# Multi-Point Search Towards Dynamic Multimodal Optimization with Frequent Solution Changes on their Locations, Moving speeds, and Numbers

Shoei Fujita
*The University of Electro-Communications*
Tokyo, Japan
shfujita@cas.lab.uec.ac.jp

Hiroyuki Sato
*The University of Electro-Communications*
Tokyo, Japan
h.sato@uec.ac.jp

Keiki Takadama
*The University of Tokyo*
Chiba, Japan
takadama@g.ecc.u-tokyo.ac.jp

***Abstract*—This paper focuses on dynamic multimodal optimization problems where multiple optimal solutions change (i) their locations, (ii) their movement speeds, and (iii) their number as time passes, and investigates how our method, NMMSO- and CMA-ES-based Continuous Optimizer (NCCO) with Population Size Adjustment, can cope with these three changes on optimal solutions. Unlike the conventional researches which mainly evaluated the methods on the issue (i) with a very slow location change of optimal solutions (e.g., optimal solutions do not move their locations until a certain number of evaluations), this paper focuses on a frequent change of optimal solutions which changes their locations (i) their locations, (ii) their movement speeds, and (iii) their number before all individuals have completed to be evaluated, and investigates a robustness of our method together with the conventional methods. The experiments on dynamic benchmark functions in the two dimensions based on the Moving Peaks Benchmark (MPB) have revealed the following implications: (1) our method showed both the smallest fitness error (which evaluates a difference between the true fitness and the best fitness found by the methods) and distance error (which evaluates a difference between the location of all true optimal solutions and that of those found by the methods) in comparison with the conventional methods; (2) By flexibly adjusting the numbers of swarms and individuals, the proposed method exhibited particularly strong performance in problems involving changes in the movement speeds and the number of optimal solutions.**

***Index Terms*—dynamical multimodal optimization, particle swarm optimization, CMA-ES, NMMSO, swarm intelligence, moving peaks benchmark**

## I. Introduction

Dynamic Optimization Problems (DOPs), in which the objective function changes over time [1], can be interpreted as changes in the solution landscape (environment). Such changes include not only variations in the (i) locations of optima, but also changes in their (ii) movement speeds and (iii) number. However, much of the existing literature assumes "discrete" changes, even within category (i), where the environment remains unchanged for a certain period after a change occurs. In other words, the environment changes only after all individuals have been evaluated and some time has elapsed. For such settings, methods such as Dynamic Multi-Swarm Fractional Best Particle Swarm Optimization (DMSFPSO) [3] and Species-based Particle Swarm Optimization with Adaptive Population and Adaptive Deactivation ($SPSO_{+AP+AD}$) [11] are effective. Nevertheless, these approaches struggle to track "continuous" changes, in which the environment evolves gradually, as commonly observed in real-world scenarios. This limitation becomes particularly critical in practical problems with expensive fitness evaluations, where the environment may change before all individuals are evaluated. Furthermore, beyond changes in the locations of optima, DOPs require robustness against changes in (ii) movement speeds and (iii) the number of optima. However, DSPSO [10] can handle only a limited range of velocity changes, while multiswarm Quantum Swarm Optimization (mQSO) [2] requires the number of optima to be specified in advance.

Against this background, this study focuses on scenarios involving extremely high-frequency changes, in which the environment changes before all individuals are evaluated, and aims to comprehensively analyze how robust various existing methods—including the proposed approach—are to the three types of environmental changes described above. Specifically, the study compares the performance of NCCO(dec+ul) [13], which extends the NMMSO- and CMA-ES-based Continuous Optimizer (NCCO) [12]—originally designed to track continuous changes under sufficiently long evaluation times—by incorporating population size adjustment mechanisms. The comparison includes $SPSO_{+AP+AD}$, regarded as a state-of-the-art method for DOPs, as well as other representative approaches such as DMSFPSO, DSPSO, and multipopulation Differential Evolution (mDE) [9]. In addition, mQSO is included as a reference, despite requiring the number of optima to be specified in advance. Here, the dec (decrement) mechanism of NCCO(dec+ul) adaptively reduces the number of individuals in a swarm to track optima with as few individuals as possible, while the ul (upper limit) mechanism imposes an upper bound on the swarm population size.

