Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Redundancy-Triggered Relocation: Decentralized Coordination of Local Searchers for Static and Dynamic Multimodal Optimization

Diego Palma

dpalmasan@gmail.com

Independent Researcher

Research Article

Keywords: multimodal optimization, dynamic optimization, niching, CMA-ES, multi-population methods, decentralized coordination

Posted Date: November 9th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-11355166/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Redundancy-Triggered Relocation: Decentralized Coordination of Local Searchers for Static and Dynamic Multimodal Optimization

Diego Palma[1*]

[1*]Independent Researcher, Concepción, Chile.

Corresponding author(s). E-mail(s): dpalmasan@gmail.com;

Abstract

Dynamic multimodal optimization asks for many optima, again after every change, and teams of local searchers waste budget when several converge to the same basin. In redundancy-triggered relocation (RTR), a searcher that perceives a fitter teammate within an exclusion radius restarts elsewhere. Each searcher perceives only the teammates within its own radius, so redundancy is judged at a small scale without a radius tuned to the problem, and no clustering or central controller is needed. We show that global exclusion caps the number of optima a team can hold, and that limited perception lifts this cap. With covariance matrix adaptation evolution strategy (CMA-ES) searchers on the organisers' instances of the CEC 2022 dynamic multimodal benchmark, RTR raises the mean peak ratio from 0.503 to 0.628, or 0.640 without two optional components, and ranks first among the dynamic baselines we ran. It trails the best published entries of the 2022–2024 competitions (0.648 to 0.686), but has the highest mean on the 10-dimensional cone landscapes. Controls with criteria fixed before their runs point to the trigger as the source of the gain, since relocating at random times at the same rate, or restarting on convergence even with an archive, is worse on 20 to 23 of 24 problems, and teammate information delayed by up to one cycle does not measurably hurt. On the CEC 2013 niching benchmark it is close to the best published results on the 5- to 20-dimensional composition functions, but behind recent methods on low-dimensional problems with many optima.

Keywords: multimodal optimization, dynamic optimization, niching, CMA-ES, multi-population methods, decentralized coordination

1 Introduction

Many design, control and model-fitting problems have several distinct good solutions. A decision maker benefits from seeing all of them, for example to choose among them on criteria not captured by the objective, or to keep an alternative ready when one becomes infeasible. Multimodal optimization formalizes this goal as locating all global optima of a function [1, 2]. When the function changes over time, as in scheduling under changing demand or tracking several targets, the optima must also be found again after each change. This setting is called dynamic multimodal optimization [3, 4].

Evolutionary and swarm algorithms are natural candidates because they maintain many solutions at once. Without intervention, however, selection drives a population into a single basin. Niching methods counteract this, through fitness sharing, crowding, clearing, species formation, clustering or repulsion [1, 2, 5–9]. In dynamic environments, multi-population methods additionally keep sub-populations apart and re-diversify them after changes [10–14].

Many of the most effective recent methods share an architecture: several local searchers, often instances of CMA-ES [15], combined with a rule that decides when two searchers are exploring the same basin and what to do about it [7, 8, 16].

This paper isolates that rule and studies the simplest decentralized version we could devise. Each searcher compares itself only with the teammates it perceives, and if a fitter one lies within an exclusion radius, the searcher is considered redundant, gives up its basin and restarts elsewhere. Otherwise it keeps its basin. Since a searcher perceives only the teammates within its own perception radius, redundancy is in effect judged within the smaller of the two radii, which keeps the test local for most searchers without a radius chosen for the problem. We call this conditional relocation redundancy-triggered relocation (RTR). The configuration evaluated in this paper, fixed before the experiments, also includes two further components, perception radii adapted by imitating teammates that hold their basins more reliably [17] and a destination chosen as the farthest of several random points from the perceived teammates. Since our ablations show that neither component is needed, we treat them as tested variants rather than as part of the rule.

The rule is close in spirit to the *exclusion* operator of multi-swarm particle swarm optimization (mQSO) [10], which re-initializes the worse of two swarms whose best points are closer than a radius. In RTR, by contrast, each searcher takes the decision from its own perception, and no populations are compared globally. This has a practical consequence, since global exclusion abandons the less fit of any two optima closer than the exclusion radius, whereas limited perception shrinks the effective radius enough for both to be held, without a smaller radius having to be set for each problem. RTR also differs from restart strategies in the reason for a restart. A searcher restarts because a fitter teammate already holds its basin, whether or not it has converged or stagnated (Section 3.7).

Decentralization matters when the searchers do not share one process. When evaluations are expensive simulations spread over many workers, or when the searchers are physical agents such as robots looking for several sources at once, keeping an up-to-date global view of all searchers is costly or impossible. A rule that needs only the positions and fitness values of nearby teammates can be served by local communication

or a neighbour query, and the searchers can decide asynchronously. Our experiments simulate the team in a single process, and they test whether the rule is competitive as an optimizer and whether it tolerates delayed information about teammates.

We ask whether this rule makes a team of local searchers competitive on standard static and dynamic multimodal benchmarks without clustering, central control or a per-problem estimate of the number of optima. The rule does rely on one default exclusion radius, derived as in mQSO from an assumed density of ten optima per domain, but this radius is the same on every problem.

Our results answer the question largely in the affirmative, within limits that we state. On the organisers' instances of the CEC 2022 dynamic multimodal suite, RTR with CMA-ES searchers ranks first among the dynamic baselines we ran (mQSO, HmSO and DE) and is ahead of three of the six published competition entries. The three best entries of the 2022–2024 competitions remain ahead in the mean, although RTR has the highest mean of all methods on the 10-dimensional cone landscapes, and without its two optional components the rule does slightly better (0.640 against 0.628). RTR also has a higher mean peak ratio than HillVallEA run from its authors' code, the two being strong on different kinds of landscape, and on the CEC 2013 niching suite it comes close to the best published results on the composition functions in 5 to 20 dimensions. It is not competitive in offline error on Moving Peaks, and it trails explicit basin-detection methods when optima are many and closely spaced. Controls show that most of the gain is lost when relocation happens at random times at the same rate, or when searchers restart on convergence or stagnation instead. Every confirmation criterion was fixed before its runs, and we also report those that were not met.

The paper makes three contributions. The first is RTR itself, a lightweight coordination rule under which a searcher relocates when it perceives a fitter teammate within a small effective radius and otherwise keeps its basin (Sections 3.1 and 3.2). The rule reads only the points and fitness values the searchers already hold and wraps the local optimizer, CMA-ES or a (1+1) evolution strategy in our experiments. The second is an analysis of exclusion, which shows that global exclusion caps the number of optima a team can hold at rest at the packing number of the optima at the exclusion radius, and that limited perception lifts this cap (Section 3.6). On four benchmark problems the cap is reached exactly (Section 5.6). The third is an evaluation that separates the components of the rule, with controls whose criteria were fixed before their runs, which distinguish the trigger from the relocation rate, from restarting on convergence and from the size of the exclusion radius, and which test the rule with stale information about teammates (Section 5.5).

The rest of the paper is organized as follows. Section 2 reviews related work, and Section 3 presents RTR, analyses it and relates it to existing designs. Sections 4 and 5 describe the experimental setup and the results on three benchmark families, Section 6 discusses the results and the limitations of the method, and Section 7 concludes.

2 Related Work

2.1 Niching for static multimodal optimization

The classical niching methods modify selection so that several sub-populations survive. *Fitness sharing* divides an individual's fitness by the number of neighbours within a sharing radius [5], *clearing* keeps only the best individuals of each niche [6], and *species conservation* copies "species seeds" into the next generation [18]. They need a niche radius that encodes how far apart the optima are, and surveys document how sensitive they are to it [1, 2, 19].

Two lines of work reduce this dependence. The first relies on local neighbourhoods: particle swarm optimization with a ring topology forms stable niches without a niching parameter [20], and differential evolution can choose mutation partners from spatial neighbourhoods [21, 22]. The second detects basins explicitly. Nearest-better clustering links each solution to its nearest better neighbour and cuts unusually long links, and NEA2 couples it with CMA-ES runs [8]. The hill-valley test [17] declares two points to lie in the same basin when no sampled point between them is worse than both, and HillVallEA clusters populations with it and runs one local optimizer per cluster [7]. Among CMA-ES-based methods, niching CMA-ES adapts a niche radius [19], and RS-CMSA repels the offspring of each sub-population from the centres of fitter sub-populations and from archived optima, through taboo regions whose size adapts. It won the 2016 GECCO and CEC niching competitions [16]. A further line turns multimodal problems into bi-objective ones whose Pareto set contains the optima [9]. The CEC 2013 niching benchmark [23] became the standard test bed for this literature, and its competition data, including the solutions submitted by entrants, are public.

2.2 Multiple populations for dynamic optimization

Dynamic optimization has been surveyed several times [24–27]. Multi-population methods are among its most successful designs [10–14], and most of them have been evaluated on the Moving Peaks benchmark [28], a landscape of cone-shaped peaks whose positions, heights and widths change at fixed intervals. Its offline error, the running average of the error of the best solution found since the last change, became the standard measure [24, 27].

mQSO [10] runs one swarm per expected peak and mixes neutral and "quantum" particles to stay diverse. Its two operators, *exclusion* and *anti-convergence*, are the closest relatives of our rule. Exclusion re-initializes the worse of two swarms whose best points are closer than a radius, and anti-convergence re-initializes the worst swarm when all have converged. HmSO [11] lets a parent swarm explore and spawns child swarms around promising points. A child swarm hibernates when it stops improving and wakes up after a change. Clustering-based methods form sub-populations by clustering at run time and so avoid fixing their number [12], and later multi-swarm methods such as FTMPSO [13] reduced the offline error on Moving Peaks substantially.

2.3 Dynamic multimodal optimization

Dynamic multimodal optimization combines both goals, since the algorithm should find all global optima in every environment. Early methods adapted static niching algorithms to changing landscapes, for example a clonal selection algorithm [29] and an evolutionary mean-shift algorithm [30]. The PopDMMO framework [3] equips population-based algorithms with mechanisms for both multimodality and change, and EvoDCMMO extends the setting to constrained problems [31]. Ahrari et al. proposed an adaptive multilevel prediction of where the optima move after a change (AMLP) [32], which they combined with the RS-CMSA family of niching methods [16, 33], and a parametric benchmark generator for the problem [34]. More recent methods include a niching-based brain storm optimization with two archives [35], an adaptive multipopulation method [36], and DAEA, which uses a density model to guide initialization, search and restarts, with prediction and memory for the changes [37].

