A Simple Strategy for Maintaining Diversity and Reducing Crowding in Differential Evolution

James Montgomery
Research School of Computer Science
Australian National University
Canberra, Australia
james.montgomery@anu.edu.au

Stephen Chen
School of Information Technology
York University
Toronto, Canada
sychen@yorku.ca

***Abstract*—Differential evolution (DE) is a widely-effective population-based continuous optimiser that requires convergence to automatically scale its moves. However, once its population has begun to converge its ability to conduct global search is diminished, as the difference vectors used to generate new solutions are derived from the current population members' positions. In multi-modal search spaces DE may converge too rapidly, i.e., before adequately exploring the search space to identify the best region(s) in which to conduct its finer-grained search. Traditional crowding or niching techniques can be computationally costly or fail to compare new solutions with the most appropriate existing population member. This paper proposes a simple intervention strategy that compares each new solution with the population member it is most likely to be near, and prevents those moves that are below a threshold that decreases over the algorithm's run, allowing the algorithm to ultimately converge. Comparisons with a standard DE algorithm on a number of multi-modal problems indicate that the proposed technique can achieve real and sizable improvements.**

I. Introduction

The effectiveness of the search process in differential evoluton (DE) depends strongly on the diversity of its population. In a typical configuration, DE's search in multi-modal search spaces is characterised by an initial period of exploration followed by rapid convergence in a good area found earlier. This rapid convergence can occur well before the algorithm's budget of function evaluations is exhausted; that is, the algorithm can converge prematurely. While all heuristic optimisation algorithms must achieve a balance between exploration and exploitation in their search, this is particularly the case with population-based approaches, since a drop in diversity in the population dramatically impacts their ability to continue exploring a solution space. DE can be affected more strongly by this than other population-based algorithms, since its mutation mechanism, in its default form, is entirely dependent on differences between population members. Once DE's population has converged to a small area there is little probability that it can explore outside that region. Despite the importance of balancing the two aspects of search, the standard mechanics of DE's mutation and selection do not explicitly control the importance of either.

Techniques to slow or prevent convergence in multi-modal search spaces are not new in evolutionary computing, being used since at least the mid-seventies [1]. Such approaches include *niching*, in which explicit mechanisms are employed to split a population's search efforts across different parts of the search space, perhaps by using multiple, minimally-interacting subpopulations. Another, older approach is *crowding*, where each new solution is compared against a subset (not necessarily proper) of the population and replaces the one most similar to it, if the new solution is better. The key parameter for crowding is the *crowding factor*, or number of current population members to which new solutions are compared. This can be small, which can cause "replacement errors" in which some unexamined solution is actually more similar, or it can be large, which can cause significant increases to the required computational effort [2].

Recent work [3] has identified a way to efficiently implement a convergence control mechanism similar to crowding in particle swarm optimisation (PSO). In PSO, particles, representing solutions in some continuous space, explore that space and remember the best position they have visited so far, their respective "personal best". Each particle has "momentum", which allows exploration, but is also attracted to its personal best location *and* "local best", which is the best personal best within its neighbourhood.[1] The key insight gained is that once a particle's personal best becomes close to its current local best, its ability to conduct exploration diminishes. Thus, by having an improved personal best position update the local best instead (if it is too close), diversity can be maintained. In DE, each solution's position evolves through a series of positions with monotonically decreasing cost (assuming a minimisation problem). Thus the DE population is similar to the population of personal bests in PSO. This paper describes the development of a simple intervention strategy that uses knowledge of how DE converges to prolong the algorithm's exploration phase. This is part of a broader effort to develop thresholding techniques to control convergence in heuristic search algorithms [4].

The following subsection describes the machanics of DE, as these are the basis for understanding its convergence behaviour. The subsequent subsection describes the various niching and crowding approaches that have been applied to

[1]For those unfamiliar with PSO, the neighbourhood is established by a communication topology at the algorithm's start and is not indicative of particles' relative positions at a particular time.

DE. Section II describes the many factors affecting DE's convergence behaviour before a mechanism to control the rate of DE's convergence is presented in Section III. Comparative results for standard and the modified DE are given in Section IV, while Section V summarises the work.

A. DE mechanics

DE is a generational evolutionary algorithm in which, at each iteration, every population member is considered as a *target* for replacement by a newly generated point in solution space. There are many variants of the DE approach, which use population members in a variety of ways to generate new solutions. This section, and the remainder of this work, considers only the highly common, and frequently effective [5], variant labelled DE/rand/1/bin. This version of the algorithm generates new solutions by adding the weighted difference between two randomly selected population members to a third population member (not the *target*) and then performing uniform crossover of that new point with the *target*. Let $P = \{x_1, x_2, \ldots, x_{N_p}\}$ be the population of Np solutions. For a given *target* solution x_{target}, a new point v is generated according to

$$v = x_{base} + F \cdot (x_{r1} - x_{r2}) \tag{1}$$

where x_{base} (or simply *base*), x_{r1} and x_{r2} are distinct, randomly selected solutions from $P \setminus \{x_{target}\}$ and F is the scaling factor, typically in $(0, 1]$ although larger values are also possible. Uniform crossover is performed on v, controlled by the parameter $Cr \in [0, 1]$, to produce the candidate solution u, according to

$$u^j = \begin{cases} v^j & \text{if} R^j \leq Cr \text{ or } j = I, \\ x^j_{target} & \text{if} R^j > Cr \text{ and } j \neq I, \end{cases} \tag{2}$$

where $R \in [0, 1)$ is a uniform random number and I is the randomly selected index of a component that must be mutated for this solution, which ensures that $u \neq x$. The *target* is replaced if the new solution is as good or better.

