single-peak	F1	5.5773e-10	5.7429e-10	4.0729e-10	2.2349e-10	2.1192e-10	2.2627e-10
	F2	0.085524	0.076789	0.067863	0.073647	0.075084	0.074041
	F3	0.00023626	2.4574e-05	1.2992e-05	1.5972e-05	1.3077e-05	2.4903e-05
	F4	5.8261e-06	4.3085e-06	1.2128e-06	2.0266e-06	2.1427e-06	2.213e-06
Multi-peak	F5	-0.29707	-0.62509	-0.63926	-0.60446	-0.60353	-0.58347
	F6	0.10815	0.08123	0.086067	0.07736	0.070655	0.094865
	F7	0.0084086	0.0076342	0.007364	0.0041768	0.0037827	0.0040005
	F8	0.0010705	0.0010039	0.00089415	0.00067405	0.00047806	0.00078329
	F9	1.283	1.0115	1.0055	0.99243	0.97903	0.96135
	F10	0.12702	0.12803	0.10708	0.1047	0.10907	0.12328
	F11	4.9952	4.65	4.3249	4.4342	4.5136	4.9187
	F12	0.0092067	0.0071333	0.0075968	0.006941	0.0068909	0.0067043
	F13	-0.97778	-0.97473	-0.98112	-0.97756	-0.99058	-0.97779
	F14	0.99975	0.99975	0.99975	0.99975	0.99975	0.99975
	F15	0.043631	0.036161	0.032631	0.036221	0.036648	0.031902
	F16	-77.2292	-77.2902	-77.2721	-77.2155	-76.9469	-77.3144
better than JADE		-	12	15	14	15	16
equal to JADE		-	0	0	0	0	0
worse than JADE		-	4	1	2	1	0
sign test result		-	$p < 0.05$	$p < 0.01$	$p < 0.01$	$p < 0.01$	$p < 0.01$

in Single-peak functions compared to the 10-generation case.

TABLE 4. Performance comparison with optimal parameters (500 iterations, 10 generations, $D = 10$

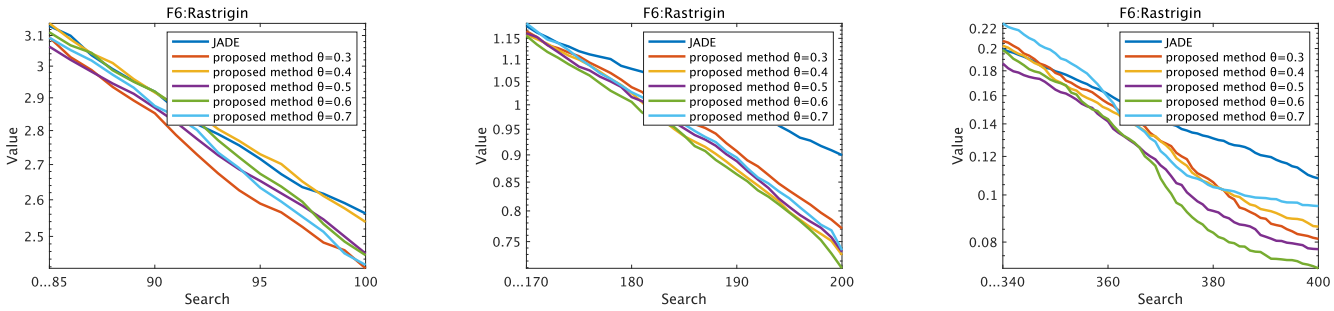
		JADE	$\theta = 0.3$	$\theta = 0.4$	$\theta = 0.5$	$\theta = 0.6$	$\theta = 0.7$
Single-peak	F1	0.2532	0.2457	0.25287	0.2479	0.2439	0.24823
	F2	1273.4221	1242.5313	1273.6702	1250.5037	1224.0291	1239.3632
Multi-peak	F6	64.9101	65.253	64.809	65.0547	65.0732	64.8208
	F7	14.8182	14.8134	14.7814	14.7164	14.7307	14.6935
	F14	23.7877	23.1131	23.7587	23.3113	22.9511	23.3409
	F15	0.015952	0.015893	0.016391	0.015718	0.015718	0.015718
	F16	-277.2216	-275.4908	-276.1529	-275.2414	-275.2414	-275.2414
better than JADE		-	5	4	5	5	6
equal to JADE		-	0	0	0	0	0
worse than JADE		-	2	3	2	2	1
sign test result		-	$p < 0.3$	$p < 0.5000$	$p < 0.3$	$p = 0.2266$	$p = 0.0625$

TABLE 5. Performance comparison with optimal parameters (1000 iterations, 20 generations, $D = 10$)