The CEC 2022 competition on seeking multiple optima in dynamic environments introduced a suite of 24 problems and a peak-ratio measure summed over environments [4]. Its problems combine eight base functions (four distance-based cone landscapes and four composition functions) with eight change modes, in 5 and 10 dimensions. The same suite and protocol were used in the 2023 and 2024 editions. AMLP-RS-CMSA-ESII won the 2022 edition, ahead of AMPSO and an HmSO-based entry (results distributed with [4]). CMA-ES-DS won the 2023 edition, ahead of mEGA [38], and DAEA was the 2024 entry [37, 39]. Section 5.1 compares RTR with the published results of all three editions.

3 Method

3.1 Overview

A team of N agents searches a box-constrained domain. Agent i holds a point x_i with fitness $f_i = f(x_i)$ (maximization) and runs a local optimizer, its *searcher*, which proposes new points around x_i and replaces x_i whenever it finds a better one.

We use an independent $(\mu/\mu_w, \lambda)$-CMA-ES per agent with standard settings [15, 40]. Its mean is initialized at x_i, it has step size σ_0, and it is *converged* when its step size times the largest standard deviation falls below a tolerance. A converged searcher spends no evaluations until it is restarted, which happens when its agent relocates, when the environment changes, or, if every searcher of the team has converged, when all of them are restarted at their current points with step size σ_0 so that the budget is not left unused. For Moving Peaks, whose peaks are cones, we also use a (1+1) evolution strategy, (1+1)-ES [41], which adds Gaussian noise to x_i, keeps the result if it is better, and multiplies the step size by 1.5 after a success and by 0.9 after a failure.

Time advances in rounds. In each round, every active searcher performs one iteration (one generation of λ evaluations for CMA-ES, one evaluation for the (1+1)-ES). Agents make their coordination decisions in *cycles* of c rounds, and an agent that has just relocated may decide early, after a probation of p rounds (Section 3.2). Fig. 1 illustrates the rule.

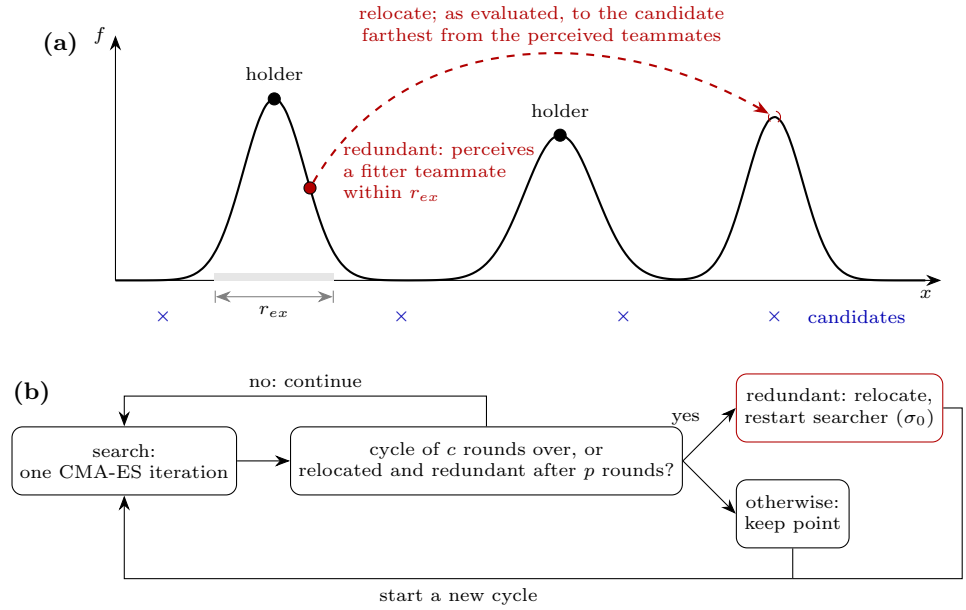


Fig. 1 The redundancy-triggered relocation rule. (a) Two agents climb the same peak; the less fit one perceives its fitter teammate within the exclusion radius r_{ex}, so it is redundant. It moves elsewhere and restarts its searcher there, while the holders keep refining their peaks; as evaluated, it moves to the candidate point (four of k shown) farthest from the teammates it perceives. (b) The decision each agent makes at the end of its cycle, or early when it is redundant after a relocation.

3.2 Redundancy and relocation

Agent i perceives the teammates within its perception radius v_i. It is **redundant** when it perceives a fitter teammate within the exclusion radius r_{ex}:

$$\text{redundant}_i \iff \exists\, j \neq i : \ \|x_j - x_i\| < r_{ex}, \quad f_j > f_i, \quad \|x_j - x_i\| \leq v_i. \tag{1}$$

Ties in fitness are broken by agent index, so that of two agents on the same point exactly one is redundant.

At the end of its cycle, an agent that is redundant **relocates**, that is, it moves to a new point and restarts its searcher there with step size σ_0, while any other agent keeps its point and continues its searcher. Relocating the redundant agent recovers the evaluations it would otherwise spend refining a basin that a fitter teammate already holds.

The rule itself does not prescribe where a relocating agent goes. In the configuration we evaluate, the agent draws k points uniformly in the domain and moves to the one farthest from its nearest perceived teammate, with the aim of spreading restarts into unexplored regions. Since a uniformly random point does as well (Section 5.5), we regard this choice as one of the tested variants. An agent whose current cycle began with a relocation may also decide early, as it relocates again at once if it is still

Algorithm 1 Redundancy-triggered relocation (RTR), in the configuration evaluated

Require: team size N, cycle c, probation p, exclusion radius r_{ex}, number of candidate points k, searcher (CMA-ES with σ_0 and σ_{change}), radius range $[v_{\min}, v_{\max}]$

place the N agents uniformly at random; start their searchers; draw the radii v_i
while the evaluation budget is not used up **do**
    if the environment changed **then**
        re-evaluate all agents; restart their searchers with σ_{change}
    end if
    for every agent i: redundant$_i \leftarrow$ a fitter teammate or archived point j exists with $\|x_j - x_i\| < r_{ex}$ and $\|x_j - x_i\| \leq v_i$
    for all agents i whose cycle ended, or whose cycle began with a relocation, has lasted p rounds, and who are redundant **do**
        (imitation variant) update the hold score h_i; copy the radius of a random teammate j if $h_j > h_i$
        if redundant$_i$ **then**
            move to a new point (as evaluated: the farthest of k random points from the perceived teammates); restart the searcher with σ_0
        end if
        start a new cycle of c rounds
    end for
    if every searcher has converged, restart all of them at their points with σ_0
    run one iteration of every active searcher; each agent keeps its best point
    (static problems) retire the point of every converged searcher to the archive; move its agent to the farthest of k random points from all teammates and archived points, restart its searcher with σ_0, and end its cycle
end while
return the agents' points (and the archive)

redundant after p rounds (the probation period) instead of waiting for the end of its cycle. The probation lets a newly relocated agent climb before it is judged, and the early decision quickly undoes a relocation into a basin that is already held. Algorithm 1 summarizes RTR in the configuration we evaluate.

3.3 Perception radius

With full perception, an agent on a peak next to a fitter peak is redundant whenever the two lie within r_{ex}, even if they are distinct optima. Limiting each agent's perception to a radius v_i prevents such false verdicts, because redundancy is then judged within $\min(r_{ex}, v_i)$, although too small a radius lets duplicates go unnoticed. The effective radius should therefore be small compared with the distance between optima (Section 3.6). A single small radius shared by all agents achieves this, but it has to be chosen for the problem. Radii drawn over a wide range, log-uniformly in $[v_{\min}, v_{\max}]$, keep it small for a sizeable share of the agents across scales without that choice. We do not claim that radii which differ between agents find more optima than

a well-chosen common radius, and on the CEC 2022 suite a common radius of 5 units indeed performs as well (Section 5.5).

The configuration we evaluate, fixed before the experiments, also adapts the radii by imitation. The ablation shows no advantage of this adaptation, so we do not count it as part of the rule, but we describe it here because every confirmation run used it. At the end of each cycle, agent i records whether it *held* its basin, meaning that its fitness is positive and either no fitter teammate lies within r_{ex} or the nearest one, j, is separated from it by a valley, $f\big((x_i + x_j)/2\big) < \min(f_i, f_j)$ (a one-point hill-valley test [7, 17]). It keeps an exponential average $h_i \leftarrow m\, h_i + (1 - m)\, \mathbf{1}[\text{held}]$ and compares it with the score h_j of a random teammate. If $h_j > h_i$, it copies that teammate's radius with multiplicative noise, $v_i \leftarrow \text{clip}(v_j e^{\varepsilon}, v_{\min}, v_{\max})$, $\varepsilon \sim \mathcal{N}(0, s^2)$.

The positivity condition was introduced for Moving Peaks, whose fitness is non-negative. On landscapes whose fitness is never positive, including the composition functions of both CEC suites, no agent ever holds, imitation is inactive, and the agents keep heterogeneous initial radii.

3.4 Environment changes

When the environment changes, every agent re-evaluates its point and restarts its searcher around it with a small step size σ_{change}, so that it re-climbs the moved peak. Changes are handled according to each benchmark's protocol, announced to the algorithm on the CEC 2022 suite and detected on Moving Peaks by re-evaluating the best point at every round.

3.5 Archive for static problems

Static problems may have more global optima than there are agents (up to 216 in our experiments), so for them we add an archive. When a searcher converges, its agent *retires* the point to the archive. Before adding the point, the agent checks the nearest archived point within r_{ex} with the hill-valley test. If the two lie in the same basin, only the better of them is kept. The agent then moves to the farthest of k uniform points from all teammates and archived points, not only the perceived ones, and restarts its searcher there with step size σ_0. It decides at the next round, as if its cycle had ended, and starts a new cycle. Archived points act as teammates that never move, so an agent that perceives a fitter archived point within r_{ex} is redundant, and the reported solution set is the union of the agents' points and the archive. The archive is independent of RTR. A team that never relocates by choice but retires converged searchers is a restart CMA-ES with a shared archive and exclusion-based restarts, and it serves as the control on static problems.

3.6 Analysis

The main result of this section, Proposition 1, characterizes the configurations that the rule leaves alone and shows what global exclusion loses. Two elementary remarks then state how long the rule lets a duplicate run and what it gains over relocating at random times, and they motivate the measurements of Section 5.6. Throughout, the landscape is fixed between changes, the team has no archive, and "fitter" is the strict

order given by fitness, with ties broken by index. An optimum o is *held* by agent i when $\|x_i - o\| < \epsilon$, and we assume that the global optima are pairwise more than 2ϵ apart, so that an agent holds at most one optimum. *Full perception* means $v_i \geq r$ for every agent, where r is the exclusion radius. T denotes the number of rounds between two environment changes. We call the team *at rest* when no agent is redundant. A team at rest never relocates by choice, so its rest states are the configurations the rule converges to between changes.