B. Niching and Crowding in DE

Two related approaches to maintaining a diverse population in evolutionary algorithms are niching and crowding [6]. In niching the purpose is to have the population split into subpopulations that explore disparate parts of the search space. It is thus useful when one wishes to find multiple, alternative optima. Niching methods in DE include Li's speciation-based DE [7], which splits the population by identifying "dominant" individuals and then determining clusters of similar individuals near them. Zaharie's Multipopulation DE [8] takes an alternative approach, dividing the search space into a number of non-overlapping subspaces with separate populations for each. The search consists of a number of "epochs", after which the subspaces are reinitialised at a more fine-grained level.

Rönkkönen [9], [10] presents a number of alternative DE algorithms including DE with local selection and local mutation (DELL), which is also a niching-based approach. The DELL algorithm does not use crossover as it would make the algorithm no longer rotationally invariant. Local selection in DE means that the *base* is the same as *target*, which naturally helps keep solutions from converging. Rönkkönen [10] acknowledges that local selection on its own is not sufficient to spread knowledge of the best solution outside its own niche.

The purpose of crowding is to prevent solutions from becoming too close in solution space, which slows the rate of convergence and can help the population identify multiple optima. Thomsen [2] describes two DE algorithms that use different mechanisms to achieve this: SharingDE and CrowdingDE. In a sharing-based scheme, as used in SharingDE, the reported quality of a solution diminishes with its proximity to other solutions. In CrowdingDE, instead of comparing a new candidate solution against the *target*, the solution is compared to the population member nearest to it in the search space (i.e., the crowding factor is Np as all solutions must be examined). If the new solution is better it replaces that nearest solution immediately, unlike standard DE which operates a generational model. SharingDE and CrowdingDE were tested on a range of multi-modal two-dimensional problems using $Np = 100$ (very large for problems of this size, as discussed in the next section), with CrowdingDE performing better than SharingDE. Given the superficial similarity of CrowdingDE to the convergence control scheme proposed in this paper—new solutions are accepted based on their distance to some other solution(s)—it is compared against the proposed approach in Section IV-C.

Each of these techniques works on solutions after they have been generated, i.e., a new point in solution space has been produced *and* evaluated. In many real-world problems of interest solution evaluation can be computationally expensive, so a convergence control technique that can intervene prior to this could be beneficial. Some of these approaches also require a large number of comparisons between new and existing solutions. However, if one considers the natural convergence behaviour of DE then alternative approaches can eventuate that address both of these issues.

II. Convergence Behaviour in DE

DE's search behaviour results from a complex interaction of population size Np, scale factor F and crossover rate Cr [11]. While the particular positions that population members occupy also have an impact—they may, for instance, establish a predominant direction for the search—the spread of the population, the magnitude of exploratory moves made and the rate at which these change are most strongly determined by these three factors Np, F and Cr.

Population size can affect convergence behaviour, with 'small' populations allowing convergence and 'large' populations retarding it [11].[2] Assuming a fixed budget of function evaluations, as in common in experimental and practical use of heuristic optimisers, the larger the population the fewer times each individual is (potentially) updated. Thus the potential

[2]It is worth noting that DE's creators recommend a population size of $10 \cdot D$ be used [12], which should be considered large, while many other studies have used much smaller populations, closer in size to D.

benefit of having a greater number of solutions exploring the search space does not necessarily lead to improvements in the final result if the algorithm is allowed to run to convergence [11].

It has been known for almost a decade that too small a value of F can lead to premature convergence. Zaharie [13] derived a relation between population variance (i.e., a measure of convergence) and the values of Cr and F, showing that the range of effective values of F is greater when Cr is low, and that F must be relatively high when Cr is high. This finding was strengthened by Montgomery [11] and Montgomery and Chen [14], who found that DE exhibits vastly different search behaviours when Cr is near its extremes (and typically performs very poorly when $Cr \approx 0.5$). At low values of Cr, DE population members effectively conduct independent searches, making *small* moves *from the target solution*. A low value of F can make these moves smaller, but it cannot lead the population to converge. Indeed, in multi-modal search spaces DE with low Cr will generally not converge within typical function evaluation limits [11].

When Cr is high, moves are largely conducted from the *base* solution, so new candidate solutions are typically closer to *base* than the *target* they replace, leading to convergence. It is for this reason that Cr is never set to exactly 1, since new solutions are then guaranteed to be $||F \cdot (x_{r1} - x_{r2})||$ distance away from the *base* used, leading to extremely fast convergence [11]. When the number of dimensions D of the problem being solved is moderate, such as $D = 20$, even with $Cr \approx 0.9$ there is still a significant probability that all components of the child vector will be produced according to the first branch of Equation 2. Even with much larger problems, mutating on average 90% of vector components using a small difference vector can lead to a significant loss of diversity.

Earlier studies of DE's behaviour in multi-modal search spaces have shown that small difference vectors applied to distant *base* solutions are a significant driver of movement of solutions between competing optima [15]–[17]. Indeed, as tighter groups of individuals form around different optima this "recruitment" behaviour becomes the *only* mechanism for movement between different basins of attraction, as any moves produced by large difference vectors fall short of areas which would improve *target* solutions. Once these groups of close individuals form, smaller difference vectors are more likely to be used in generating new solutions, leading to a cascade of converence.