A key challenge in environments that change before all individuals are evaluated is the trade-off between repeatedly updating a small number of individuals to rapidly adapt to

979-8-3315-3431-8/25/$31.00 ©2025 IEEE

changes and maintaining a sufficiently large population to reliably discover and track optima. If the population size is too small, the probability of failing to locate or follow optima increases. Under such conditions, problems involving simultaneous changes in the (i) locations, (ii) movement speeds, and (iii) number of optima become significantly more difficult than conventional DOPs. In this study, these environmental changes are realized using the Moving Peaks Benchmark (MPB), and the robustness of each method is systematically compared.

The remainder of this paper is organized as follows. Section II introduces NCCO(dec+ul), an algorithm capable of tracking continuous changes while adaptively adjusting population size. Section III presents experimental comparisons between the proposed method and conventional approaches, and Section IV discusses the obtained results. Finally, Section V concludes the paper.

## II. NMMSO and CMA-ES based Continuous Optimizer (NCCO(dec+ul))

### A. Overview

NCCO(dec+ul) enhances NCCO, which was originally proposed for continuously changing multimodal DOPs, by incorporating a population size adjustment mechanism, thereby enabling adaptation to more frequent environmental changes. NCCO integrates the niching mechanism of the Niching Migratory Multi-Swarm Optimizer (NMMSO) [5] with dynamic optimization algorithms based on Particle Swarm Optimization (PSO) [6] and Covariance Matrix Adaptation Evolution Strategies (CMA-ES) [4], namely Tracking-PSO (TPSO) and Tracking-CMA-ES (TCMA-ES). By dynamically adjusting the number of swarms through swarm generation, separation, and merging based on NMMSO, NCCO enables the search for optima whose number varies over time. At the same time, assigning both TPSO and TCMA-ES individuals to each swarm allows simultaneous estimation of movement directions and local exploitation, thereby achieving effective tracking of continuously moving optima.

In addition, NCCO(dec+ul) introduces decrement and upper-limit processes that adjust the number of individuals according to the search status of each swarm. A "tracking state", in which a swarm is regarded as successfully following an optimum, is detected when the step size $\sigma$ of TCMA-ES converges. Once a swarm enters the tracking state, the upper-limit process prevents an increase in the population size through the merging mechanism, while the decrement process reduces the number of individuals by one at each iteration. Tracking failure is identified based on a decreasing trend in fitness values.

### B. Swarm

*1) TCMA-ES:* TCMA-ES is one of the individual types assigned to each swarm in NCCO and is a dynamic optimization method that extends CMA-ES [4] to generate individuals in the direction of the moving optimum. Specifically, individuals (dimension $\times$ 2) are placed on the boundary of a Gaussian distribution (ellipsoid), enabling evaluations in a wide range of directions. This design prevents TCMA-ES from losing track of moving optima.

TCMA-ES updates its parameters based on elite individuals within a swarm. When the best individual in TPSO, $S.tpso_{cgbest}$, achieves a better fitness value than the best individual in TCMA-ES, $S.tcmaes_{cgbest}$, $S.tpso_{cgbest}$ is incorporated into the parameter update process. This mechanism allows TCMA-ES to exploit high-quality solutions found by TPSO when its own search progress becomes insufficient.

*2) TPSO:* TPSO is one of the individual types assigned to each swarm in NCCO and is designed to track dynamically moving optima. Instead of using the "historical" personal best $pbest$ and global best $gbest$ employed in standard PSO, TPSO promotes movement toward the "current" swarm-best individual, denoted as $cgbest$. This modification is necessary because, after an environmental change, historical best solutions may guide the search in directions that are entirely unrelated to the current optima. Accordingly, the velocity update is formulated as in Equation (1). Here, $tpso_i$ denotes the $i$-th TPSO individual, $\boldsymbol{r}_3$ is a vector of uniformly distributed random variables in $[0, 1]$, and $w$ and $c_3$ are parameters.

$$tpso_i.\boldsymbol{v}_i^t = wtpso_i.\boldsymbol{v}_i^{t-1} + c_3\boldsymbol{r}_3(S.cgbest^{t-1} - tpso_i^{t-1}) \quad (1)$$

When the best TPSO individual, $S.tpso_{cgbest}$, is inferior to the best TCMA-ES individual, $S.tcmaes_{cgbest}$, the update is directed toward $S.tcmaes_{cgbest}$, as shown in Equation (2).