Proposition 1 (Exclusion bound) *Let O be the set of global optima, and let $P_s(O)$ be the size of the largest subset of O whose points are pairwise more than s apart.*

(a) With full perception and exclusion radius r, the agents of a team at rest are pairwise at least r apart, and at most $P_{r-2\epsilon}(O)$ optima are held at rest.
(b) With limited perception, two agents at distance $d < r_{ex}$ can both be at rest only if the less fit one has a perception radius $v_i < d$.

Proof. Let two agents be at distance $d < r$. One of them is strictly fitter, and the other is redundant if it perceives it, that is, if its radius is at least d. With full perception this always holds, so at rest every two agents are at least r apart. With limited perception it fails exactly when the less fit agent's radius is below d, which gives (b). For the bound in (a), let $o \neq o'$ be held at rest by agents i and j. Then $i \neq j$, because an agent holds at most one optimum. By the triangle inequality, $\|o - o'\| \geq \|x_i - x_j\| - \|x_i - o\| - \|x_j - o'\| > r - 2\epsilon$. The held optima are thus pairwise more than $r - 2\epsilon$ apart, and there are at most $P_{r-2\epsilon}(O)$ of them. □

Global exclusion therefore caps the number of optima a team can hold at rest, whatever the team size and budget, at the number of optima that can be picked pairwise more than $r - 2\epsilon$ apart, which is close to r when ϵ is small. When two distinct optima lie closer than that, one of them is necessarily discarded. The same argument bounds the number of optima held by the best points of the swarms of mQSO, whose exclusion compares exactly these points [10]. Limited perception lifts the cap for every pair whose less fit holder perceives less than its distance to the other holder. If the radii are drawn log-uniformly in $[v_{\min}, v_{\max}]$, as they are initially, the expected fraction of agents with a radius below d is $\ln(d/v_{\min})/\ln(v_{\max}/v_{\min})$ for $v_{\min} \leq d \leq v_{\max}$.

The rest states also explain how the team divides its work. Suppose the landscape has K basins whose maxima are at least r_{ex} apart, perception is full, and every searcher has converged to the maximum of its basin. Two agents in the same basin are then at distance $0 < r_{ex}$, so at most one agent per basin is at rest, and at most K agents in all. If $N > K$, the other $N - K$ agents are redundant and relocate (Remark 1). The team thus splits into *holders* and *scouts*, and the number of scouts adapts to the number of basins. Neither the number of sub-populations, as in mQSO [10], nor the number of free explorers, as in mQSO's anti-convergence, is a parameter.

Remark 1 (Bounded redundancy) Let c be the cycle length and p the probation period.

(a) An agent that is redundant at every round from round t on relocates no later than round $t + c$.
(b) If its current cycle began with a relocation at round s, it relocates no later than round $\max(t, s + p)$.
(c) Without relocation, the coordination layer never ends a redundancy episode: it ends only when the searchers' own moves or an environment change take the agent out of the exclusion region of every fitter teammate it perceives.

Proof. (a) Every agent decides at the end of each cycle, so its decisions are at most c rounds apart, and an agent that is redundant when it decides relocates. (b) This is the early decision of Section 3.2, by which such an agent decides at every round from round $s + p$ on at which it is redundant. (c) Without relocation, no decision moves an agent. □

A redundancy episode thus costs at most $c\lambda$ evaluations, and a duplicate gets a fresh start at least $\lfloor T/c \rfloor$ times per environment. Without the rule it gets none, and since a change restarts every searcher around its own point (Section 3.4), two searchers on one peak may follow it together across changes.

Remark 2 (Trigger versus random relocation) Compare RTR with a team that relocates at random times.

(a) Under RTR, an agent that is not redundant is never relocated by the rule.
(b) Suppose instead that each agent relocates at each decision with probability q, independently of redundancy and of the other decisions. Then an agent that holds an optimum is relocated at none of its next m decisions with probability $(1 - q)^m$, and so it is displaced from the optimum by one of them with probability $1 - (1 - q)^m$.

Proof. (a) is the rule. (b) follows from the independence of the m decisions. □

An agent that is not relocated makes at least $m = \lfloor T/c \rfloor$ decisions per environment, so at RTR's own rate q, random relocation displaces a holder before the environment ends with probability at least $1 - (1 - q)^{\lfloor T/c \rfloor}$, whereas RTR relocates a holder only when it is redundant. Matching the relocation rate of RTR is therefore not enough. The remark is about displacement. The final loss of optima is smaller when displaced optima are found again and larger when other causes add to it, so the displacement probability estimates the loss without bounding it.

As for cost, the redundancy test uses only the points and fitness values the agents already hold, so it costs $O(N^2 D)$ arithmetic per round and no function evaluations. Radius imitation costs at most one evaluation per agent per cycle (the hill-valley midpoint), a fraction of at most $1/(c\lambda)$ of the evaluations, and the archive adds at most one evaluation per retirement (the hill-valley midpoint), besides the evaluation of the new point that every relocation needs.

RTR guarantees neither coverage nor a preference for global optima. Relocation targets are random, so a narrow basin may never be visited, and a lower basin whose holder perceives no fitter teammate is held as firmly as a global one. Moreover, agents

only read each other's points and fitness values and exchange neither search distributions nor messages, so the rule coordinates the searchers without making them cooperate in the sense of sharing information about the landscape.

3.7 Relation to existing methods

None of the ingredients of RTR is new on its own. The exclusion radius is that of mQSO [10], and keeping new searches away from known solutions is common to restart-based niching [16, 42]. What sets it apart is who decides that a searcher is redundant, from which information, and when (Table 1).

In mQSO, whenever two swarm attractors are closer than the exclusion radius, the worse swarm is re-initialized, so two optima much closer than this radius cannot both be held at rest (Proposition 1). In RTR, the test is run by the searcher that would be displaced, against the teammates it perceives, so RTR can be read as exclusion with a radius $\min(r_{ex}, v_i)$ that belongs to each agent. Both optima of a close pair can then be held without changing r_{ex} and without knowing how far apart the optima are, and no step compares all pairs. Section 5.5 tests this by giving every agent full perception, and it also shows the limit of the argument, since one smaller radius shared by all agents does almost as well.

Among repelling and restart methods, RS-CMSA is the closest design. It repels its sub-populations from the centres of fitter sub-populations and from archived optima [16]. This is the same comparison with fitter neighbours, but it acts inside the optimizer, by rejecting offspring in taboo regions, against every fitter sub-population. RTR leaves the optimizer unchanged and acts only through restarts. Restart strategies for CMA-ES, from IPOP-CMA-ES [43] to repelling restarts [42], start a new search when the current one has converged or stagnated, and the recent ones steer it away from the optima already found. RTR restarts a searcher for a different reason, and earlier, namely as soon as a fitter teammate holds its basin, whether or not it has converged (Remark 1). Section 5.5 separates this trigger from the number of restarts, by relocating at random times at RTR's rate, and from restarting on convergence or stagnation.

Clustering methods such as NEA2 and HillVallEA decide which points share a basin by clustering a sample, with nearest-better clustering or with hill-valley tests that cost extra evaluations [7, 8]. These tests are more precise than a distance threshold, and on problems with many closely spaced optima the methods built on them find far more optima than RTR (Section 5.3). RTR makes the cheaper choice, as its test uses only distances and fitness values that are already known.

4 Experimental Setup

4.1 Benchmarks and performance measures

CEC 2022 dynamic multimodal suite [4].

The suite has 24 problems built from eight base functions: F1–F4, distance-based cone landscapes with 4 global peaks (F1 also has 4 local ones), and F5–F8, composition functions with 6–8 global optima. P1–P8 apply change mode C1 to F1–F8 in 5-D,

Table 1 How related methods handle overlapping searchers.

method	overlap decided by	redundant searcher	number of optima needed	local optimizer
fitness sharing / clearing [5, 6]	population-wide niche radius	loses fitness / is cleared	via radius	GA operators
mQSO [10]	global pairwise comparison of swarm bests	worse swarm re-initialized uniformly	yes (number of swarms)	PSO
HmSO [11]	parent–child distance test	child swarm removed or hibernated	no	PSO
HillVallEA [7]	hill-valley clustering of a sample	merged into one cluster	no	estimation-of-distribution algorithm
RS-CMSA [16]	distance to fitter sub-populations and archived optima (adaptive taboo radius)	offspring repelled from taboo regions	no	CMSA-ES
RTR (ours)	**each searcher, from the teammates within its perception radius**	**restarts elsewhere**	**no**	**any restartable optimizer (CMA-ES, (1+1)-ES tested)**

P9–P16 apply change modes C1–C8 to F8 in 5-D, and P17–P24 repeat P1–P8 in 10-D. Each run has 60 environments of $5000 \cdot D$ evaluations, and changes are announced to the algorithm. At the end of each environment the algorithm reports its current solutions, and a global optimum counts as found when a reported solution lies within distance 0.05 of it and within an accuracy $\epsilon \in \{10^{-3}, 10^{-4}, 10^{-5}\}$ of its value. The *peak ratio* is the number of optima found, summed over environments, divided by the total number of optima. We report $\epsilon = 10^{-4}$.

P8 and P9. P8 and P9 are the same problem (F8 with C1 in 5-D) and, since the organisers seed every problem's dynamics identically, the same instance, so every method, including the published entries, has identical results on both. We keep both to follow the protocol, so P8/P9 counts twice in every per-problem tally and mean. Counting it once changes no conclusion of this paper. Section 5.2 lists the affected numbers.

CEC 2013 niching suite [23].

The suite has 20 static problems in 1–20 dimensions with 1–216 global optima, run under the official evaluation budgets (50 000 to 400 000). The measure is the official peak ratio, at accuracies 10^{-1} to 10^{-5}, of the final solution set.

Moving Peaks [28].

We use scenario 2 (5-D, 10 cone peaks, heights in [30, 70] starting at 50, widths in [1, 12], shift length 1, a change every 5000 evaluations, 100 changes) and nine standard variations, with 20, 50 or 100 peaks, a change every 1000, 2500 or 10 000 evaluations, a

shift length of 2 or 5, or 10 dimensions. The measures are the offline error [27] and the share of the currently visible peaks that have an agent within distance 1 (*coverage*).

Implementations.

All benchmarks and baselines were implemented in Python. The CEC 2013 implementation matches the organisers' reference code to within $2 \cdot 10^{-11}$ (relative) on 2000 random points per problem. The CEC 2022 implementation reproduces the organisers' rules, constants and data, but its random stream, which moves peaks and rotates composition components, comes from a different generator, so it draws its own instances from the same generator. The organisers' own instances, exported from their reference implementation, can be replayed exactly: our implementation reproduces the optimum values of all 60 environments of every problem. All head-to-head comparisons run on identical instances.