In summary, when using a low value of Cr, premature convergence is not a concern as individuals effectively conduct independent searches, but the global search the algorithm performs is likely not to be as successful in complex multi-modal search spaces, as little information about distant areas is transmitted between individuals. When using high Cr, the scale factor F and population size can both affect the rate of convergence, but choosing the best values to achieve a desired rate is difficult. Small difference vectors, which are an emergent property of the current state of the population (and which indicate that some convergence has already occurred) do appear to lead to further convergence of the population. In particular, they are most likely to place a new solution near the *base* used in its construction. This suggests a way in which convergence can be controlled while leaving the values of Np, F and Cr at typical levels, which is discussed in the next section.

III. Developing a Simple Mechanism for Convergence Control in DE

Given that convergence of a DE population when using $Cr \approx 0.9$ is likely to come from a new point being generated near to the *base* solution used in its construction, it is this event at which a simple convergence control mechanism may appropriately intervene. Note that this is similar to the previous work of the authors with PSO [3], which identified with which single other point a new solution is likely to form a crowd: in the case of of PSO it is the particle's local best attractor, whereas in DE it is the new solution's *base*.

At least three alternative convergence control schemes are possible. In each scheme a decreasing *threshold* is used, although the threshold does not apply to the same distance in each scheme. The new, candidate solution is referred to as *new*.

1) Prevent moves produced by difference vectors whose magnitude ($||x_{r1} - x_{r2}||$) is less than *threshold*.
2) Prevent moves where the actual distance between *new* and *base* is less than the threshold, i.e, $||new - base|| < threshold$.
3) As in (2), but instead of preventing the move, if $f(new) < f(base)$,[3] have *new* replace *base*.

Scheme (1) is computationally efficient as the difference vector magnitude can be calculated contemporaneously with generation of *new*. However, it is indirect, using difference vector magnitude as a proxy for distance between *new* and *base* even though the number of mutated components may have been low due to the effects of crossover. The third scheme is most similar to the convergence control mechanism proposed for PSO in [3], in which another solution's "personal best" position may be replaced. It is also similar to CrowdingDE, while retaining standard DE's generational solution replacement. However, since the DE generation model compares each new solution against its parent, such a scheme is potentially very complex. If *new* replaces *base* instead of *target*, convergence in some dimensions is almost guaranteed, as *new* will likely share some components with *target*. Furthermore, *base* is also a *target* for replacement, and its own new candidate solution may be better than the *new* with which this scheme would replace *base*.

Given the complexities of implementing scheme (3) under a standard DE generational model, and the indirect nature of scheme (1), the second approach was selected for implementation and testing. As the selected convergence control mechanism considers the distance between *new* and *base*, but

[3] All problems considered here have minimisation objectives.