$$\boldsymbol{v}_i^t = \begin{cases} \text{if } f(S.tcmaes_{cgbest}^{t-1}) > f(S.tpso_{cgbest}^{t-1}) \\ \quad w\boldsymbol{v}_i^{t-1} + c_4(S.tcmaes_{cgbest}^{t-1} - S.tpso_{cgbest}^{t-1}) \\ \text{else} \\ \quad w\boldsymbol{v}_i^{t-1} \end{cases} \quad (2)$$

### C. Tracking State

The upper-limit and decrement processes must be applied only after a swarm has secured a sufficient number of individuals; otherwise, an insufficient population may prevent the swarm from successfully tracking an optimum. To this end, a "tracking state", in which a swarm is considered to be successfully following an optimum, is determined using the step size $\sigma$ of TCMA-ES. The step size $\sigma$ represents the scale of the individual distribution: it is large during the search phase and decreases after an optimum has been located. Therefore, once $\sigma$ converges, it is assumed that the swarm can track the optimum with a sufficient number of individuals. In this study, a swarm is defined to be in the tracking state if the standard deviation of $\sigma$ over the $t_w$ loop of NCCO is less than or equal to $\tau_{tr}$.

While tracking-state detection based on the standard deviation of $\sigma$ is suitable for identifying the onset of successful tracking, it is ineffective for detecting tracking failure. This is because, once $\sigma$ converges, the TCMA-ES individuals generated from it also converge, and the value of $\sigma$ rarely exhibits large fluctuations. Accordingly, NCCO(dec+ul) uses the standard deviation of $\sigma$ to judge tracking success and the fitness value of each swarm's $cgbest$ to judge tracking failure.

Specifically, tracking failure is detected when the fitness value of $cgbest$ decreases monotonically for $t_w$ consecutive iterations and the fitness difference over these $t_w$ iterations exceeds a threshold $\tau_{failure}$. When tracking failure is detected, the lower bound on the number of TPSO individuals in the swarm, $ng_{min}$, is incremented by one, the tracking state is canceled, and the TCMA-ES of the swarm is reinitialized.

### D. Mechanism

*1) Merge:* In the merge mechanism, when two swarms are judged to be exploring the same optimum—either (i) the distance to the nearest swarm is less than $TOL$, or (ii) the fitness value evaluated at the midpoint between the two nearest swarms is better than the $cgbest$ of either swarm—one of the swarms becomes redundant, and the two swarms are merged into a single swarm. Upon merging, all individuals from both swarms are sorted in descending order of fitness, and the top $ng_{max}$ individuals are retained as the new swarm. The inter-swarm distance and the midpoint are computed using the current best individuals, $cgbest$, of the two swarms.

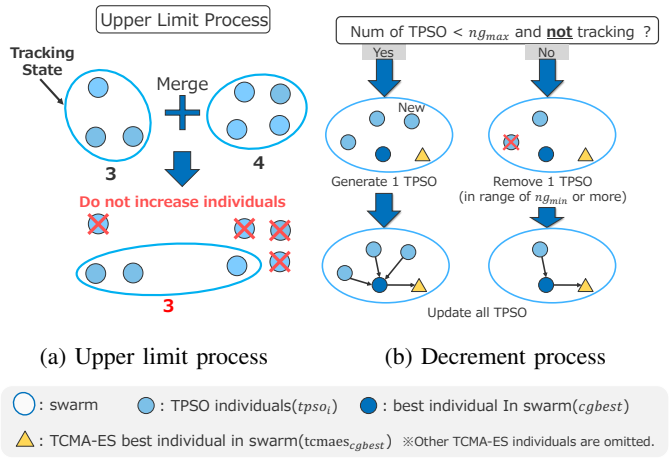
Furthermore, the upper-limit process imposes a cap on the population size after merging for swarms in the tracking state, preventing an increase in the number of individuals. For example, as illustrated in Fig. 1a, under the conventional scheme, merging a swarm of three individuals with a swarm of four individuals yields a new swarm of seven individuals (when the maximum swarm size $\geq 7$). In contrast, in NCCO(dec+ul), if the swarm with three individuals is in the tracking state, the merged swarm maintains the population size at three, corresponding to the size at the onset of tracking. Individuals to be removed are selected in ascending order of fitness.

*2) Increment:* In the increment mechanism, when the population size of a swarm is smaller than $ng_{max}$, a new individual is generated in the vicinity of $S.cgbest$ and added to the swarm. Subsequently, all TPSO individuals in the swarm are updated. Each TPSO individual is updated according to Equation (1) or Equation (2) described in Section II-B2.