4.2 Compared methods

- **No relocation (ablation).** The same agents and searchers that never relocate by choice, keeping the archive on CEC 2013. This is the primary control: it isolates the effect of RTR.
- **mQSO** [10]: 10 swarms of 5 neutral and 5 quantum particles, with exclusion and anti-convergence radii derived from the expected number of peaks [10]. Our implementation reaches an offline error of 1.52 on Moving Peaks scenario 2 (30 runs, against 1.75 published).
- **HmSO** [11], by default with a parent swarm of 5 particles, child swarms of 10, and the published radii. Our implementation reaches an offline error of 1.39 on scenario 2 (30 runs, against 1.46 published). On the CEC 2022 suite, the published radii put child swarms to sleep about 0.1 away from a peak, far coarser than the required accuracy. We therefore also report *tuned HmSO*, with its radii tuned per function family on development runs of one cone problem (P2) and one composition problem (P5), both in 5-D, which gave a child-swarm and exclusion radius of 10 units and a convergence radius of 10^{-4} units for the cone landscapes, and the published radii with a convergence radius of 10^{-5} units for the composition functions. Tuned HmSO was added after the pre-specified confirmation and is not part of its criteria.
- **DE/rand/1/bin** with $N = 100$, $F = 0.5$, $CR = 0.5$: the baseline distributed with the CEC 2022 suite.
- **HillVallEA** [7], a leading niching method for static problems, run through the authors' C++ code with its default settings. It has no change handling, so each environment is a separate run with that environment's budget, either from scratch (*restart*) or with the previous environment's elites added to its first sample (*seeded*). Its own termination can end a run before the environment's budget is used, earlier in the seeded variant. We did not change it, and the remaining evaluations re-evaluate the best point. On CEC 2013, our build matches the authors' submitted solution files (0.892 at accuracy 10^{-4}).
- **Published results**, used for context only and never in statistical tests. These are the entries of the 2022–2024 competitions on the CEC 2022 suite [4, 38, 39], the 2013 competition entrants on CEC 2013 [23] together with RS-CMSA-ESII

and HillVallEA19 from later competitions, which we re-scored from their submitted solution files with our implementation of the official measure, and the published offline errors on Moving Peaks [11, 13].

4.3 Parameter settings

All RTR settings were fixed on Moving Peaks development instances before any confirmation run (Table 2). The searcher was chosen per benchmark family during development: CMA-ES for the CEC suites (composition functions), and the (1+1)-ES for Moving Peaks (cone peaks). Distances are expressed in units of 1/100 of the domain's side length. With $c = 40$, radius imitation uses at most $1/(c\lambda)$ of the evaluations (Section 3.6): below 0.7% for any $\lambda \geq 4$, and 0.3% in 5-D, where $\lambda = 8$.

Table 2 Settings of RTR (identical in every experiment unless stated).

component	setting	value
team	agents N	20
coordination	cycle c; probation p	40 rounds; 5 rounds
	exclusion radius r_{ex}	side length / $(2 \cdot 10^{1/D})$: half the side of a hypercube holding one tenth of the domain's volume (10 expected optima) [10]
	candidate destinations k	8
perception	initial radii	log-uniform in $[v_{\min}, v_{\max}] = [0.5, 100]$ units
	imitation noise s; memory m	0.2; 0.7
CMA-ES searcher	population λ; weights	$4 + \lfloor 3 \ln D \rfloor$; standard [40]
	σ_0; σ_{change}; tolerance	10; 2; 10^{-10} units
(1+1)-ES (Moving Peaks)	initial step; step after relocation	2; 5 units
	step update on success / failure; bounds	$\times 1.5$ / $\times 0.9$; $[10^{-9}, 20]$ units
archive (CEC 2013 only)	retirement	on searcher convergence

4.4 Protocol and statistics

The success criteria of each confirmation (Table 8) were written down before its runs, following the practice of pre-registration [44]. Development used separate random seeds. We ran 30 independent runs per problem on the CEC 2022 suite (15 in the pre-specified confirmation, extended to 30 afterwards), 50 on CEC 2013, as in the official protocol, and 30 on Moving Peaks.

Per-problem comparisons use two-sided Mann–Whitney U tests [45] on per-run scores, with Holm correction across the problems of a suite [46] at $\alpha = 0.05$. The direction of a significant difference is that of the rank statistic (A_{12} above or below 0.5), which the test assesses. On problems where most runs reach the maximum, a single outlying run can place the difference of means on the other side. Alongside the tests we report the Vargha–Delaney effect size A_{12} [47], the probability that a run of the first method beats a run of the second, and 95% bootstrap confidence intervals of the mean difference.

Multi-method comparisons use Friedman mean ranks over problems, with the Friedman test and Holm-corrected post-hoc comparisons of every method against RTR [48–50].

With 10 runs per method and Holm's correction over 24 problems, the ablations declare a difference only when it is large, since the first and strictest comparison needs $A_{12} \geq 0.90$ (or ≤ 0.10) and even the last needs $A_{12} \geq 0.77$ (or ≤ 0.23), without ties. A variant without significant differences is therefore not shown to be equivalent to RTR, as small differences may go undetected. The sensitivity study (5 runs per problem) is descriptive and uses no tests.

5 Results

5.1 CEC 2022 suite: the organisers' instances

The main results on the CEC 2022 suite use the organisers' own instances, on which the competition results were obtained, with 30 runs per problem. The pre-specified confirmation ran earlier on our own draws of the instances. Section 5.2 reports it, together with the comparison with HillVallEA and the robustness to the draw. Before the runs on the organisers' instances, we registered for them the two criteria of the confirmation, with tuned HmSO in place of default HmSO. P8 and P9 are the same instance and count twice, as in the protocol (Section 4.1).

Compared with the same searchers without relocation, RTR raises the mean peak ratio at accuracy 10^{-4} from 0.503 to 0.628 (Fig. 2). It is significantly better on 20 of 24 problems, with $A_{12} \geq 0.85$ on 17, but significantly worse on two, so the registered criterion (better on at least 16, worse on none) is *not met*. Both losses are small. On P2, RTR's mean is higher (by 0.007, 95% CI [−0.002, +0.024]), but the ablation is perfect in 29 of 30 runs while RTR misses an occasional optimum, so the ranks favour the ablation ($A_{12} = 0.37$). On P22, RTR trails by 0.020 (95% CI [−0.037, −0.002], $A_{12} = 0.25$). The gains are largest where independent searchers crowd onto the same peaks: on the cone landscapes with a local peak (0.75 against 0.44 on P1 and 0.66 against 0.35 on P17), on the 10-D cone landscapes (P19, P20: +0.20 and +0.29), and on every composition problem except P22 (+0.02 to +0.19). Table S1 in Online Resource 1 gives every difference with its confidence interval.

Among the dynamic baselines we ran on these instances, RTR has the best Friedman mean rank, 1.04, against 2.33 for mQSO, 3.08 for tuned HmSO and 3.54 for DE (Friedman test $p < 10^{-10}$, Holm-adjusted post-hoc $p \leq 0.0006$ against each), and it is better than tuned HmSO on 23 of 24 problems and worse on one (P1), so the second registered criterion is met. Tuned HmSO is strong on the 5-D cone landscapes (0.925 against 0.937), where its radii were tuned, but its radii do not transfer to 10 dimensions (0.012 on P17–P20).

Table 3 compares RTR with the published results of the three editions of the competition, which used the same suite and protocol. Three entries are ahead of RTR: DAEA (2024, 0.686), AMLP-RS-CMSA-ESII (2022, 0.654) and CMA-ES-DS (2023, 0.648). RTR (0.628) is ahead of the other three: AMPSO (0.514), the HmSO entry (0.346) and mEGA (0.325). Per problem, RTR has the higher mean than DAEA on 5 problems, all in 10-D (P17, P19, P20, P23, P24), than CMA-ES-DS on 10 and than

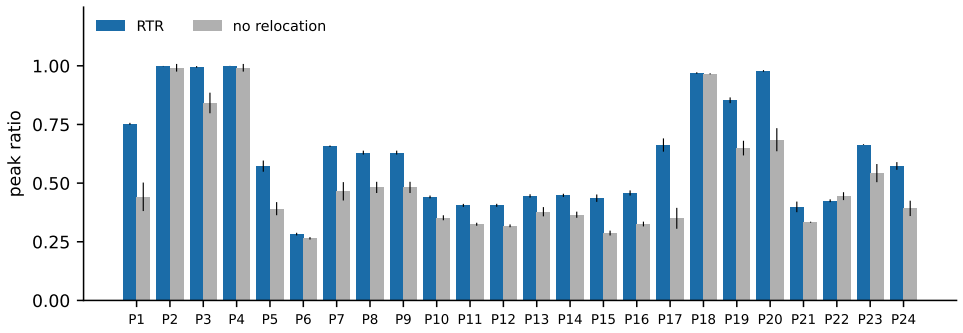


Fig. 2 Mean peak ratio at accuracy 10^{-4} on the 24 problems of the CEC 2022 suite, organisers' instances (30 runs, 95% confidence intervals): RTR (blue) and the same CMA-ES searchers without relocation (grey).

AMLP-RS-CMSA-ESII on 14. On the 10-D cone landscapes its group mean (0.866) is the highest of all methods, whereas the leading entries are clearly stronger on F8 under the eight change modes (P9–P16: 0.459 for RTR, against 0.548 to 0.605). The published results give only per-problem means, so these comparisons are descriptive, not statistical. The organisers' repository contains no entrant code, and we did not run the entrants ourselves.

All the results above are for the configuration fixed before the experiments, which includes radius imitation and the far-point destination. The rule as we recommend it, with uniform destinations and the initial log-uniform radii kept fixed, was run afterwards on the same instances (30 runs per problem). We expected, before the runs, a mean within 0.02 of the evaluated configuration and significant differences on at most 2 problems. It reaches 0.640 against 0.628 and is significantly better on three problems (P5, P10, P13) and worse on none, so the expectation is not met, in the rule's favour. Against the searchers without relocation it is better on 21 problems and worse on the same two, P2 and P22, and its Friedman rank among the dynamic baselines is again 1.04. It is still behind the three leading entries (Table 3), with a higher mean than AMLP-RS-CMSA-ESII on 15 problems, CMA-ES-DS on 11 and DAEA on 6.