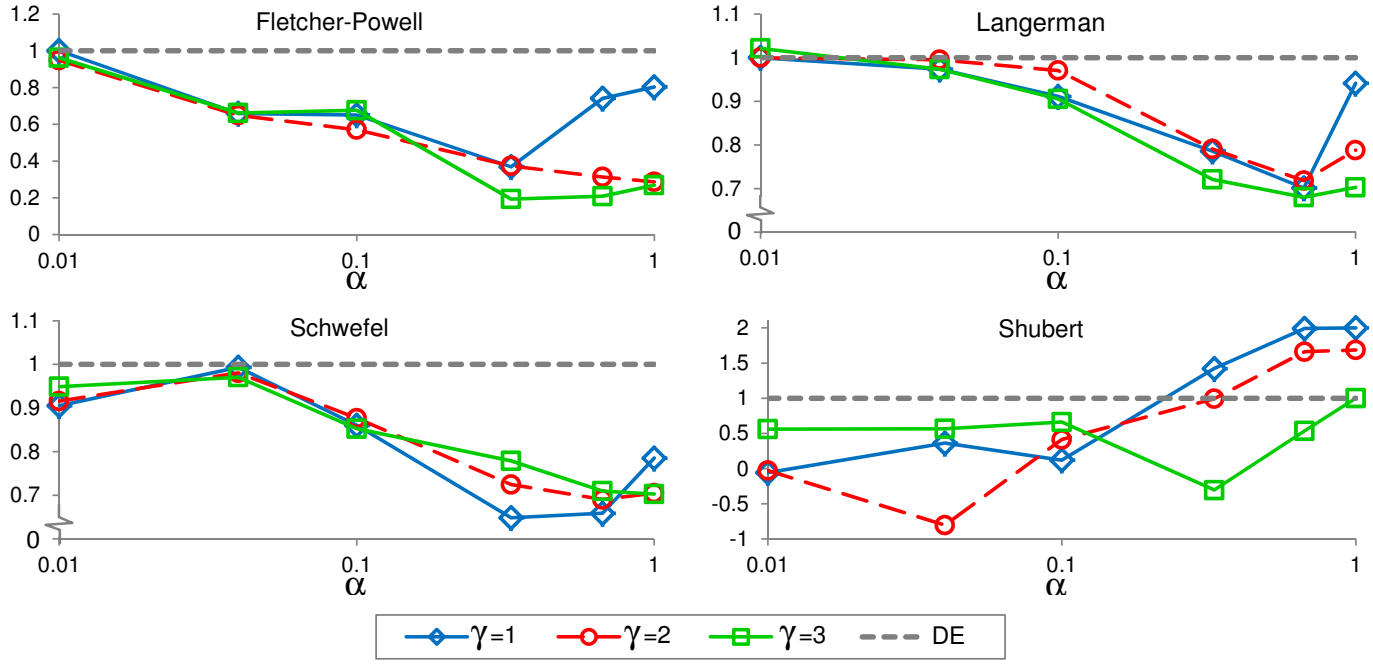

Fig. 1. Relative mean performance of modified DE with varying initial threshold α, using linear ($\gamma = 1$), quadratic ($\gamma = 2$) and cubic ($\gamma = 3$) decay functions. Mean values have been scaled such that standard DE is 1, with values below this indicating improved performance. Values only exist at the lozenges; lines have been added as a visual guide to trends. The y-axes for Langerman and Schwefel have been truncated to show greater detail.

not their relative cost, and it is solution evaluation that is often the most costly component of optimisation problems of interest, when a potential move is rejected an alternative move is generated (randomly selecting a new *base*, x_{r1} and x_{r2}). This allows the algorithm to make progress when the range of potential moves includes those both above and below the threshold. If no acceptable move can be generated after five attempts then no child solution is produced for that *target* that generation. Note, however, that in the current implementation this solution still counts *as if it had been evaluated*. While allowing up to four additional attempts to create an acceptable solution adds to the processing time of the algorithm, it still results in fewer distance calculations than would be required under CrowdingDE, especially as population size is increased to tackle higher dimensional problems. Moreover, the distance calculation can be performed concurrently with solution generation: each dimension j of *new* is either $|F \cdot (x^j_{r1} - x^j_{r2})|$ units or $|target^j - base^j|$ units from *base*.

The threshold is governed by a similar rule to that used in a prior PSO scheme [3], in which an initial threshold is selected and then decays over the course of a run:

$$threshold_t = (\alpha \cdot d) \cdot \left(\frac{n-t}{n}\right)^{\gamma} \tag{3}$$

where $\alpha \in [0, 1]$ is the fraction of the main space diagonal d that represents the initial value of *threshold*, n is the total number of iterations, t is the current iteration and γ controls the decay rate of the threshold.

IV. Experimental Results

The performance of DE/rand/1/bin was compared with an equivalent DE algorithm with the convergence control mechanism described above. The shared settings for the two algorithms were $Np = 20$, $Cr = 0.9$ and $F = 0.8$, with 5,000·D =100,000 function evaluations. Initial testing found that larger population sizes resulted in the algorithm failing to converge within the allowed number of function evaluations. If the algorithm is set up so that it doesn't converge, a diversity control mechanism is unnecessary. However, since DE's local search ability requires that the population converge, Np and F were chosen so that, across the range of problems studied, it was likely to converge within the evaluations limit.

A. Multi-modal problems

The following multi-modal functions were used in a first set of experiments: Fletcher-Powell [18] ($x \in [-\pi, \pi]$), Langerman (with number of Gaussian ripples $m = 7$, $x \in [0, 10]$), Schwefel ($x \in [-500, 500]$) and Shubert ($x \in [-10, 10]$). In each case the objective was minimisation.

The threshold parameters examined were from the sets $\gamma \in \{1, 2, 3\}$ and $\alpha \in \{0.01, 0.04, 0.1, 0.33, 0.67, 1.0\}$. Note that $\alpha = 1$ initially excludes *all* possible moves, although the threshold declines thereafter. This extreme value was included so that the modified algorithm's behaviour as α is varied could be observed.

The standard DE and modified DE (with each combination of threshold parameters) were run across 25 random seeds. Fig. 1 shows how the performance of the modified DE varies with α and γ. The results for each function have been scaled so

TABLE I
DETAILS FOR BEST COMBINATIONS OF α AND γ BY PROBLEM.

	Standard DE		Modified DE		Parameters			
Problem	mean	stdev	mean	stdev	α	γ	%-diff	*t*-test
Fletcher-Powell	16852.4	22693.6	3516.7	22972.2	0.67	3	**79%**	**0%**
Langerman	-0.31	0.11	-0.41	0.08	0.67	3	**32%**	**0%**
Schwefel	2352.6	487.7	1549.8	765.0	0.67	1	**34%**	**0%**
Shubert	-6.4e+21	1.0e+22	-1.8e+22	5.8e+21	0.04	2	**180%**	**2%**

that the mean DE result is 1. Values below 1 indicate improved performance.

Table I shows the mean and standard deviation (stdev) for results for standard DE and the best achieving modified DE. Also shown is the percent improvement between standard and modified DE (%-diff), where positive values indicate that the modified approach performed better. Paired *t*-tests were used to compare the distributions of results for standard and modified algorithms.[4] Statistically significant improvements with a p-value of 5% or lower are bolded in this and subsequent tables. Although only the best results are presented in the table, most of the combinations that produced an improvement (see Fig. 1) were statistically significant at the 1% level, with many others significant at the 5% level.

Examination of the plots in Fig. 1 shows that each problem has a "preferred" value of the initial threshold α, below which the modification has less impact on the algorithm's behaviour and above which it is likely too effective in preventing convergence (and thus retards the exploitation of the best found region). Nevertheless, the range of effective values is quite broad on these problems, indicating that the modification has a real, positive effect. Moreover, adjustment of the intial threshold α is more important than the particulars of the decay function used.