		JADE	$\theta = 0.3$	$\theta = 0.4$	$\theta = 0.5$	$\theta = 0.6$	$\theta = 0.7$
Single-peak	F1	0.048384	0.050119	0.049786	0.047966	0.046336	0.04613
	F2	189.8659	185.8044	184.1023	179.8713	175.9344	176.3578
Multi-peak	F6	47.1463	47.2088	47.0388	46.9733	46.9733	46.9733
	F7	9.5015	9.562	9.6123	9.4389	9.4159	9.3581
	F14	5.3546	5.5107	5.4807	5.3169	5.1703	5.1517
	F15	0.0079815	0.0079729	0.0079115	0.0079115	0.0079115	0.0079115
	F16	-306.0463	-306.5388	-306.307	-305.8934	-305.8934	-305.8934
better than JADE		-	3	4	6	6	6
equal to JADE		-	0	0	0	0	0
worse than JADE		-	4	3	1	1	1
sign test result		-	$p < 0.5000$	$p < 0.5000$	< 0.1	$p < 0.1$	$p < 0.1$

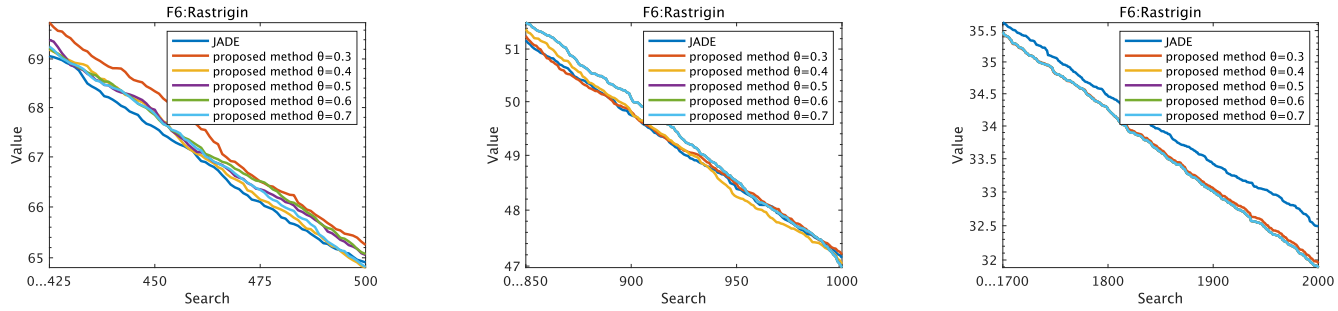
TABLE 6. Performance comparison with optimal parameters (2000 iterations, 40 generations, $D = 10$)

		JADE	$\theta = 0.3$	$\theta = 0.4$	$\theta = 0.5$	$\theta = 0.6$	$\theta = 0.7$
Single-peak	F1	0.0022816	0.0022621	0.0022868	0.0021285	0.0019929	0.0020055
	F2	18.6844	17.7435	16.8862	15.7372	16.484	16.2504
Multi-peak	F6	32.49	31.9522	31.8902	31.8902	31.8902	31.8902
	F7	3.774	3.7454	3.716	3.6519	3.6002	3.5882
	F14	1.2053	1.2036	1.2071	1.1887	1.1796	1.1805
	F15	0.0043033	0.0042164	0.0042164	0.0042164	0.0042164	0.0042164
	F16	-346.7529	-348.0947	-348.7955	-348.8092	-348.9026	-348.9026
better than JADE		-	7	5	7	7	7
equal to JADE		-	0	0	0	0	0
worse than JADE		-	0	2	0	0	0
sign test result		-	$p < 0.01$	$p < 0.3$	$p < 0.01$	$p < 0.01$	$p < 0.01$



100 iterations (10 generations) 200 iterations (20 generations) 400 iterations (40 generations)

FIGURE 1. Detailed Analysis of Final Convergence Results by Generation in 2D Space (F6:Multi-peak)



500 iterations (10 generations) 1000 iterations (20 generations) 2000 iterations (40 generations)

FIGURE 2. Detailed Analysis of Final Convergence Results by Generation in 10D Space (F6:Multi-peak)

The experimental results for two-dimensional problems (TABLE 1 to TABLE 3) demonstrate the basic performance of the proposed method. For 10 generations (TABLE 1), performance improvements over JADE were observed in all 16 functions with $\theta = 0.7$ ($p < 0.01$), particularly for Multi-peak functions. The timing of population reduction was appropriately scaled to the maximum number of generations, ensuring control based on search progress. With 20 generations (TABLE 2), all functions showed improvements over JADE at $\theta = 0.3$ and $\theta = 0.4$ ($p < 0.01$), with particularly notable gains in Single-peak functions compared to the 10-generation case.

To better understand the previous quantitative results, we visualized the transition of function values during the final stages of the search for representative functions. Figures 1 and 2 show the changes in function values during the final stages of the search for the Multi-peak function (F6: Rastrigin) in 2D and 10D. Although the convergence behavior varies depending on the setting of the parameter θ, performance equal to or better than that of JADE was visually confirmed.