In addition, through the decrement process, a swarm in the tracking state removes one TPSO individual at each iteration in the increment mechanism, as long as the number of TPSO individuals does not fall below the lower bound $ng_{min}$. The lower bound on the population size is adjusted based on the tracking failure detection described in Section II-C. Specifically, as illustrated in Fig. 1b, when the number of TPSO individuals in a swarm is smaller than $ng_{max}$ and the swarm is not in the tracking state, one TPSO individual is generated; otherwise, one TPSO individual is removed.

*3) Separate:* In the Separate mechanism, TPSO individuals that are judged to be tracking a different optimum within a swarm are split off to form new swarms. This judgment is made using the fitness value at the midpoint between $cgbest$ and the candidate individual, in the same manner as in the Merge mechanism. For each newly created swarm, $2\times$dimension TCMA-ES individuals are also generated.

*4) Generate:* In the Generate mechanism, a new swarm consisting of one TPSO individual and $2\times$dimension TCMA-ES individuals is created either at a random location in the search space or at a crossover position between the $cgbest$ of two swarms.




(a) Upper limit process (b) Decrement process

Fig. 1: Upper limit process and Decrement process

**Algorithm 1** NCCO(dec+ul)

1: $S \leftarrow$ **InitSwarm**$(\lambda)$
2: $P \leftarrow [S]$
3: **while** All environments are not finished **do**
4: **for all** $S_i \in P$ **do**
5: $S_i.tr \leftarrow$ **IsTrackingState**$(S_i)$
6: $S_i.tr, S_i.ng_{min} \leftarrow$ **FailureDetection**$(S_i)$
7: **end for**
8: **Merge(ul)**$(P, ng_{max}, TOL)$
9: **TCMA-ESUpdate**$(P)$
10: **Increment(dec)**$(P, ng_{max}, MAX\_INC, w, c_3, c_4)$
11: **Separate**$(P, TOL)$
12: **Generate**$(P)$
13: **for all** $S \in P$ **do**
14: Evaluate $S.tpso_{cgbest}$
15: **end for**
16: **end while**

### E. Algorithm

The overall procedure of NCCO(dec+ul) is summarized in Algorithm 1. At the beginning of each iteration, all swarms $S_i$ perform tracking-state detection and tracking-failure detection (Section II-C) (lines 4–7). The algorithm then sequentially executes the merge operation (Section II-D1), TCMA-ES update (Section II-B1), increment operation (Section II-D2), separation (Section II-D3), and swarm generation (Section II-D4).

## III. Experiments

### A. Experimental Setup

The performance of NCCO(dec+ul) is compared with existing dynamic optimization algorithms. The compared methods include mQSO, DMSFPSO, DSPSO, SPSO$_{+AP+AD}$, and mDE. Note that mQSO requires the number of swarms to be specified

in advance and, therefore, is not strictly comparable with the other methods.

The Moving Peaks Benchmark (MPB) [7] is employed as the objective function. The MPB parameters are based on the P3R setting in [8], with the following modifications: the dimensionality is set to $d = 2$, the correlation coefficient of peak movement is $\lambda_{mpb} = 0.5$, the environmental change frequency is $\vartheta = 25, 50$, and the total number of environments is $T = 50000/\vartheta$. This study investigates scenarios in which the number of optima $m$ and the movement severity of optima $\tilde{s}$ either remain fixed or vary over time. When fixed, the number of optima is set to $m = 5$; when varied, $m$ is incremented or decremented by one every ten environmental changes. When fixed, the movement severity is set to $\tilde{s} = 1.0$; when varied, $\tilde{s}$ is randomly sampled from the range $[1.0, 5.0]$ at each environmental change. Each problem instance is denoted as MPB$_{ff}(\vartheta)$, where the first $f$ indicates whether $m$ is fixed (fix) or variable (var), and the second $f$ indicates whether $\tilde{s}$ is fixed or variable.