5.2 CEC 2022 suite: our draws, confirmation and robustness

The pre-specified confirmation took place on our own draws of the instances, where the two criteria were met at the registered 15 runs per problem: RTR was better than the searchers without relocation on 20 of 24 problems and worse on none, and it had the best Friedman mean rank, 1.08, among RTR, default HmSO, mQSO and DE, better than HmSO on all 24 problems. With 30 runs, the same two problems as on the organisers' instances, P2 and P22, become significantly worse, with the same pattern: RTR raises the mean from 0.505 to 0.645 and is better on 20 problems, worse on 2. Table S3 and Fig. S1 in Online Resource 1 give the per-problem results on this draw. Among the five methods we ran on these draws (Table S2 in Online Resource 1), RTR again has the best Friedman mean rank, 1.12, against 2.54 for mQSO, 3.12 for default

Table 3 Mean peak ratio at accuracy 10^{-4} on the organisers' instances of the CEC 2022 suite, by problem group: RTR and the baselines run by us (30 runs per problem), and the published results of the 2022–2024 editions of the competition [4, 38, 39], sorted by mean.

method	P1–P4 cones 5-D	P5–P8 comp. 5-D	P9–P16 F8, C1–C8	P17–P20 cones 10-D	P21–P24 comp. 10-D	all 24
RTR (evaluated configuration)	0.937	0.536	0.459	0.866	0.515	0.628
RTR as recommended[†]	0.937	0.558	0.473	**0.875**	0.523	0.640
no relocation	0.817	0.401	0.354	0.663	0.428	0.503
mQSO	0.689	0.342	0.063	0.148	0.192	0.249
HmSO (tuned)	0.925	0.225	0.032	0.012	0.061	0.214
DE	0.101	0.006	0.001	0.165	0.024	0.050
DAEA (2024)	0.938	**0.586**	**0.605**	0.852	0.530	**0.686**
AMLP-RS-CMSA-ESII (2022 winner)	0.904	0.562	0.548	0.817	**0.545**	0.654
CMA-ES-DS (2023 winner)	**0.942**	0.560	0.561	0.772	0.490	0.648
AMPSO (2022 runner-up)	0.941	0.469	0.281	0.740	0.375	0.514
HmSO entry (2022)	0.898	0.420	0.126	0.323	0.182	0.346
mEGA (2023 runner-up)	0.799	0.227	0.020	0.634	0.247	0.325

[†] Uniform destinations and fixed log-uniform perception radii, run after the confirmation.

HmSO, 3.79 for tuned HmSO and 4.42 for DE (30 runs per problem and 15 for default HmSO, with a Friedman test $p < 10^{-11}$ and Holm-adjusted post-hoc $p \leq 0.002$ against each).

To check that the results do not depend on one draw, we also ran RTR on four further draws of the instances (5 runs per problem), where its mean peak ratio stayed between 0.626 and 0.633, and above that of the searchers without relocation (0.506–0.509) on every draw. The organisers' instances (0.628) fall within this range, and the draw used for the confirmation (0.645) at its upper end. The ablations of Section 5.5 ran on the confirmation draw.

We also compared RTR with HillVallEA on the same draw (30 runs per problem), under three criteria fixed before the runs. The first, that our build of HillVallEA reproduces the published one on CEC 2013 within 0.03, is met (0.892 against 0.892). The second, that RTR have the best Friedman mean rank among RTR and the two HillVallEA variants, is not met: seeded HillVallEA ranks 1.69, RTR 1.71 and restarted HillVallEA 2.60 (Friedman test $p = 0.001$). The post-hoc tests separate RTR from restarted HillVallEA (Holm-adjusted $p = 0.004$) but not from seeded HillVallEA ($p = 0.94$). The third, that RTR be significantly better than each variant on at least 12 problems, is met: it is better than each on 14 problems, and worse than the restarted variant on 5 (P5, P6, P12, P14, P16) and than the seeded one on 9 (P5, P6, P10 to P14, P16, P21). RTR has the higher mean peak ratio (0.645, against 0.572 seeded and 0.512 restarted), but the two methods are strong in different places. RTR is far ahead on the cone landscapes (0.955 against at most 0.781 in 5-D, and 0.953 against at most 0.609 in 10-D), including P2 and P18, where HillVallEA finds about half of the optima (0.50 to 0.55). Seeded HillVallEA is ahead on the 5-D composition functions (0.566 against 0.524) and on F8 under the eight change modes (0.483 against 0.461), and level on the 10-D ones (0.517 against 0.516). Its explicit basin detection pays off on the composition functions, which suggests combining it with the trigger (Section 7). Finally, no conclusion of this section or the previous one changes when P8/P9 is

counted once. On the organisers' instances, RTR is better than the searchers without relocation on 19 of 23 problems and worse on 2, and the means become 0.628 for RTR against 0.686, 0.656 and 0.649 for the three leading entries. The Friedman ranks of the baselines keep their order. On our draws, RTR is better on 19 of 23 problems at both 15 and 30 runs, and worse on 0 and 2, and the means become 0.646 for RTR and 0.507 without relocation. In the comparison with HillVallEA, the ranks keep their order (1.67, 1.74 and 2.59). The ablation counts of Section 5.5 each lose at most the duplicated problem.

5.3 CEC 2013 niching suite

On the CEC 2013 suite, where both RTR and the searchers without relocation keep the archive, the two pre-specified criteria are met. RTR is significantly better on 8 of the 20 problems and worse on 2, against a required 7 better and at most 2 worse, and its mean peak ratio at 10^{-4} rises from 0.736 to 0.777. Both solve the six easiest problems (F1–F5 and F10), and Table 4, together with Fig. S2 in Online Resource 1, shows where they differ. RTR gains most on the Vincent functions (F7 and F9), whose optima are spaced on a logarithmic scale, and on the higher-dimensional composition functions (F19 and F20). It loses on the Shubert functions (F6 and F8), whose optima come in close pairs, because a single exclusion radius covers both members of a pair and the second member is judged redundant and abandoned. This is the false-redundancy failure that limited perception is meant to prevent. In the evaluated configuration the perception radii are adapted by imitation on these landscapes, whose optima have positive fitness, but they do not become small enough for such fine spacing within the budget. The archive itself is essential where optima outnumber the agents, and without it RTR reaches only 0.728 (0.49 against 0.76 on F6, and 0.09 against 0.35 on F9).

Among the 15 entrants of the 2013 competition, the two re-scored recent methods and RTR, RTR has the third-best Friedman mean rank (5.78), behind RS-CMSA-ESII (3.60) and HillVallEA19 (3.88) and level with NEA2 (5.80). It is strongest on the composition functions in 5 to 20 dimensions (F16–F20), where it comes within 0.002 of the best published values on F17 and F18. On F16, F19 and F20 it trails RS-CMSA-ESII by 0.08 to 0.14, but it is ahead of HillVallEA19 on F19 and F20 and of NEA2 on F16, F17 and F20. On the low-dimensional functions with many optima (F6–F9), by contrast, it is far behind the specialized niching methods, which detect basins explicitly and find nearly all optima, while the relocations of RTR are random.

5.4 Moving Peaks: a measure RTR does not target

Moving Peaks is the standard benchmark for dynamic optimization, and its usual measure, the offline error, rewards tracking the *highest* peak quickly after each change, not holding many peaks. It therefore tests what the rule costs when only the best peak counts. The criteria were fixed in advance, and RTR, with (1+1)-ES searchers, fails all three (Table 5). In offline error it is better than mQSO on 1 of 10 instances and worse on 3, and better than HmSO on 1 of 10 and worse on 2. Its coverage, the measure closer to its aim, is higher than both on 5 of 10 instances, short of the 7 required.

Table 4 CEC 2013, mean peak ratio at accuracy 10^{-4} (50 runs). The verdict column compares RTR with no relocation (Holm-corrected Mann–Whitney): +, better; −, worse; =, no significant difference. NEA2 is the 2013 competition winner; RS-CMSA-ESII and HillVallEA19 were re-scored from their submitted solution files.

F	D	optima	RTR	no relocation	RTR, no archive	verdict	NEA2	RS-CMSA-ESII	HillVallEA19
F1–F5	1–2	1–5	1.000	1.000	1.000	=	1.000	1.000	1.000
F6	2	18	0.764	0.907	0.487	−	0.950	1.000	1.000
F7	2	36	0.759	0.432	0.468	+	0.914	1.000	1.000
F8	3	81	0.194	0.252	0.182	−	0.240	0.996	0.975
F9	3	216	0.346	0.192	0.087	+	0.581	0.985	0.972
F10	2	12	1.000	1.000	1.000	=	0.988	1.000	1.000
F11	2	6	0.980	1.000	0.983	=	0.960	1.000	1.000
F12	2	8	0.787	0.740	0.800	+	0.840	1.000	1.000
F13	2	6	0.917	0.790	0.823	+	0.957	0.977	1.000
F14	3	6	0.813	0.717	0.797	+	0.807	0.847	0.923
F15	3	8	0.728	0.740	0.723	=	0.718	0.750	0.750
F16	5	6	0.697	0.670	0.687	=	0.673	0.833	0.723
F17	5	8	0.748	0.710	0.738	+	0.695	0.748	0.750
F18	10	6	0.667	0.647	0.667	=	0.667	0.667	0.667
F19	10	8	0.632	0.530	0.625	+	0.667	0.708	0.593
F20	20	8	0.502	0.395	0.500	+	0.360	0.618	0.480
mean			**0.777**	0.736	0.728		0.801	0.906	0.892

Relocation is nevertheless essential: without it, the offline error is 2 to 7 times higher (scenario 2: 8.43 vs. 1.63).

RTR does not do better on offline error because it does not aim at the highest peak. RTR spreads its budget evenly over the peaks it holds, and gives the agent on the highest peak no more evaluations than the others. The perception radius does not help either: imitation-based perception did not differ significantly from full perception in offline error on any of the ten instances.

Methods designed for offline error, such as FTMPSO (0.67 on scenario 2 [13]), remain far ahead.

Table 5 Moving Peaks: mean offline error (lower is better) and coverage (higher is better), 30 runs.

instance	RTR	no relocation	mQSO	HmSO	RTR coverage	mQSO coverage	HmSO coverage
scenario 2	1.63	8.43	1.52	1.39	0.870	0.761	0.815
20 peaks	1.48	6.38	1.90	1.62	0.738	0.415	0.634
50 peaks	1.12	3.96	1.06	1.09	0.378	0.170	0.421
100 peaks	0.97	2.39	0.71	1.03	0.193	0.085	0.275
change every 1000	3.92	9.60	4.39	4.58	0.713	0.627	0.588
change every 2500	2.33	8.67	2.29	2.49	0.818	0.726	0.749
change every 10 000	1.16	8.22	0.82	0.93	0.912	0.792	0.855
shift length 2	2.60	9.29	2.08	2.12	0.706	0.703	0.746
shift length 5	4.74	10.59	4.07	4.26	0.562	0.557	0.564
10-D	3.61	12.68	4.17	5.95	0.660	0.602	0.561

5.5 Controls and component ablation

Each variant in Table 6 changes one thing in RTR and was run on all 24 problems of the CEC 2022 suite (our confirmation draw, 10 runs per problem), and compared with RTR on the same runs. The first group was run in the main ablation. Of these, only the comparison with random relocation had a criterion fixed before its runs (better on at least 12 problems, worse on at most 2). The other variants were added after the confirmation, each with a decision rule fixed before its runs. With 10 runs, only large differences are detected (Section 4.4), so a variant with no significant difference is not shown to be equivalent to RTR.