1) Local Optima Analysis: To further examine the impact of the modified solution generation scheme additional tests were conducted in which a local optimiser was used to determine the number of distinct local optima covered by the population at each iteration. The particular local optimiser used conducts a deterministic dimension-by-dimension search with an initial step size of 1% of the range, which is decreased in magnitude by 10% when a non-improving move is made. The local optimiser proceeds through 70 diminishing step sizes, which was found experimentally to find local optima quite reliably. To count the number of distinct local optima, each newly accepted solution has a copy locally optimised, with the position of this optimised version compared with the local optima found for other population members.

The left side of Fig. 2 plots the average number of distinct local optima at each iteration for standard DE and the best performing modified DE. The right side of the figure plots, for the same problems, the average value of the best local optimum covered by the population at each iteration. The plots show averages across 25 runs of each algorithm to illustrate general trends rather than the behaviour of any individual run. These results confirm that the modified DE maintains the ability to reach a greater number of local optima for longer than the standard algorithm, with variability depending on the problem being solved. In all but the Langerman function the modified algorithm eventually converges to a single locally optimal area. However, in the case of the Langerman function the population does converge to 6% of its original spread; the search surface of this function has many small local optima. The plots on the right indicate that this prolonged period of exploration in each problem allows the algorithm to locate improved areas in which to focus its later exploitative search.

B. Black-box Optimization Benchmarks

Although the modified DE is intended to improve DE's peformance on multi-modal problems, it is worth examining its peformance on a range of other problems, including both uni-modal problems and multi-modal problems with little overall structure (that could be exploited by a coarse-grained search to identify the best regions of the search space). Thus additional tests were conducted on the Black-Box Optimization Benchmarking (BBOB) functions [19]. The BBOB problems are divided into five sets: (1) separable functions; (2) functions with low or moderate conditioning; (3) unimodal functions with high conditioning; (4) multi-modal functions with adequate global structure; and (5) multimodal functions with weak global structure. Sets 4 and 5 are of greatest interest in this work.

To match both the experiments described in the previous section and typical usage of the BBOB set, the number of dimensions $D = 20$, with standard DE or the modified variant run 25 times (five trials on each of the first five instances of each BBOB function)[5] for 5,000$\cdot D$ = 100,000 function evaluations. As the optimal value is known for each function–instance combination, results are reported as the absolute difference between the best solution found and the relevant optimum. The threshold parameters examined with these problems were from the sets $\gamma \in \{1, 2, 3\}$ and $\alpha \in \{0.01, 0.04, 0.1, 0.33\}$. As standard DE (and the modified algorithm) can solve function 1 (sphere) in every case, no comparison data is reported for that function.

Table II presents the same comparison information as in the previous section: mean and standard deviation for standard and modified DE algorithms, the best performing settings for α and γ, the percentage difference and result of a paired *t*-test. Across all problems, the modified DE only failed to achieve

[4]Runs of the two approaches using the same random seed (hence, same initial population) were paired for the purposes of statistical comparison.

[5]Each instance of a BBOB problem is shifted within the search space bounds so that its optimum is in a different location.