The influence of the control parameter θ showed different trends depending on problem characteristics. Two-dimensional problems tended to show better results with relatively high θ values ($0.6 - 0.7$), while ten-dimensional problems showed better results with moderate θ values ($0.5 - 0.6$). This difference is thought to reflect the varying search characteristics due to the dimensionality of the search space. The performance improvement in high-dimensional problems particularly suggests that efficient control of population size functions effectively against the expansion of the search space.

These results demonstrate the effectiveness of the continuous control of the number of search points by the sigmoid function. In particular, the reduction timing changes according to the maximum number of generations, which results in a stable performance improvement according to the progression level and confirms its versatility for a wide range of problems.

Based on these experimental results, key features of the proposed method include flexible adjustment of search points using a sigmoid function, adaptive control of re-

duction timing based on the maximum generations, and broad applicability through generalization. In particular, the reduction timing is appropriately scaled with the maximum generations, enabling efficient search based on progress. This control mechanism demonstrated performance improvements, regardless of problem dimensionality or search space characteristics.

5. Conclusions

In this research, we proposed a continuous control mechanism based on the sigmoid function to generalize population size control in Differential Evolution. Compared to conventional discrete control methods, this approach achieves more flexible adjustment of population size according to search progress. In particular, the introduction of the control parameter θ enabled adaptive control based on problem characteristics.

Experimental evaluation demonstrated the effectiveness of the proposed method in both two-dimensional and ten-dimensional optimization problems. In two-dimensional problems, performance improvements were widely confirmed across 16 benchmark functions, with particularly significant improvements in Multi-peak functions. Similarly, stable performance improvements were achieved in ten-dimensional problems, demonstrating the versatility of the proposed method.

The continuous control using the sigmoid function enabled smooth adjustment of population size from the initial stage to the final stage of search, achieving efficient exploration. This feature proved to be particularly advantageous in problems with multiple local optima.

Future challenges include application to higher-dimensional problems and development of automatic adjustment mechanisms for control parameters, through which further development of the proposed method is expected.

Acknowledgements

This work was supported by JSPF KAKENHI Grant Numbers JP18K11473, 22K12182.

References

[1] R. Storn and K. Price, “Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces,” Journal of Global Optimization, Vol. 4, pp. 341–359, 1997.

[2] R. Storn and K. Price, “Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces,” Technical Report, TR-95-012, ICSI, 1995.

[3] S. Maeda, T. Kato, K. Ikushima, and M. Shibahara, “Prediction of welding deformation of large-scale structure using inherent deformation computed by using machine learning,” 13th International Seminar on Numerical Analysis of Weldability, 2023.

[4] Osaka Metropolitan University, Shibahara Laboratory. Available: https://www.omu.ac.jp/eng/shibahara/

[5] T. Matsuki, A. Notsu, K. Honda, T. Kato, and M. Shibahara, “Control of JADE population in limited number of searches for realistic situations,” Proceedings of the 2024 IEEE Congress on Evolutionary Computation (CEC), pp. 1–8, 2024.

[6] J. Zhang and A. C. Sanderson, “JADE: Adaptive differential evolution with optional external archive,” IEEE Transactions on Evolutionary Computation, Vol. 13, No. 5, pp. 945–958, 2009.

[7] C. M. Bishop, Pattern Recognition and Machine Learning, Springer, 2006.

[8] D. Hush and B. Horne, “Progress in supervised neural networks,” IEEE Signal Processing Magazine, Vol. 10, No. 1, pp. 8–39, Jan. 1993.

[9] M. F. Ahmad, N. A. M. Isa, W. H. Lim, and K. M. Ang, “Differential evolution: A recent review based on state-of-the-art works,” Alexandria Engineering Journal, Vol. 61, Issue 5, pp. 3831–3872, May 2022.

[10] A. Notsu, M. Sakakibara, S. Ubukata, and K. Honda, “Setting of candidate solutions considering confidence intervals in differential evolution,” Proceedings of the 2018 International Conference on Fuzzy Theory and Its Applications, pp. 7–11, 2018.

[11] A. Notsu, J. Tsubamoto, Y. Miyahira, S. Ubukata, and K. Honda, “Randomness selection in differential evolution using Thompson sampling,” Proceedings of the 2020 Joint 11th International Conference on Soft

Computing and Intelligent Systems and 21st International Symposium on Advanced Intelligent Systems, pp. 1–5, 2020.

[12] J. Cheng, Z. Pan, H. Liang, Z. Gao, and J. Gao, "Differential evolution algorithm with fitness and diversity ranking-based mutation operator," Swarm and Evolutionary Computation, Vol. 61, 2021.

[13] J. Brest, S. Greiner, B. Boskovic, M. Mernik, and V. Zumer, "Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems," IEEE Transactions on Evolutionary Computation, Vol. 10, No. 6, pp. 646–657, 2006.