Table 6 Controls and component ablation on the CEC 2022 suite, our confirmation draw: mean peak ratio at accuracy 10^{-4} over 24 problems (10 runs per problem), and the number of problems on which RTR is significantly better or worse than the variant (Holm-corrected Mann–Whitney). RTR reaches 0.645 and the searchers without relocation 0.502 on the same runs.

variant	mean	RTR better	RTR worse
when to relocate			
random relocation at RTR's rate	0.217	23	0
restart on convergence or stagnation, no trigger*	0.492	22	0
the same, keeping left points until the change*	0.496	20	0
the size of the exclusion region			
full perception (global exclusion, default radius)	0.570	11	0
global exclusion at 5 units*	0.633	6	1
fixed perception radius of 5 units*	0.634	2	0
tested variants of the evaluated configuration			
uniform destination	0.656	0	1
sign-independent imitation*	0.634	0	0
stale information about teammates			
snapshot every 5 rounds*	0.644	0	0
snapshot every 20 rounds*	0.640	0	0
snapshot every 40 rounds (one cycle)*	0.636	0	0

* Added after the confirmation, with a decision rule fixed before its runs.

The first controls concern when to relocate. Relocating at random times, with the per-problem relocation rate that RTR showed in development runs (from 5% to 89% of the decisions), lowers the mean to 0.217, below never relocating. RTR is better on 23 of 24 problems and worse on none, so the criterion is met. The one problem without a difference, P21, has the lowest relocation rate. Random relocation moves holders as readily as duplicates (Remark 2).

Restarting a searcher at the farthest of k random points from its teammates when its CMA-ES converges, or when its best point has not improved for $10 + \lceil 30D/\lambda \rceil$ generations, with no redundancy trigger, is the usual restart strategy for multimodal search. It reaches 0.492, no better than never relocating, and RTR is better on 22 of 24 problems and worse on none, which meets the rule fixed for it (better on at least 12, worse on at most 2). Without an archive, a searcher that has converged on a distinct optimum leaves it too, so part of this loss could come from forgetting optima rather than from the missing trigger. A second control, registered after the first, therefore keeps every point a searcher leaves as a reported solution until the next change, and counts it as occupied for later destinations. It reaches 0.496, hardly more, and RTR

is again better on 20 of 24 problems and worse on none, which meets the same rule. A restart on convergence does not single out duplicates, and a duplicate restarts only after spending its evaluations on converging. On CEC 2013, where the corresponding control keeps an archive of converged points, RTR is also better (Section 5.3). The benefit of RTR therefore comes from which searchers relocate and when they do so.

The next three variants change the size of the exclusion region. Full perception at the default radius lowers the mean to 0.570 and is worse on 11 problems. On four cone problems it is held exactly at the exclusion bound of Proposition 1 (Section 5.6). Reducing the radius of global exclusion to 5 units removes these losses, as the bound predicts (1.000 on P2, P3 and P18, 0.993 on P19), and raises the mean to 0.633. It remains worse than RTR on six composition problems (P11 to P14, P16, P24) and better on one (P22), so its rule, which would attribute the gain of RTR over global exclusion to the scale of the radius (at most 2 losses and a mean within 0.02), is not met. The difference is small, however. A fixed perception radius of 5 units reaches 0.634 and is worse on two problems (P12, P16), which meets its rule: under the tested conditions, radii that differ between agents show no advantage over one small common radius. Its redundancy verdicts are those of global exclusion at 5 units, and the two do not differ significantly on any problem. These results are consistent with a simple account: what matters is a small effective exclusion radius, which the radii of RTR provide without choosing one for the problem. A first version of the fixed-radius variant, run with the main ablation, still let the radius grow on cone landscapes and gave a similar result (0.637, worse on the same two problems).

Of the two components we no longer count as part of the rule, uniform destinations do as well as the farthest of k candidates (0.656, and better on P10), and imitation with a hold test that does not depend on the sign of the fitness, so that it is active on every landscape, changes no problem (0.634). As fixed before the latter run, imitation is therefore not part of the rule.

Finally, in a distributed implementation, a searcher would see its teammates' positions and fitness with a delay. We simulated this with snapshots of the team refreshed every k rounds and after every announced change, each agent knowing its own state exactly. With $k = 5$, 20 and 40 rounds (40 is one full cycle, and about a quarter of an environment in 5-D), RTR reaches 0.644, 0.640 and 0.636, does not differ significantly from RTR with current information on any problem, and remains better than never relocating on 16 or 17 problems and worse on none. The rule fixed before these runs (for $k = 5$ and 20: mean within 0.05 of RTR, better than no relocation on at least 16 problems, worse on at most 2) is met: the rule degrades gracefully when information about teammates is delayed.

5.6 Checks of the analysis

The checks below use the runs of Section 5.5 and, for Remark 1, a small set of instrumented runs. Except where noted, they were made after the runs they use.

To check Proposition 1, we computed the bound $P_{r_{ex}-2\epsilon}(O)$ from the geometry of the 60 environments of each problem, without running any algorithm, with $\epsilon = 0.05$, the distance within which the suite counts an optimum as found (its optima are at least 1.26 apart, far more than 2ϵ). The proposition concerns rest states, whereas the

peak ratio is measured at the end of each environment, so the comparison checks the proposition without following from it. The peak ratio of full perception never exceeds the bound, and on four cone problems it equals it, while RTR exceeds it (Table 7). On P2, for example, the closest optima lie 2.24 apart and $r_{ex} = 3.15$, and 72% of RTR's initial perception radii are expected to be below 2.24. At 5 units, the bound is 1 on every problem. We recorded before its results were read that global exclusion at that radius should lose no optima to false redundancy, and it lost none on the four cone problems (Section 5.5). On the composition problems the bound (0.79 to 1.00) lies well above what any variant reaches, so the losses of full perception there have another cause.

Table 7 Exclusion bound of Proposition 1 on the problems of the CEC 2022 suite where it is reached: the fraction of optima that can be held at rest with full perception (from the geometry of the instances), and the measured mean peak ratio at accuracy 10^{-3} (10 runs per problem).

problem	closest optima	bound	full perception	RTR
P2	2.24	0.500	0.500	1.000
P3	2.24	0.750	0.750	0.998
P18	2.53	0.500	0.500	1.000
P19	3.16	0.750	0.749	0.982

For Remark 1, counters added to the implementation, which reproduced every run exactly, recorded each round in which an agent was redundant, for RTR and for the searchers without relocation on six problems (P1, P2, P5, P10, P17 and P21, with 3 runs each). All four expectations fixed before these runs held. The counters changed no result, and no episode under RTR lasted longer than Remark 1(a) allows ($c + 1$ rounds counting both ends, while the longest lasted 40). Episodes were shorter under RTR on every problem (2.7 to 9.2 rounds on average, against 27 to 65), and the share of agent-rounds spent redundant was lower (1% to 19%, against 53% to 89%). Without relocation, the longest episode of a run lasted 620 to 1390 rounds on average, many times the length of an environment, so duplicates do persist across changes.

For Remark 2, we computed for each problem the displacement probability $1 - (1 - q)^m$ of Remark 2, from RTR's relocation rate q (5% to 89% of the decisions) and $m = \lfloor T/c \rfloor$ (3 in 5-D, where $T \approx 156$, and 6 in 10-D, where $T \approx 250$). It ranges from 0.24 to almost 1. Since this is a displacement probability and the loss can differ from it, we compared it with the measured relative loss, $1 - \mathrm{PR}_{\mathrm{random}}/\mathrm{PR}_{\mathrm{RTR}}$, by rank: it orders the 24 problems almost as the loss does (Spearman $\rho = 0.96$). It is higher than the loss on 21 problems, as rediscovery would make it, and lower on three (P5, P7, P15, by up to 0.09). A first, cruder model, which counted misallocated decisions ($2q(1 - q)$ per decision), did not track the loss ($\rho = 0.28$).

5.7 Sensitivity to the settings

We changed one setting at a time on eight problems that cover the families of the suite (P1, P5, P7, P10, P14, P16, P17, P21), with 5 runs per problem (Fig. S3 in Online Resource 1). With the default settings, the mean peak ratio on these problems is 0.560 at accuracy 10^{-4} (0.564 at 10^{-3}). Halving or doubling the cycle length (20 or 80 rounds) or the exclusion radius, or setting σ_0 to 5 or 20 units, changes the mean by at most 0.03, so RTR is not sensitive to these settings within a factor of two.

The team size matters more. With 10 agents the mean drops by 0.02. With 40 agents it drops by 0.23 at accuracy 10^{-4} but only by 0.06 at 10^{-3}, and at 10^{-5} it falls to 0.207. The evaluation budget per environment is fixed ($5000 \cdot D$), so a larger team gives each searcher fewer generations: 40 agents leave each searcher about 78 generations per environment in 5-D and 125 in 10-D, against 156 and 250 for the default of 20. The searchers still locate the peaks, but too few of them refine their points to the required accuracy before the next change, and five of the eight problems lose more than 40% of their peak ratio. As a rule of thumb, the team size should leave each searcher about 150 generations per environment: $N \approx B/(150\,\lambda)$ for a budget of B evaluations per environment, which gives the default of 20 agents in 5-D.

5.8 Summary of the pre-specified criteria

Table 8 lists the success criteria fixed before the confirmation runs and their outcomes, followed by those of the experiments added after the confirmation, each fixed before its runs.

6 Discussion

Independent searchers started at random points tend to crowd onto the basins with the largest attraction regions, and this is where RTR gains most, on the cone landscapes that mix global and local peaks (P1 and P17). The controls in Section 5.5 indicate that relocation pays off only when redundancy triggers it. Relocating at random times at RTR's own rate does worse than never relocating, and so does the usual restart strategy, which restarts a searcher once it has converged or stagnated, even when the points it leaves are kept until the next change. The trigger singles out searchers that have a fitter teammate within a small effective radius. When that radius is small, the teammate already holds their basin, so their departure costs the team little, and they leave before spending further evaluations on converging to an optimum that is already held. Where a relocated searcher restarts seems to matter much less, since uniform destinations performed as well as the farthest candidate.