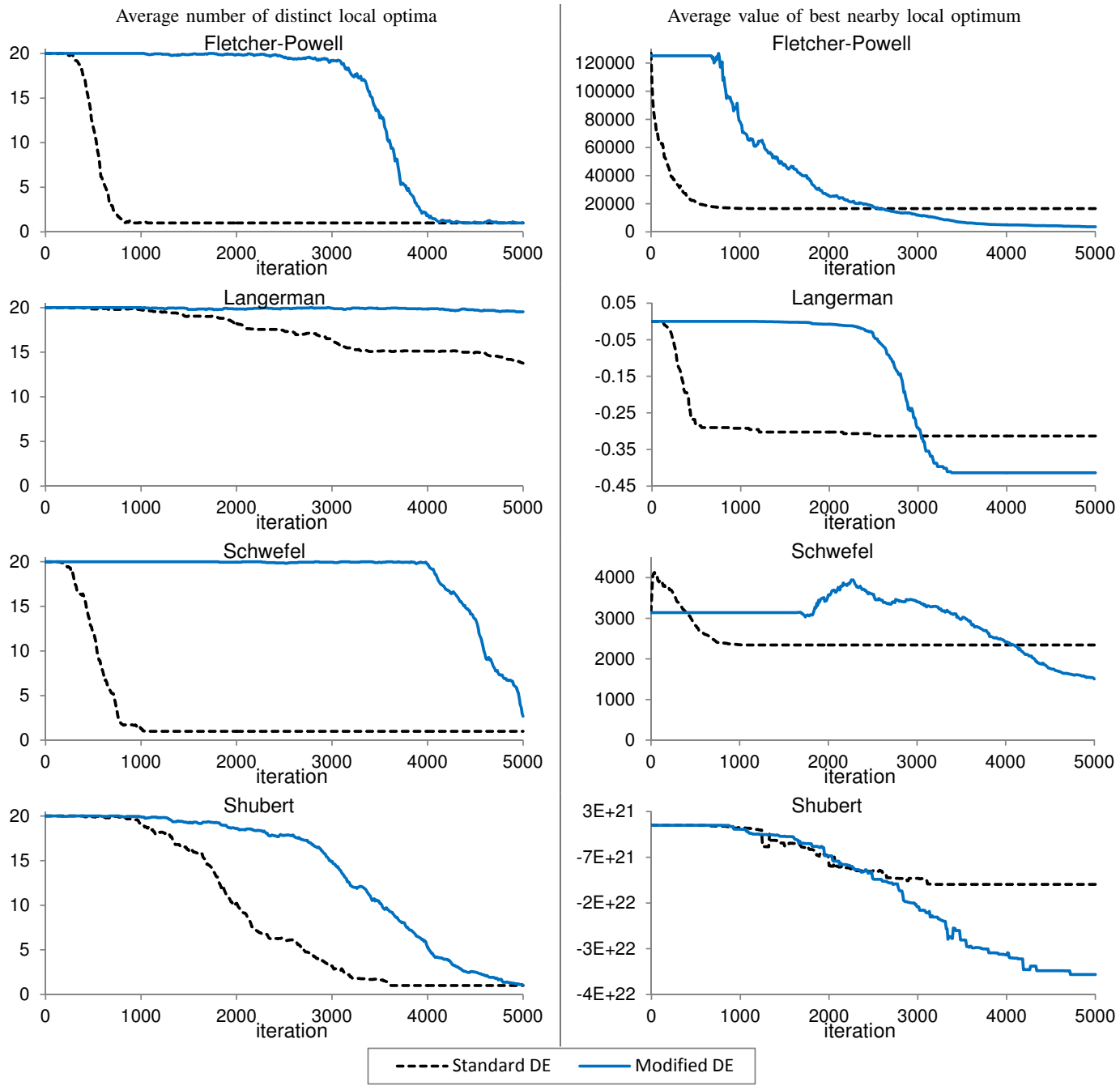

Fig. 2. Number of distinct local optima covered by standard and modified DE populations. The results are averages (by iteration) across 25 runs, smoothed by plotting every 10^{th} iteration. The settings used by the modified DE correspond to the best performing combinations shown in Table I.

an improved result on one problem. Across the remaining problems it can achieve sizable improvements, with 11 of these statistically significant at or below the 5% level. In the two multi-modal, non-separable sets (4 and 5) the modified DE can achieve statistically significant improvements of 16–62% in seven of the 10 problems.

While there is some variability in which values of α and γ produce the best performance, low values of α (0.01–0.04) were frequently effective. In those cases where the largest test value of 0.33 worked best it appears likely that it is due to the function being solved. For instance, functions 21 and 22 are essentially the same, but with a different number of Gaussian peaks.

Analysis of the number of distinct local optima "covered" by the two algorithms during their runs revealed that standard DE does not converge within the function evaluation limit for functions 19, 23 and 24 (which contain a large number of small local optima). The modified DE achieved only slight improvements on these problems, compared with large improvments on the other multi-modal problems. This suggests

TABLE II
BEST COMBINATIONS OF α AND γ BY PROBLEM FROM BBOB

		Standard DE		Modified DE		Param.			
Set	fn	mean	stdev	mean	stdev	α	γ	%-diff	t-test
	1	0.00e+0	0.00e+0						
	2	6.44e+1	2.93e+2	8.12e-6	4.68e-6	0.1	2	100%	15%
1	3	2.29e+1	1.09e+1	1.42e+1	5.96e+0	0.1	3	**38%**	**0%**
	4	2.62e+1	8.33e+0	1.90e+1	6.08e+0	0.04	1	**28%**	**0%**
	5	2.16e+0	8.59e+0	0.00e+0	0.00e+0	0.1	1	100%	11%
	6	1.51e+1	2.99e+1	2.74e+0	9.69e+0	0.33	2	**82%**	**1%**
2	7	2.14e+0	2.89e+0	1.30e+0	7.23e-1	0.04	3	**39%**	**8%**
	8	4.85e-1	1.29e+0	3.49e-1	9.66e-1	0.01	3	28%	30%
	9	1.00e+1	1.58e+1	8.67e+0	2.06e+0	0.01	3	13%	33%
	10	8.76e+2	2.54e+3	6.95e+2	9.69e+2	0.04	3	21%	31%
	11	2.15e+0	1.57e+0	1.70e+0	9.45e-1	0.01	3	**21%**	**9%**
3	12	2.16e+4	1.06e+5	3.47e+0	5.05e+0	0.33	3	100%	16%
	13	2.08e+0	1.79e+0	5.63e-1	4.75e-1	0.04	2	**73%**	**0%**
	14	6.69e-5	2.78e-5	1.93e-4	7.76e-5	0.01	3	-189%	0%
	15	6.79e+1	3.35e+1	4.72e+1	3.13e+1	0.04	3	**30%**	**2%**
	16	1.62e+1	3.81e+0	1.36e+1	5.57e+0	0.01	1	**16%**	**0%**
4	17	5.98e-2	1.42e-1	2.28e-2	5.73e-2	0.1	3	**62%**	**5%**
	18	2.85e-1	2.72e-1	2.12e-1	2.07e-1	0.01	2	**26%**	**1%**
	19	4.74e+0	5.71e-1	4.60e+0	4.94e-1	0.04	3	3%	11%
	20	9.65e-1	2.43e-1	6.73e-1	3.02e-1	0.04	1	**30%**	**0%**
	21	5.84e+0	8.65e+0	2.44e+0	1.64e+0	0.33	3	**58%**	**3%**
5	22	4.20e+0	5.15e+0	2.03e+0	1.81e+0	0.33	1	**52%**	**3%**
	23	1.97e+0	3.57e-1	1.92e+0	2.27e-1	0.1	1	3%	28%
	24	1.31e+2	1.22e+1	1.28e+2	1.27e+1	0.01	1	2%	10%

that if standard DE fails to converge in a particular search landscape then any mechanism to further delay converge is of limited benefit.