### B. Evalution Metrics

Fitness-Error ($E_F$) and Distance-Error ($E_D$) are used as performance metrics. The definition of $E_F$ is given in (3). Here, $\boldsymbol{x}^{\star(t)}$ denotes the global optimum in the $t$-th environment, $T$ is the total number of environments, $\vartheta$ is the number of evaluations available before an environmental change, $c$ is the current evaluation index within each environment, and $\boldsymbol{x}^{(a)}$ represents the best individual obtained at the $a$-th evaluation. This metric computes the difference between the objective function value of the global optimum $\boldsymbol{x}^{\star(t)}$ in the $t$-th environment and that of the best individual $\boldsymbol{x}^{((t-1)\vartheta+c)}$ obtained at evaluation $((t-1)\vartheta + c)$, and then averages this difference over all evaluation points. Thus, $E_F$ evaluates the ability of an algorithm to track the global optimum.

$$E_F = \frac{1}{T\vartheta}\sum_{t=1}^{T}\sum_{c=1}^{\vartheta}\left(f^{(t)}\left(\boldsymbol{x}^{\star(t)}\right) - f^{(t)}\left(\boldsymbol{x}^{((t-1)\vartheta+c)}\right)\right) \quad (3)$$

The definition of Distance-Error ($E_D$) is given in (4). Here, $M$ denotes the total number of optima, and $N_m$ is the number of individuals belonging to the swarm closest to the $m$-th optimum. The vector $\boldsymbol{x}_m^{\star(t)}$ represents the $m$-th optimum in the $t$-th environment, and $\boldsymbol{x}_{mi}^{((t-1)\vartheta+c)}$ denotes the $i$-th individual in the swarm closest to the $m$-th optimum at evaluation $((t-1)\vartheta + c)$. The term $d(\boldsymbol{x}_m^{\star(t)} - \boldsymbol{x}_{mi}^{((t-1)\vartheta+c)})$ represents the Euclidean distance between the optimum $\boldsymbol{x}_m^{\star(t)}$ and the individual $\boldsymbol{x}_{mi}^{((t-1)\vartheta+c)}$. $E_D$ evaluates the ability to track all optima.

$$E_D = \frac{1}{T\vartheta}\sum_{t=1}^{T}\sum_{c=1}^{\vartheta}\sum_{m=1}^{M}\frac{\sum_{i=1}^{N_m} d\left(\boldsymbol{x}_m^{\star} - \boldsymbol{x}_{mi}^{((t-1)\vartheta+c)}\right)}{N_m} \quad (4)$$

### C. Parameter Settings

For all algorithms, the total number of individuals is set to 50, assuming that each optimum is tracked by 10 individuals. For NCCO(dec+ul), the exclusion distance is set to $TOL = 0.1$; the TPSO update parameters are $w = 0.5$, $c_3 = 1.0$, and $c_4 = 1.0$; the number of iterations used for tracking-state detection is $t_w = 5$; the threshold of std($\sigma$) for tracking-state detection is $\tau_{tr} = 1.0$; and the fitness threshold for tracking-failure detection is $\tau_{failure} = 10$. The parameters of the other methods are set according to their original papers.

### D. Results

Fig. 2 presents the results for MPB$_{ff}$. The horizontal axis represents the algorithms, and the vertical axis shows $E_F$ or $E_D$. Algorithms that exhibit a statistically significant difference from NCCO(dec+ul) according to the Wilcoxon signed-rank test are marked with an asterisk. NCCO(dec+ul) achieves the best performance in all cases except for $E_F$ on MPB(2, 50) (Fig. 2a1). Notably, for $E_F$ on MPB(2, 25), where the change frequency is higher (Fig. 2b1), NCCO(dec+ul) yields the best result.

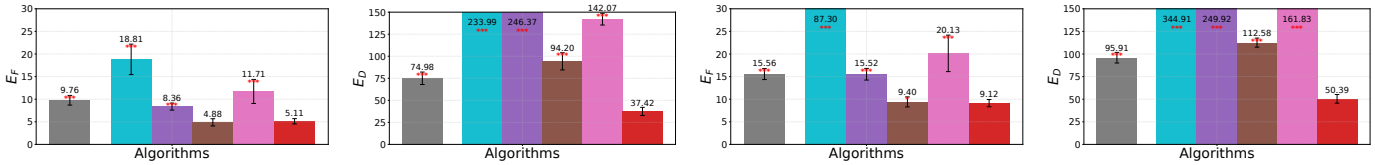
Fig. 3 shows the results for MPB$_{vf}$. Similar to MPB$_{ff}$, NCCO(dec+ul) attains the best performance in all settings except for $E_F$ at $\vartheta = 50$ (Fig. 3a1). In contrast, the performance of DSPSO, for example in terms of $E_F$ at $\vartheta = 25$, degrades substantially compared to its performance on MPB$_{ff}$.