The size of the exclusion region plays the other main role. A small effective radius avoids false redundancy verdicts between distinct optima that lie close together. With the default radius, global exclusion stays exactly at its bound wherever optima are close, whereas a single small global radius removes those losses and comes within 0.012 of RTR in the mean. We therefore see the benefit of limited perception in sparing the user the choice of a radius for each problem, rather than in finding more optima than

Table 8 Pre-specified success criteria and outcomes.

criterion	benchmark	threshold	outcome
relocation vs. none	CEC 2022	better on $\geq 16/24$, worse on none (15 runs)	**met** (20/24, none); 30 runs: 20 better, 2 worse
vs. baselines	CEC 2022	best mean Friedman rank; better than HmSO on $\geq 12/24$	**met** (rank 1.08 among the four pre-specified methods*; 24/24 vs. default HmSO)
instance draws	CEC 2022	stable within 0.05; better than ablation on every draw	**met** (0.626–0.633; 4/4)
trigger vs. random relocation	CEC 2022	better on $\geq 12/24$, worse on ≤ 2	**met** (23/24, none)
relocation vs. none	CEC 2013	better on $\geq 7/20$, worse on ≤ 2	**met** (8/20, 2)
mean peak ratio vs. none	CEC 2013	higher	**met** (0.777 vs. 0.736)
offline error vs. mQSO	Moving Peaks	better on $\geq 7/10$, worse on ≤ 1	not met (1/10, 3)
offline error vs. HmSO	Moving Peaks	better on $\geq 7/10$, worse on ≤ 1	not met (1/10, 2)
coverage vs. both	Moving Peaks	better than both on $\geq 7/10$	not met (5/10)
added after the confirmation, each fixed before its runs			
global exclusion at 5 units	CEC 2022	worse on $\leq 2/24$ and mean within 0.02 (gain attributed to radius scale)	not met (worse on 6, better on 1; mean -0.012)
fixed radius 5 units	CEC 2022	worse on $\leq 2/24$ and mean within 0.02 (no advantage of per-agent radii shown)	**met** (worse on 2; mean -0.011); not a test of equivalence
sign-independent imitation	CEC 2022	kept if better on $\geq 1/24$ and worse on none	not kept (no difference); imitation optional
duplicate lifetime	CEC 2022	four expectations on 6 problems (Section 5.6)	**met** (4/4)
restart on convergence (control)	CEC 2022	RTR better on $\geq 12/24$, worse on ≤ 2	**met** (22/24, none)
stale information, $k = 5, 20$	CEC 2022	mean within 0.05 of RTR; better than no relocation on $\geq 16/24$, worse on ≤ 2	**met** (0.644, 0.640; 16 and 17, none)
restart control with archive	CEC 2022	RTR better on $\geq 12/24$, worse on ≤ 2	**met** (20/24, none)
rule as recommended, organisers' instances	CEC 2022	mean within 0.02 of the evaluated configuration; differs on $\leq 2/24$	not met in its favour (0.640 vs. 0.628; better on 3, worse on none)
relocation vs. none, organisers' instances	CEC 2022	better on $\geq 16/24$, worse on none (30 runs)	not met (20/24, 2 worse: P2, P22)
vs. baselines, organisers' instances	CEC 2022	best mean Friedman rank; better than tuned HmSO on $\geq 12/24$	**met** (rank 1.04; 23/24)
HillVallEA build valid	CEC 2013	mean within 0.03 of the authors' solution files	**met** (0.892 vs. 0.892)
vs. HillVallEA: rank	CEC 2022	best mean Friedman rank among RTR and two HillVallEA variants	not met (1.71 vs. 1.69 for seeded HillVallEA)
vs. HillVallEA: problems	CEC 2022	better than each variant on $\geq 12/24$	**met** (14/24 vs. each)

* RTR, default HmSO, mQSO and DE, 15 runs per problem. Section 5.2 reports 1.12 among five methods at 30 runs (15 for default HmSO), including tuned HmSO, which was added after the confirmation.

a well-chosen common radius would. Whether radii drawn over a wide range are also more robust than one fixed radius across very different suites remains untested.

Because the rule only needs the positions and fitness values of nearby teammates, it lends itself to local communication and asynchronous decisions. In our simulations, teammate information up to one cycle old did not change the results measurably, although a genuinely distributed implementation, with message delays and losses, was not evaluated.

RTR fails in two situations. When optima are closely spaced, as in the Shubert functions of CEC 2013, a single exclusion radius merges distinct optima, and the perception radii are not small enough to keep them apart. Methods that detect basins explicitly do better there, and on the composition functions of the CEC 2022 suite HillVallEA and the leading competition entries are mostly ahead. A basin test against the fitter teammate before relocating would be a natural remedy. When the measure rewards only the best peak, as the offline error of Moving Peaks does, RTR is at a disadvantage because it spreads its effort evenly over the peaks it holds.

The study has several limitations. The confirmation, the controls and the ablations ran on our own draw of the CEC 2022 instances, and the organisers' instances were replayed afterwards. The conclusions agree on both. mQSO and HmSO are our re-implementations, and although their offline errors on Moving Peaks are at or below the published values on most instances (1.52 and 1.39 on scenario 2, against 1.75 and 1.46), small differences of protocol cannot be ruled out. We did not run the leading competition entries ourselves, so the comparison with them is descriptive. The searcher type was chosen per benchmark family during development, and the default exclusion radius assumes about ten optima per domain. Halving or doubling it changed the mean peak ratio on the CEC 2022 suite by less than 0.02, but suites with many closely spaced optima would need a smaller radius. The team size has to be matched to the evaluation budget per environment (Section 5.7). Finally, the confirmations used a configuration with two components, radius imitation and the far-point destination, that brought no measurable benefit. The rule without them was later run on the organisers' instances (0.640, against 0.628), but not on CEC 2013 or Moving Peaks.

7 Conclusion

Conditional relocation, *relocate when a fitter teammate is perceived within the exclusion radius*, is an effective and lightweight way to coordinate a team of local searchers for multimodal optimization. On the organisers' instances of the CEC 2022 suite it raises the mean peak ratio of independent CMA-ES searchers from 0.503 to 0.628, ranks first among the dynamic baselines we ran, and has the best mean of all methods on the 10-dimensional cone landscapes, while the best published competition entries remain ahead overall. Without its two optional components it reaches 0.640. On the CEC 2013 suite it improves on the same searchers and archive without relocation, and it is close to the best published results on the composition functions in 5 to 20 dimensions. The gain comes from the trigger, which neither relocating at random times nor restarting on convergence reproduces, together with a small effective exclusion radius. The analysis shows why global exclusion cannot match it where optima lie close together. The rule needs no clustering, no central control and no per-problem

estimate of the number of optima, beyond one default radius that assumes a moderate density of optima, and it tolerates delayed information about teammates. It is not designed for offline error, and it trails explicit basin-detection methods when optima are many and closely spaced.

Several directions remain open. A redundant agent could contest its verdict before leaving, to recover closely spaced optima, either through an explicit basin test against the fitter teammate or archived point that makes it redundant, such as a hill-valley check at one evaluation per verdict, or by ignoring the verdict at random, which costs no evaluation but cannot tell a false verdict from a true one. When the measure rewards the best peak, more of the budget could go to the best held basins. It also remains to study how the perception radius should scale with dimension and with the density of optima.

Declarations

- **Funding:** No funding was received for conducting this study.
- **Competing interests:** The author has no relevant financial or non-financial interests to disclose.
- **Ethics approval and consent to participate:** Not applicable.
- **Consent for publication:** Not applicable.
- **Data availability:** No new data were created in this study; all benchmarks are public and were obtained from their original authors. The CEC 2013 niching benchmark [23], its reference code and the competition results used for comparison are available at https://github.com/mikeagn/CEC2013. The CEC 2022 benchmark on seeking multiple optima in dynamic environments [4], its problem data and the published competition results are available at https://github.com/MiLabHITSZ/dmmops. The Moving Peaks benchmark is defined in [28] and its instances are generated from that definition. The per-run results obtained in this study are available with the code (below).
- **Code availability:** The source code of RTR, of the benchmark implementations and of the baselines, the scripts that run every experiment and score its criteria, and the raw per-run results are available at https://github.com/dpalmasan/rtr.
- **Author contribution:** D. Palma conceived the study, designed and implemented the method and the experiments, analysed the results and wrote the manuscript.

References

[1] Li, X., Epitropakis, M.G., Deb, K., Engelbrecht, A.: Seeking multiple solutions: An updated survey on niching methods and their applications. IEEE Transactions on Evolutionary Computation **21**(4), 518–538 (2017) https://doi.org/10.1109/TEVC.2016.2638437

[2] Das, S., Maity, S., Qu, B.-Y., Suganthan, P.N.: Real-parameter evolutionary multimodal optimization – a survey of the state-of-the-art. Swarm and Evolutionary Computation **1**(2), 71–88 (2011) https://doi.org/10.1016/j.swevo.2011.05.005

[3] Lin, X., Luo, W., Xu, P., Qiao, Y., Yang, S.: PopDMMO: A general framework of population-based stochastic search algorithms for dynamic multimodal optimization. Swarm and Evolutionary Computation **68**, 101011 (2022) https://doi.org/10.1016/j.swevo.2021.101011

[4] Luo, W., Lin, X., Li, C., Yang, S., Shi, Y.: Benchmark Functions for CEC 2022 Competition on Seeking Multiple Optima in Dynamic Environments. arXiv:2201.00523 (2022)

[5] Goldberg, D.E., Richardson, J.: Genetic algorithms with sharing for multimodal function optimization. In: Proceedings of the Second International Conference on Genetic Algorithms, pp. 41–49 (1987)

[6] Pétrowski, A.: A clearing procedure as a niching method for genetic algorithms. In: Proceedings of IEEE International Conference on Evolutionary Computation, pp. 798–803 (1996). https://doi.org/10.1109/ICEC.1996.542703

[7] Maree, S.C., Alderliesten, T., Thierens, D., Bosman, P.A.N.: Real-valued evolutionary multi-modal optimization driven by hill-valley clustering. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '18), pp. 857–864 (2018). https://doi.org/10.1145/3205455.3205477

[8] Preuss, M.: Improved topological niching for real-valued global optimization. In: Applications of Evolutionary Computation (EvoApplications 2012). Lecture Notes in Computer Science, vol. 7248, pp. 386–395 (2012). https://doi.org/10.1007/978-3-642-29178-4_39

[9] Wang, Y., Li, H.-X., Yen, G.G., Song, W.: MOMMOP: Multiobjective optimization for locating multiple optimal solutions of multimodal optimization problems. IEEE Transactions on Cybernetics **45**(4), 830–843 (2015) https://doi.org/10.1109/tcyb.2014.2337117