C. Comparison with Crowding DE

Given the surface similarity of the proposed modified DE to CrowdingDE, additional experiments were conducted with a re-implementation of the CrowdingDE algorithm. The initial work on CrowdingDE [2] applied the technique to a range of 2D problems using a population of 100. Adapting this to the 20D problems considered here results in a population of 1000 solutions, which tests demonstrated to be unable to perform well (in comparison to standard or the modified DE) due to the fixed number of function evaluations and consequent small number of updates per population member. Limited parameter tuning found that the algorithm's performance was improved by reducing the population size to the 20 used here, and reducing the scale factor F from 0.8 to 0.5.

Table III compares the mean solution value found by CrowdingDE with the modified DE results from Tables I and II. A positive percentage difference indicates that the modified DE performed better than CrowdingDE.[6] Although only the best modified DE results are used in the comparison, on the four multi-modal problems examined in Section IV-A the modified algorithm performed statistically significantly better than CrowdingDE across *all* configurations except on Langerman when $\alpha \leq 0.33$, where it performed equivalently. On the BBOB set the modified DE performs better than CrowdingDE on most problems. Those where the modified DE was clearly poorer than CrowdingDE include four (fn 6, 12–14) that are unimodal and hence not the target kind of problem for either algorithm.

[6]Both CrowdingDE and the modified DE were able to find the optimal solution on every run for BBOB functions 1 and 5, so no t-test was carried out.

TABLE III
CROWDINGDE RESULTS, DIFFERENCE FROM MODIFIED DE

	CrowdingDE			
Problem	mean	stdev	%-diff	t-test
BBOB 1	0.00E+00	0.00E+00	0%	
BBOB 2	4.42E-03	2.17E-02	100%	16%
BBOB 3	7.78E+01	1.59E+01	**82%**	**0%**
BBOB 4	9.27E+01	1.48E+01	**80%**	**0%**
BBOB 5	0.00E+00	0.00E+00	0%	
BBOB 6	5.18E-04	6.05E-04	-5e+5%	9%
BBOB 7	5.05E-01	5.15E-01	-157%	0%
BBOB 8	7.22E+00	2.97E+00	**95%**	**0%**
BBOB 9	1.10E+01	2.83E+00	**21%**	**0%**
BBOB 10	1.18E+03	7.73E+02	**41%**	**4%**
BBOB 11	3.80E+00	2.94E+00	**55%**	**0%**
BBOB 12	2.52E+00	4.59E+00	-38%	26%
BBOB 13	2.76E-02	5.98E-02	-1942%	0%
BBOB 14	1.05E-04	4.37E-05	-84%	0%
BBOB 15	1.24E+02	8.96E+00	**62%**	**0%**
BBOB 16	1.80E+01	1.88E+00	**24%**	**0%**
BBOB 17	6.42E-02	5.57E-02	**64%**	**0%**
BBOB 18	4.58E-01	3.74E-01	**54%**	**1%**
BBOB 19	4.94E+00	3.65E-01	**7%**	**1%**
BBOB 20	2.56E+00	1.48E-01	**74%**	**0%**
BBOB 21	2.82E+00	2.69E+00	13%	30%
BBOB 22	1.57E+00	1.32E+00	-30%	12%
BBOB 23	2.08E+00	2.66E-01	**8%**	**1%**
BBOB 24	1.47E+02	8.07E+00	**13%**	**0%**
Fletcher-Powell	7.90E+04	2.23E+04	**96%**	**0%**
Langerman	-3.32E-01	1.10E-01	**24%**	**1%**
Schwefel	3.70E+03	4.55E+02	**58%**	**0%**
Shubert	-7.32E+15	8.85E+15	**2.4e+8%**	**0%**

The comparatively poor performance of CrowdingDE is plausibly explained by the differing design intentions between it and the modified DE proposed here. CrowdingDE is intended to identify and maintain multiple optima. Yet the nature of DE's search means that local improvements must come at the expense of global search; conversely if multiple good regions are maintained then little local improvement can take place. The modified DE proposed here is intended to prolong global search before allowing the algorithm to converge "naturally". Thus the explicit, decreasing threshold is a key difference between the two approaches that appears to provide an advantage in finding good, single solutions.

D. Discussion

The proposed modification to DE has demonstrable effect in slowing the rate of population convergence (see Fig. 2). The resultant prolonged period of exploration has benefits across a range of different problems, including most of the multi-modal problems tested, which are those for which it was specifically intended. On the multi-modal problems where it provided only marginal improvements over a standard DE implementation, standard DE is found to converge very slowly anyway, so the additional slowing of convergence the modification provides is less beneficial.

Two key differences exist between the proposed approach

and prior techniques for slowing convergence or maintaining exploration in DE. First is the identification of the most likely existing population member with which a new solution may form a crowd: the *base* solution used in the new solution's generation. This allows for an extremely efficient distance calculation to be performed to determine if the new solution should be kept, *before a potentially costly function evaluation is performed*. Existing crowding techniques examine solutions after they have been evaluated and may need to examine a large number of pairs of solutions for similarity. The second key difference is the use of a decaying distance threshold around the *base* solution, within which the new solution may not be placed. This use of a threshold confers two advantages: it provides explicit control over how close population members are allowed to be (crowding techniques can replace distant solutions merely because they happen to be the closest); and by decaying it allows the population to converge, which is a necessary part of DE's search behaviour.