Fig. 4 presents the results for MPB$_{fv}$. For MPB$_{fv}$, NCCO(dec+ul) consistently achieves the best performance across all parameter settings. Although SPSO$_{+\text{AP+AD}}$ performs well in terms of $E_F$ at $\vartheta = 50$ in other problem settings, its tracking performance deteriorates when the movement severity of optima becomes larger and random.

Across all experimental settings, NCCO(dec+ul) demonstrates particularly strong performance in terms of $E_D$ compared with existing dynamic optimization algorithms, indicating a superior ability to comprehensively track multiple optima. Furthermore, when comparing $\vartheta = 50$ and $\vartheta = 25$, NCCO(dec+ul) shows a more pronounced performance advantage over the other algorithms under the more rapidly changing environment ($\vartheta = 25$), suggesting its suitability for high-frequency environmental changes.

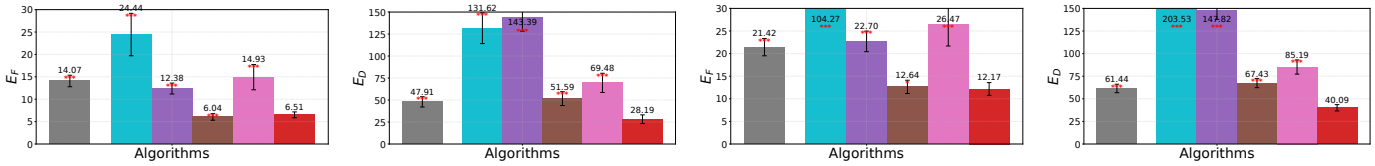
## IV. Discussion

This section discusses the reasons why DSPSO exhibits a substantial degradation in $E_F$ between MPB$_{ff}(25)$ and MPB$_{vf}(25)$, whereas the degradation of NCCO(dec+ul) is comparatively small. Fig. 5 illustrates the search behavior of DSPSO and NCCO(dec+ul) on MPB$_{vf}(25)$ for a representative random seed. The top three figures show the search process of DSPSO, and the bottom three figures show that of NCCO(dec+ul); time progresses from left to right. Green stars indicate optima, while the other points represent individuals. The largest star denotes the global optimum. The leftmost figures correspond to the moment just before the number of optima increases from one to two, at which point both algorithms successfully track the global optimum. The middle figures show the situation shortly after the number of optima has increased to two. NCCO(dec+ul) quickly identifies and tracks the new global optimum, whereas DSPSO fails to do so. This behavior arises because, when only a single optimum



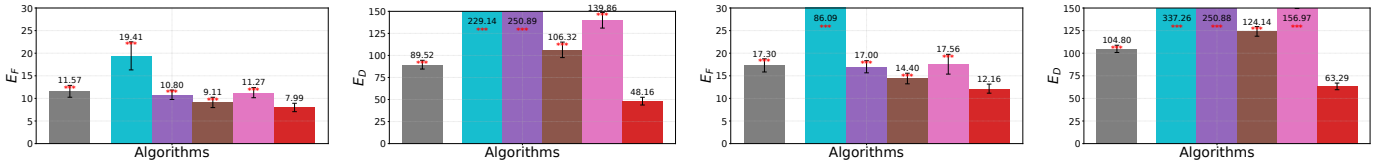

(a1) $E_F$ on MPB$_{ff}(50)$ (a2) $E_D$ on MPB$_{ff}(50)$ (b1) $E_F$ on MPB$_{ff}(25)$ (b2) $E_D$ on MPB$_{ff}(25)$

Fig. 2: Results for MPB$_{ff}$




(a1) $E_F$ on MPB$_{vf}(50)$ (a2) $E_D$ on MPB$_{vf}(50)$ (b1) $E_F$ on MPB$_{vf}(25)$ (b2) $E_D$ on MPB$_{vf}(25)$

Fig. 3: Results for MPB$_{vf}$ (functions with a varying number of optima)




(a1) $E_F$ on MPB$_{fv}(50)$ (a2) $E_D$ on MPB$_{fv}(50)$ (b1) $E_F$ on MPB$_{fv}(25)$ (b2) $E_D$ on MPB$_{fv}(25)$

Fig. 4: Results for MPB$_{fv}$ (functions with random movement severity of optima)