[10] Blackwell, T., Branke, J.: Multiswarms, exclusion, and anti-convergence in dynamic environments. IEEE Transactions on Evolutionary Computation **10**(4), 459–472 (2006) https://doi.org/10.1109/tevc.2005.857074

[11] Kamosi, M., Hashemi, A.B., Meybodi, M.R.: A hibernating multi-swarm optimization algorithm for dynamic environments. In: Second World Congress on Nature and Biologically Inspired Computing (NaBIC), pp. 363–369 (2010). https://doi.org/10.1109/nabic.2010.5716372

[12] Li, C., Yang, S.: A general framework of multipopulation methods with clustering in undetectable dynamic environments. IEEE Transactions on Evolutionary Computation **16**(4), 556–577 (2012) https://doi.org/10.1109/TEVC.2011.2169966

[13] Yazdani, D., Nasiri, B., Sepas-Moghaddam, A., Meybodi, M.R.: A novel multi-swarm algorithm for optimization in dynamic environments based on particle swarm optimization. Applied Soft Computing **13**(4), 2144–2158 (2013) https://doi.org/10.1016/j.asoc.2012.12.020

[14] Yazdani, D., Cheng, R., Yazdani, D., Branke, J., Jin, Y., Yao, X.: A survey of evolutionary continuous dynamic optimization over two decades – part B. IEEE Transactions on Evolutionary Computation **25**(4), 630–650 (2021) https://doi.org/10.1109/TEVC.2021.3060012

[15] Hansen, N., Ostermeier, A.: Completely derandomized self-adaptation in evolution strategies. Evolutionary Computation **9**(2), 159–195 (2001) https://doi.org/10.1162/106365601750190398

[16] Ahrari, A., Deb, K., Preuss, M.: Multimodal optimization by covariance matrix self-adaptation evolution strategy with repelling subpopulations. Evolutionary Computation **25**(3), 439–471 (2017) https://doi.org/10.1162/evco_a_00182

[17] Ursem, R.K.: Multinational evolutionary algorithms. In: Proceedings of the 1999 Congress on Evolutionary Computation, vol. 3, pp. 1633–1640 (1999). https://doi.org/10.1109/CEC.1999.785470

[18] Li, J.-P., Balazs, M.E., Parks, G.T., Clarkson, P.J.: A species conserving genetic algorithm for multimodal function optimization. Evolutionary Computation **10**(3), 207–234 (2002) https://doi.org/10.1162/106365602760234081

[19] Shir, O.M., Bäck, T.: Niche radius adaptation in the CMA-ES niching algorithm. In: Parallel Problem Solving from Nature – PPSN IX. Lecture Notes in Computer Science, vol. 4193, pp. 142–151 (2006). https://doi.org/10.1007/11844297_15

[20] Li, X.: Niching without niching parameters: Particle swarm optimization using a ring topology. IEEE Transactions on Evolutionary Computation **14**(1), 150–169 (2010) https://doi.org/10.1109/TEVC.2009.2026270

[21] Qu, B.-Y., Suganthan, P.N., Liang, J.J.: Differential evolution with neighborhood mutation for multimodal optimization. IEEE Transactions on Evolutionary Computation **16**(5), 601–613 (2012) https://doi.org/10.1109/tevc.2011.2161873

[22] Epitropakis, M.G., Plagianakos, V.P., Vrahatis, M.N.: Finding multiple global optima exploiting differential evolution's niching capability. In: 2011 IEEE Symposium on Differential Evolution (SDE) (2011). https://doi.org/10.1109/SDE.2011.5952058

[23] Li, X., Engelbrecht, A., Epitropakis, M.G.: Benchmark functions for CEC'2013 special session and competition on niching methods for multimodal function optimization. Technical report, Evolutionary Computation and Machine Learning Group, RMIT University (2013)

[24] Nguyen, T.T., Yang, S., Branke, J.: Evolutionary dynamic optimization: A survey of the state of the art. Swarm and Evolutionary Computation **6**, 1–24 (2012) https://doi.org/10.1016/j.swevo.2012.05.001

[25] Yazdani, D., Cheng, R., Yazdani, D., Branke, J., Jin, Y., Yao, X.: A survey of evolutionary continuous dynamic optimization over two decades – part A. IEEE Transactions on Evolutionary Computation **25**(4), 609–629 (2021) https://doi.org/10.1109/TEVC.2021.3060014

[26] Jin, Y., Branke, J.: Evolutionary optimization in uncertain environments – a survey. IEEE Transactions on Evolutionary Computation **9**(3), 303–317 (2005) https://doi.org/10.1109/tevc.2005.846356

[27] Branke, J.: Evolutionary Optimization in Dynamic Environments. Genetic Algorithms and Evolutionary Computation, vol. 3. Kluwer Academic Publishers, Boston (2002). https://doi.org/10.1007/978-1-4615-0911-0

[28] Branke, J.: Memory enhanced evolutionary algorithms for changing optimization problems. In: Proceedings of the 1999 Congress on Evolutionary Computation, pp. 1875–1882 (1999). https://doi.org/10.1109/CEC.1999.785502

[29] Luo, W., Lin, X., Zhu, T., Xu, P.: A clonal selection algorithm for dynamic multimodal function optimization. Swarm and Evolutionary Computation **50**, 100459 (2019) https://doi.org/10.1016/j.swevo.2018.10.010

[30] Cuevas, E., Gálvez, J., Toski, M., Avila, K.: Evolutionary-mean shift algorithm for dynamic multimodal function optimization. Applied Soft Computing **113**, 107880 (2021) https://doi.org/10.1016/j.asoc.2021.107880

[31] Lin, X., Luo, W., Qiao, Y., Xu, P., Zhu, T.: EvoDCMMO: Benchmarking and solving dynamic constrained multimodal optimization problems. Swarm and Evolutionary Computation **75**, 101184 (2022) https://doi.org/10.1016/j.swevo.2022.101184

[32] Ahrari, A., Elsayed, S., Sarker, R., Essam, D., Coello Coello, C.A.: Adaptive multilevel prediction method for dynamic multimodal optimization. IEEE Transactions on Evolutionary Computation **25**(3), 463–477 (2021) https://doi.org/10.

1109/TEVC.2021.3051172

[33] Ahrari, A., Elsayed, S., Sarker, R., Essam, D., Coello Coello, C.A.: Static and dynamic multimodal optimization by improved covariance matrix self-adaptation evolution strategy with repelling subpopulations. IEEE Transactions on Evolutionary Computation **26**(3), 527–541 (2022) https://doi.org/10.1109/TEVC.2021.3117116

[34] Ahrari, A., Elsayed, S., Sarker, R., Essam, D., Coello Coello, C.A.: A novel parametric benchmark generator for dynamic multimodal optimization. Swarm and Evolutionary Computation **65**, 100924 (2021) https://doi.org/10.1016/j.swevo.2021.100924

[35] Jin, H., Wang, X., Cheng, S., Sun, Y., Zhang, M., Lu, H., Wu, H., Shi, Y.: Solving dynamic multimodal optimization problems via a niching-based brain storm optimization with two archives algorithm. Swarm and Evolutionary Computation **89**, 101649 (2024) https://doi.org/10.1016/j.swevo.2024.101649

[36] Fujita, S., Ishizawa, R., Sato, H., Takadama, K.: Adaptive multi-population dynamic optimization for multimodal dynamic function optimization. In: Proceedings of the Genetic and Evolutionary Computation Conference (GECCO '25), pp. 1540–1548. ACM, New York, NY, USA (2025). https://doi.org/10.1145/3712256.3726473

[37] Zhu, Y., Xu, P., Huang, J., Lin, X., Luo, W.: Density-assisted evolutionary dynamic multimodal optimization. ACM Transactions on Evolutionary Learning and Optimization **6**(1), 1–30 (2026) https://doi.org/10.1145/3723171

[38] Lin, X., Luo, W., Li, C., Yang, S., Shi, Y.: CEC 2023 Competition on Seeking Multiple Optima in Dynamic Environments: Results. https://github.com/MiLabHITSZ/dmmops (2023)

[39] Lin, X., Luo, W., Li, C., Yang, S., Shi, Y.: CEC 2024 Competition on Seeking Multiple Optima in Dynamic Environments: Results. https://github.com/MiLabHITSZ/dmmops (2024)

[40] Hansen, N.: The CMA Evolution Strategy: A Tutorial. arXiv:1604.00772 (2016)

[41] Beyer, H.-G., Schwefel, H.-P.: Evolution strategies – A comprehensive introduction. Natural Computing **1**(1), 3–52 (2002) https://doi.org/10.1023/A:1015059928466

[42] Nobel, J., Vermetten, D., Kononova, A.V., Shir, O.M., Bäck, T.: Avoiding redundant restarts in multimodal global optimization. In: Parallel Problem Solving from Nature – PPSN XVIII. Lecture Notes in Computer Science, pp. 268–283. Springer, Cham (2024). https://doi.org/10.1007/978-3-031-70068-2_17

[43] Auger, A., Hansen, N.: A restart CMA evolution strategy with increasing population size. In: 2005 IEEE Congress on Evolutionary Computation, pp. 1769–1776 (2005). https://doi.org/10.1109/CEC.2005.1554902

[44] Nosek, B.A., Ebersole, C.R., DeHaven, A.C., Mellor, D.T.: The preregistration revolution. Proceedings of the National Academy of Sciences **115**(11), 2600–2606 (2018) https://doi.org/10.1073/pnas.1708274114

[45] Mann, H.B., Whitney, D.R.: On a test of whether one of two random variables is stochastically larger than the other. The Annals of Mathematical Statistics **18**(1), 50–60 (1947) https://doi.org/10.1214/aoms/1177730491

[46] Holm, S.: A simple sequentially rejective multiple test procedure. Scandinavian Journal of Statistics **6**(2), 65–70 (1979)

[47] Vargha, A., Delaney, H.D.: A critique and improvement of the CL common language effect size statistics of McGraw and Wong. Journal of Educational and Behavioral Statistics **25**(2), 101–132 (2000) https://doi.org/10.3102/10769986025002101

[48] Friedman, M.: The use of ranks to avoid the assumption of normality implicit in the analysis of variance. Journal of the American Statistical Association **32**(200), 675–701 (1937) https://doi.org/10.1080/01621459.1937.10503522

[49] Demšar, J.: Statistical comparisons of classifiers over multiple data sets. Journal of Machine Learning Research **7**, 1–30 (2006)

[50] Derrac, J., García, S., Molina, D., Herrera, F.: A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms. Swarm and Evolutionary Computation **1**(1), 3–18 (2011) https://doi.org/10.1016/j.swevo.2011.02.002

Supplementary Files

This is a list of supplementary files associated with this preprint. Click to download.

- ESM1.pdf