The introduction of a threshold does introduce additional parameters to the algorithm. However, considering the range of effective values illustrated in Fig. 1, there is one important new parameter, the initial threshold α, with the decay rate γ providing a crude way to control its decay. Given that no one algorithm, and hence no single set of control parameters, will suit every optimisation problem [20], some tuning of an additional parameter is not particularly onerous given the large benefits that can result. In order that the γ parameter can be removed, future work will examine the use of adaptive techniques that set the threshold based on the current state of the population or recent algorithm behaviour.

V. Summary

Population diversity is critically important to maintaining exploration in a DE search. Previous approaches to slowing convergence in DE, adapted largely from the crowding and niching stable of techniques, can be computationally costly or fail to compare new solutions with the most appropriate existing population member. This paper proposed a simple intervention strategy that compares each new solution with the population member it is most likely to be near, the *base* solution used in its construction. Moves that are below a threshold are disallowed, a decision that can be taken before the new solution has been evaluated, which can be computationally costly in some real world problems. By having the threshold decrease during the algorithm's run, the algorithm is allowed to ultimately converge and thus refine the best solution(s) found during exploration. Comparisons with a standard DE algorithm and CrowdingDE on a number of multi-modal problems indicate that the proposed technique can achieve real and sizable improvements.

References

[1] K. De Jong, "An analysis of the behavior of a class of genetic adaptive systems," PhD thesis, Dept. of Computer and Communication Sciences, University of Michigan, 1975.

[2] R. Thomsen, "Multimodal optimization using crowding-based differential evolution," in *IEEE Congress on Evolutionary Computation, CEC 2004*. Portland, Oregon: IEEE Press, 2004, pp. 1382–1389.

[3] S. Chen and J. Montgomery, "A simple strategy to maintain diversity and reduce crowding in particle swarm optimization," in *24th Australasian Joint Conference on Artificial Intelligence (AI2011)*. Perth, Australia: Springer, 2011.

[4] S. Chen, C. Xudiera, and J. Montgomery, "Simulated annealing with thresheld convergence," in *IEEE CEC 2012*, 2012.

[5] E. Mezura-Montes, J. Velázquez-Reyes, and C. A. Coello Coello, "A comparative study of differential evolution variants for global optimization," in *Genetic and Evolutionary Computation Conference (GECCO'06)*, Seattle, Washington, USA, 2006, pp. 485–492.

[6] S. W. Mahfoud, "Niching methods for genetic algorithms," PhD thesis, University of Illinois, 1995.

[7] X. Li, "Efficient differential evolution using speciation for multimodal function optimization," in *Genetic and Evolutionary Computation Conference (GECCO 2005)*, Washington DC, USA, 2005, pp. 873–880.

[8] D. Zaharie, "A multipopulation differential evolution algorithm for multimodal optimization," in *10th MENDEL International Conference on Soft Computing*, Brno, Czech Republic, 2004, pp. 17–22.

[9] J. Rönkkönen and J. Lampinen, "On determining multiple global optima by differential evolution," in *Evolutionary and Deterministic Methods for Design, Optimization and Control, Proceedings of Eurogen 2007*, Jyväskylä, Finland, 2007, pp. 146–151.

[10] J. Rönkkönen, "Continuous multimodal global optimization with diferential evolution based methods," PhD thesis, Lappeenranta University of Technology, 2009.

[11] J. Montgomery, "Crossover and the different faces of differential evolution searches," in *IEEE CEC 2010*. Barcelona, Spain: IEEE, 2010, pp. 1804–1811.

[12] K. Price, R. Storn, and J. Lampinen, *Differential Evolution: A Practical Approach to Global Optimization*. Berlin: Springer, 2005.

[13] D. Zaharie, "Critical values for the control parameters of differential evolution algorithms," in *MENDEL 2002, 8th International Conference on Soft Computing*, R. Matoušek and P. Ošmera, Eds., Brno, Czech Republic, 2002, pp. 62–67.

[14] J. Montgomery and S. Chen, "An analysis of the operation of differential evolution at high and low crossover rates," in *IEEE CEC 2010*. Barcelona, Spain: IEEE, 2010, pp. 881–888.

[15] J. Montgomery, "Differential evolution: Difference vectors and movement in solution space," in *IEEE CEC 2009*. Trondheim, Norway: IEEE Press, 2009, pp. 2833–2840.

[16] ——, "Erratum: Differential evolution: Difference vectors and movement in solution space," Swinburne University of Technology, Tech. Rep. SUTICT-TR2009.02, 16 June 2009.

[17] ——, "The effect of different kinds of move in differential evolution searches," in *4th Australian Conf. on Artificial Life*, ser. LNAI, K. Korb, M. Randall, and T. Hendtlass, Eds., vol. 5865. Melbourne, Australia: Springer, 2009, pp. 272–281.

[18] R. Fletcher and M. J. D. Powell, "A rapidly convergent descent method for minimization," *Comput. J.*, vol. 6, pp. 163–168, 1963.

[19] N. Hansen, S. Finck, R. Ros, and A. Auger, "Real-parameter black-box optimization benchmarking 2009: Noiseless functions definitions," INRIA, Technical Report RR-6829, 2009.

[20] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Trans. Evol. Comput.*, vol. 1, no. 1, pp. 67–82, 1997.