The results of the Wilcoxon signed-rank test indicate a significant difference between NCCO(dec+ul) and the other methods when $p < 0.05$ (*), $p < 0.01$ (**), and $p < 0.001$ (***), with the corresponding symbols placed next to the numerical values.


exists, DSPSO concentrates its search in the lower region of the space and is therefore not prepared to explore newly emerging optima. As a result, although $E_F$ can be kept relatively small in MPB$_{ff}(25)$—where the number of optima does not change and a sufficiently dispersed population is adequate—$E_F$ deteriorates markedly in MPB$_{vf}(25)$, where the number of optima increases and decreases over time. The rightmost figures show the moment immediately after the number of optima decreases from two to one; in this case, the individuals of NCCO(dec+ul) promptly move away from the location of the former optimum.

We further discuss why NCCO(dec+ul) exhibits a smaller performance degradation than the other algorithms when comparing MPB$_{fv}$ with MPB$_{ff}$. Fig. 6 shows the average number of individuals per swarm for each method on MPB$_{fv}(25)$ with a representative random seed. As observed, NCCO(dec+ul) maintains the smallest average swarm size. By aggressively reducing the number of individuals in highly dynamic environments, this method can repeatedly evaluate and update the same individuals. Consequently, even when the movement severity of optima increases in MPB$_{fv}$, NCCO(dec+ul) is able to mitigate performance degradation.

## V. Conclusion

This study addresses dynamic multimodal optimization problems in which the (i) locations, (ii) movement speeds, and (iii) number of optima change over time, with a particular focus on high-frequency changes where the environment shifts before all individuals are evaluated. We evaluate how robust NCCO(dec+ul), which incorporates a population size adjustment mechanism, is against these three types of changes by comparing it with conventional methods. NCCO(dec+ul) detects successful tracking using the step size $\sigma$ of TCMA-ES and detects tracking failure based on a decreasing trend in fitness values. By adaptively adjusting the numbers of swarms and individuals through the decrement and upper-limit mechanisms, the method achieves effective tracking of multiple continuously moving optima.

Experimental results on two-dimensional dynamic benchmark functions based on the Moving Peaks Benchmark (MPB)

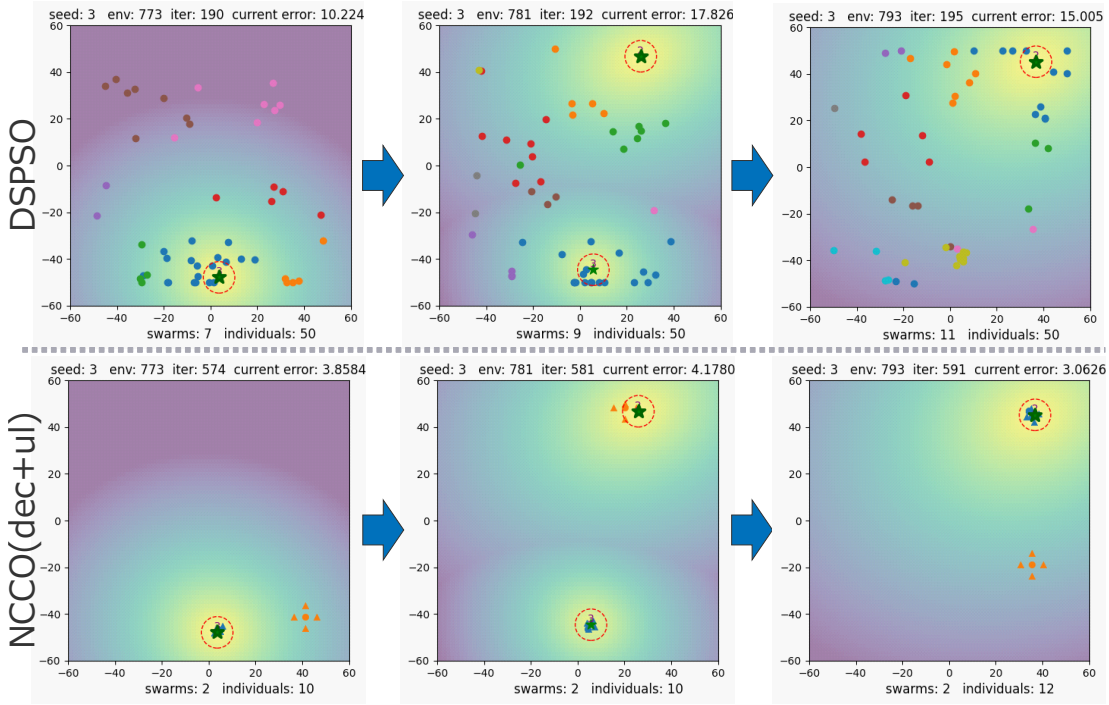



Fig. 5: Search-space behavior of DSPSO and NCCO(dec+ul) for a representative seed on $\text{MPB}_{vf}(25)$

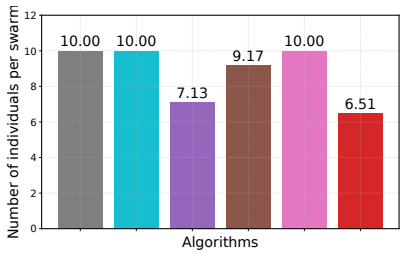



Fig. 6: Average number of individuals per swarm for each method on $\text{MPB}_{fv}(25)$

show that (1) the proposed method achieves smaller fitness error and distance error than conventional methods in most problem settings, demonstrating superior tracking performance for all optima, and (2) by flexibly adjusting the numbers of swarms and individuals, it outperforms existing methods, particularly in scenarios where the movement speed and the number of optima vary over time. These results demonstrate the effectiveness of NCCO(dec+ul) in dynamic multimodal environments with high-frequency changes.

## REFERENCES


[1] J. Branke, "Evolutionary approaches to dynamic environments - updated survey," in *GECCO Workshop on Evolutionary Algorithms for Dynamic Optimization Problems*, pp. 27–30, 2001.

[2] T. Blackwell and J. Branke, "Multiswarms, exclusion, and anti-convergence in dynamic environments," *IEEE transactions on evolutionary computation*, vol. 10, no. 4, pp. 459–472, 2006.

[3] S. Dennis and A. Engelbrecht, "Dynamic multi-swarm fractional-best particle swarm optimization for dynamic multi-modal optimization," in *IEEE Symposium Series on Computational Intelligence (SSCI2020)*, pp. 1549–1556, 2020.

[4] N. Hansen and A. Ostermeier, "Adapting arbitrary normal mutation distributions in evolution strategies: the covariance matrix adaptation," in *Proceedings of the IEEE International Conference on Evolutionary Computation (CEC96)*, pp. 312–317, 1996.

[5] J. E. Fieldsend, "Running up those hills: Multi-modal search with the niching migratory multi-swarm optimiser," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC2014)*, pp. 2593–2600, 2014.

[6] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of the International Conference on Neural Networks (ICNN95)*, vol. 4, pp. 1942–1948, 1995.

[7] J. Branke, "Memory enhanced evolutionary algorithms for changing optimization problems," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC99)*, vol. 3, pp. 1875–1882, 1999.

[8] S. A. G. van der Stockt and A. P. Engelbrecht, "Analysis of selection hyper-heuristics for population-based meta-heuristics in real-valued dynamic optimization," *Swarm and Evolutionary Computation*, vol. 43, pp. 127–146, 2018.

[9] D. Yazdani, M. N. Omidvar, J. Branke, T. T. Nguyen and X. Yao, "Scaling up dynamic optimization problems: A divide-and-conquer approach," *IEEE Transactions on Evolutionary Computation*, vol. 24, no. 1, pp. 1–15, 2019.

[10] D. Parrott and X. Li, "Locating and tracking multiple dynamic optima by a particle swarm model using speciation," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 4, pp. 440–458, 2006.

[11] D. Yazdani, D. Yazdani, D. Yazdani, M. N. Omidvar, A. H. Gandomi and X. Yao, "A species-based particle swarm optimization with adaptive population size and deactivation of species for dynamic optimization problems," *ACM Transactions on Evolutionary Learning and Optimization*, vol. 3, no. 4, pp. 1–25, 2023.

[12] S. Fujita, R. Ishizawa, H. Sato and K. Takadama, "Limitation of Adapting to Continuously Changing Optimization Problem and Its Solution in Swarm Optimization," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC)*, pp. 20–25, 2025.

[13] Anonymous, "Dynamic Multimodal Optimization with Frequent Changes Needs Population Size Adjustment for Multi-Point Search," in *The Genetic and Evolutionary Computation Conference (GECCO 2026)*, submitted.